Fusion-layer-based Machine Vision for Intelligent Transportation Systems

by

Yajun Fang

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineer and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2010

Author .....
Department of Electrical Engineering and Computer Science
May.21, 2010

Certified by .....
Berthold K.P. Horn
Professor of Electrical Engineer and Computer Science Department
Thesis Supervisor

Accepted by.....
Terry Orlando
Chairman, Department Committee on Graduate Students

Fusion-layer-based Machine Vision for Intelligent Transportation Systems

by

Yajun Fang

Submitted to the Department of Electrical Engineering and Computer Science
on May.21, 2010, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Electrical Engineer and Computer Science

Abstract

Environment understanding technology is very vital for intelligent vehicles that are expected to automatically respond to fast changing environment and dangerous situations. To obtain perception abilities, we should automatically detect static and dynamic obstacles, and obtain their related information, such as, locations, speed, collision/occlusion possibility, and other dynamic current/historic information. Conventional methods independently detect individual information, which is normally noisy and not very reliable. Instead we propose fusion-based and layered-based information-retrieval methodology to systematically detect obstacles and obtain their location/timing information for visible and infrared sequences. The proposed obstacle detection methodologies take advantage of connection between different information and increase the computational accuracy of obstacle information estimation, thus improving environment understanding abilities, and driving safety.

Thesis Supervisor: Berthold K.P. Horn
Title: Professor of Electrical Engineer and Computer Science Department

Acknowledgments

I am heartily thankful to my supervisor, Dr.Ichiro Masaki and Prof.Berthold Horn, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject. I really appreciate the wonderful research opportunities, great research projects, and creative research environment that they provide me. It is them who show me the wonder of doing research in this field and teach me the rigorous research attitude and habit. I could not express how much I appreciate their guidance, sharp insights, creative feedback and comments that provide me many eye-opening ideas on my research. It is my honor to be able to learn from them on how to summarize, how to think clearly, how to actively listen and communicate, how to prepare and give presentation, and most importantly, what kinds of attitude we should hold for research, career and future life. Without such training, endless care and strong support, it would be impossible for me to finish my work and to leave with three international awards. It is my great pleasure to study and grow under their guidance in different aspects. I am impressed by their wide interests, passion and curiosity toward the world. Their research methodologies, life attitude, endless patience, calm and inner peace, sincere care and wise guidance have great impact on me. I am so glad to have them as my role model that I can continuously learn from, for now, and for future in my life.

The time at MIT is an unforgettable experience carrying so much life significance in my life. I benefit so much from its excellent programs and made huge progress on skill development in academic as well as many other aspects. It is my great fortune to study and to grow at MIT and to have huge improvement on personality and characters. I have met many wonderful people at MIT who help me to be a better and stronger person than I was, and make my MIT experience so wonderful. I sincerely thank all MIT faculties/staffs/classmates/friends/students who have helped me to go through the life journal. Their happy smile, sincere care and encouragement, wonderful support and vision always warm me up and mean a lot to me.

Thank Prof. William Grimson, Prof. Seth Teller for all their advices, comments, suggestions and encouragement on my thesis work, defense and area/oral exams. Thank Prof.

Alexandre MEGRETSKI, Prof. Alan S Willsky, for their counseling, mentoring and encouragement on my course study. Thank Dr. Keiichi Yamada, Dr. Yoshiki Ninomiya, Mr. Sumio Yokomitsu for the wonderful collaboration experience from which I learned a lot. Thank Dr. Uwe Franke, Prof. Alberto Broggi, Dr. Michel Parent, Prof. Umit Ozguner and Prof. Charles Thorpe for their advice on ITS research and their support for my career development. Thank Prof.William M Wells, Prof.Freeman Bill for their teaching and mentoring on machine vision research and on my area exams.

Thank Lynn Roberson, Blanche Staton, Toni P. Robinson, Jane Dunphy, Prof. Polina Golland, and all other wonderful people who support me and other MIT female students whenever we feel puzzled about our life. Thank Ms. Marilyn A. Pierce, Ms. Janet Fischer, Ms. Margaret Flaherty, Ms. Maysoon Hamdiyyah, Ms. Debroah Hodges-Pabon and Ms. Valerie DiNardo, for taking care of my academic needs and emotional support all these years. Thank Love Nicole, Chapman Robbin, Tamara Williams, Niu Chaowei for all their companionship, help, care and support whenever I need. Thank Orcun Kurugol, Matthew T Farrell, William Frederick Herrington, Berkin Bilgic, Marcelo Mizuki, Jason Bergendahl, Duangmanee Putthividhya, Fiore Paul, Cheng Haiqian, Chen Bikui for all the wonderful discussion, group learning and all the support when working together.

Thank all members of my big family for their long term financial and emotional support. Thank them for their trust, encouragement and blessing. Thank Dr.Yan Xin for teaching YXLST® that gives me energy, courage and inner peace, happiness and positive attitude during my thesis work. Thank all members of my special YXLST® family for learning together and being there for me whenever and wherever I need.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the project.

Contents

1 Introduction **37**

1.1 The Challenges of Current Transportation Systems 37

1.2 Intelligent Transportation Systems 39

1.3 Expectation on and Challenges of Reliable Collision Avoidance Technologies 40

1.3.1 The status of current safety enhancement research 40

1.3.2 Active driver Assistance Systems 41

1.3.3 Development of Auto Electronics 44

1.3.4 Cooperated Sensors and Automatic Environment Understanding . . 45

1.4 Fusion-based and Layer-based Scheme to Obtain Obstacle Information . . . 46

1.5 Other ITS Components 48

1.5.1 An Integrated Information/Physical Infrastructure[32] 48

1.5.2 Advanced Transportation Management[32] 50

1.5.3 ITS Deployment 52

2 Target detection and 3D segmentation based on motion and distance range **57**

2.1 Introduction 58

2.1.1 ITS requirements 58

2.1.2 Typical vision-based segmentation algorithms 58

2.1.3 Fusion of binocular stereo and radar for 3D information 60

2.2 Information Fusion: Motion-based Target Depth Detection 63

2.2.1 Motion-based Correspondence Matching Criteria 64

2.2.2 Depth Range Detection through Disparity Histogram 69

2.2.3 Target Depth Detection Results 69

2.3 Fusion-based Layer-based 2D Segmentation 73
2.3.1 Step 1: Depth-Based Edge Layer Separation 74
2.3.2 Step 2: Depth-Based Target Contour Discrimination 75
2.3.3 Step 3: Depth-Based Target Boundary Determination 76
2.3.4 Segmentation Results 76
2.3.5 Performance comparison 79
2.4 Discussion 81
2.4.1 Two correspondence matching 81
2.4.2 Robustness of the algorithm 81
2.4.3 Computational loads 82
2.4.4 Limitations and possible solutions 83

3 Target detection and 3D segmentation under heavy background noise based on motion and distance range 85

3.1 Introduction 85
3.2 Background Detection and Removal 87
3.2.1 The definition of distance-based background 87
3.2.2 Distance-based Background Separation 87
3.2.3 The Impact of Background Noise 90
3.2.4 Distance-based Background Removal 93
3.3 Motion-based target segmentation 94
3.3.1 Similarity and saliency of motion vectors for target objects in distance-based edge layers 96
3.3.2 Expansion of segmentation region based on motion similarity . . . 97
3.4 Discussion: the principle of layer-based and fusion-based algorithms 100
3.4.1 Fusion of depth and motion information 101
3.4.2 Static-block elimination and moving-target detection 102
3.4.3 Ability to detect multiple objects of different shape 104
3.4.4 The features of our fusion algorithms 105
3.5 Summary 105

4 Fusion-based Layer-based Pedestrian Detection based on infrared cameras 109

4.1 Introduction . . . 111

4.1.1 Performance Index . . . 111

4.1.2 Challenges and Previous works on Pedestrian Detection with Infrared Images . . . 113

4.1.3 The Methodology and Principle for Fusion-based Layer-based Shape-Independent Pedestrian Detection . . . 119

4.2 Automatic Pedestrian Segmentation . . . 120

4.2.1 Horizontal Segmentation . . . 120

4.2.2 Vertical Segmentation within Horizontally Segmented Image Stripes 124

4.3 Classification . . . 128

4.3.1 Histogram-based Classification . . . 128

4.3.2 Inertia-based Classification . . . 130

4.3.3 Contrast-based Classification . . . 132

4.3.4 Multi-dimensional Classification Feature . . . 135

4.3.5 Comparison with Conventional Classification Feature . . . 137

4.4 Performance Evaluation . . . 138

4.4.1 Test Sequences . . . 139

4.4.2 Segmentation Performance . . . 140

4.4.3 Classification Performance . . . 141

4.5 Summary and Future work . . . 143

5 A Layered-based Fusion-based Approach to Detect and Track the Movements of Pedestrians through Partially Occluded Situations 159

5.1 Introduction . . . 159

5.1.1 Traditional Algorithms for Initial Segmentation and their Limitation 160

5.1.2 Traditional Algorithms for Human Tracking and their Limitations . 161

5.1.3 Layered-based and Fusion-based Dynamic Tracking of Pedestrians 162

5.2 Layered-based Image Separation and Initial Segmentation . . . 163

5.2.1 Initial Horizontal Segmentation for Visible Images from Single Camera . . . 163
5.2.2 Initial Vertical Segmentation within Horizontal Segmented Stripes . 188
5.3 Fusion-based Layered-based Tracking . . . 189
5.3.1 Template with different poses . . . 191
5.3.2 Layered-based Tracking . . . 193
5.3.3 Prediction Models . . . 194
5.3.4 Fusion-based Tracking . . . 196
5.4 Results . . . 198
5.4.1 Performance Evaluation . . . 200
5.5 Discussion: the Features & Advantages of Proposed Algorithms . . . 203
5.5.1 The Advantage of Layered-based Processing . . . 204
5.5.2 The Advantage of Fusion-based Principle . . . 205
5.6 Conclusion . . . 206

6 Fusion-based hierarchical method for direct gradient-based time-to-contact estimation 209

6.1 Introduction . . . 210
6.1.1 Definition . . . 210
6.1.2 Traditional methods and the limitation . . . 214
6.2 Direct method for time-to-contact . . . 217
6.2.1 Linear and nonlinear solutions for different cases . . . 220
6.2.2 Case (I): Translational motion along the optical axis towards a plane perpendicular to the optical axis . . . 222
6.2.3 Case (II): arbitrary translational motion relative to a plane perpendicular to the optical axis . . . 222
6.2.4 Case (III): translational motion along the optical axis relative to an arbitrary plane . . . 224
6.2.5 Case (IV): arbitrary translational motion relative to an arbitrary plane 225
6.2.6 Summary: comparison between different computational models . . 227

6.2.7 Identification of different setups in real applications 228
6.3 Factors affecting computational results 229
6.3.1 Numerical method for partial derivatives 229
6.3.2 Computational areas: full images vs. segmented regions 230
6.3.3 Threshold for time derivative of brightness 231
6.3.4 Four various computational models 232
6.3.5 Different subsampling rates 233
6.4 Hierarchical Time-to-Contact estimation based on fusion of multi-scale subsampling 235
6.4.1 Multi-scale TTC fusion based on condition numbers 235
6.5 Experiments 236
6.5.1 Test image sequences 236
6.5.2 TTC "ground-truth" and evaluation/test scheme 239
6.5.3 Synthetic image sequences: time-to-contact computation based on different computational models and different subsampling rates . . 242
6.5.4 Stop-Motion image sequences with translational motion along the optical axis 252
6.5.5 Stop-Motion image sequences with translational motion off the optical axis 264
6.5.6 Video from a camera mounted on a car 273
6.6 Discussion 278
6.6.1 The impact of different parameters on TTC computation 278
6.6.2 Comparison of proposed algorithm and traditional size-based TTC estimation 282
6.6.3 Multiple-scaled TTC fusion based on condition number 283
6.7 Conclusions 285
6.8 Acknowledgments 287

7 Discussion and Summary: the principle of fusion-based and layered-based schemes 289

7.1 The principle of layered technology: Divide and conquer 290
7.1.1 Disparity-Range-based Edge Layer Separation and Background Removal 291
7.1.2 Vertical-Projection based Layer Separation 292
7.2 Other Layer-based Processing 293
7.2.1 Similarity of Motion Vectors in Separated Layers 293
7.2.2 Layer-based Dynamic Tracking Model 294
7.3 Structural Fusion among Different Operations 294
7.3.1 Segmentation based on Fusion of Segmentation and Classification Features 295
7.3.2 Detection based on Fusion of Initial Segmentation and Dynamic Tracking 296
7.3.3 Detection based on Fusion of Estimation/Update Step for Kalman Filtering 296
7.4 Other Related Fusion technologies 297
7.4.1 Fusion in Time Dimension 297
7.4.2 Fusion in Spatial Dimension 298
7.4.3 Fusion in Physical Connection among Obstacle Features 299
7.5 Conclusion 300

8 Future Work **303**

8.1 Possible in-depth research topics for discussed systems 303
8.1.1 Application of "fusion-based layer-based 3D segmentation and detection" for tilted obstacles 303
8.1.2 Research on pedestrian detection in very cluttered scenes for both visible and infrared sequences 304
8.1.3 Possible research on "time-to-contact estimation" 304
8.2 Additional fusion schemes 305
8.2.1 Application of "horizontal first, vertical scheme" scheme for general obstacles 305

8.2.2 Application of "fusion-layer-based dynamic tracking" scheme for infrared sequences and general obstacles 305
8.3 Other possible applications and summary 306

List of Figures

1-1 Distribution of cost for traffic accidents. For each death, 18 people are hospitalized and 400 are medically attended injuries[70] 38

1-2 (a) General framework of fusion-and-layer based methodology. (b) Framework for 3D segmentation. (c) Framework for night-vision based pedestrian detection. (d) Framework for fusing historical information and dynamic estimation. 55

2-1 An example where radar is not enough. 58

2-2 (a) Traditional method of target detection (b) Proposed depth-based target detection: fusing radar and stereo 62

2-3 Target depth detection system based on fusing binocular frames and motion. Detail for Figure (2-2)(b). 64

2-4 Disparity Constraint for Binocular and Trinocular Stereo. 65

2-5 Examples of positive and negative edges. 65

2-6 Similarity of MV-CE-Curves and Texture Index for two pair of corresponding edge pixels in binocular stereo images. (a) Top/bottom: binocular stereo images and corresponding points from left/right cameras. (b)(c) Left/right: Variation of correlation errors when searching for motion vectors (MV-CE-Curves) for yellow/red points, top/bottom, marked in (a). . . 68

2-7 Binocular stereo images and their corresponding edge-maps. (a) Images from left(top) & right(bottom) cameras at frame No.57. (b) Edge-maps for (a). 70

2-8 Similarity of motion vectors for binocular stereo images. 70

2-9 Disparity histogram for edge pixels in Figure (2-7)(b) and corresponding maching relationship. (a)(b) x coordinate: disparity value. y coordinate: the number of edge pixels within the disparity range. (a1)(a2) Motion-based binocular stereo algorithm using frames No.57 in Fig. (2-7)(a) and No.58 in Fig. (2-8)(a). (b1)(b2) Trinocular algorithm. 71

2-10 Histogram of disparity for Figure (2-7)(frame No.57). 77

2-11 Edge layers corresponding to different disparity ranges for Figure (2-7)(frame No.57) (a) 7-9 (b) 18-20 (c) 22-24 77

2-12 Procedure to locate targets within a depth range which corresponds to disparity range 22-24 for Figure (2-7) (frame No.57). (a) Edge dilation result (b) Edge closing result (c) Segmentation result 78

2-13 Example of detecting the right vehicle at frames No.62, No.69, No.73, No.80. 78

2-14 Example of locating targets at different disparity ranges. Top row: disparity range 7-9 at frame No.57 (Figure (2-7). Bottom row: disparity range 7-9 at frame No.31. Left column: Edge closing operation result. Right column: segmentation result. 79

2-15 Example of locating targets with disparity range 5-6 (frame No.31) (a) Edge Closing Operation Result. (b) Segmentation Results. 79

2-16 Target segmentation results within disparity range 18-20 for frame No.57 (Figure 2-7). An example of "single-sided limitation" where only part of the target is captured, which leads to target size detection error. It happens when only part of target is in the image. 83

2-17 An example of detecting both ball and pedestrian using same initial parameters. 83

3-1 Binocular stereo frames and edge images. (a)(b) Left/Right images at time n. (c)(d) Left/Right images at time $n-1$. (e)(f) Left/Right edge images at n. Images (a)-(d) are courtesy of Daimler-Chrysler Research and Technology. 88

3-2 Disparity distribution 89

3-3 Distance-range-based image separation based on figure (3-2) for figure (3-1)(a)(b). The top and bottom rows correspond to two possible separation. (a1): Background layer at disparity 0-5. (b1): Object layer at disparity range 17-19. (c1): Cleaned object layer at disparity range 17-19. (a2): Background layer at disparity 0-9. (b2): Object layer at disparity range 17-19. (c2): Cleaned object layer at disparity range 17-19. 90

3-4 Distance-range-based segmentation results. (a) Left/Right matching result at disparity range 17-19 (from Figure (3-2) for Figure (3-1)(a)(b). (b) Morphological operation result on (a). (c) Edge-length filtering and segmentation result from (b). (d) Segmentation result drawn on Figure (3-1)(a). . . . 91

3-5 Distance-range-based segmentation results subject to background removal. Case 1: Removed background: Disparity 0-5. Case 2: Removed background: Disparity 0-9. (a) Left/Right matching results at disparity range 17-19 (from Figure (3-2) for Figure (3-1)(a)(b). (b) Morphological operation results on (a). (c) Edge-length-based filtering results on (b). (d) Segmented regions drawn on Figure (3-1)(a). 95

3-6 Comparison of motion vectors for distance-based target edge layer (after background removal)(a) and for the full edge map (b) 96

3-7 Motion-based segment expansion results for cleaned target edge layers that remove background pixels from target edge layers. (a) Bounding region expansion based on aggregating pixels of similar motions in cleaned target edge layers. (b) Segmented regions drawn on Figure (3-5)(c). (c) Segmented regions drawn on Figure (3-1)(a). Case 1 and case 2 respectively correspond to two definitions of background layers. Case 1: background layer with disparity 0-5. Case 2: background layer with disparity 0-9. . . . 99

3-8 Flow chart for segmentation algorithm 101

3-9 Segmentation results based on static box removal in target edge layers and motion region expansion. (a) The result of static block removal for Figure (3-4). (b) The result of motion-based region expansion for seed regions from (a). Top row: Segmented regions, the results of static block removal and motion-based expansion, are plotted on Figure (3-4)(c), the results of Morphological operation and edge-length-based filtering. Middle row: Segmented regions are plotted on Figure (3-4)(a), the original target edge layers. Bottom row: Segmented regions are plotted on Figure (3-1)(a), the original image. 103

4-1 Vision Degradation for Seniors. (Image Source: MIT Age Lab.) 110

4-2 Segmentation/Classification Performance Index Definition. (a): Segmentation side accuracy/efficiency definition. (b): ROC boundary/curve definition for multi-dimensional-feature-based classification: false_alarm_rate (X axis) vs. detection_rate (Y axis). Different points correspond to multi-dimensional-feature-based classification results using different multi-dimensional thresholds. Solid curve is ROC boundary, the upper/left boundary of all classification performance points. Dashed and dotted curves are ROC curves for 1D-feature-based classification. 112

4-3 Algorithm and Performance Comparison for Different Pedestrian-Detection Methods. (a): Detection Performance Comparison. (b): Detection Algorithm Comparison. (c): Default Pedestrian Template for MIT Shape-Independent Method. Image size: 58×21. 118

4-4 The Feature of Bright-Pixel-Vertical-Projection Curves for Infrared Images and Brightness-based Vertical Segmentation Results. For (a)(c): Winter results. For (b)(d): Summer results. (a): Top row: original infrared image in winter. Center row and Bottom row: bright-pixel-vertical-projection curves when using two different thresholds. (b): Top row: original infrared image in summer. Center row: bright-pixel-vertical-projection curve. Bottom row: horizontally segmented image stripes based on projection curve. Note that Several separated stripes shown in the center row seem to be connected. For (c)(d): Brightness-based vertical segmentation results. For all projection curves: X axis: Image column position. Y axis: Number of bright pixels in each column. 146

4-5 Pedestrian segmentation based on two different methods. (a): Bodyline-based geometric pedestrian-size model. (b): Bodyline image. (c): Candidate pedestrian region estimation based on (a) and (b). (d): Bodyline-based segmentation result. 147

4-6 Properties of Brightness-histogram curves for Pedestrian/non-Pedestrian ROIs. (a0): Pedestrian from summer data. Used as default template in our algorithm. (a1): Pedestrian ROIs with different poses. (a2): Non-pedestrian ROIs. (a1)(a2) are segmentation results for winter data. For (b1)(b2): Brightness histograms for (a1)(a2). (b1): demonstrates histogram similarity among winter pedestrian ROIs. (b2): demonstrates the histogram variation among winter non-pedestrian ROIs. For (c1)(c2): Solid lines: Average brightness histogram for winter pedestrian ROIs (b1) and winter non-pedestrian ROIs (b2) respectively. Dashed lines: Histogram curve for summer pedestrian (a0). (c1): demonstrates the histogram similarity between winter pedestrians and summer pedestrian template. (c2): demonstrates the disparity between winter non-pedestrian ROIs and summer pedestrian template. For (b1)(b2)(c1)(c2): X axis: Image intensity range (0-255). Y axis: brightness histogram. 148

4-7 For (a1)(a2): Sample pedestrian regions from 2 sequences (every $five$ frames) to show the variation of pedestrian poses and sizes, which correspond to the Seq2 and Seq3 shown in Figure (4-17)(b1)(c1) and Table 4.1, two pedestrian-detection examples in Section 4.4. (b): Left: two brightness-histogram curves with brightness i (X axis) vs. $h^n(i)$ and $h^m(i)$ (Y axis) that are pixel numbers with brightness i from two image regions. Right: Definition for "histogram variation curve" with all possible histogram variance value $h^n(i) - h^m(i)$ (X axis) vs. variation frequency (Y axis). (c): Collective "histograms variation curve" for 911 pedestrian samples (in 7 sequences), with all possible histogram variation value (X axis) vs. the distribution of histogram variation value from all pedestrian histogram curves and their mean (Y axis). 149

4-8 (a): ROI inertia definition. (b): Distribution of inertia values for all 911 pedestrian samples. X axis: Inertia value. Y axis: Distribution percentage. 150

4-9 Contrast-based non-Pedestrian ROI Removal for ROIs in Figure (4-5)(d). For (a)(b): ROIs and their vertical neighborhood regions on edge map, i.e., vertical-neighborhood-contrast property. (a): For detected non-pedestrian ROIs. (b): For remaining ROIs. (c): Remaining ROIs on the original image. 151

4-10 Contrast Feature Vectors for Pedestrian ROIs and non-Pedestrian ROIs from Seq3 shown in Figure (4-17)(c1) (details in Section 4.4, Table 4.1). Circles: pedestrian ROIs. Dots: non-pedestrian ROIs. (a): 2D "upper-contrast-index" for ROIs. X axis: ROI row-edge index. Y axis: upper row-edge index. (b): 2D "lower-contrast-index" for ROIs. X axis: ROI row-edge index. Y axis: lower row-edge index. 152

4-11 2D Feature Vectors for Pedestrian ROIs and non-Pedestrian ROIs from Seq1 shown in Figure (4-17)(a1) (details in Section 4.4, Table 4.1). For (a)(b): 2D inertia/histogram feature vectors for pedestrian ROIs and non-pedestrian ROIs respectively. X axis: Inertia feature. Y axis: "Histogram Difference," for ROIs and pedestrian template in Figure (4-6)(a0). For (c)(d): 2D inertia/pixel-comparison-based feature vectors for pedestrian ROIs and non-pedestrian ROIs respectively. X axis: Inertia feature. Y axis: "Image-Intensity-Difference" between ROIs and pedestrian template in Figure (4-6)(a0). In Figure (4-11)(a) and (b), histogram feature points for 19.64% of pedestrian ROIs and 16.13% of non-pedestrian ROIs overlap in their data ranges. In Figure (4-11)(c) and (d), pixel-comparison-based feature points for all pedestrian ROIs and 85.74% of non-pedestrian ROIs overlap in their data ranges. The comparison between (a)(b) and (c)(d) shows the advantages of the histogram feature over the shape-dependent pixel-comparison-based feature. 153

4-12 2D Inertia/Histogram Feature Vectors for Pedestrian ROIs and non-Pedestrian ROIs from Seq2 shown in Figure (4-17)(b1) (details in Section 4.4, Table 4.1). X axis: Inertia feature. Y axis: Histogram feature. 154

4-13 2D Inertia/Histogram Feature Vectors for Pedestrian/non-pedestrian ROIs from Seq3 shown in Figure (4-17)(c1) (details in Section 4.4, Table 4.1) after "contrast-based non-pedestrian ROI-removal." Rectangular box: data range of feature vectors for pedestrian ROIs. (a): For removed non-pedestrian ROIs. (b): For original pedestrian ROIs. (c): For remaining non-pedestrian ROIs. X axis: Inertia feature. Y axis: Histogram feature. 154

4-14 Classification Ability Comparison. For (a)-(d): feature points of different definitions that measure the similarity between ROIs in Figure (4-6)(a1)(a2) and the default pedestrian template shown in Figure (4-6)(a0). Circles and dots respectively denote pedestrian ROIs and non-pedestrian ROIs. (a): Conventional 1D Pixel-comparison-based Feature. (b): 1D Histogram-based Feature. (c): 1D Inertia-based Feature. (d): 2D Histogram/Inertia Feature. For (a)-(c): Bottom/Top lines: data ranges for pedestrian/non-pedestrian ROIs. X axis: feature value. The Y axis is only used to separate two lines. The overlap percentage between them over non-pedestrian data range: 48.22%(a), 0%(b), $3.87\%(c)$. For (d): X axis: inertia value. Y axis: histogram difference value. 155

4-15 Segmentation Evaluation for 3 Sample Sequences. Detection Accuracy vs. Efficiency. X axis: frame segmentation side accuracy. Y axis: frame segmentation side efficiency. 155

4-16 Classification Performance Evaluation for Three Sample Sequences. X axis: false_alarm_rate. Y axis: detection_rate. (a): Seq1: ROC for histogram-based classification. (b): Seq2: Histogram/inertia-based classification. Dashed line: ROC for inertia-based classification. Dotted line: ROC for histogram-based classification results. Block points: Histogram/inertia-based classification result. (c1) Seq3: ROC for histogram/inertia-based classification. Dashed line: ROC for inertia-based classification. Dotted line: ROC for histogram-based classification results. Star points: Histogram/inertia-based classification result. (c2) Seq3: ROC for histogram/inertia/contrast-based classification. Dashed line: ROC for inertia/contrast-based classification. Dotted line: ROC for histogram/contrast-based classification results. Star points: Histogram/inertia/contrast-based classification result. . . 156

4-17 Pedestrian-detection performance for Seq1, Seq2, and Seq3. (a1)(b1)(c1): original images. (a2)(b2)(c2): Segmentation results. (a3)(b3)(c3): Classification results. 157

5-1 Three synthetic continuous image frames, their Backward Difference-Images(BDI), Forward Difference-Images(FDI), and Combined Difference-images(CDI) when their foregrounds (ellipses) move at two different speeds: case (1) and (2). (a) The top, middle and bottom rows are three continuous synthetic images frames: *previous, current and next.* Two columns (a1)(a2) respectively correspond to situations when foreground moves fast(a1) and slow(a2). (b) Top/bottom: BDI and FDI for (a). (c)(d) The 1st and 2nd definition of CDI for both cases. The shapes for CDI are equal to the shapes for the *current* frames. 169

5-2 Three synthetic continuous image frames, their Backward Difference-Images(BDI), Forward Difference-Images(FDI), and Combined Difference-images(CDI) when their foregrounds (pedestrians) move at three different speeds: case (1), (2) and (3). (a) The top, middle and bottom rows are three continuous synthetic images frames: *previous, current and next.* Three columns (a1)(a2)(a3) respectively correspond to situations when people walk extremely fast(a1), normal(a2) and extremely slow(a3). For the top row, the horizontal regions of pedestrian images in three frames do not overlap. (b) Top/bottom: BDI and FDI for (a). (c)(d) The 1st and 2nd definition of CDI for three cases. The shapes for CDI are equal to the shapes for the *current* frames. 170

5-3 Example of horizontal segmentation based on the CD-images and CDI-VP-curves. (a) Three continuous sample images with pedestrians (b) Top two figures: BDI and FDI for (a). Bottom two figures: two definitions of CDIs for (a), Def_1, Def_2. (c) Horizontal segmentation based on the vertical projection of CDIs. Top row: CDI-VP curves. Middle row: Edge maps for CDI. Bottom row: Segmentation results. 171

5-4 (a) CDI (1st Def) for uniform images (b) CDI (1st Def) for general cases (c) The comparison of CDI images and original frames. 177

5-5 The three continuous video frames with two different image patterns(a/b) and 12 different movement speeds. From (1) to (12), the relative motion increases. 182

5-6 The changing trend of CD-Images and CDI-VP-curves(Def_1) for three continuous video frames defined in cases (a1)-(a6), (b1)-(b6) of Figure (5-5). CD-Images are plotted in the top row of each figure. CDI-VP curves are plotted in the red curve is the bottom row for each figure. In the bottom rows, the curves in blue, cyan, and green colors respectively correspond to the vertical projections for three original continuous frames, for BDI, FDI. . 183

5-7 The comparison of two definition of CDI-VP-curves for three continuous video frames defined in cases (a1)-(a6), (b1)-(b6) of Figure (5-5). 184

5-8 The comparison of two definition of CDI-VP-curves for three continuous video frames defined in cases (a6)-(a12), (b6)-(b12) of Figure (5-5) 185

5-9 Examples that the horizontal locations of pedestrians correspond to transition peaks in projection-curves. Dotted lines correspond to the boundaries of intersection regions. Solid lines correspond to the locations of pedestrians' heads. We notice that the horizontal locations of people's head are exactly at the transitional peaks of the projection-curves, and the horizontal locations of merged regions at accurately determined by the peak boundaries of projection-curves. 187

5-10 The changing trend of CDI-VP-Curves and the results of segmentation and tracking before and after two people intersected for Sequence VID_1. . . . 188

5-11 (a) Algorithm flowchart. (b1) Flowchart for CDI, CDI-VPC computation. (b2) Flowchart for fusion-based tracking. 190

5-12 The similarity of body-trunk signature for pedestrians with different poses.(a) Segmentation for whole pedestrians. (b) Body-trunk regions. 192

5-13 The process of pedestrian tracking (a) For non-occluded situations where CDI-VPC curves are separated waves. (b) For non-occluded situations where CDI-VPC curves are merged waves. Top row: For Img$(k-1)$. Bottom row: For Img(k).Left part of (a) and (b): CDI-VPC. Right part of (a) and (b): Two continuous image frames. 194

5-14 The segmentation/tracking results before and after two people intersected for Sequence VID_2. One person started to bow during walking. 200

5-15 The segmentation and tracking results before and after two people intersected Sequence VID_3. One person turned 90 degree and changed the direction of walking. 201

5-16 The segmentation and tracking results before and after two people intersected Sequence VID_4. One person first walked backward and then forward. 201

5-17 Tracking results for Sequence VID_1 in Figure (5-10). Left: Pedestrian 1. Right: Pedestrian 2. 202

5-18 Tracking results Sequence VID_2 in Figure (5-14). Left: Pedestrian 1. Right: Pedestrian 2. 202

5-19 Tracking results Sequence VID_3 in Figure (5-15). Left: Pedestrian 1. Right: Pedestrian 2. 202

5-20 Tracking results Sequence VID_4 in Figure (5-16). Left: Pedestrian 1. Right: Pedestrian 2. 202

6-1 Time-to-contact. 210

6-2 Example of FOE(focus-of-expansion). 213

6-3 Projection relationship for a linear object model. S: Linear object size. f: Principal distance. When a linear object is at two different distances Z_1 and Z_2, the image size corresponding to the linear object is respectively s_1 and s_2. 215

6-4 The situation when there is arbitrary translational motion between camera and an plane with arbitrary orientation. 218

6-5 Four cases of relative motion. (a) Case (I): Translational motion along the optical axis towards a planar surface perpendicular to the optical axis. (b) Case (II) Translational motion in an arbitrary direction, with the optical axis perpendicular to a planar surface. (c) Case (III) Translational motion along the optical axis relative to an arbitrary plane. (d) Case (IV) Arbitrary translational motion relative to an arbitrary plane. 221

6-6 Computing cell. The involved cubics E_{xyt} are the elements involved in Equation (6.43). (x, y) corresponds to the location. The z is along the time dimension, t. 229

6-7 The impact of different thresholds of brightness derivative ($E_t\_Threshold$) on TTC computation for four computational models and different sequences. X axis: subsampling parameters which change from 1 to 120. Y axis: TTC values computed based on two continuous image frames (No.3 and No.4 frames). Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. Figure (a)(b): TTC for synthetic sequences SEQ_1 in Figure (6-9) and SEQ_2 in Figure (6-10). (c)(d): TTC for stop-motion sequences with relative motion along optical axis, SEQ_3 in Figure (6-24) and SEQ_4 in Figure (6-25). (e)(f): TTC for stop-motion sequences with relative motion off optical axis, SEQ_5 in Figure (6-32) and SEQ_6 in Figure (6-35). 232

6-8 The impact of different subsampling rates on TTC computation for four computational models and two synthetic sequences, SEQ_1 in Figure (6-9) and SEQ_2 in Figure (6-10). X axis: subsampling parameters which change from 1 to 120. Y axis: TTC values computed based on two continuous image frames (No.3 and No.4 frames). Top row: for SEQ_1 in Figure (6-9). Bottom row: for SEQ_2 in Figure (6-10). Each column, from the left to the right, respectively corresponds to computational case (I), (II), (III), (IV). 234

6-9 Sample frames of Sequence SEQ_1, synthetic image sequence "tipper" (0 – 224 frames). Frame seq: 0, 40, 80, 120, 160, 200. 242

6-10 Sample frames of Sequence SEQ_2, synthetic image sequence "newman" ($0 - 149$ frames). Frame seq: 0, 25, 50, 75, 100, 125. 242

6-11 TTC calculated based on 4 assumptions at subsampling rate 4×4 for Sequence SEQ_1, synthetic image sequence "tipper" compared to true TTC (dashed black line). x axis: frame number; y axis: TTC. All figures are plotted with the same scales. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. 243

6-12 Parameter computation for object motion and object orientation based on Case (II), (III) and (IV) at subsampling rate 4×4 for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: FOE and (p, q) parameters. The FOE coordinates are determinded in a coordinate system with its origin at the left/top corner in images. Figures in the same column are plotted with the same scales. (a)/(c): FOE for case (II)/(IV). FOE coordinates x_0 and y_0 are respectively represented by red and blue lines. (b)/(d): object orientation parameters for case (II)/(IV). Parameters p/f and q/f are respectively represented by red and blue lines. 243

6-13 FOE results plotted on different image frames at subsampling rate 4×4 for Sequence SEQ_1, synthetic image sequence "tipper." (a1)(b1)(c1): FOE results at frame 0, 100, 200 from Case (II). (a2)(b2)(c2): FOE results at frame 0, 100, 200 from Case (IV). This Figure helps to identify the FOE directly by observing image pixels at the same location corresponding to the same object points. 244

6-14 TTC computation (for Case (I)(II) and Case (III)(IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: TTC. Dashed black line: TTC ground-truths. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. All figures are plotted with the same scales. 249

6-15 TTC comparison at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for four cases (Sequences: tipper). Horizontal axis: frame number. Vertical axis: TTC. Dashed black line: TTC ground-truths. 1×1: Red thick lines. 2×2: Green thick lines. 4×4: Blue thick lines. 8×8: Magenta thick lines. 16×16: Cyan thick lines. 32×32: Yellow thick lines. 64×64: Blue thin lines. All figures are plotted with the same scales. 250

6-16 TTC computation (for all cases) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_2, synthetic image sequence "newman." x axis: frame number; y axis: TTC. Dashed black line: TTC ground-truths. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. All figures are plotted with the same scales. 251

6-17 TTC comparison at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for all cases (Sequences: newman). Arbitrary planar orientation and arbitrary translational motion. Horizontal axis: frame number. Vertical axis: TTC. Dashed black line: TTC ground-truths. 1×1: Red thick lines. 2×2: Green thick lines. 4×4: Blue thick lines. 8×8: Magenta thick lines. 16×16: Cyan thick lines. 32×32: Yellow thick lines. 64×64: Blue thin lines. All figures are plotted with the same scales. 251

6-18 FOE computation (for Case (II) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: FOE coordinates x_0 and y_0 respectively represented by red and blue lines. All figures are plotted with the same scales. 253

6-19 FOE comparison for Case (II) and Case (IV) at different subsampling rates(1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64). FOE points are plotted on the first image of synthetic image sequence(tipper). Top two rows: Case (II). Bottom two rows: Case (IV) 254

6-20 FOE computation (for Case (II) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_2, synthetic image sequence "newman." x axis: frame number; y axis: FOE coordinates x_0 and y_0 respectively represented by red and blue lines. All figures are plotted with the same scales. 255

6-21 FOE comparison for Case (II) and Case (IV) at different subsampling rates(1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64). FOE points are plotted on the first image of Sequence SEQ_2, synthetic image sequence "newman." Left column: Case (II). Right column: Case (VI) 256

6-22 Estimated plane orientation parameters (for Case (III) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: parameter $(p/f, q/f)$ respectively represented by red and blue lines. All figures are plotted with the same scales. 257

6-23 Estimated plane orientation parameters (for Case (III) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_2, synthetic image sequence "newman." x axis: frame number; y axis: $(p/f, q/f)$ values respectively represented by red and blue lines. All figures are plotted with the same scales. 258

6-24 Sample frames and their segmentation for Sequence SEQ_3, stop-motion image sequence "new_Bus_Front_5mm"(104 frames). Frame seq: 1, 21, 41, 61, 81, 101. 259

6-25 Sample frames and their segmentation for Sequence SEQ_4, stop-motion image sequence "newCam_Bus_Front_5mm"(81 frames). Frame seq: 1, 16, 31, 46, 61, 76. 259

6-26 The comparison of TTC computation based on full image and the segmented area for different cases at subsampling rate 2×2 for Sequence SEQ_3, stop-motion sequence "new_Bus_Front_5mm," and Sequence SEQ_4, stop-motion sequence "newCam_Bus_Front_5mm." x axis: frame number. y axis: TTC. Dashed black line: TTC ground-truths. TTC thin/thick lines: based on full image or segmented areas. TTC results based on full images/segmented regions: Case (I): Magenta/Cyan lines, Case (II): Blue thin/thick lines, Case (III): Red/Cyan lines, Case (IV): Green thin/thick lines. All figures are plotted with the same scales. 259

6-27 TTC deviation rates based on full image and the segmented area with different cases at subsampling rate 2×2 for stop-motion Sequence SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm." Top row: SEQ_3. Bottom row: SEQ_4. x axis: frame number. y axis: TTC deviation rate (TTC-truth)/truth. TTC thin lines: based on full image. TTC thick lines: based on segmented areas. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. All figures are plotted with the same scales. 260

6-28 The transition image frames for Sequence SEQ_3, stop-motion sequence "new_Bus_Front_5mm(104 frames)." Frame seq: 61, 71, 95. 262

6-29 FOE based on full images and segmented areas for Case (II) and (IV) at subsampling rate 2×2 for stop-motion sequences SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm." x axis: frame number; y axis: FOE coordinates x_0 and y_0 respectively represented by red and blue lines. (a1)(b1)(a2)(b2): based on segmented areas. (c1)(d1)(c2)(d2): based on full images. Left column: Case (II). Right column: Case (IV). All figures are plotted with the same scales. 265

6-30 FOE results plotted on different image frames for stop-motion sequences SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm." The computation is based on full image and the segmented area for Case (II) and (IV) at subsampling rate 2×2. (a1)(b1)(a2)(b2): based on full image. (c1)(d1)(c2)(d2): based on the segmented area. Left column: Case (II). Right column: Case (IV). 266

6-31 Orientation parameters (p, q) for stop-motion sequences SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm," based on full image and segmented area for Case (II) and (IV) at subsampling rate 2×2. (a1)(b1)(a2)(b2): based on full image. (c1)(d1)(c2)(d2): based on the segmented area. x axis: frame number; y axis: $(p/f, q/f)$ values respectively represented by red and blue lines. Left column: Case (III). Right column: Case (IV). All figures are plotted with the same scales. 267

6-32 Sample frames and their segmentation for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf" (103 frames). Frame seq: 1, 21, 41, 61, 81, 101. 269

6-33 The comparison of TTC computation based on full image and the segmented area for different cases at subsampling rate 2×2 for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf." x axis: frame number. y axis: TTC. Dashed black line: TTC ground-truths. TTC thin/thick lines: based on full image or segmented areas. TTC results based on full images/segmented regions: Case (I): Magenta/Cyan lines, Case (II): Blue thin/thick lines, Case (III): Red/Cyan lines, Case (IV): Green thin/thick lines. All figures are plotted with the same scales. 270

6-34 The comparison of individual TTCs at different subsampling rates and related TTC fusion results based on given segmentation, hand labeling and auto segmentation, for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf." x axis: frame number; y axis: TTC. Top row: results for segmentation based on hand labeling. Bottom row: results for segmentation based on auto segmentation. (a)(b): TTC results computed at different subsampling rates for segmented areas. (c) Solid lines: Fusion results based on (a) and (b). Dashed black lines in (a)(b)(c): TTC ground-truths. (a) TTC dotted lines: 1×1 (Red), 2×2 (Green), 4×4 (Blue), 8×8 (Magenta). (b) TTC dotted lines: 16×16 (Cyan), 32×32 (Yellow), 64×64 (Blue). All figures are plotted with the same scales. . . . 270

6-35 Sample frames and their segmentation for Sequence SEQ_6, stop-motion image sequence "newCam_slant_front_5mm"(106 frames). Frame seq: 1, 21, 41, 61, 81, 101. 271

6-36 Sample frames and their segmentation for Sequence SEQ_7, stop-motion image sequence "newCam_slant15_front_5mm"(107 frames). Frame seq: 1, 21, 41, 61, 81, 101. 271

6-37 Sample frames and their segmentation for Sequence SEQ_8, stop-motion image sequence, "newCam_slant25_front_5mm"(107 frames). Frame seq: 1, 21, 41, 61, 81, 101. 271

6-38 The comparison of individual TTCs at different subsampling rates and related multi-scale-based TTC fusion results based on given segmentation for stop-motion image sequences, SEQ_6, "newCam_slant_front_5mm," SEQ_7, "newCam_slant15_front_5mm," and SEQ_8, "newCam_slant25_front_5mm." x axis: frame number; y axis: TTC. (a)(b) TTC results computed at different subsampling rates for segmented areas. (c) Solid lines: Fusion results based on (a) and (b). Dashed black lines in (a)(b)(c): TTC ground-truths. (a) TTC dotted lines: 1×1 (Red), 2×2 (Green), 4×4 (Blue), 8×8 (Magenta). (b) TTC dotted lines: 16×16 (Cyan), 32×32 (Yellow), 64×64 (Blue). All figures are plotted with the same scales. 272

6-39 Sample frames and their segmentation for Sequence SEQ_9, stop-motion image sequence "camera3CUT1-horn"(299 frames). Frame seq: 100, 159, 218, 277, 336, 395. 275

6-40 Size-based TTC estimation for Sequence SEQ_9, stop-motion image sequence "camera3CUT1-horn." x axis: frame number. y axis: (a) Smoothed size information based on labeled foreground regions. (b) Log of smoothed size. (c) Difference between the log of smoothed size between every 16 frames. (d) Size-based TTC estimation results. 276

6-41 The comparison of TTC fusion results and traditional size-based TTC estimation for Sequence SEQ_9, stop-motion image sequence "camera3CUT1-horn." x axis: frame number; y axis: TTC. TTC dotted lines: results at different subsampling rates for segmented areas. TTC thick line: fusion results. Green circle: size-based TTC measurement. (a1)(a2) TTC results computed at different subsampling rates. TTC dotted lines: 1×1 (Red), 2×2 (Green), 4×4 (Blue). (b1)/(b2) Fusion results based on (a1)/(a2) respectively. (a1)/(a2) Results based on full images vs. labeled regions. All figures are plotted with the same scales. 277

6-42 The comparison of size-based TTC measurement and proposed gradient-based algorithms. x axis: frame number; y axis: TTC. TTC solid lines: TTC fusion results based on multi-scale subsampling at 1×1, 2×2, 4×4, 8×8. (b) 16×16, 32×32, 64×64. Dashed black line: TTC ground-truth. Green circle: size-based TTC measurement. Blue lines: (a) Fusion results based on full images. (b) Fusion results based on segmented regions. All figures are plotted with the same scales. 283

6-43 The condition-number-based TTC fusion for SEQ_8. (a) Condition-number-based fusion results based on full images. (b) Condition-number-based fusion results based on segmented regions. x axis: frame number; y axis: TTC. TTC solid lines: TTC fusion results based on multi-scale subsampling at 1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64. Dashed black line: TTC ground-truth. Green circle: size-based TTC measurement. (c)/(d) condition numbers for TTCs at different subsampling rates using full images/segmented regions. Dotted lines: condition numbers at different subsampling rates. Red solid line: threshold for condition numbers. If condition numbers are larger than threshold, their corresponding TTC results will be ignored in TTC fusion. 286

6-44 Sample frames and their segmentation for Sequence SEQ_9, stop-motion image sequence, slanted_Bus_5mm(93 frames). Frame seq: 1, 18, 35, 52, 69, 86. 287

List of Tables

2.1 Sensor Performance Comparison 61

3.1 Possible category of edge pixels 91

4.1 Sequence Information for three Examples 140
4.2 Segmentation Algorithms & Performance for three Examples 140
4.3 Classification Algorithms & Performance for three Examples 140

5.1 Summarization of test video sequences and results 199

6.1 Summarization of computational models and relative assumptions 227
6.2 Summarization of computational sequences and related tests 241
6.3 The error rate (percent) of TTC estimation in the bottom row of Figure (6-26) for Sequence SEQ_3. 261
6.4 The error rate (percent) of TTC estimation in the bottom row of Figure (6-26) for Sequence SEQ_4. 262
6.5 TTC estimation error (percent) in Figure (6-34) for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf." 269
6.6 Performance Evaluation for TTC with different case models 278
6.7 The different factors's impact on TTC 278
6.8 The different factors's impact on FOE 278
6.9 The different factors's impact on (p, q) 279

Chapter 1

Introduction

Under the pressure of increasing population, crowded traffic, the energy crisis, and environment concerns, current transportation systems have run into serious challenges in the following respects: safety, security, efficiency, mobile access, and the environment [32]. With the availability of faster computers, better sensor technology, and wider coverage of wireless communication network, Intelligent Transportation Systems (ITS) are gradually being seen as a crucial innovation that can satisfy the above needs. New ITS technologies will help to provide seamless, end-to-end inter-model travel for passengers and freight, to offer better transportation modes, to increase reliability and safety, and to contribute to the improvement of the environment. In this chapter we will discuss current transportation challenges, current ITS products, future research status, and the tasks we would like to address in this thesis.

1.1 The Challenges of Current Transportation Systems

Safety has always been the top priority and is one of most challenging issues. According to the National Highway Traffic Safety Administration (NHTSA) in US (Traffic Safety Facts 2008)[71], more than 5.8 million police-reported motor vehicle crashes occurred in the United States in 2008, leading to 26,689 death and 2.12 million people injured.

Collision with other moving vehicles is the most common cause of fatal injury, and

property-damage-only crashes. It is reported by NHTSA [1] that "Collisions with fixed objects and non-collisions accounted for only 19 percent of all crashes, but they accounted for 44 percent of fatal crashes." As shown in Figure(1-1), the cost of traffic accident is very high. The economic cost of these crashes is estimated around $164 billion annually, including property damages, lost earnings, medical costs, emergency services, legal costs and travel delays[70]. Around 75 percent of vehicular crashes are caused by inattentive drivers [69].

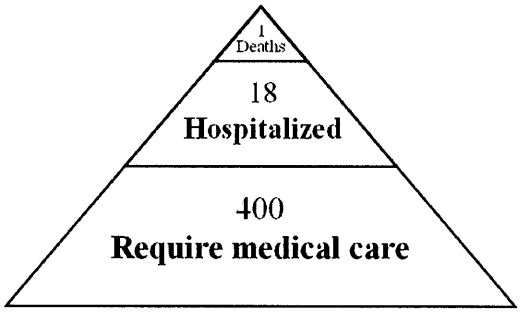


Figure 1-1: Distribution of cost for traffic accidents. For each death, 18 people are hospitalized and 400 are medically attended injuries[70]

Furthermore, age-related decreases in vision, cognitive functions, and physical impairments affect the driving ability of senior drivers(Owsley 1999). A 50-year-old driver needs twice as much light to see as does a 30-year-old driver[31]. The fatality rate for seniors is 17 times as high as the 25-65 age group[72]. There is an increase of the number of senior drivers who have difficulties in driving. According to the U.S. Census Bureau, in 1994 one out of every eight Americans was age 65 or older. In 2050, one out of every five Americans will be over aged 65[73]. According to NHTSA data, in 2008, US senior population (over age 65) is around 34 million [72]. There were over 31 million older licensed drivers in 2007, around 15 percent of all licensed drivers in 2007, compared with 14 percent in 1997[72]. Between 1997 and 2007, the increase of licensed senior drivers is 19-percent in contrast to 13-percent for total drivers[72].

While transportation systems should prevent accidents and/or decrease the loss and injuries involved in accidents, the transportation system development also needs to increase

[1] http://www.asse.org/Newsroom/naosh07/docs/trans_statistics.pdf

the national security to avoid unexpected situation such as the 9.11 event. Besides, the focus of transportation system development has been expanded to include making better use of existing systems as well as new construction (such as highways, railroads, etc.)[32]. New technologies are expected to reconfigure, refine, or expand the current transportation infrastructure in order to increase transportation speeds, ease congestion, and increase the efficiency of existing transportation systems[32]. Finally, mobility and information accessibility are attracting people's attention with the advent of the new Internet age. Energy, pollution control, and environment issues are also of concern[32].

In summary, human driving errors lead to dangerous outcomes and driving safety becomes more and more challenging for aging drivers and the whole society. At the same time, new issues always emerge to challenge current transportation systems.

1.2 Intelligent Transportation Systems

In general, Intelligent Transportation Systems (ITS) are seen as viable solutions to address the challenges in today as well as future transportation systems. ITS consists of the following three key components with different priorities among different countries [32]:

- Transportation infrastructures and facilities[32]

  ITS is expected to integrate all three essential components of transportation systems, people, road and vehicles, into a seamless information/physical infrastructure[32]. These elements should be integrated, performance-oriented, driven by customer needs, and should be managed and controlled under all circumstances[32].

- High-quality, high-relevance information[32]

  With the integration of vehicle and infrastructure for all ITS customers, advanced transportation management systems are needed to effectively manage the relationship between infrastructure and vehicles, and to process relevant information and structures. High-quality and high-relevance information, such as, destination/routing information, real-time road information, and weather information, can be relayed to

drivers and other interested parties so that quick responses are possible during crisis situations, and efficient management can be achieved in normal circumstances.

- In-vehicle technologies

  Safe, well-integrated in-vehicle technologies, such as, damage mitigating technologies, selective automatic enforcement, active driver assistance systems, are expected to increase the safety and the efficiency of all types of vehicles. Collision avoidance technologies in particular attract lots of attention.

The third component, *Reliable Collision Avoidance Technologies*, is the focus of this thesis. We will first discuss the expectation and challenges for the collision avoidance technologies in Section 1.3, and then propose our fusion-based and layer-based scheme to obtain obstacle information in Section 1.4. For the sake of completeness, we briefly discuss the first two key components of ITS, the *Integrated Information/Physical Infrastructure* and *Advanced Transportation Management* later in this chapter.

1.3 Expectation on and Challenges of Reliable Collision Avoidance Technologies

1.3.1 The status of current safety enhancement research

Since 1980, researchers have been working on technologies to enhance vehicle safety and driver comfort as summarized in [69]. Early research focused on improving the capabilities of vehicles, such as electronic fuel injection, TUFFUP tire technology for motorcycles, air pressure monitors, and cruise control, etc. Current research can be categorized into the following three categories: 1) damage mitigation when crashes happen, 2) selective automatic enforcement, 3) enhancement of drivers' judgment,

Damage mitigating technologies are expected to mitigate damages and to reduce potential injuries when accident/crashes are unavoidable. Such in-vehicle technologies include antilock braking systems, injury reduction bumpers, occupant protection/advanced air-bags, etc. The technologies can be considered as passive safety systems.

Selective Automatic Enforcement is developed for some critical situations since drivers might make mistakes even if they realize potential danger. There are many possibilities for automated processes. It is important to first do research on drivers' behavior in order to clarify how much a vehicle can do autonomously, and how much the infrastructure can do for the vehicle. Furthermore, driver qualification systems can automatically determine driver fitness to enhance security and safety. Some systems prevent accidents by compensating for at least some driver errors, for example, *Collision velocity reduction systems, Rollover stability warning and control systems, Cornering Speed Regulation System.*

Active driver assistance systems that aim to enhance the drivers' visual ability and to detect potential danger are the major focus of most ITS research due to the social acceptance and the reliability limitation of current safety techniques. Typical products include adaptive headlamps, blind spot monitoring, autonomous cruise control, lane departure warning, driver drowsiness alert, night vision, etc[78]. Since most police-report crashes are caused by driver mistakes, it is very important to automatically warn of the presence of pedestrians and motorcycles, and to compensate for recognition and judgment errors. Because of the importance, we will devote the next subsection to discuss how "active driver assistance systems" enhance drivers' visual ability and judgment in urgent and normal driving situations.

1.3.2 Active driver Assistance Systems

Since it is hard to enhance drivers' vision and judgment in all circumstances, ITS technologies first focus on the most dangerous situations and design specific systems to eliminate the most common collisions. Furthermore, information from the integrated infrastructure can also help to reduce collisions. The active driver assistance systems can be roughly classified into the following three categories, *Crash-based Vision Enhancement & Warning Systems*, *General Vision Enhancement Systems*, and *Infrastructure-assisted Hazard Warning*. The sensors involved are radar, infrared cameras, visible cameras, and navigation systems. Warning signal delivery mechanisms will include head-up displays, voice, and mechanical stimulation.

Crash-based Vision Enhancement & Warning Systems

Eighty percent of police reports [32] cited driver errors as the primary cause of vehicle crashes. It is estimated that 60% crashes at intersections and about 30% head-on collisions could be avoided if drivers had an extra half-second to respond [77]. To reduce the most common types of collisions, specific systems are designed as shown below in order to alert drivers in the most life-threatening situations.

1. Rear-end collisions

   It is expected that the distance of obstacles in front of the vehicle will be detected so that Adaptive Cruise Control (ACC) devices can adjust speed to maintain safe distances. Currently available ACC systems from different companies use different obstacle detection sensors. Subaru uses camera-based sensors; Toyota uses laser-based sensors; and Mercedes-Benz uses radar-based sensors. Sometimes, special pedestrian detection systems will be needed to ensure safety.

2. Roadway departure crashes

   Lane departure avoidance systems help to keep vehicles within lanes and provide warning signals to drivers whenever there is dangerous roadway departure detected by lane marker recognition systems. Lane departure warning systems are commercially available from many companies. For example, Iteris AutoVue lane departure warning system which emit distinctive rumble strip sounds as warning signals. Lane Keeping Support (LKS) and the Lane Departure Avoidance (LDA) from Nissan are designed for monotonous driving, and they only operate on "straight-ish roads" [1] and above a minimum defined speed. The LDA system can reduce road departure crashes through a vibrating steering wheel, audible warnings, and some degree of control.

3. Intersection collisions

   Infrastructure support, such as advanced traffic signs and signals, can be used to minimize intersection collisions. For example, vehicles will be able to receive signals

[1] Roads with curvatures more than 1000m.

from smart traffic lights that alert individual drivers to the possibility of collision with other vehicles, the approach of a red-light runner, or the presence of a pedestrian.

4. Lane change or merge crashes

   The side-rear warning systems can monitor the speed and location of nearby vehicles to advise drivers of potential collisions.

General Vision Enhancement Systems

Though it is hard to enhance drivers' vision and judgment in all circumstances, there are many technologies that can provide drivers better visual information and enhance safety in normal driving situations. These techniques are discussed below.

"Indirect Visual Field Systems" offer object information from nose-view, side-view and rear-view to drivers. For example, if drivers are provided an aerial view of the vehicle and surrounding area, the parking process will be more convenient. For Parking Assistant Systems in Japan, a virtual image viewed from an arbitrary angle is synthesized from multiple images taken from cameras mounted on vehicles in order to extend vision.

Night vision offers drivers a better view of what lies ahead at night. The extra vision extends three-to-five times the range of low-beam headlights and doubles the range of high-beam headlights. At 60 miles per hour, normal headlights provide a driver about 3.5 seconds to react to an object ahead. With Night Vision, the driver will have up to 15 seconds to react. The system also can help drivers see beyond the headlight glare from oncoming vehicles.

Since accidents are more likely to happen in bad weather or lighting conditions, sensors that work in these situations are especially needed. New headlight technologies will offer better views for drivers by improving lighting situations. Possible new headlights include active headlights (self-aiming headlights) and light distribution control headlights. Better cameras with wide dynamic range can provide higher quality images. A new imaging sensor was developed at MIT [26] to implement a novel brightness adaptive algorithm, which significantly improves the imaging performance.

Infrastructure-assisted Hazard Warning

In-vehicle devices alone only offer limited warning ability. Integrated information systems can help to offer more reliable and faster warning signals. The information exchange between in-vehicle sensors and infrastructure can enhance the performance of the sensors, while communication networks help vehicles to be actively connected with ITS infrastructure. The possible examples include road surface monitors, curve warning systems, cooperative intersection decision support systems, highway-rail intersection warnings and crash avoidance systems, blind side collision avoidance systems through cross-traffic vehicle information, on-board diagnostic systems for roadside inspection, etc. Different from crash-based vision enhancement and warning systems whose purpose is to develop in-vehicle devices to detect potential collisions and to provide in-time warning, the current system provides drivers with relevant information from infrastructure to enhance their vigilance and safety. There might not be any real potential collision.

1.3.3 Development of Auto Electronics

It is estimated that implementing collision-avoidance systems in vehicles could prevent 1.1 million accidents in US each year – 17 percent of all traffic accidents, which could save 17,500 lives (compared to the 10,500 lives saved by seatbelts and airbags) and $26 billion in accident-related costs [69].

Besides these above safety enhancement in-vehicle electronics, information products and diagnostic prognostic products are also developed. These technologies focus on accident prevention based on information, communication, and sensing technologies while conventional safety measures aimed at protection during a crash. The development of better in-vehicle electronics and the full cooperation between the vehicle and infrastructure significantly improves transportation situations.

The demand for in-car electronic products is increasing. Actually around 35 percent of the cost of car assembly comes from electronics[74]. According to Strategy Analytics, the market for automotive grade semiconductors in 2006 will reach $18 billion, a year-to-year growth of 10 percent. Revenues in the U.S. will be $29 billion by 2013[74]. Additional

vehicle networking will create an annual volume compound growth rate for semiconductor bus transceiver chips of 16 percent, resulting in a market worth $1 billion per year in 2014. However these above systems are far behind the drivers' need.

1.3.4 Cooperated Sensors and Automatic Environment Understanding

The current phase of ITS technologies is to develop individual ITS components to detect the static and dynamic information about driving environment. The *static* information includes: location, obstacle/foreground at different distance ranges, etc. The *dynamic* information includes: speed, time-to-contact, possible cross over or occlusion, obstacle/foreground at different time-to-contact range, collision/occlusion possibility, and other current/historical information, etc. Current methods independently detect individual piece of information, which lead to many independent in-vehicle sensor devices. However, all dynamic and static obstacle information should be integrated to provide an accurate description of current driving environment, such as, are there any obstacles (pedestrian, vehicles, etc.)? Where are they in the camera? How far away? Whether and when will the current vehicle run into these obstacles? In order to aid drivers, we need a highly coordinated and integrated system that will work cooperatively with drivers [69] to provide obstacle information.

The infusion of the uncoordinated in-vehicle devices may overwhelm drivers, disturb drivers' attention & alertness, thus may degrade driving safety and performance[69]. As a matter of fact, the safety devices may become safety hazards themselves. Furthermore, civil engineers and electronic engineers also focus on different aspects of ITS systems. Civil engineers pay more attention to the systematic traffic design in order to improve the ITS performance, while electrical engineers pay more attention to component technologies on enhancing individual abilities of sensors.

In order to avoid the above extreme situations and to help drivers to respond better to fast changing environment and dangerous situations, system level research is required to balance the requirements for individual sensors. All safety-enhanced devices should be coordinated within a framework to manage & display information in order to develop a

balanced approach to ITS systems. Such integrated systems can help to decrease the heavy computational load required for individual algorithms. At the same time, the research on component technology will offer more feasibility for system management.

In this thesis we are proposing fusion-based and layer-based schemes to systematically detect and combine obstacle information. The additional information from other sensors helps to simplify original individual complicated task. Such integrated systems can enhance the overall performance, reliability, and robustness in order to meet the ITS requirements.

1.4 Fusion-based and Layer-based Scheme to Obtain Obstacle Information

Sensor fusion is a common technology to improve the detection performance by combining information from different resources. The technology can be divided into three categories: feature-level fusion, data-level fusion, and decision-level fusion[75][76]. Feature-level fusion takes advantage of specific information from one sensor and design special algorithms to incorporate the additional information in another sensor's process. Data-level fusion combines raw data in similar format from different resources statistically, for example, through voting technique, Bayesian, and Dempster Shafer, etc. Decision-level fusion takes all sensors' decision (after individual's process) into account when making final decisions at a higher level.

Our proposed framework, shown in Figure (1-2)(a), employs two principles: the principle of fusion techniques, and the principle of layered techniques or divide and conquer. Our fusion belongs to feature-level fusion category. In our framework, information obtained from other detectors/algorithms is incorporated into the segmentation process to match corresponding obstacles' information, which improves detection performance. Instead of separately estimating static/dynamic information, including distance ranges, segmentation, motion, and classification features, our scheme makes use of the physical connections among these features. Additional information such as distance, motion, and dynamic history can all be used to enhance the segmentation accuracy. The additional information can

be of low quality, yet they can improve overall performance [96] [97] [100] [101]. Furthermore, the accurate segmentation information can be used to improve the detection accuracy of these fused information and other timing information, such as, time to contact.

Our scheme is not only *fusion-based* but also *layer-based.* With the additional information, a complex task can be divided into several simple ones. Extra information helps to split one complex task into several simple tasks in different layers, which are easier to solve than the complex one. Different signal combinations in Figure (1-2)(a) lead to different applications as highlighted by different color shading blocks.

The first application, shown in Figure (1-2)(b), is target detection and 3D segmentation which detects obstacles at different distance ranges and provides complete 3D information for the road situation [96] [97] [99]. The segmentation scheme incorporates both distance range and motion information to improve obstacle segmentation performance. Our method does not assume objects' shape and size, which means our method has a wide range of applications. Detailed discussion will be in Chapter 2 and Chapter 3.

The second application, shown in Figure (1-2)(c), is a pedestrian segmentation/identification scheme for infrared images. The scheme takes advantage of classification features to enhance segmentation accuracy for infrared images [100], which is to be discussed in Chapter 4.

The third application, shown in Figure (1-2)(d), is to incorporate historical information or predicted information from a dynamic model into segmentation process in order to understand complicated scenario and to obtain time-to-contact information for obstacles. This application is to be discussed in Chapter 5 and Chapter 6.

In summary, the essence of the framework is to convert a complex segmentation task into several simple ones so that the time-consuming full-image search can be avoided. Before we present the details of our proposed framework in different chapters for the rest of the thesis, we would like to first discuss briefly the two other important ITS components in order to demonstrate the big picture of ITS and the role of collision avoidance technologies.

1.5 Other ITS Components

After presenting the current status, the demanding requirements of collision avoidance technology, and our proposed fusion-based and layer-based scheme, we would like to describe briefly the other two important ITS components, the *Integrated Information/Physical Infrastructure* and *Advanced Transportation Management*. Since ITS has not only technology components but also social aspects, we will touch upon social acceptance of ITS deployment and discuss the current deployment of ITS in different countries with different priorities at the very end of this section.

1.5.1 An Integrated Information/Physical Infrastructure[32]

The infrastructure of transportation includes the physical infrastructure, people, road and vehicles, as well as an electronic information infrastructure[32]. The integration of physical and information infrastructure serves as the supporting platform in order to detect and respond to emergencies and to maximize efficiency and utilization[32]. It is expected to provide full coordination among urban buses, rail transit, railway, highway and arterial systems[32]. Such seamless integration not only improves the transportation infrastructure and in-vehicle devices, but also provides effective traffic management and information services.

Transportation related information should be available on all types of media[32]. Related information includes online mapping, driving directions, en-route variable message signs and kiosks, and personal subscription services, as well as real-time information for pre-trip planning, ride-sharing and en-route modifications, and current and expected conditions for all relevant modes. Furthermore, vehicles' operators should be provided with tailored weather information and its impacts. There should be new tools for data collection, storage and analysis. Probe-car system is a new data collection method, in which several vehicles act as moving sensors to collect real-time traffic information. Because of privacy concerns, drivers may have concern about providing probe-car data while awaiting traffic information from ITS centers. However, the centers need probe-car data for traffic management. "Traffic Information Exchange Market" Projects at MIT [26] introduced a

traffic information exchange market to solve the puzzle where third parties are expected to gather/exchange traffic information and prices will be adjusted by market mechanisms.

Today, *Telematics* is one of important in-vehicle products as part of the integrated information network. Telematics refers to the consumer products, services, and supporting systems that deliver information, communications, and entertainment to in-vehicle and mobile devices. Several automobile companies provide telematics products and have offered communication network services in different levels.

For example, for *Onstar* system, developed at GM, *Safe & Sound* plan provides safety and security assistance, and *Directions & Connections Plan* provides directions, guidance and assistance. ATX (Ford) system automatically notifies emergency operators if an airbag is deployed and provides 24-hour emergency assistance and route guidance. Also, Lincoln VCS are equipped with three buttons, "SOS," "i," and "Phone," for emergency services and Roadside Assistance, route assistance and points-of-interest, and hands-free calling and information services. Toyota's G-Book service allows drivers to probe Communication Traffic Information based on membership-based information system GAZOO and uses data from other vehicles equipped with G-BOOK to provide the drivers with highly precise information on traffic congestion. G-BOOK provides Map-on-Demand features for automatically delivering differential map data to car navigation systems, which creates an user-friendly and comprehensive telematics service.

Spatial geo-location/routing technologies can help to track movement so that *Electronic Maps* and *Video Surveillance* can be generated. To provide useful visual information, vision sensors are needed for traffic and accident monitoring, environment understanding. Right now, traffic image information transmission based on compression/decompression is limited by communication bandwidth. Image transportation is costly yet the compression/decompression process is slow. To address this issue, "Image Sensor Network" Projects at MIT [27] [26] developed a technique called "object recognition without decompression." This technique increases processing speed while decreasing bandwidth requirement. Mobile agents associated with this technique tailor the transmission mode based on users' goals, and will help to reduce the number of images transmitted and decrease the response time for ITS infrastructure.

1.5.2 Advanced Transportation Management[32]

By providing the related information to operators and users, ITS can enhance management and operations of existing highways, public transportation, and railroad infrastructure. It will help to ease congestion, respond to crises, reduce energy consumption, and increase the effective capacity of systems. The focus of transportation system development has been expanded to include performance-oriented operations as well as new construction. There are three basic modes of advanced transportation management, including *Advanced Monitoring and Supervision Systems, Advanced Emergency Response Technologies* and *Advanced Automation Systems.*

Advanced Monitoring and Supervision Systems[32]

The focus of these systems is on direct management of the infrastructure and the communication of useful information to travelers and vehicles. Important functions involved are "Area-wide surveillance and detection," "rapid acquisition and evaluation of traffic flow data," and "operational responses to traffic flow changes."

Advanced Emergency Response Technologies[32]

Another measurement to improve safety is enhance the ability of incident detection and fast response to emergency when crashes do occur. It is expected that the traffic environment will not deteriorate due to traffic accidents. This ability is important in cases of national crisis.

Timely incident notification needs to automatically detect the crises (severity, precise location) and to quickly provide related information, such as suitable hospitals based on the degree of the injuries, and routing to these locations. When necessary, integrated information infrastructure has the ability to offer emergency medical services through real-time voice, visual and data communication. One example is the *Fast Emergency Vehicle Preemption Systems* (FAST) developed in Japan. FAST can prioritize signal control and provide optimum routes for emergency vehicles. Infrared beacons close to traffic lights can identify emergency vehicles through their vehicle IDs from in-vehicle units. When emergent situa-

tion happens, the Traffic Control Center quickly designs optimal route and controls related traffic lights based on vehicle ID. Thus, emergency vehicles can follow the shortest routes without stopping at intersections and process emergencies as early as possible in order to reduce injuries and increase efficiency.

Advanced Automation Systems[32]

When the infrastructure and vehicles communicate and interact, the Cooperative Vehicle-Highway Automation Systems (CVHAS) are expected to automate all or part of driving tasks for individual vehicles and to safely increase the capacity and flow of existing infrastructure. Typical automatic technologies include weigh-in-motion technology, automatic vehicle content check-up, electronic payment for transport, mobile Internet and public transport, automatic vehicle locations and other vehicle remote control technologies. It is also important to automatically check the identity and fitness of drivers through security systems. These systems need cooperation across jurisdiction.

For example, *Electronic Toll Collections*(ETC) deploy various communications and electronic technologies to support the automated collection of payment at tollbooths. Collectively, these technologies increase system throughput, improve customer service, enhance safety, and reduce environmental impacts. The first generation of electronic toll collection has been completed. The next stage is to develop ETC into an instrument of public travel policy and transportation management. It can also be the basis for a wide variety of modest-sized electronic payments. In Europe, Smart Card technology provides safe electronic payment for transport, mobile Internet, public transport, and so on. It can be the base for dynamic road pricing.

It is noticed that ITS users value change with elapsed time and sudden changes in network load. Based on a bidding function concept, an auction mechanism can help to resolve competition for network resources among individuals utilizing communication networks for ITS, which is proposed in *Auction and Bidding Algorithms for Resolving Network Resource Competition Projects at MIT [28]*

Several other important ITS elements include: precision docking of public transportation vehicles, dedicated lanes for automated trucks, automatic guidance of snow removal

and other maintenance, and fully automated passenger vehicles in the future. For example, *Intelligent Multimode Transit System*(IMTS) is a new-concept bus system developed by Toyota. On dedicated inter-city roads, buses run automatically. On normal roads, the buses are operated manually.

1.5.3 ITS Deployment

Unlike other techniques, ITS depends heavily on social acceptance. The development and deployment of ITS are strongly connected to many non-technical factors, such as the culture of transportation system management and operations, the roles and funding of the public sectors, government policies and initiative to encourage the private sectors' involvement and other human factors [32].

Successful deployment of ITS technology depends on the effort from both technical progress and social acceptance promotion, including forging new forms of cooperation within and among the public sectors at all levels and the private sector in its broadest sense. Development steps should focus on both pure technical ITS challenges and social acceptance in order to successfully achieve the ITS vision. Transition processes from the present situation to the targeted system architecture, and the proposal of deployment processes, will also be of great importance. It is necessary to clarify the goals, the obstacles to the goal realization, and the means of deployment and then to decide on the necessary infrastructure to build up. In the remaining section, we will discuss ITS deployment in US and Europe.

USA ITS Deployment [32]

The Model Deployment Initiative (MDI) was initiated by the U.S. Department of Transportation (DOT) in 1996. MDI was designed to bring increased levels of service to the traveling public through the integration of several key systems: traffic signal control; transit, freeway, and incident management; emergency services management; regional, multimodal traveler information services; and electronic toll collection and fare payment. MDI helps to evaluate the benefits of integrating ITI infrastructure in a metropolitan area. Phoenix (AZtech), Seattle (SmarTrek), San Antonio (TransGuide), and New York City/New Jer-

sey/Connecticut (iTravel) were selected as MDI test sites, with different focuses, including:

– Enhancing traffic and transit management operations and developing an extensive traveler information system.

– Enhancing traffic management and emergency response operations and creating a comprehensive traveler information system.

– Implementing an innovative emergency medical services management system, enhancing traffic management operations and creating a multi-modal traveler information system.

– Developing a highly integrated, multi-modal traveler information system.

The MDI Program has allowed the sites to deploy and integrate several ITS systems. Although the MDI program is nearly complete, all sites have funded future system expansions and enhancements.

At the beginning of 2001, the Intelligent Transportation Society of America proposed a ten-year development vision, *National Intelligent Transportation Systems Program Plan.* In this plan, safety, security, efficiency/economy, mobility/access and energy/environment are chosen as the five most important goals. A series of programmatic and enabling themes are developed to reach these goals, which include: 1) An integrated network of transportation information; 2) Advanced crash avoidance technologies; 3) Automatic crash and incident detection, notification, and response; 4) Advanced transportation management.

European ITS Deployment [25]

In Europe, Intelligent Transportation Systems and Smart Cards for the transportation sector are among the eleven main actions of the "eEurope 2002 initiative." The ITS challenges are to meet the growing demand for mobility within the transportation infrastructure networks. One of the deployment barriers is the fragmentation of transportation infrastructure management among member countries.

Proposed projects of Information Society Technologies (IST) plan to enhance the development of systems and services for intelligent transportation infrastructures. The first objective is to improve mobility management and to focus on advanced surveillance and control systems on the wide range of tunnels and railways, intelligent inter-urban and ur-

ban transportation management systems, advanced ITS-based systems for logistics, and cooperative management of resources. The second objective is to develop intelligent vehicle systems and to focus on the promotion of advanced driver assistance systems and tele-service systems in areas such as maintenance, dependability, remote diagnostics, and vehicle performance.

EU plans to equip 50% of major European larger cities with "travel planning and traffic information services" and to equip 50% of the main European road network with "congestion and accident detection and management systems." EU also plan to install new vehicles sold in Europe with "active safety and driver assistance systems" and to enhance emergency service number "112" with location information in order to offer full emergency services.

After having a full picture of ITS, in the following chapters of this thesis, we discuss how the fusion-based and layer-based principles are used in our different applications. In Chapter 2 and Chapter 3, we first discuss how to separate an image into several distance-based image-layers so that each layer contains potential obstacles at a distance-range or background objects that are far away. Thus we can remove distance-based background noises and detect targets of interest. In Chapter 4 and Chapter 5, we discuss how to separate an image into several vertical stripes that contain potential pedestrian obstacles in infrared images and in visible images. In Chapter 6, we discuss how to obtain time-to-contact information for targets of interest.

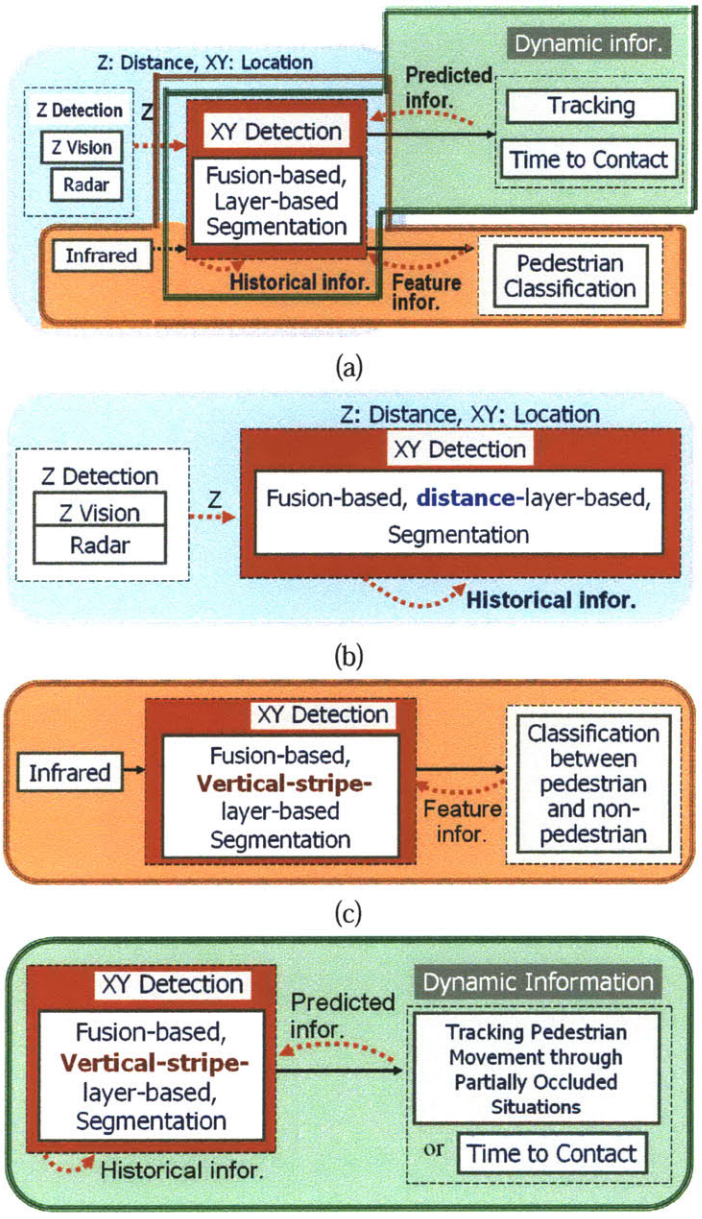


Figure 1-2: (a) General framework of fusion-and-layer based methodology. (b) Framework for 3D segmentation. (c) Framework for night-vision based pedestrian detection. (d) Framework for fusing historical information and dynamic estimation.

Chapter 2

Target detection and 3D segmentation based on motion and distance range

Segmentation is an integral component of the interpretation of a dynamic environment. One is required to segment target objects, obtain lane information, and detect objects within given distance ranges. Current segmentation algorithms face tradeoffs between performance and computational speed.

In this and the next chapter, we will apply the concept of fusion-based and layer-based methodology to obtain 3D segmentation of interesting obstacles which is the first application shown in Figure (1-2)(b). In this chapter, we will discuss a general fusion-based layer-based method which segments objects in distance ranges of interest. In the next chapter, we further discuss how to remove the effect of backgrounds at specific distance ranges. Both distance and motion information are incorporated into segmentation in order to improve the accuracy and robustness and to decrease computational load. Our proposed methods improve the performance of static image segmentation, thus lay down a good foundation for tracking of other information in video sequences.

2.1 Introduction

2.1.1 ITS requirements

For Intelligent Transportation Systems (ITS) and Intelligent Vehicle (IV) applications, a real-time highway environment interpretation system is expected to provide complete 3D information for the targets in the driving environment, i.e., the target sizes, locations, and depth (distances). For the system in Figure (2-1), for example, drivers need to know the horizontal locations of the leading vehicle ($w1$ and $w2$) as well as the distance information for safe driving. Such information helps with short-range frontal detection of elevated objects and provides warning function in city traffic or during "stop and go" driving on highways. A vehicle can use an Intelligent Cruise Control system to adjust speed based on the distance between the vehicle and the nearest preceding vehicles, and on the types of these vehicles.

In order to fulfill these dynamic scene interpretation requirements, image segmentation is required in order to partition an image into different regions associated with generic labels or informational labels. Each region consists of groupings of image pixels with similar data feature values. The need of dynamic scene processing in real time brings high requirement on sensors in intelligent transportation systems.

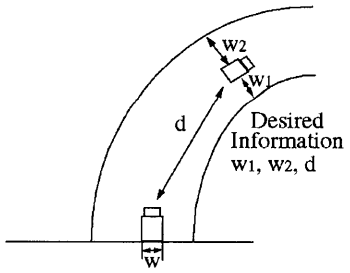


Figure 2-1: An example where radar is not enough.

2.1.2 Typical vision-based segmentation algorithms

There are several traditional image segmentation algorithms to obtain locations, size and depth of target objects:

- Characteristic feature thresholding or clustering [4] [8]

  Popular k-means [9] and ISODATA algorithms are examples of clustering algorithms. These methods do not exploit spatial information and their segmentation performance is limited.

- Boundary detection [4] [8]

  Boundary Detection methods [8] [4], such as edge detection algorithms, are typically used together with other techniques.

- Region growing [4]

  Region growing is a procedure that groups pixels or sub-regions into larger regions. The method needs to have seed points or seed regions to grow into larger segmentation regions.

- Motion Based Methods [3] [5]

  Motion-based segmentation methods [3] [5] segment images based on the target motion pattern (e.g. optical flow vectors). The optical flow vectors for one object represent a pattern that can be used to segment the object. A classical approach to motion segmentation is to estimate a dense motion field followed by a segmentation of the scene based only on the motion information. As in [5], object enclosing boxes are used to cluster the estimated vectors before passing them to the obstacle test. Detailed introduction and comments on such algorithms can be found in [3].

- Tracking Based Methods [10] [29] [12]

  Real-time applications usually adopt tracking-based segmentation methods [10] [29] [12]. These algorithms assume similarity of target features in consecutive frames and use this to supress initial segmentation errors. This method first detects objects with decreased accuracy requirements and segments multiple rectangular image regions which are the candidate position of objects. The image regions which do not correspond to any object can be easily eliminated by tracking several video frames because

the information from tracking soon shows up the error caused by coarse static segmentation. The extreme situation in this category of segmentation methods is feature tracking. By successfully tracking feature points, it is possible to segment each specific frame with better results. However, variation of image positions and sizes of objects brings difficulties for tracking.

- Symmetry Based Methods [1]

  The symmetry-based segmentation [1] is another typical segmentation method for IV systems. The method assumes the symmetry of typical IV targets (such as vehicles, pedestrians) and asymmetry of background in the videos for Intelligent Transportation Systems, and uses a symmetry finder to detect candidate target regions. The candidates for a leading car are located by detecting the vertical axis of symmetry, which is a feature for measuring the leading car's relative lateral displacement in consecutive images.

  The performance is invariant under nodding movements of the camera and under changes of target sizes. However, these assumptions do not necessarily hold in real situations, which limits their applications. This method can not deal with a vehicle with arbitrary angles and other asymmetric objects.

The last three methods are more typically used in intelligent vehicle systems than other methods.

2.1.3 Fusion of binocular stereo and radar for 3D information

In real time ITS applications, a single sensor is typically not adequate to provide reliable information for autonomous driving guidance in real time because of weather, ambient lighting, reliability, and other limitations. We need to obtain 3D information for the current road situation, i.e., the X, Y position and the distance Z of objects in images. While vision-based 2D segmentation algorithms can provide spatial resolution, vision systems (binocular stereo) are not very reliable in depth detection because of mis-correspondence problems. They also fail to function well in bad weather.

Table 2.1: Sensor Performance Comparison

Characteristics	radar	vision
Distance Resolution	Good	Limited
Horizontal Resolution	low	Good

Radar systems offer relatively accurate depth information and robustness in bad weather, but their spatial resolution is poor. Usually, there are three types of radar systems: broad single beam systems not having much horizontal resolution (type 1); multi-beam systems having limited horizontal resolution (type 2); and scanning radar systems including electronic scanning and mechanical scanning (type 3). For the system in Figure (2-1), a simple radar could not provide enough information to assist drivers to pass leading cars.

As the horizontal resolution of radar systems improves, the cost of radar systems goes up significantly. Even for expensive specialized imaging radar that provides both depth and spatial resolution, the horizontal resolution remains worse than typical vision systems. Furthermore, radar systems used in IV are designed to detect only moving targets to avoid false alarms from static objects *along* a road. Such systems might also ignore dead (static) vehicles *on* the road. Finally, distance resolution for lower performance radar systems is not satisfactory because of reflection. Though developing more advanced image radar is one possible solution to meet the requirements, it would be more feasible and economically efficient to develop sensor fusion system that is composed of several low cost, low performance sensors, i.e., simple radar and stereo cameras.

The comparison between radar sensors and vision systems is shown in Table (2.1) in which the advantages and disadvantages of two sensors are complementary. If requirements for one system concentrate on only one of the characteristics in Table (2.1), either distance accuracy or horizontal resolution, we could use only one sensor. To meet the high requirements from ITS, sensor fusion systems are expected to take advantage of the benefits of both sensors at the upper left corner and lower right corners in Table (2.1).

Typical intelligent vehicle systems detect depth and target segmentation information separately from low-cost radar systems and video sequences respectively, and match the complementary information at the final stage, as shown in Figure (2-2)(a). Video cameras

are used to detect the accurate location, size and depth of target objects such as vehicles and possible pedestrians on the road. The information is fused with the distance information provided by radar systems, thus providing comprehensive 3D measurements, the dynamic size, location, and distance information of all target objects. Sensor fusion will improve the reliability of systems, especially in bad weather.

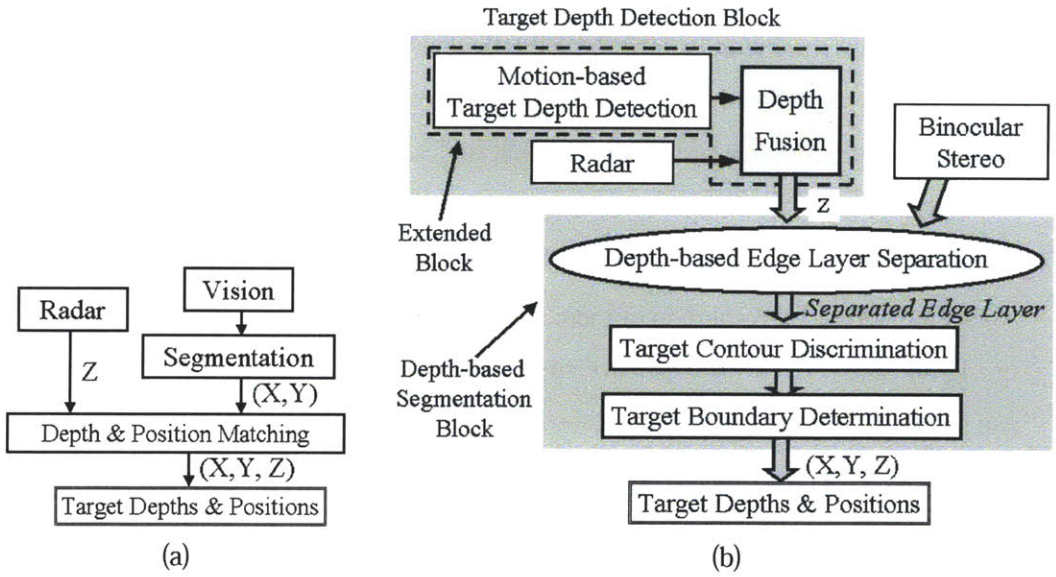


Figure 2-2: (a) Traditional method of target detection (b) Proposed depth-based target detection: fusing radar and stereo

The systems shown in Figure (2-2)(a), however, do not take advantage of the close relationship between depth and stereo images, and still suffer from the limitation of traditional segmentation methods. Instead of simply mixing two complementary information sets together, we are presenting a data fusion strategy that actively fuses the depth information into image segmentation process.

Our proposed depth-based segmentation algorithm is shown in Figure (2-2)(b). By providing binocular stereo systems with coarse target depth information from sensors, we are able to split an edge-map into n depth-based edge layers, and to decompose the original task of multiple-target segmentation into n segmentation tasks on each depth-based target feature layer. Our fusion scheme belongs to the feature-level fusion, in which the depth-based segmentation block is the layer-based 3D segmentation. With the added distance

information from radar, a binocular stereo system could significantly improve its object segmentation performance.

In Section 2.2, we first explain how to obtain the required target depth range when radar information is not available, In Section 2.3, we then describe how to implement the layer-based segmentation based on calculated depth ranges. At the end, we discuss the feature of our methods in Section 2.4.

2.2 Information Fusion: Motion-based Target Depth Detection

The first step of our fusion scheme in Figure (2-2)(b) is to detect the targets' depth information and obtain the corresponding disparity ranges. Given a radar system, we can directly obtain a series of depth data for n targets, say, $d_1, d_2, ..., d_n$. Assuming the resolution from the radar is Δd, the real depth of the ith target will be between $[d_i - \Delta d, d_i + \Delta d]$. For binocular stereo, the depth ranges can be translated into the disparity ranges using equation (2.1). For simplicity, we respectively call the image taken by the left and the right camera of stereo system *the left image* and *the right image*. Their edge-maps are called *the left edge-map* and *the right edge-map*.

$$x_l - x_r = b\frac{f}{z}, \tag{2.1}$$

where x_l, x_r are the x-coordinates of target pixels in the left and right images of stereo system with baseline distance b and focal length f, $x_l - x_r$ is the disparity, and z is the actual distance between the target and the stereo system.

Since the real depth of the ith target in the given radar data series is between $[d_i - \Delta d, d_i + \Delta d]$, the corresponding disparity for target i will be in the following ranges:

$$[\frac{bf}{d_i + \Delta d}, \frac{bf}{d_i - \Delta d}]. \tag{2.2}$$

When radar information is not available, we can directly obtain target depth ranges and disparity ranges from other sensors or other vision-based algorithms. For example,

the disparity information can be achieved through an edge-based trinocular method [2]. Here we are developing a motion-based binocular method to obtain the disparity ranges for targets. The scheme is an important part of depth-based target detection system as shown in Figure (2-2)(b)). The structure of the proposed depth-detection algorithm is shown in Figure (2-3). In Section 2.2.1, we first introduce "motion-based correspondence matching criteria" to obtain the disparity histogram, the histogram of the disparity for all edge pixels. Then we discuss how to analyze disparity histogram and to detect target disparity ranges in Section 2.2.2.

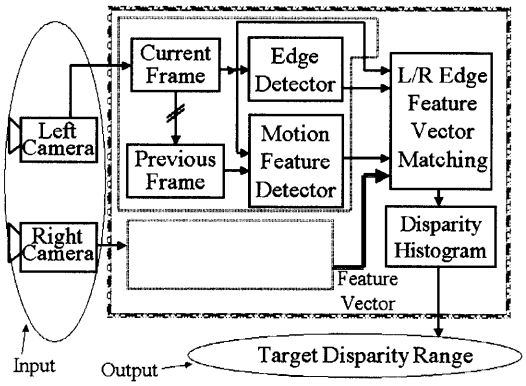


Figure 2-3: Target depth detection system based on fusing binocular frames and motion. Detail for Figure (2-2)(b).

2.2.1 Motion-based Correspondence Matching Criteria

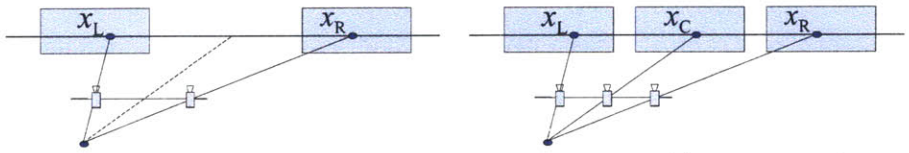
In order to obtain disparity histograms for stereo image frames, we apply the following three correspondence constraints to compute correspondence disparity information for all edge pixels.

- *Binocular Stereo Epipolar Line Constraints*

For binocular stereo image pairs, there exists a strong positional constraint, i.e., matching feature pixels from stereo images should be on corresponding epipolar lines. For aligned cameras, the epipolar lines are frame rows as shown in Figure (2-4)(a).

Usually the epipolar line constraint alone is not sufficient enough to locate corresponding feature points, so traditional binocular stereo methods need other complex feature con-

straints to further limit correspondence searching. Extra feature constraints typically include local content constraints, for examples, cornerness, edge, color, intensity, etc., and global constraints (continuity). The different choices of feature constraints lead to different algorithms.



(a) Binocular Stereo (b) Trinocular Stereo

Figure 2-4: Disparity Constraint for Binocular and Trinocular Stereo.

Imposing global constraints is complex and time consuming, and algorithms based on local constraints are not reliable. Trinocular stereo [2] as shown in Figure (2-4)(b) is one solution to meet both simplicity and effectiveness requirements, which imposes epipolar line constraints among *three* aligned camera images, i.e., the horizontal image position of a target point from the center camera must be the midpoint of image positions of the target point in the left and right images as followed:

$$x_C = \frac{1}{2}(x_L + x_R) \tag{2.3}$$

For a binocular stereo setting, there is no similar simple and effective method. Instead, we propose to impose the following three additional constraints in order to improve correspondence reliability.

- *Edge-feature Constraints*

(a) Positive edge (b) Negative edge

Figure 2-5: Examples of positive and negative edges.

To remove correspondence ambiguity, we only match vertical edge pixels from stereo images, and differentiate "positive edges" from "negative edges." As shown in Figure (2-5), positive edges are where image intensities change *from darkness to brightness* most

rapidly, negative edges are where image intensities change *from brightness to darkness* most rapidly.

• *Motion Constraints*

Because of rigidity, motion information for the same target points calculated from the left and right images should be similar. As shown in Figure (2-8), motion vectors for both stereo video frames show similar patterns even if motion vector detection involves noises. For a feature point at (x, y) in Seq# (k), to determine its motion vectors between frames at Seq# (k) and $(k+1)$, we define its template patch with size (8×8) in Seq# (k) and a search window with size (16×16) in Seq# $(k+1)$ both of which center around (x, y). Then we search within the search window for a patch with minimum correlation errors from template. Though for boundary feature points, the template patches contain visual features from different background regions, to some extent, there still exist the similarities of motion information for stereo corresponding points as shown in Figure (2-8). It is because the sizes of template patches are small enough. Motion constraints will be combined with other contraints to provide the histogram of disparity, which do not expect high reliability and accuracy for estimation of motion vectors.

• *Texture Constraints*

When locating motion vector for a given pixel, the variation curve of the correlation errors at different candidate positions represents the texture information for that pixel, which is named as *Motion-Vector Correlation-Error Curve*(MV-CEC or MV-CE-Curves). The large variation of MV-CE-Curves in Figure (2-6)(b) corresponds to a feature point, while the regular pattern of MV-CE-Curves in Figure (2-6)(c) corresponds to a point in uniform region. We define the function of the difference between the maximum and the minimum correlation errors as *Texture Index* in order to reflect the texture information in the neighborhood region surrounding the pixel. If there are significant changes within the image patch surrounding a given pixel, its corresponding texture index is large. Otherwise, the texture index is small.

For a pair of corresponding edge pixels in binocular stereo images, there exists the similarity between two corresponding MV-CE-Curves in both shape and numerical values. The texture index for correspondence-pixel-pairs in stereo images are also similar. The

similarity of MV-CEV-Curves for correspondence-pixel-pairs are shown in Figure (2-6) in which the top and the bottom rows are for two pairs of points marked in yellow diamond regions and in red rectangular regions in Figure (2-6)(a). For template patch with size 8×8 and search window with size 16×16, there are $81 (= (16-8+1) \times (16-8+1))$ candidate positions and 81 corresponding correlation errors.

The similarities of texture index also somehow apply to a pair of boundary feature points, though their neighborhood regions are not quiet the same due to the different background regions. It is because that the maximum and minimum correlation errors are close when the sizes of template patches are small enough. Even if the texture index for some boundary feature points might not exactly the same, it would not matter since the purpose of motion-based correspondence matching is to obtain disparity histogram and the disparity ranges for interested obstacles.

Taking advantage of the above contraints, we define multidimensional feature vectors including location constraints, motion constraints, texture constraints, and brightness constraints for all edge pixels. For a pixel at position (i, j), its feature vector $\vec{V}(i, j)$ includes the following information:

$$\vec{V}(i,j) = [MV_x(i,j)\ MV_y(i,j)\ E_{cor}(i,j)\ i_u(i,j)\ I(i,j)].' \tag{2.4}$$

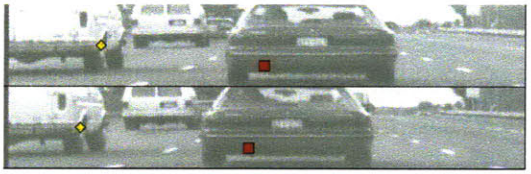
where I is the pixel's intensity value, $(MV_x,\ MV_y)$ are motion vectors calculated based on correlation methods, E_{cor} is the minimum correlation error when searching for motion vectors between two continuous images, i_u is uniform index.

The *matching error index* $D_{match}(i_l, i_r, j)$ between pixel (i_l, j) in the left image and pixel (i_r, j) in the right image is defined as the weighted sum of their respective feature vector component differences as below:

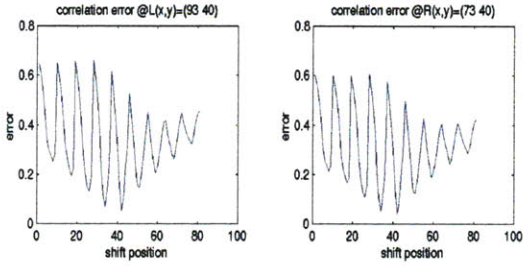
$$D_{match}(i_l, i_r, j) = \sum_k w_k * [\vec{V_l}(i_l, j) - \vec{V_r}(i_r, j)]_k, \tag{2.5}$$

where k represents the vector index and w_k is the weight.

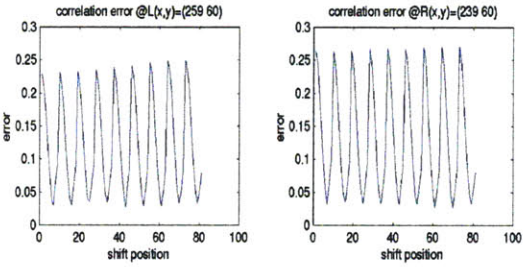
For edge pixel EL of left images, we search for its matching correspondence pixel among multiple candidates, edge pixels ER_1, ..., ER_m, ..., ER_n, at the same row as EL



(a) Two pairs of corresponding points in binocular stero images. Top/bottom: left/right images.



(b) Left/right: MV-CE-Curves for yellow points (top/bottom) marked in (a).



(c) MV-CE-Curves for red points marked in (a)

Figure 2-6: Similarity of MV-CE-Curves and Texture Index for two pair of corresponding edge pixels in binocular stereo images. (a) Top/bottom: binocular stereo images and corresponding points from left/right cameras. (b) (c) Left/right: Variation of correlation errors when searching for motion vectors (MV-CE-Curves) for yellow/red points, top/bottom, marked in (a).

in right-camera images. We compare multidimensional feature vectors for pair EL and ER_m, search for pairs with minimum matching errors, and record corresponding disparities $x_l - x_{r_m}$ in a disparity histogram. If there are multiple possible matches in the right image for a feature in the left image, we pick the disparity with smallest values and consider the pixel as background pixel. Such conservative choice ensure the histogram peaks with large disparity are noise free. A disparity histogram reflects the distribution of disparities for stereo visions.

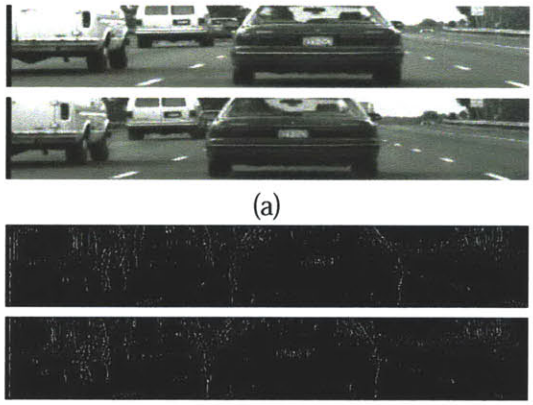
2.2.2 Depth Range Detection through Disparity Histogram

The concentration of edge points at peaks in the disparity histogram comes from targets in the same distance ranges. Normally, peaks at large disparity locations correspond to close targets and peaks at small disparity locations correspond to targets that are far away. Compared to vehicles that are far away, closer vehicles occupy larger image area (excluding the possibilities of toy cars on the road). Therefore, for obscale edges, the number of pixels with large disparity will be larger than the number of pixels with small disparity. Among the peaks corresponding to real obstacles, the higher peaks are normally located to the right of the lower peaks in the disparity histogram as shown in the matcing relationship between Figure (2-9)(a2)(b2) and Figure (2-9)(c). Occasionally, due to the complexity of targets, farther vehicles might have more edge pixels in images and have higher peaks in disparity histogram than closer vehicles. However, generally, to identify a peak in a disparity histogram as a target other than noise, we have to choose high threshold if the peak is at the location with large disparity as shown in Figure (2-9)(a2)(b2)(c). The larger the peak's disparity, The higher the threshold should be. Based on these common characteristics of the histogram peak distribution shape, target depth information can be derived.

2.2.3 Target Depth Detection Results

In this subsection, we will present a detailed example of motion-based target depth detection for binocular stereo images in Figure (2-7) (frame No.57 in our video sequence), in which the three vehicles are named "left car," "middle car," and "right car." We need to first

detect three depth ranges for the three vehicles.



(a)

(b)

Figure 2-7: Binocular stereo images and their corresponding edge-maps. (a) Images from left(top) & right(bottom) cameras at frame No.57. (b) Edge-maps for (a).

Motion information is computed based on two consecutive frames, Frame No.57 and Frame No.58 (Figure (2-8)(a)). Based on the motion similarity shown in figure (2-8)(b) for given binocular image sequences, we obtain the motion-based correspondence matching result for stereo frames No.57 (Figure (2-7)(a)). The disparity histogram is shown in Figure (2-9)(a). For the comparison purpose, we also compute disparity histogram based on trinocular algorithm [2] in Figure (2-9)(b).

(a) Images from left & right cameras at frame No.58

(b) Motion vector between frames No.58 and No.57 (Fig. (2-7).

Figure 2-8: Similarity of motion vectors for binocular stereo images.

In Figure (2-9)(a), the three highest peaks correspond to the three nearest vehicles in frame No.57 with detailed information as shown in Table 2.2.3. For Figure (2-9)(a), the peak for the left most car (marked by green circle) is higher than for the right most car

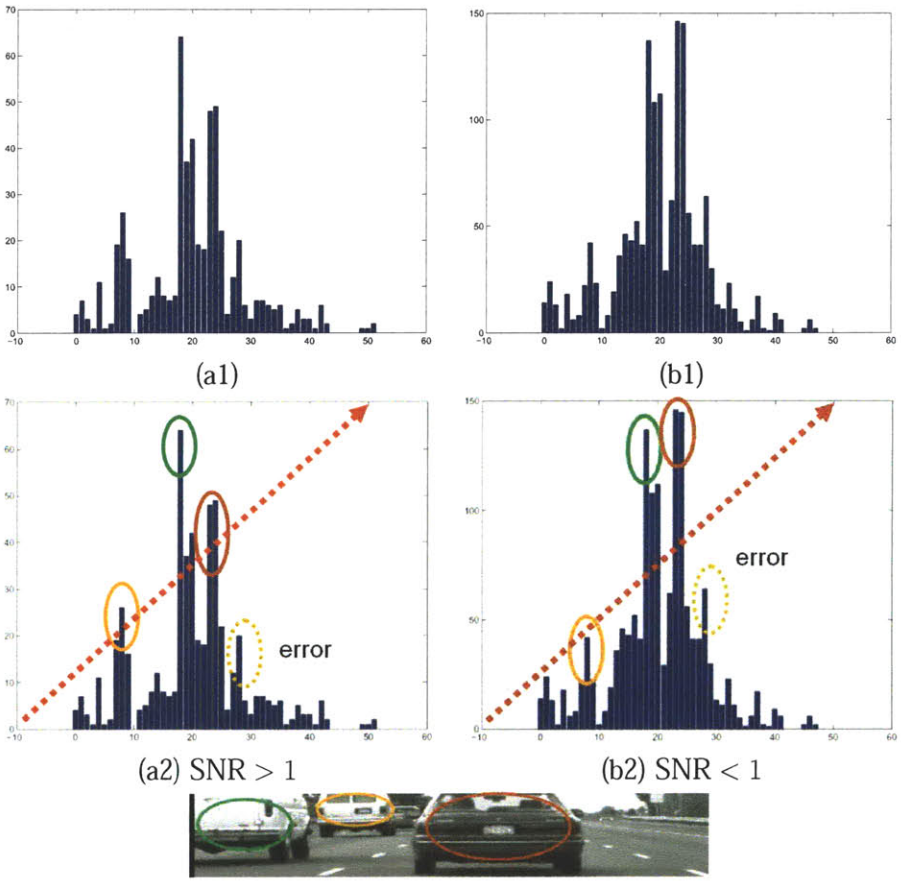


Figure 2-9: Disparity histogram for edge pixels in Figure (2-7)(b) and corresponding maching relationship. (a)(b) x coordinate: disparity value. y coordinate: the number of edge pixels within the disparity range. (a1)(a2) Motion-based binocular stereo algorithm using frames No.57 in Fig. (2-7)(a) and No.58 in Fig. (2-8)(a). (b1)(b2) Trinocular algorithm.

(marked by red circle) because the depths of the left car and the right car in Figure (2-7)(a) (frame No.57) are similar while the left car has more complex projection in the edge-map shown in Figure (2-7)(b). The lower peaks to the right of the three highest peaks are noises.

category	Disparity	Closeness	Edge Pixel #
Left Car	18-20	2nd nearest	most
Middle Car	7-9	3rd nearest	3rd most
Right Car	22-24	1st nearest	2nd most

For disparity histogram based on trinocular algorithm [2] as shown in Figure (2-9)(b), three highest peaks, is in decreasing order (from high to low), are at disparity ranges $23-24$, $18-20$, 28. The trinocular-based histogram misses the target at disparity range $7-9$ while introducing a false target (peak) at disparity 28. The detection error is caused by mis-correspondence. For both case in Figure (2-9)(a) and (b), the dotted lines can help to differentiate meaningful peaks from noises. However, the robustness for both cases is different.

We define the signal to noise ratio (SNR) as the ratio of the lowest peak corresponding to real vehicles over the highest peak corresponding to false alarms. For the trinocular stereo algorithm, the SNR is around 0.75, less than *one*, while the SNR from our proposed algorithm is around 1.5, which offers better performance. The histogram comparison in Figure (2-9) shows the performance improvement of motion-based target detection over trinocular algorithms.

Our proposed target detection algorithm can be fused with radar information as in Figure (2-2). Target information from radar would help to differentiate target peaks from noise peaks in choosing peaks from disparity histograms and to give out more reliable target depth information. Fusing depth information from two sources would help to reduce reflection errors from radar systems and mis-correspondence from vision systems. The calculated vehicle disparity ranges are later used in Section 2.3.4 to segment vehicles.

Combining the proposed "motion-based target depth detection system" with "depth-based segmentation system (Section 2.3)" as in shown in Figure (2-2), our proposed system can offer both depth and location information for targets without introducing too much complexity.

2.3 Fusion-based Layer-based 2D Segmentation

Given extra disparity ranges for targets, we propose a feature-level fusion scheme to identify target locations. The key component of our 2D segmentation is edge layer separation, which leads to our fusion-based layer-based 2D segmentation methodology. The edge-map of a real scene image is composed of edge pixels that belong to different targets at different depth ranges. Given a series of disparity ranges from either radar or pure vision systems, we first split an edge-map of a binocular image into n edge-layers that contain the edge pixels of targets corresponding to n different given target disparity ranges. It is the relationship between target depth and its correspondence-disparity that imposes constraints on searching for corresponding groups of target feature points at different depth ranges from the edge-maps of original stereo images. The detailed algorithm is in Section 2.3.1.

Each depth-based feature layer has fewer targets than the original whole image. In other words, the original multiple-target segmentation task is decomposed into several simple and relatively easy single-target segmentation tasks on several depth-based target feature layers, thus the segmentation performance will be improved. It is easier to sequentially locate the corresponding positions of n targets at different distance ranges in n separated edge layers than in the original segmentation task. More accurate and robust target segmentation can be achieved.

The depth-based segmentation algorithm needs only coarse target depth information to separate depth-based feature layers and imposes minimum performance requirements on depth detection sensors. A low-quality radar system is adequate since accurate depth information for every pixel is not needed. When the information of distance ranges from a radar is not available, we have devised a pure vision-based target depth-range detection algorithm in Section 2.2 to detect the number of targets and target depth information. For real time applications, it is better to combine depth information from both radar and "vision-based target depth detection" algorithm to achieve ideal performance.

Sensor fusion system in Figure (2-2)(b) can provide better target detection than separately detecting segmentation regions and depth information and matching them afterward in Figure (2-2)(a). Depth-based segmentation algorithm helps to enhance target segmen-

tation performance. On the other hand, finding the correspondence between segmented regions on stereo images can also improve the estimation of target depth. The performance of each sensor in the fusion system would be better than it is being used alone.

Our proposed fusion-based layer-based segmentation algorithm as shown in Figure (2-2) (b) consists of three steps. We will first focus on the layer-based concept to discuss depth-based edge-layer separation, and then explain the fusion-based target boundary detection.

2.3.1 Step 1: Depth-Based Edge Layer Separation

Depth-based edge layer separation is feasible due to the relationship between the target distance and the correspondence disparity of target in binocular stereo images. The object distance range imposes constraints when searching for corresponding feature points in the left and right stereo images. Such constraint helps to separate edge layers at different distance ranges. Binocular vision systems can make use of object distance information Z from radar to identify corresponding target objects in that distance range, and to provide the X, Y position information for targets.

The objective of this first step is to sequentially locate image edge pixels corresponding to targets at given disparity ranges. To obtain the corresponding depth-based edge layer for each depth range, we locate target edge pixels that simultaneously satisfy depth constraints and other binocular stereo correspondence constraints. These correspondence constraints are defined as followed:

- *Binocular Stereo Epipolar Line Constraint*

For aligned cameras, matching feature pixels from stereo images should be on the same rows.

- *Depth/Disparity Constraints*

For a binocular stereo system, target depth z imposes constraints on its correspondence disparity. For the edge pixels of targets within a distance range, the differences of their x-coordinates in binocular stereo images $x_l - x_r$ should be within the given disparity ranges, which can be found using equation (2.2) for radar systems or disparity ranges for pure vision systems or both.

- *Edge-feature Constraints*

Similarly, we also adopt a "positive and negative edge" scheme instead of binary edge-maps to decrease mismatching.

Given disparity range information, we apply the above three constraints to choose edge pixels in the left images for which there exist corresponding edge-feature pixels in the right edge-map. For each edge pixel in the left edge-map, we search for edge pixels on the corresponding epipolar lines in the right edge-maps with disparities falling the range defined by Equation (2.2). We do not differentiate among multiple candidates from the right edge-map. As long as there exists corresponding edge pixels in the right edge-map, we keep the pixel in the left edge-map, which make up a depth-based edge layer. Some edge pixels can be chosen at different depth edge layers. All chosen feature pixels are very helpful in segmenting objects within the given disparity range. Applying this step at different disparity ranges, corresponding edge layers are obtained and we separate the original stereo edge-map into different distance layers. We further detect whether there are any objects and where objects are at each distance-based edge layer. Noises due to edge layer separation and the residual background noises can be later removed by depth-range-based filtering.

2.3.2 Step 2: Depth-Based Target Contour Discrimination

In each separated edge layer, target pixels are clustered, while pixels from other depth ranges scatter randomly throughout the layer. Many segmentation algorithms determine target regions by locating where most edge pixels cluster. The performance of such algorithms is sensitive to the choices of thresholds, and is not robust enough.

Instead of applying segmentation algorithms directly, we adopt edgeline-based morphological operation to connect all edge pixels into long lines. In edge detection process, we register all vertical edge pixels in different edge lines. Then we delineate candidate objects in an edge layer through a morphological "closing" operation or depth-based target contour discrimination. More particularly, we first dilate vertical edge lines in the edge layer to increase their length. The neighbor edge-lines can be connected at the neighborhood of their

terminals. The aim is to connect all the boundaries of targets into long boundaries with lengths longer than those of noisy edges. We then apply an erosion operation to remove isolated edge pixels or short edge-lines caused by noises. After these two operations, long meaningful edge-lines are kept and interested targets should stand out since the boundaries of our targets are typically longer than noisy edges coming from other depth layers.

2.3.3 Step 3: Depth-Based Target Boundary Determination

Because target contours are manifest in separated edge layers after the "discrimination" step, the final step can create rectangular bounding boxes for targets through finding the left and right locations of longest boundary lines and pixel aggregation with seed points being chosen from accentuated candidate points.

Repeating the proceeding steps at different disparity ranges, we can sequentially detect whether there are objects within the different depth ranges and where objects are, i.e., the locations and the sizes of targets. Examples of segmentation results are presented in the next section.

2.3.4 Segmentation Results

In this subsection, we will present detailed segmentation results for binocular stereo images shown in Figure (2-7). Without losing generality, we show only results for the left images from the segmentation step. We detect three vehicles' locations in Figure (2-7)(a) after applying the depth range determination (step 0) and the three steps of 2D Segmentation (step 1, 2, 3).

Step 0: Depth Range Determination

We obtain the depth information, disparity histogram, shown in Figure (2-10) based on our motion-based binocular methods discussed in Section 2.2. The three highest peaks from the disparity histogram as shown in Figure (2-10) represent three nearest vehicles whose disparity ranges are 22-24, 17-19, 7-9. The larger the disparity range is, the closer the vehicle is to the observer.

Step 1: Depth-Based Edge Layer Separation

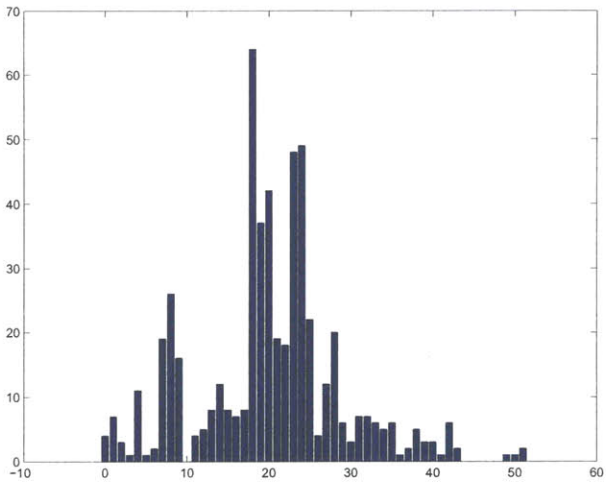


Figure 2-10: Histogram of disparity for Figure (2-7)(frame No.57).

The edge-map shown in Figure (2-7)(b) (for left image) is decomposed into three edge layers corresponding to the three disparity ranges 7-9, 18-20, and 22-24. The three edge layers corresponding to these disparity ranges are shown as in Figure (2-11).

Though there are edge pixels overlapping among the three edge layers, the three separate vehicles are distinct. The depth-range-based target contour discrimination and determination in step 2 and 3 will remove residual noise pixels from other distance-range layers.

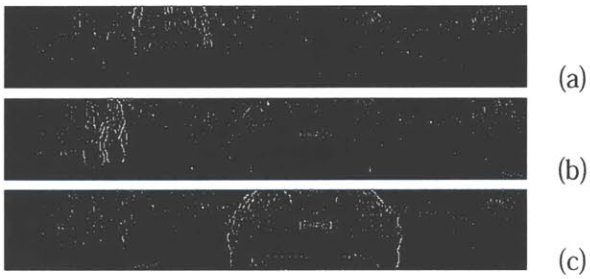


Figure 2-11: Edge layers corresponding to different disparity ranges for Figure (2-7)(frame No.57) (a) 7-9 (b) 18-20 (c) 22-24

Step 2: Depth-Based Target Contour Discrimination

For the edge layer shown in Figure (2-11)(c), the dilation operation result is shown in Figure (2-12)(a), and the closing result (from applying both dilation and erosion operations)

is shown in Figure (2-12)(b). The morphological operations successfully remove most error edge lines, thus the target vehicle stands out and can be easily detected. Our edge-line based operation is effective in enhancing visibility of interested targets and segmentation performance.

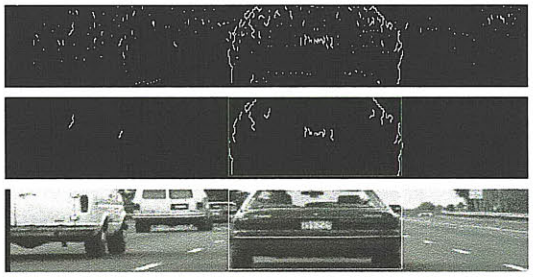


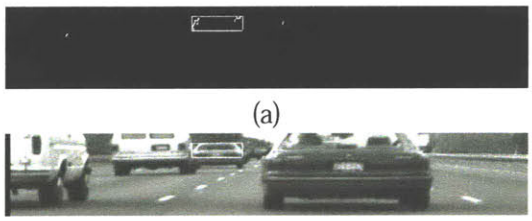
Figure 2-12: Procedure to locate targets within a depth range which corresponds to disparity range 22-24 for Figure (2-7) (frame No.57). (a) Edge dilation result (b) Edge closing result (c) Segmentation result

Step 3: Depth-Based Target Boundary Determination

Figure (2-12)(b) and (c) plot segmented regions for the right vehicle (at disparity range 22-24) in frame No.57 in video sequence shown in Figure (2-7). Additional segmentation results for the right vehicle in frames 62, 69, 73, and 80 are shown in Figure (2-13). The segmentation results at disparity ranges 7-9 for frame No.57 and No.31 are shown in Figure (2-14). Segmentation results for the new vehicle with disparity range 5-6 in frame No.31 are shown in Figure (2-15).

Figure 2-13: Example of detecting the right vehicle at frames No.62, No.69, No.73, No.80.

Figure 2-14: Example of locating targets at different disparity ranges. Top row: disparity range 7-9 at frame No.57 (Figure (2-7). Bottom row: disparity range 7-9 at frame No.31. Left column: Edge closing operation result. Right column: segmentation result.



(a)

(b)

Figure 2-15: Example of locating targets with disparity range 5-6 (frame No.31) (a) Edge Closing Operation Result. (b) Segmentation Results.

Our algorithm uses the same initial parameters to segment targets at different depth ranges and in different video frames as in Figures (2-12), (2-13), (2-14), (2-15), and Figure (2-16). These results illustrate target detection with accuracy and robustness.

Our algorithm does not impose high requirements on the accuracy of target depth determination and edge detection. We use several different disparity ranges, $22-24$, $23-25$, $22-25$, etc., and different Sobel edge-detection thresholds to yield the same segmentation results. At the same time, the segmented regions from stereo images can also improve depth estimation when the resolution of given depth is not satisfactory.

2.3.5 Performance comparison

Our morphological operation in segmentation algorithms differs from other traditional morphological operation in the following aspects. First, traditional methods identify targets in the original edge-map that involves heavy background noises. For our method, in separated edge layers, there are less targets, less noises and cleaner backgrounds than in the original image. Segmentation on each layer is simpler and easier than on the original edge image.

Secondly, our morphological operation is edge-line based instead of pixel based. The

purpose is to ensure the vertical edges at horizontal boundaries of interested targets to be longer than at interior regions and at background regions in the separated edge layers. It is feasible for most obstacles in driving situations since their longest vertical edge-lines are normally at their left/right boundaries and edge-based separation removes majority of long edges from obstacles in other depth ranges. Thus the boundary locations of interested targets can be determined by searching for horizontal locations of the left most and right most long edge-lines in separated edge maps. The detection can be obtained with high accuracy using edge-line based operation and is robust to different choices of initial parameters.

Some traditional segmentation algorithms locate targets based on 3D information and cluster pixels with similar depth information. Such operation is somehow similar to obtain Figure (2-11) and to aggregate pixels that have similar disparities and are close to each other. We have tried to detect targets in Figure (2-11) in this way. We search for a region containing majority of pixels with similar depth/locations. Such method is very sensitive to the initial parameters. Generally, there are noises involved when obtaining 3D reconstruction from binocular stereo images using simple correspondence matching algorithms as shown in Figure (2-11). Thus final segmentation is not very reliable.

2.4 Discussion

2.4.1 Two correspondence matching

Our depth-based segmentation algorithm does not impose high requirements on depth accuracy. Only coarse target depths are needed, and these can be from either radar with low resolution or simple vision-based algorithms.

In our proposed method, we apply correspondence matching twice. The first time, we apply motion-based correspondence matching to obtain disparity histograms. We define feature vectors of edge pixels to find corresponding edge pairs from multiple candidates. Our motion-based correspondence matching takes advantages of the extra information that binocular stereo video (multiple frames) provides when compared to static binocular images, which increases reliability. Among multiple matching candidates, we choose the match with smallest disparity. The purpose of the first correspondence matching is to provide the statistical distribution of disparity.

The second correspondence matching operation is to separate edge-maps into several edge layers using the disparity ranges obtained in the first correspondence matching. Then in each layer, we locate edge pixels with matching correspondence edge pairs. We do not need to differentiate multiple matching candidates and allow one pixel to be chosen at multiple layers. The requirement on the accuracy of the second correspondence match during "Depth-Based Edge Layer Separation" is lower than that needed to obtain disparity histogram.

2.4.2 Robustness of the algorithm

As shown in Section 2.3.4, depth-range-based segmentation is very robust to the impact of noises and initial parameters. The key step of the algorithm is to identify edge pixels corresponding to specific disparity ranges with edge-layer separation and to intensify vertical edges at the left/right boundaries of interested targets with edge-line-based morphological operation. In the separated edge layers, the vertical edges at targets' left and right boundaries are longer than in the background regions, and targets can be easily detected. Thus,

the noises introduced at different steps would not have huge impact on final segmentation performance as long as the long vertical edges at targets' boundaries are connected and stand out in the background regions.Depth-based edge-layer separation algorithm is robust to different choices of initial parameters. It is not necessary to have high quality edge detection, strict edge layer separation, and accurate disparity ranges. We have chosen the simple and fast Sobel edge detection algorithm. Low edge-detection thresholds are used to provide sufficient pixels for reliable target discrimination in the next step. In Chapter 3, we will show how to reduce the impact of background noises due to non-rigid objects such as trees.

2.4.3 Computational loads

Our layer-based method only detects targets within the depth ranges of interest for the purpose of autonomous driving, and does not try to recover accurate 3D shape information for all objects in the images, which really enhances the efficiency and performance.

Among the four operations discussed above, most computation time is spent in correspondence searching at the second step of "depth edge layer separation." In each edge layer, the correspondence searching time for each edge pixel is in direct proportion to the size of its disparity range instead of the typical image width. The computational load of our algorithm is in direct proportion to the product of target numbers, the number of edge pixels and the size of disparity ranges. The correspondence searching process can be implemented simultaneously at different rows or even at different pixels, which makes parallel VLSI hardware implementation possible and further increases calculation speed.

Due to the light computational load of our segmentation method, we can afford to apply such segmentation to every frame for target detection. Unlike other tracking-based segmentation, such segmentation algorithm does not need initialization time. Thus, our algorithm has the potential to quickly detect new targets (cut-ins) and respond to rapidly changing environment in real-time driving situations. Better initial static segmentation always improves the performance of an overall segmenting/tracking system. In summary, our fusion-based layer-based algorithms can significantly enhance the performance of static segmentation as

well as the performance of target detection and tracking in a video sequence.

2.4.4 Limitations and possible solutions

In general, our proposed methods do not make assumptions on target shapes or the symmetry of targets, and have wide range of applications. Our algorithm is limited in situations where only single sides of targets are in images, as in Figure (2-16). Depth-based segmentation needs to capture target boundaries. If there is only a single side of a target in an image, we capture only partial target regions. Such problems can be solved using the motion-based region expansion described in the next Chapter. With several modifications, our algorithm can detect targets of various shapes, such as pedestrians as shown in Figure (2-17), which is to be discussed in the next chapter.

Figure 2-16: Target segmentation results within disparity range 18-20 for frame No.57 (Figure 2-7). An example of "single-sided limitation" where only part of the target is captured, which leads to target size detection error. It happens when only part of target is in the image.

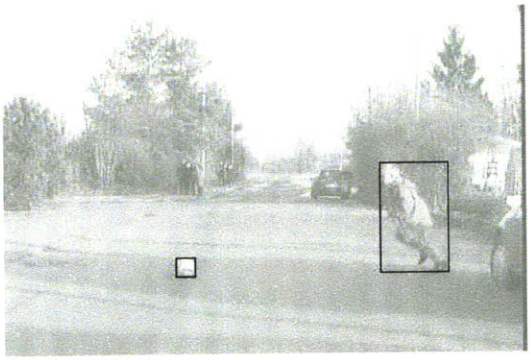


Figure 2-17: An example of detecting both ball and pedestrian using same initial parameters.

There still exist several other issues to be further pursued. For example, one may desire the ability to segment multiple objects within the same distance range. The distance-based

segmentation algorithm detects target boundaries and removes correspondence errors by growing contours and removing short edges. The edge-length threshold can be adjusted based on different distance ranges. However, when there are several objects of different sizes within the same distance range, a fixed-length threshold might cause troubles. If we use a large edge-length threshold to remove shorter-edged noises, we might remove short contour information that belongs to our target objects. We would not detect accurate bounding boxes for all individual targets.These issues will be addressed in the next chapter.

Chapter 3

Target detection and 3D segmentation under heavy background noise based on motion and distance range

3.1 Introduction

Segmentation algorithms are typically challenged by noisy static background and the variation of targets' locations and sizes. This leads to various segmentation errors. Static segmentation usually has limited performance and one depends on tracking algorithms to remove the errors in initial static segmentation. Tracking algorithms do not work well when dealing with background noise which changes over successive video frames.

In order to reduce or remove the impact of heavy background noise on segmentation performance, many segmentation algorithms first detect background information and then subtract background images from the original video frames. It becomes relatively easier and more reliable to segment objects in processed images than in the original images. "Background subtraction" is one widely used method whose performance depends on the quality of reference images. There are two basic techniques in acquiring reference frames [59]. The first basic technique is for situations such as indoor video surveillance and monitoring. Reference images can be obtained by directly taking pictures when there

are no target objects. The limitation of the technique is that the reference frame might not always be available. Reference frames have to be updated from time to time in order to match the current lighting levels. But reference frames under different lighting levels are not always available. The second technique treats the previous frames as reference images for sequential frames when changes between successive video frames are small. "Adaptive background detection" [44] improves the quality of background detection based on Gaussian models. The two techniques apply only to controlled situations and might not function well when video frame backgrounds change from frame to frame, such as for backgrounds containing non-rigid objects like trees. For video frames captured in intelligent vehicles, the frame backgrounds are constantly changing and unpredictable. Thus, a new algorithm is needed in order to offer satisfying results in segmentation or motion detection.

In this chapter, we propose a fusion-based layer-based method to suppress background noise and to increase segmentation reliability. In Chapter 2, we have presented a new distance-range-based segmentation method to reduce correspondence errors so that the accuracy and robustness of target detection can be improved. Our proposed algorithms work well for our test sequences in highway environment which are usually less affected by background noise than the image sequences taken in urban driving environment. In this chapter, we further discuss how to apply the principle of layer-based and fusion-based segmentation to reduce the impact of background noise. The chapter has two parts. In the first part, a distance-based background detection algorithm is proposed to reduce the influence of noisy backgrounds without using reference frames. In the second part, a motion-based segmentation and adjustment is presented to detect objects of different sizes. The background detection/removal algorithm is layer-based. The depth/motion-based segmentation algorithm is fusion-based. Our proposed methods overcome the limitations of traditional segmentation, while increasing the accuracy, reliability, and robustness of object segmentation in the presence of heavy background noise.

3.2 Background Detection and Removal

3.2.1 The definition of distance-based background

In order to effectively remove the impact of the background, we need to clarify the definition of backgrounds which vary in different application scenario. If the objective is to detect moving objects, static objects are considered background. When one tries to detect distance using binocular stereo systems, the objects in other distance ranges are considered noise which can introduce correspondence errors and lead to erroneous distance information. For the purpose of target detection in intelligent transportation systems (ITS), it is appropriate to define backgrounds as objects whose distance ranges are much farther than the one of interest to us. In other words, objects beyond specific distance ranges are less important than close objects. For example, for the scenario shown in Figure (3-1)(a)-(d), trees and pedestrians far away are considered as backgrounds since they can be temporarily ignored in most driving situations. Such background objects contribute a majority of edge pixels in the edge images shown in Figure (3-1)(e)(f). These background pixels are noise which leads to matching errors for binocular stereo correspondence and for motion vector computation when comparing either left/right edge maps or two consecutive edge frames. Such errors lead to difficulties in achieving accurate segmentation.

3.2.2 Distance-based Background Separation

In order to remove the impact of these background pixels, we take advantage of the depth-based image layer separation proposed in Chapter (2) to split background layers from the original edge maps.

With algorithms of motion-based binocular stereo correspondence matching introduced in Chapter 2.2, we obtain disparity histograms. Object pixels with small correspondence disparity are farther away than object pixels with large disparity. When object disparities are almost zero, objects are infinitely far away. Background objects that are far away have smaller disparity than target objects. Thus, pixels with small disparity represent the characteristics of the background.

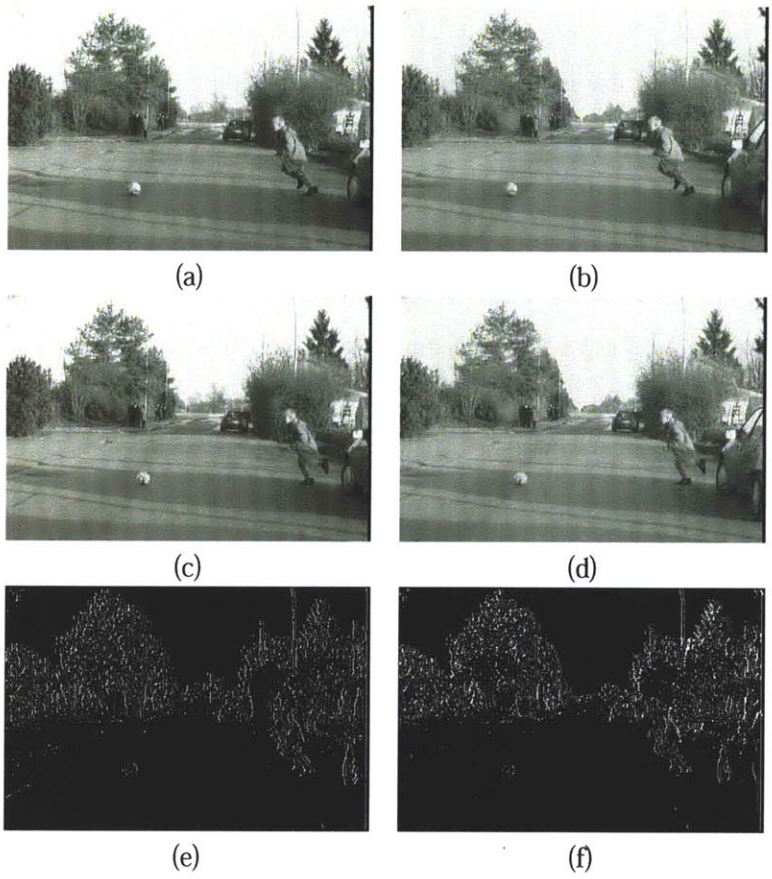


Figure 3-1: Binocular stereo frames and edge images. (a)(b) Left/Right images at time n. (c)(d) Left/Right images at time $n-1$. (e)(f) Left/Right edge images at n. Images (a)-(d) are courtesy of Daimler-Chrysler Research and Technology.

For frames in Figure (3-1), the disparity histogram of binocular stereo matching is shown in Figure (3-2). Video frames in Figure (3-1)(a)-(d) are courtesy of Daimler-Chrysler Research and Technology. The large peak at disparity $3-4$ on a disparity histogram means that there are many background pixels. Another peak at disparity range $17-19$ corresponds to the distance range where the boy and the ball are situated. If we are only interested in targets in this distance range, we can define background distance range to be around disparity range 0-5, case (I), or 0-9, case (II). Case (I) is more conservative choice than case (II).

The results of depth-based layer separation are shown in Figure (3-3). Separated background edge layers for case (I) and case (II) are respectively shown in Figure (3-3)(a1) and (a2). Both background layers capture most noisy edge pixels due to trees and pedestrians far away, and look quite similar. Separated target edge layer with disparity range 17-19 is shown in Figure (3-3)(b). Even though the segmentation on this targe edge layer is already easier than on the original edge layer, the heavy background noise still poses difficulties in target segmentation. We will demonstrate that the segmentation accuracy can be improved by combining the information of background edge pixels in Figure (3-3)(a1)(a2) and the separated target edge layer in Figure (3-3)(b1)(b2).

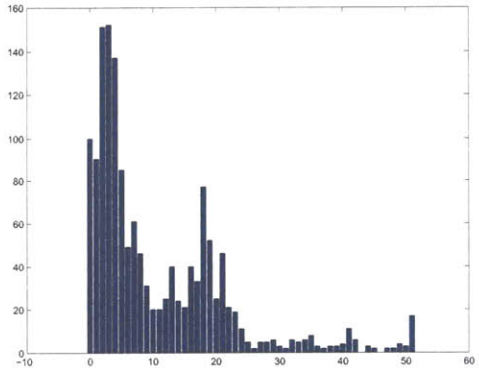


Figure 3-2: Disparity distribution

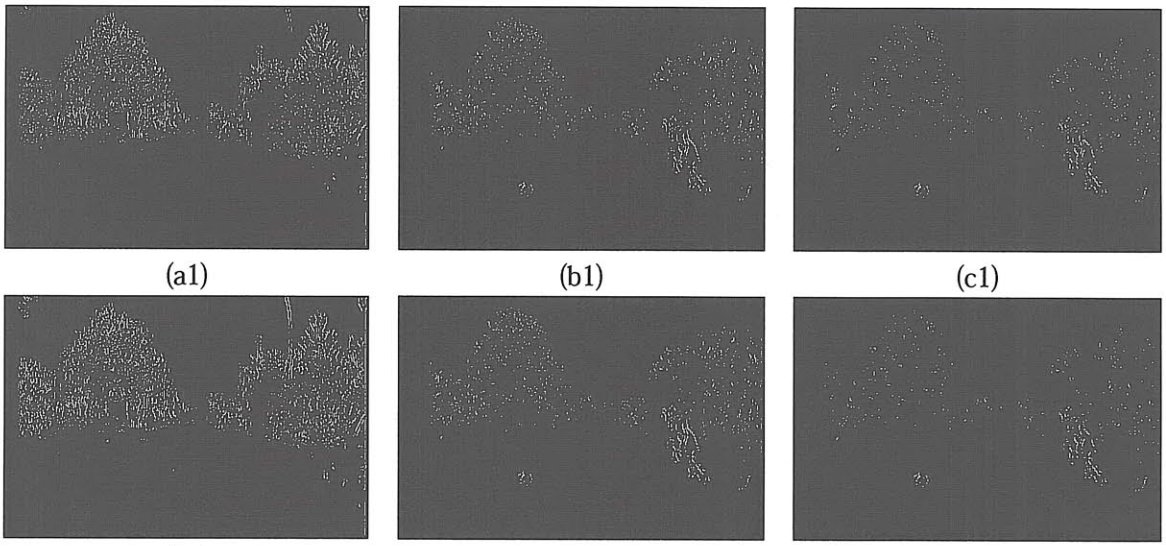


Figure 3-3: Distance-range-based image separation based on figure (3-2) for figure (3-1)(a)(b). The top and bottom rows correspond to two possible separation. (a1): Background layer at disparity 0-5. (b1): Object layer at disparity range 17-19. (c1): Cleaned object layer at disparity range 17-19. (a2): Background layer at disparity 0-9. (b2): Object layer at disparity range 17-19. (c2): Cleaned object layer at disparity range 17-19.

3.2.3 The Impact of Background Noise

Before discussing how to take advantage of distance-based background layers to remove the impact of background noise, we first investigate how background noise would affect the performance of applying distance-based target detection described in Chapter 2 to target edge layer with disparity range 17-19 in Figure(3-3)(b). The results of individual processing steps are shown in Figure (3-4). When target boundary contours are elongated in morphological operation, isolated noise pixels from background trees are connected as shown in Figure (3-4)(b). The huge number of tree edge pixels leads to many false edge lines in background areas which might be even longer than the edge lines for target objects. Thus it is hard for an edge length filter to eliminate false edge lines in background areas without affecting target objects. The noisy background pixels add difficulties in simultaneously segmenting objects with different sizes, for example, the boy and ball. As shown in Figure (3-4)(c) and (d), it is unavoidable to have false boundary edges and false bounding boxes in background areas.

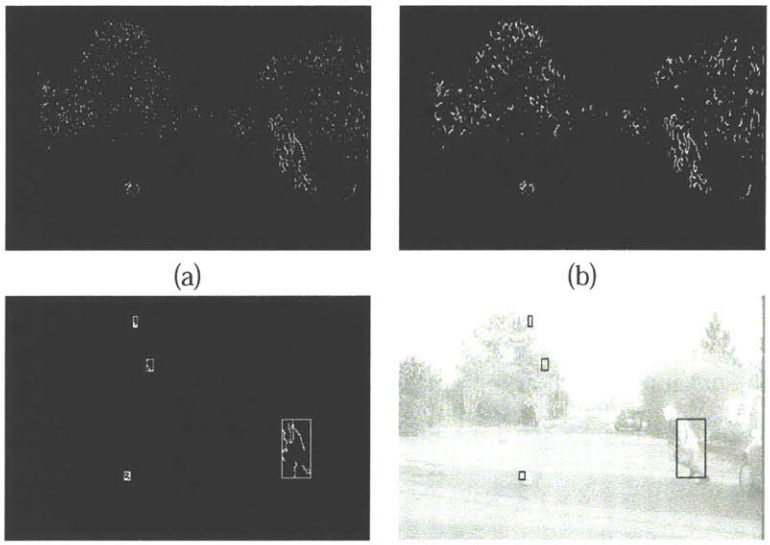


Figure 3-4: Distance-range-based segmentation results. (a) Left/Right matching result at disparity range 17-19 (from Figure (3-2) for Figure (3-1)(a)(b). (b) Morphological operation result on (a). (c) Edge-length filtering and segmentation result from (b). (d) Segmentation result drawn on Figure (3-1)(a).

	given object distance range	other object distance range	background range
edge category 1	$\surd$	no	no
edge category 2	no	$\surd$	no
edge category 3	no	no	$\surd$
edge category 4	$\surd$	$\surd$	no
edge category 5	$\surd$	no	$\surd$
edge category 6	no	$\surd$	$\surd$
edge category 7	$\surd$	$\surd$	$\surd$

Table 3.1: Possible category of edge pixels

The impact of background noise on segmentation in our distance-based edge layers arises because of the correspondence matching process when we separate distance-based edge layers. To obtain each distance-based edge layer, we search for edge pixels in the left images with correspondence edge pairs in the right images along the epipolar lines with disparities within the range defined by Equation (2.2) in Chapter 2. Edge pixel in the left images might be matched to several edge points with different disparity ranges in the right images, which corresponds to multiple choices of edge layers at different distance ranges. Without further information it is hard to decide among multiple choices. Some algorithms simply pick one distance randomly, which might not be the correct one. To avoid making the wrong decision among multiple solutions, our algorithm allows edge pixels in the left images to be matched to *several* edge points with different disparity ranges, and to be assigned to multiple edge layers in several distance ranges as shown in Table 3.1. For a given distance range the chosen edge pixels can be in categories 1, 4, 5, and 7 as shown in Table 3.1, while edge pixels in categories 4, 5, and 7 are also chosen in edge layers for different distance ranges. Our distance-based edge layer separation keeps all possible edges and includes as much target edge information as possible since it cannot afford to lose target edge pixels, which might be important component in target's contours. However, edge pixels in categories 4, 5, and 7 might introduce ambiguities since they might actually belong to background layers or other distance ranges containing other targets.

In most situations, the ambiguity caused by pixels from rigid objects in other distance-based layers is limited. The disparities of rigid objects with clear contours can normally be determined. It will be less likely for the edge pixels of rigid objects to be chosen in other distance ranges than for non-rigid objects. Even if it happens, the false edge pixels might not constitute a long edge and can be eliminated by edge length filters. Thus, there should not be too much noises from rigid objects in edge layers at different distance layers, including at background.

However, when matching non-rigid background objects in left and right images, there will be lots of correspondence matching ambiguity caused by non-rigid background objects, such as trees, which have high variance of texture and yield many edges. In the step of binocular stereo matching for different disparity ranges, pixels in the non-rigid back-

ground objects can easily be matched to multiple pixels of non-rigid background objects. For example, a leaf in one tree can be easily matched to multiple leaves in multiple trees. Thus a large amount of pixels in the non-rigid background objects will be chosen in almost every interested target distance layer. These falsely chosen background edge pixels bring in difficulties to segmentation tasks in edge layers at different object distance ranges. As shown in Figure (3-4), these misassigned edges are close to each other and will be connected together in morphological operations, thus leading to long noisy edges that cannot easily be filtered by an edge length filter.

3.2.4 Distance-based Background Removal

In order to obtain ideal segmentation accuracy and reliability when background noise is heavy, our distance-based target detection algorithm needs to remove background pixels introduced by non-rigid background objects in our interested target edge layers. We first identify the edge layers corresponding to backgrounds as shown in section 3.2.2. Then we compare the background layer and edge layers at target distance ranges, and subtract the common pixels from target edge layers. After these operations, edge pixels in categories 5 and 7 shown in Table 3.1 are removed, and only edge pixels in categories 1 and 4 remain.

The results of distance-based background removal for frames in Figure (3-1) are shown in Figure (3-3). Figure (3-3)(a) and (b) respectively correspond the separated background edge layers and target edge layer with disparity range 17-19. Figure (3-3)(c1)(c2) are the result of removing background pixels from edge layers at given distance ranges, disparity 17-19. In other words, the common pixels that exist in both Figure (3-3)(a) and Figure (3-3)(b) are removed from Figure (3-3)(b). We have named background layers with two disparity ranges 0-5 and 0-9 to be case (I) and case (II). The top and the bottom rows in Figure (3-3) respectively correspond to processes of distance-based background removal for case (I) and case (II). Background noise in processed target layers in Figure (3-3)(c1)(c2) is much less than those in the original separated target edge layers in Figure (3-3)(b1)(b2). Most tree pixels (background) are removed while preserving all the pixels of target objects including the ball.

The left and the right columns in Figure (3-5) respectively correspond to the results of target segmentation for case (I) and case (II). Figure (3-5)(a) are the target edge layers after background removals, and Figure (3-5)(b)-(d) show the results from morphological operation, edge-length filtering, and target locating. The final segmentation results in Figure (3-5)(d) show the ability of our algorithm to suppress the influence of background noise.

However, when eliminating background pixels to make the target pixels prominent, we also lose some edge pixels or even pixels at the boundaries of target objects. For example, some edge pixels in the heads of pedestrians are missing as shown in the comparison between Figure (3-3)(b1)(b2) and Figure (3-3)(c1)(c2). The chosen edge pixels are not complete enough to preserve the full boundaries of target objects, leading to smaller final segmented target regions than its real sizes. The captured region of the pedestrian in Figure (3-5)(d1)(d2) is even smaller than Figure (3-4)(d). Since the definition of background layers for case (II) is more aggressive for case (I), background removal results for case (II) eliminates more background pixels than for case (I), and also lose more useful pixels than for case (I). Thus segmentation results for both cases are smaller than real sizes and the segmentation results for case (II) are even smaller than for case (I). In summary, the operation of removing background noise is at the cost of losing some target edge pixels. In the next section, we will propose an algorithm to compensate for the information loss and performance degradation.

3.3 Motion-based target segmentation

In order to regain the lost target pixels when background edges are being removed, we have to expand shrunk segmentations by taking advantage of the similarity of motion vectors for target objects in their distance-based edge layers.

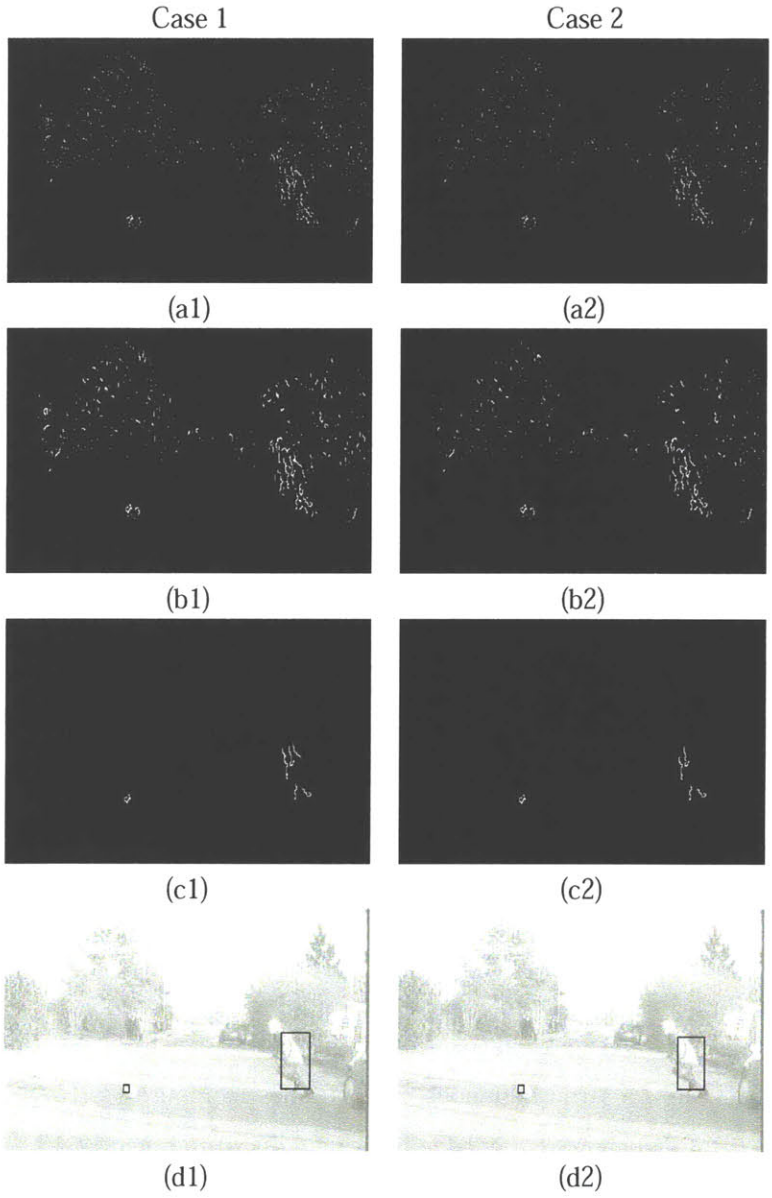
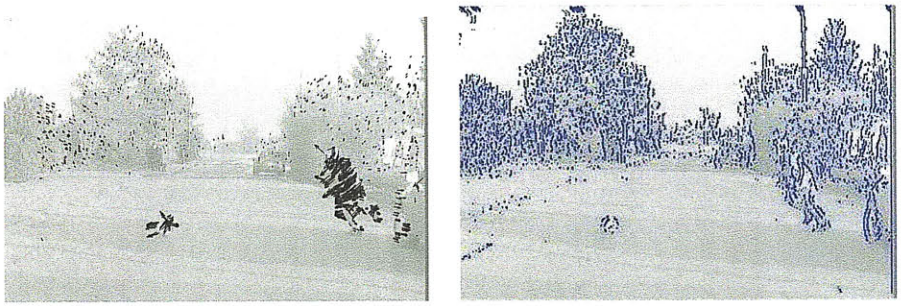


Figure 3-5: Distance-range-based segmentation results subject to background removal. Case 1: Removed background: Disparity 0-5. Case 2: Removed background: Disparity 0-9. (a) Left/Right matching results at disparity range 17-19 (from Figure (3-2) for Figure (3-1)(a)(b). (b) Morphological operation results on (a). (c) Edge-length-based filtering results on (b). (d) Segmented regions drawn on Figure (3-1)(a).



(a) For target edge layer (b) For full edge map

Figure 3-6: Comparison of motion vectors for distance-based target edge layer (after background removal) (a) and for the full edge map (b)

3.3.1 Similarity and saliency of motion vectors for target objects in distance-based edge layers

In order to expand shrunk segmentations, we can take advantage of the similarity and saliency of motion vectors for target objects in distance-based edge layers to improve the performance of our distance-based target detection. This is a special feature for our separated distance-based edge layers.

After separating distance-based target edge layers from the original edge maps, we can compute motion vectors for all the chosen edge pixels, and plot them on the top of the original video images. We observe that motion vectors for the same target objects are also similar in their magnitude though their directions may vary. At the same time, motion vectors for target object and non-rigid background noise can be clearly differentiated.

For example, for two continuous video frames in Figure (3-1) (a-d), we first obtain Figure (3-3) (b) that shows target edge layer with disparity range 17-19. Then we plot the motion vectors for pixels in that target edge layer in Figure (3-6) (a). The motion vectors plotted on top of the original images provide the information that both the boy and the ball are moving toward the left side of images. Motion vectors for the same target objects are quite similar in magnitude and have significantly larger magnitude than the motion vectors for pixels in non-rigid background noise.

When we talk about motion vectors, we do not have to assume that the objects are

moving. The magnitude of motion vectors for the same static targets should be similar. Due to the impact of different distance ranges, motion vectors for close static target objects are supposed to be much larger than for static background objects with large distances. Motion vectors for static background objects that are far away are supposed to be small. This observation holds true for most rigid objects and pedestrians.

However, due to noisy features for motion vectors, especially for non-rigid background objects, the motion vectors for some non-rigid background pixels might also have large magnitudes. If we have many pixels from non-rigid background objects, when plotting together motion vectors for target objects and these background pixels, it would be hard to differentiate the motion vectors for targets and background objects with large motion vectors. In contrast, for pixels in the original edge maps shown in Figure (3-1)(e), we also compute their motion vectors and plot them in the original images in Figure (3-6)(b). Motion vectors in Figures (3-6)(a) and (b) are plotted in different adaptive scale for visualization effect. We notice that the differences between motion vectors for target objects and for noise are not clear and it is hard to distinguish target objects and surrounding background pixels based on motion vectors. Because of the noise introduced in computing motion vectors for background pixels, it is difficult to accurately segment targets based only on motion vectors.

The differences between Figure (3-6)(a) and (b) shows the advantages of our distance-based edge layer separation. In our separated edge layers, motion vectors for targets and objects have different magnitudes while the motion vectors for the same objects are similar in its magnitude. Actually the motion vectors for static objects in the same distance ranges are also similar. But we did not take advantage of this observation.

3.3.2 Expansion of segmentation region based on motion similarity

Since background removal process might eliminate possible target pixels when cleaning up the distance-based target edge layers, the initial target segmentation in the cleaned target edge layers might provide smaller target regions than their original sizes. But these initial segmentation regions in Figure (3-5) containing edge pixels of the same objects can be used

as initial object segmentation seeds from which segmentation boundary boxes will expand based on the similarity of motion vectors for the same objects.

When enlarging segmentation regions, we decide whether we accept new edge pixels in the growing boxes by comparing the similarity of motion vectors for segmentation seed boxes and for these new edge pixels in the neighborhood. The magnitudes of motion vectors for new candidate edge pixels are compared with the average magnitude of motion vectors for all edge pixels in segmentation seed boxes. If the difference is small, the candidate points are absorbed into the bounding boxes. Bounding boxes will keep growing as long as they still absorb new pixels. When bounding boxes reject new pixels, the length of the reject gap along the directions (up, down, left, and right) will be recorded. If the gap length reaches a threshold, the bounding box stops growing in that direction. After such region growing, we regain target edge pixels lost in background removal (in Section 3.2.4) without introducing too much background noise.

Given two possible initial segmentation results for case (I) and case (II) in Figure (3-5)(d1)(d2), the results of expanding operation for case (I) and case (II) are respectively shown in the left and the right columns of Figure (3-7). Results for both cases capture the accurate sizes of the small round ball and the whole body of the running boy, which is not rigid. The final results for case (I) and case (II) in Figure (3-7)(c1)(c2) show similar accurate bounding regions in spite of different initial segmentation seed regions in Figure (3-5)(d1)(d2), which indicates our segmentation results are not sensitive to the choice of background disparity ranges.

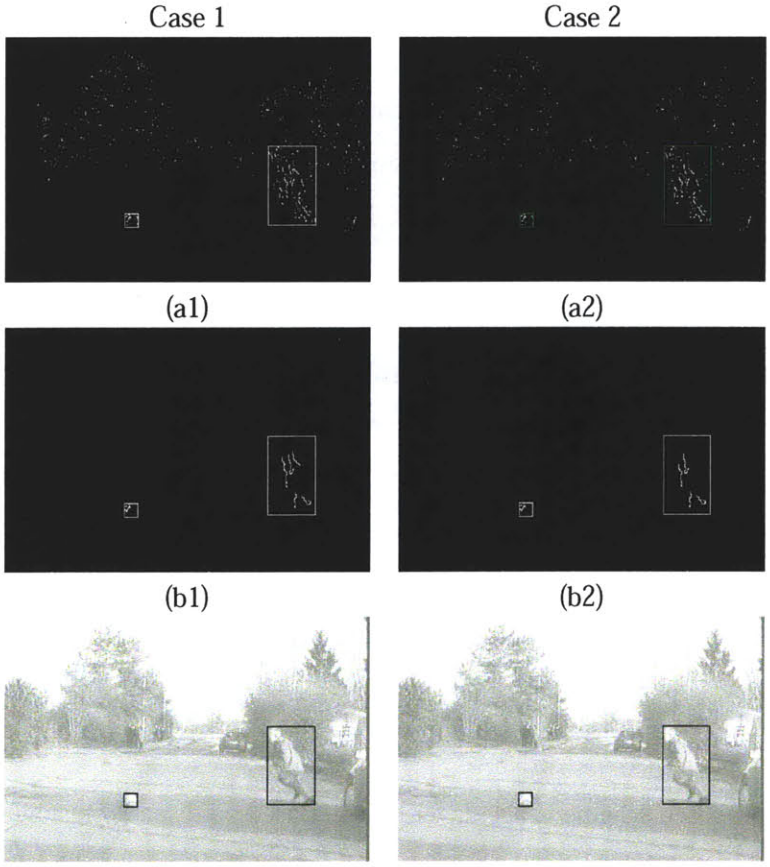


Figure 3-7: Motion-based segment expansion results for cleaned target edge layers that remove background pixels from target edge layers. (a) Bounding region expansion based on aggregating pixels of similar motions in cleaned target edge layers. (b) Segmented regions drawn on Figure (3-5)(c). (c) Segmented regions drawn on Figure (3-1)(a). Case 1 and case 2 respectively correspond to two definitions of background layers. Case 1: background layer with disparity 0-5. Case 2: background layer with disparity 0-9.

3.4 Discussion: the principle of layer-based and fusion-based algorithms

For sensor fusion methods, in general, data from all sensors are of similar characteristics and are sent to the final processing algorithms depending on mathematical methods, for example, Kalman filtering, Fuzzy logic, Dempster-Shafer Evidential Reasoning. In the process, the relationship among information from different sensors is typically ignored. We have proposed a feature-level fusion scheme which instead communicates information among sensors ahead of the final processing period. Special algorithms are developed to take advantage of additional distance information from another process for better interpretation of the environment. Distance range information is introduced into the segmentation process for both target object detection and background noise removal. Distance range adds an extra constraint during correspondence matching. Motion information is introduced into the first motion-based correspondence matching and the motion-based target region expansion.

Our proposed scheme is also layer-based. Based on disparity histogram derived from binocular stereo images, we first define target disparity ranges and background disparity ranges. We then apply our algorithm of distance-based edge-layer separation to split the original full edge map into several edge maps of different distance ranges, including background layers. In the separated edge layers at given distance ranges, possible background noise can be further removed to create clean edge layers in which target boundaries can be detected. Sometimes, target boundaries might be smaller than their original sizes due to the loss of target boundary pixels during background removal procedure. We compensate for the information loss by applying motion-based target boundary enlargement to ensure or even enhance the performance of segmentation process. Figure (3-8) summarizes the flow chart of our segmentation algorithm.

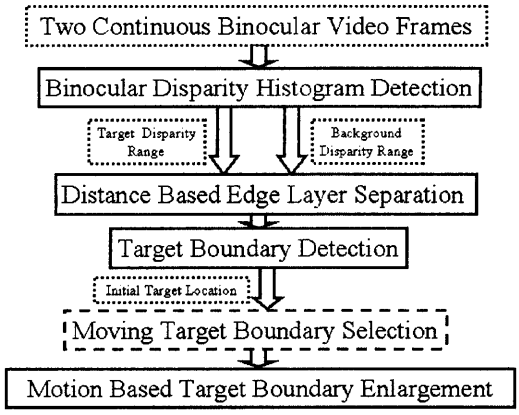


Figure 3-8: Flow chart for segmentation algorithm

3.4.1 Fusion of depth and motion information

In our fusion scheme, motion stereo information is incorporated as an additional fusion dimension in several tasks which takes advantage of the similarity of the objects' features: pixel depth and motion vector.

To compute binocular disparity histogram, the first correspondence matching algorithm specifically takes advantage of the similarities of target motion and disparity information between binocular stereo images for the same targets. The proposed motion-based target depth detection algorithm improves depth detection accuracy. High reliability could be achieved if the detected target depth information can be combined with additional radar information.

The motion information also helps to improve accuracy and reliability of segmentation in the presence of heavy background noise. To expand interested target sizes, the algorithm of motion-based expansion absorbs target neighborhood pixels with similar motion in separated distance-based edge layers, which implicitly assumes the similarities of depth and motion for pixels for one target.

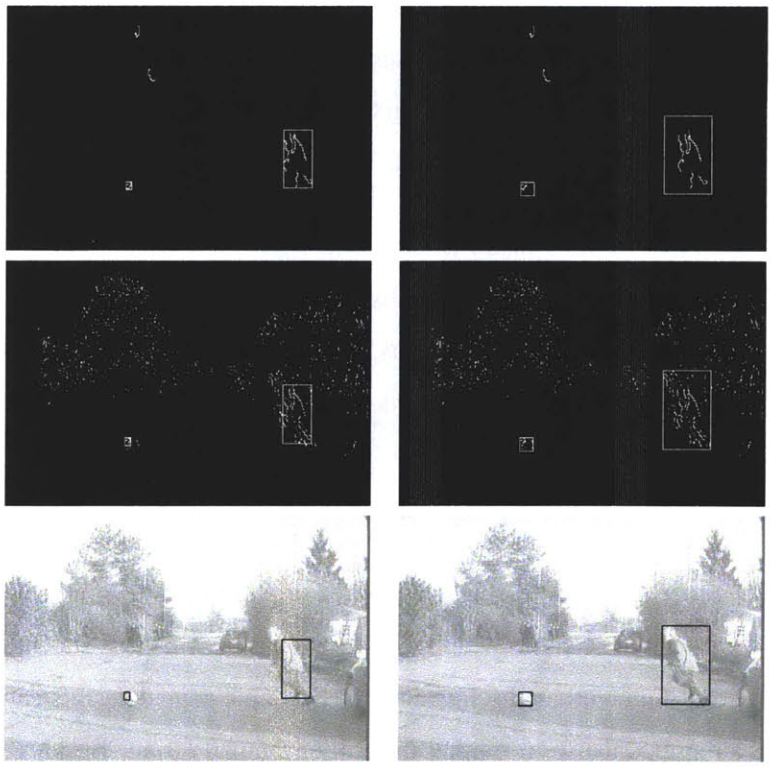
Finally, the additional motion information helps to identify static objects which will be discussed in Section 3.4.2.

3.4.2 Static-block elimination and moving-target detection

Segmentation algorithm based on both depth and motion constraints can improve the reliability and robustness of differentiating and locating both static and moving targets in the presence of noise.

If we want to detect moving targets, we can apply the motion constraint at an early stage to eliminate static blocks. For multiple segmentation regions in Figure (3-4), we calculate average magnitude of motion vectors for edge pixels in each segmentation region. If the average magnitude of motion vectors for one segmentation region is much smaller compared to motion vectors for other regions in the edge layers at given distance ranges, the region might only contain static objects or noise in which we are not interested and thus can be eliminated. the region might be from static objects or noise in which we are not interested and thus can be eliminated. The results after eliminating static blocks are shown in Figure (3-9)(a). It can be seen that two false boxes from the background regions are removed. Thereafter, we can apply the motion-based expansion to aggregate pixels of similar motions and enlarge small bounding boxes to their real sizes. The result of final segmentation results are shown in Figure (3-9)(b), which fully captures two moving objects. Motion-based region expansion can be directly applied to the original distance-based target edge layers shown in Figure (3-3)(b), and there is no need to apply background removal to such layers. The results are the same whether we apply background removal or not since we already remove the error blocks based on the magnitude of their motion vectors. The operation is shown as "moving target boundary selection" in Figure (3-8). It is presented in a dashed box since this part is only for detecting moving objects. Under general situations when we do not know anything about the movements of the targets, we need to apply background removal to reduce possible false alarm due to background noise.

Many segmentation algorithms also locate moving objects based on their salient motion vectors. However, as shown in Figure (3-6), the error of detecting optical flow makes it difficult to identify the accurate boundaries of moving objects, especially in uniform regions or in the presence of heavy background noise, thus capturing only part of a moving object instead of the whole target region.



(a) (b)

Figure 3-9: Segmentation results based on static box removal in target edge layers and motion region expansion. (a) The result of static block removal for Figure (3-4). (b) The result of motion-based region expansion for seed regions from (a). Top row: Segmented regions, the results of static block removal and motion-based expansion, are plotted on Figure (3-4)(c), the results of Morphological operation and edge-length-based filtering. Middle row: Segmented regions are plotted on Figure (3-4)(a), the original target edge layers. Bottom row: Segmented regions are plotted on Figure (3-1)(a), the original image.

3.4.3 Ability to detect multiple objects of different shape

The distance-based segmentation algorithm in Chapter 2 assumes that the longest edges are located at objects' boundary. Since boundary edge pixels for vehicles can be easily connected into long edge lines, the distance-based segmentation algorithm works well in highway driving scenarios. The algorithms do not assume symmetry of detected objects. For example, as shown in Figure (2-14) in Chapter 2, even if the car is not in a position that leads to a symmetric image, our algorithm still captures the vehicle accurately.

In this chapter, we investigate whether the segmentation algorithm can achieve ideal results for other rigid objects, such as balls, whose edge pixels cannot be easily connected into long edge lines, and non-rigid objects, such as pedestrians. Because non-rigid objects can change shapes, it is normally hard to detect the whole region of non-rigid objects based only on motion.

Furthermore, it is usually hard to accurately segment objects of different sizes. It might happen that either the smallest object is ignored or part of a large object is ignored. Therefore, we consider the scenario where a child is chasing a ball as shown in Figure (3-1), which can occur in the urban driving situations. Simple edge detectors cannot reveal ideal edge lines along boundaries for either the ball or the pedestrian. Besides, it is not easy to differentiate objects with different sizes based only on edge-length as we did before.

Our proposed depth/motion-based segmentation can effectively segment multiple objects of different sizes and objects without long boundary edges, including non-rigid targets, such as pedestrians and rigid targets, such as balls. The distance-based background detection helps to remove background pixels and prevents random background pixels from clustering and connecting into long edge lines. Thus the seed edge lines of interested targets, especially short lines in small targets, can still be captured in their distance-based edge layers. With the help of motion-based expansion, object with various sizes can finally be determined accurately. The algorithms accurately capture the region of the small ball, and also capture the whole body of the boy whose posture changes in two consecutive frames.

3.4.4 The features of our fusion algorithms

We have proposed a new depth-based target detection method based on fusing target depth information and binocular stereo images. Our sensor fusion scheme allows information communication among sensors before final processing to enhance the processing ability of each sensor based on the close relationship between information from different sensors. Sensors using additional information can offer better information than individual isolated sensors. Our sensor fusion system instead has the two following characteristics:

Task Oriented We focus on task requirements for 3D segmentation, and come up with a sensor fusion scheme on how to set up sensors, how to associate and fuse data among various sensors. The sensor set-up and fusion method are tailored to the given task. Task-related information from one sensor (coarse depth detection algorithm) is used to tailor processing of another sensor (segmentation) in fusion systems and to improve the reliability and accuracy of the informed segmentation algorithm.

Object Oriented Our method focuses on descriptive information, the distance and location, on our interest targets, instead of the accurate 3D reconstruction for our targets or information for other frame pixels. The original multi-target segmentation task is decomposed into several segmentation tasks oriented around single-target in individual depth-based target feature layers. Such scheme greatly decreases computational loads.

3.5 Summary

Our proposed fusion-based layer-based algorithm has the following characteristics:

1. *Ability to deal with heavy background noise*

   When background is very noisy (which is usually the case), the concept of fusion-based layer-based also helps to eliminate background edge pixels and enlarge the segmentation region to compensate the lost boundary pixels. Our proposed algorithm in Figure (3-8) significantly reduce the influence of background noise.

2. *Robustness to initial parameters*

   We use the same initial parameters to obtain both results in Figure (2-12) and Figure (2-14) in Chapter 2, i.e., the same parameters to locate objects at different distance ranges and in different video frames. The robustness to initial parameters is important in intelligent transportation systems. Our algorithm does not need accurate distance range information either. To detect vehicles in Figure (2-12), we could, for example, also use disparity ranges, $23-25$ or $22-25$, to obtain the same result. It means that we do not require high performance radar in order to detect corresponding vehicles accurately, which offers us much flexibility. If the resolution of distance ranges from low performance radars is not satisfying, the corresponding segmented images can also help to improve the distance estimation.

3. *Ability to work in fast changing environment.*

   Our algorithms can be implemented in parallel in hardware to increase the computational speed. The light computational load makes real-time implementation possible in practical situations. So far, our focus is to improve the performance and robustness in static image segmentation and environment interpretation while maintaining low computational load. The satisfying accuracy of static image segmentation offers a good basis for further information tracking in video sequences. The method does not assume the appearance of same objects with similar shape and sizes in several consecutive video frames. So it can be applied to situation when the dynamic scene changes fast. Therefore our algorithms can quickly provide correct segmentation results without any initialization time.

4. *Potential to solve occlusion problems*

   Partial occlusion of one object by another in road scenes adds difficulties in segmentation. Our distance-based segmentation algorithm would be effective in this situation because fusion of distance constraints and binocular correspondence constraints helps to differentiate one object from occluded objects since these are in different distance ranges.

Special thanks to Dr.Franke Ume and Dr.Stefan Heinrich for providing the video frames in Figure (3-1)(a)-(d), courtesy of DaimlerChrysler Research and Technology.

Chapter 4

Fusion-based Layer-based Pedestrian Detection based on infrared cameras

In the past two chapters, we have applied the concept of fusion-based and layer-based methodology to the 3D segmentation of obstacles of interest in visible images for day time situations. In this chapter, we will apply the principle of fusion-based and layer-based processing to detect pedestrians in infrared images for night time driving situations.

Automatic obstacle detection in both day time and night time is an integral component of the dynamic interpretation of the environment. Indeed, automatic detection of pedestrians at night has attracted more and more attention. Night-time driving is more dangerous than day-time driving — particularly for senior drivers. Because depth perception, color recognition, and peripheral vision are all impaired after sundown, three to four times as many traffic deaths occur during night-time than day-time [31]. Around eighteen percent of pedestrian fatalities happen between 12am to 6am. While there are only 28% total traffic at night, the rate of serious accidents at night reaches as high as 55%. Furthermore, people's visual capabilities deteriorate substantially as they age, as shown in Figure (4-1), which compares the visual ability of a driver of age 60 with those of a driver of age 20. A 50-year-old driver needs twice as much light to see well as a 30-year-old[31].

Because pedestrians stand out as bright regions for infrared images, current night vision systems used for infrared cameras provide a visual aid projected on a heads-up display to enhance safety. However drivers have to switch attention from driving to read the displayed

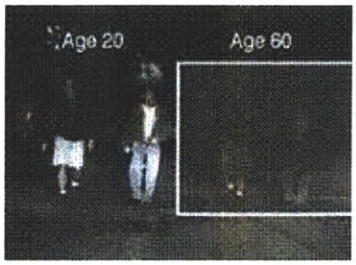


Figure 4-1: Vision Degradation for Seniors. (Image Source: MIT Age Lab.)

information and to make judgment about road situations. It is likely for drivers to ignore potential dangers or fail to respond in time. To improve the safety of night driving in the long run, automatic environment understanding and warning has drawn increasing attention. Automatic pedestrian detection based on infrared images is envisioned so that drivers can respond promptly without being distracted by added gadgetry. Nevertheless, pedestrian detection in infrared images is by no means trivial — many of the known difficulties carry over from visible images, such as image variability occasioned by pedestrians being in different poses. Furthermore, pedestrian detection is difficult because of poor infrared image quality (low resolution, low contrast, few distinguishable feature points, little texture information, etc.) and misleading signals.

Compared to the vast research on pedestrian detection based on visible images as summarized in [39][40][41], work on infrared-based pedestrian-detection research has only begun a few years ago.

We systematically compared different properties of visible and infrared images and noted several unique features of infrared-based pedestrian detection. We investigated the statistical properties of these features and now propose a novel layer-based fusion-based shape-independent pedestrian-detection scheme including automatic pedestrian image size estimation and multi-dimensional shape-independent classification. In Section 4.1, we first discuss how we evaluate detection performance and define performance index, then review previous work, analyze challenges associated with automatic pedestrian detection using infrared images, and propose our fusion-based and layer-based design principles. In Section 4.2, we describe our fusion-based layer-based automatic pedestrian segmentation. In

Section 4.3, we discuss our fusion-based shape-independent multiple dimensional classification. In Section 4.4, we analyze three infrared image sequences using our proposed scheme. Finally we will summarize and discuss the advantages of our design as well as differences from conventional methods in Section 4.5.

4.1 Introduction

Pedestrian detection [35] [36] includes two phases: *segmentation* locates multiple regions of interest (ROIs) from infrared images, and *classification* identifies pedestrians in the ROIs. In this section, we first define a Performance Index to evaluate both segmentation and classification processes, review the challenges and previous works for both processes, and then introduce the general principles for our layer-based fusion-based scheme.

4.1.1 Performance Index

To evaluate segmentation performance, we define two new performance indices: *segmentation side accuracy* and *segmentation side efficiency* as shown in Figure (4-2)(a). **Segmentation side accuracy** is defined as the square root of the ratio of the detected pedestrian region area, $S_{overlap}$, over the entire pedestrian area, $S_{pedestrian}$, which indicates how much of the pedestrian region is captured. If, for example, the segmentation side accuracy is 50%, then the width and height of the detected region might be only half of the actual pedestrian's width and height. **Segmentation side efficiency** is defined as the square root of the ratio of the detected pedestrian area, $S_{overlap}$, over the entire ROI area, S_{ROI}, which indicates how efficient the selection of the ROI region is. If, for example, the segmentation side efficiency is 50%, then the width and height of the detected pedestrian region is only half of the actual ROI's width and height.

Both performance measures lie in the range $[0,\ 1]$. The best segmentation performance is achieved when both measures are 1, which means that the ROI and actual pedestrian region overlaps completely. High segmentation accuracy with low efficiency indicates that, while most pedestrian regions are detected, this is at the cost of unnecessarily large ROI areas. Conversely, low segmentation accuracy with high efficiency indicates that the ROIs

capture only a small portion of the pedestrians, though most ROI regions are within pedestrian regions.

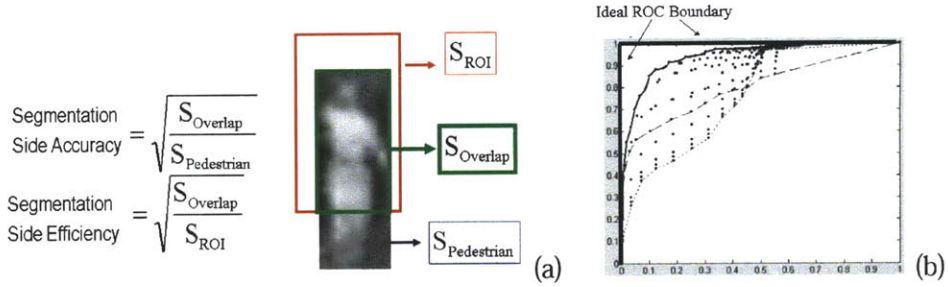


Figure 4-2: Segmentation/Classification Performance Index Definition. (a): Segmentation side accuracy/efficiency definition. (b): ROC boundary/curve definition for multi-dimensional-feature-based classification: false_alarm_rate (X axis) vs. detection_rate (Y axis). Different points correspond to multi-dimensional-feature-based classification results using different multi-dimensional thresholds. Solid curve is ROC boundary, the upper/left boundary of all classification performance points. Dashed and dotted curves are ROC curves for 1D-feature-based classification.

To evaluate the performance for multi-dimensional-feature-based classification, for different choices of classification thresholds, we plot corresponding false-alarm/detection rates as points in a 2D performance space (X axis: false_alarm_rate/Y axis: detection_rate) as shown in Figure (4-2)(b). We can choose 1D histogram-based or 1D inertia-based classification, 2D histogram/inertia-based classification, or 3D histogram/inertia/contrast-based classification with details in section 4.3.4. Performance points for multi-dimensional classification are scattered around as shown in Figure (4-2)(b). Classification performance improves when the performance point moves toward the upper left area. However, if one point is in the upper right direction of another point, we cannot easily compare their performance. Thus, the upper/left boundary of classification performance points, as shown in solid curves in Figure (4-2)(b), is used to demonstrate the classification ability of an algorithm, and is called the **receiver operating characteristics (ROC) boundary** in this chapter. The ROC boundary for 1D-based classification degrades to the conventional **ROC curve** as shown by the dotted and dashed curves in Figure (4-2)(b). All ROC curves/boundaries include two points, $(0,0)$ and $(100\%, 100\%)$, which can be achieved by rejecting all or accepting all. Obviously the best classification performance is in the upper/left corner —with a 100% de-

tection rate and 0% false-alarm rate. The ideal ROC curve/boundary is a vertically flipped "L" shape as shown in Figure (4-2)(b).

In this chapter, detection/false-alarm rates on the ROC curves are shown for ROIs rather than for image frames. We do not plot any ROC curve for frame detection/false-alarm rates since such curve also depends on segmentation performance and consequently does not necessarily pass through the $(100\%, 100\%)$ detection/false-alarm rate point, which is different from typical ROC curves.

However, for the purpose of performance comparison with other published results, we do calculate frame detection/false-alarm rates based on conventional definition in several cases. To calculate the number of detected frames, we count frames in which all pedestrians are detected, and empty frames (with no pedestrian) in which there is no false alarm.

4.1.2 Challenges and Previous works on Pedestrian Detection with Infrared Images

Pedestrian detection using infrared images has its own advantages as well as disadvantages [39][40][41][42] when compared with detection using visible images. In general, pedestrians emit more heat than static background objects, such as trees, roads, etc. In far-infrared images, pedestrian brightness tends to be less affected by lighting, color, texture, and shadow information than it is in visible images, and is generally also somewhat brighter than the background. There exist intensity similarity among arbitrary pedestrians. Thus, pedestrians tend to stand out more against the background in infrared images than in visible images.

However, infrared image intensities depend not only on object temperature but also on object surface properties (emissivity, reflectivity, and transmissivity), surface orientation, and, wavelength. Infrared images have particular characteristics that lead to detection difficulties. First, non-pedestrian objects, such as animals, vehicles, transformers, electric boxes, roads, construction areas, light poles, etc., produce additional bright areas in infrared images, especially in summer. These additional *sources of image clutter* make it impossible to reliably detect pedestrians based simply on their brightness. Secondly, the image

intensities of a given object are not uniform. Pedestrian orientation, clothes, accessories (such as backpacks), etc., all have an impact on the observed image intensity patterns. Body-trunk areas are generally darker than head and hand areas, especially when pedestrians wear heavy coats or carry backpacks. The upper parts of light poles appear brighter than the lower parts because of contrast phenomena in typical far-infrared cameras. *Non-homogeneous optical properties* add to detection difficulties. Thirdly, most infrared image intensities have a smaller intensity range than do comparable visible images. This leads to *low image quality*: blur, poor resolution and clarity, low foreground/background contrast, fewer feature points and less texture information, etc.

Thus, current infrared-based pedestrian-detection research [37][39][40][41][42][43] is still limited. Both segmentation accuracy and classification reliability of early night vision research need to be significantly improved for it to be of practical use [41][40]. For example, in winter, to have a false-alarm rate around 2.63% [41], the detection rate has to be limited to only 35%. In summer, to have a 75% to 90% detection rate, the false-alarm rate has to be raised to 100% [40] as shown in Figure (4-3)(a). Below we will discuss inherent difficulties in two phases and then review related previous work.

Challenges and Reviews for ROI Segmentation

It is difficult to segment pedestrians in real-world video images captured by cameras mounted on moving vehicles. Pedestrians have a variety of poses, sizes, and appearances, and the background is changing rapidly as the cameras move through the environment. Many conventional fast segmentation algorithms have been developed for stationary cameras, such as *background subtraction*[44], *motion calculation*, and *tracking*. These methods assume similar backgrounds or feature points, and need some initialization time.

Conventionally, segmentation based on depth information is more straightforward than other methods, and multi-scale brute force searching can be avoided. However, binocular infrared camera setup is not widely used in most night vision research, except by Tsuji (2001) [37]. There might be reliability concerns because of the properties of infrared images discussed above and the nodding movement of cameras on vehicles [42]. Besides, high cost of far infrared camera might be another concern. Thus it is expected that, for

now, pedestrian detection for intelligent vehicles will rely only on a single static image instead of multiple-image-based (motion-based) algorithms.

If detailed pedestrian contours can be extracted, pedestrians can be identified by using *contour-based shape model* [33][34][35], such as pedestrian shapes hierarchy[35] or human walking model[34]. Besides, *human component features* [45][35][46][47][48], such as skin hue, eyes, faces, etc., also help when segmenting pedestrians in visible images. However, the above well-known fast segmentation features are not applicable to far-infrared images because of the images' unique properties.

It is also hard to segment pedestrians by grouping bright spots belonging to pedestrians only based on their pixel intensities. Using one fixed brightness threshold, for example, leads to several separated bright spots at both pedestrian regions and other noise sources in background regions which are highly sensitive to the choice of brightness thresholds. When template-shape-based multi-scale brute-force searching is introduced, as happens in some night vision algorithms (as shown in Figure (4-3)(b)), segmentation ROI outputs are all candidate-pedestrian patches of different sizes and aspect ratios, at multiple initial locations.

Typically, several different pedestrian templates have to be used in order to deal with a range of poses. The total number of ROIs for completely blind multi-scale brute force searching is as follows:

$$\begin{aligned} n_{\text{ROI}} &= \sum_{i=1}^{n_{\text{scale}}} n^{i}_{\text{center-pos}} \cdot n_{\text{template}} \\ &\propto n_{\text{row}} \cdot n_{\text{column}} \cdot n_{\text{scale}} \cdot n_{\text{template}} \end{aligned} \tag{4.1}$$

where n_{scale} is the number of scales in estimating pedestrian sizes, n_{template} is the number of templates for pose matching, $n^{i}_{\text{center-pos}}$ is the number of initial ROI center positions that must be tried when testing at the ith scale. The variable $n^{i}_{\text{center-pos}}$ is proportional to the image size($n_{\text{row}} n_{\text{column}}$). The large search space for blind searching is a severe limitation.

Different segmentation algorithms take advantage of different features to decrease n_{ROI} and to expedite the searching process. To decrease $n_{\text{center-pos}}$, [41] searches infrared im-

ages for bright and round regions as potential pedestrian heads. In [39] hot ROIs with specific size and aspect ratio are searched based on the assumed *symmetry property* of pedestrians and their brightness[1]. To decrease n_{scale}, [41] and [39] assume flat roads so that pedestrians' distance can be estimated based on pedestrians' vertical positions in images. In [41], road surface boundaries are first detected in order to estimate pedestrian size and height and impossible pedestrian size/position combinations are later removed. In [39], infrared cameras are calibrated to build correspondences between image lines and distances in the 3D world for pedestrian size estimation. [39] calibrates infrared cameras to build correspondences between image lines and distances in the 3D world for pedestrian size estimation.

No assumptions are made in [40] where one searches for only three pedestrian sizes in a multi-scale brute force approach and its segmentation accuracy is limited compared with [39][41]. For real-world applications, segmentation algorithms need to further improve speed and accuracy and make fewer assumptions on the driving environment.

Challenges and Review for Classification

In far-infrared images, pedestrians yield widely varying image patterns because of variations in pedestrian pose and the imaging complexities mentioned before. Among multiple candidate image regions, differentiating pedestrians from non-pedestrian regions is difficult. Typically the decision is made based on the similarity between ROI regions and multiple pedestrian templates with various poses and appearances. Similarity can be computed either directly or indirectly.

Typical direct methods compare image intensity pixel-by-pixel and compute the *image-intensity difference* between two patches, i.e., the Frobenius norm of image pixel intensity differences. The classification methods depend heavily on shape matching. As a result they are sensitive to segmentation errors and variations in pedestrian poses. A template probabilistic model is defined in [40] to encode the shape information of pedestrians and the variations that the shape can undergo by describing the possibility of foreground and background at each pixel based on training data. In [39] pedestrians are identified through matching candidates with a simple model that encodes morphological characteristics of a

pedestrian. The shape-dependent filter removes candidates that do not present a human shape or do not have regions as bright as expected for a pedestrian.

For indirect similarity comparison, shape-dependent pedestrian-intensity-arrays are used to train classifiers to capture the similarity between pedestrian training samples and ROIs, for example, Support Vector Machine (SVM) [36] [41], Neural Network [47][38], posteriori detection (including polynomial classifiers, multi-layer perceptrons, and radial-basis functions), etc. [41] proposed SVM classifiers for three types of pedestrians for infrared images.

These brightness-similarity-comparison-based classification methods are shape-dependent, and might miss pedestrians with unusual poses even if multiple pedestrian-pose templates or training samples are used. In summary, speed, reliability, and performance robustness to pose-changes and segmentation errors are serious concerns for real-world night vision systems.

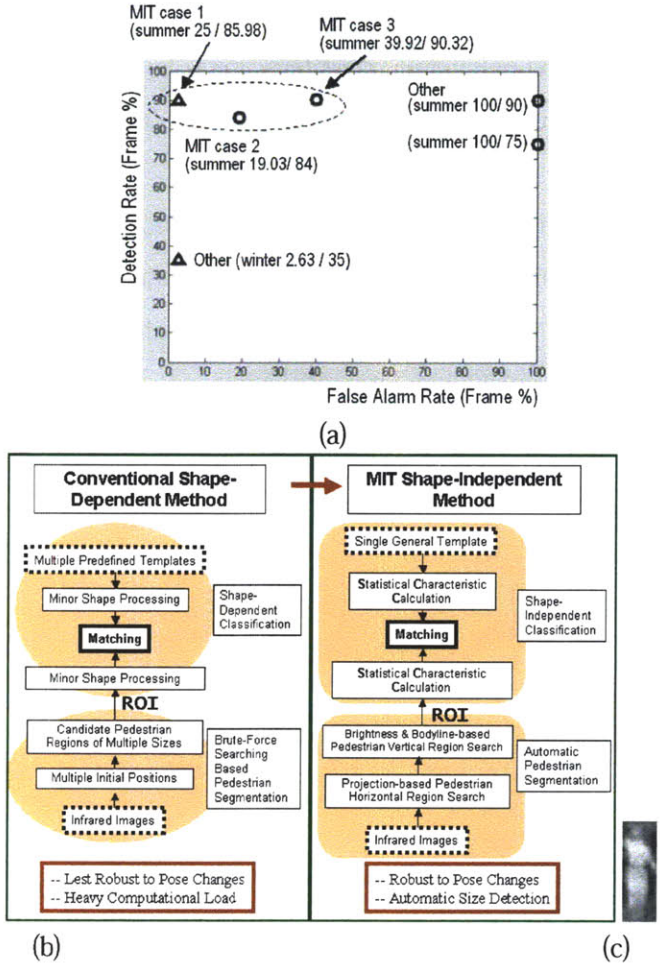


Figure 4-3: Algorithm and Performance Comparison for Different Pedestrian-Detection Methods. (a): Detection Performance Comparison. (b): Detection Algorithm Comparison. (c): Default Pedestrian Template for MIT Shape-Independent Method. Image size: 58×21.

4.1.3 The Methodology and Principle for Fusion-based Layer-based Shape-Independent Pedestrian Detection

Because of the above-mentioned difficulties involved in shape-dependent and/or brute-force-searching-based methods, the performance of present pedestrian-detection systems is limited. As shown in Figure (4-3)(a), in the summer, in order to reach 75% and 90% of detection rate, the false alarm rate reaches 100%. In the winter, with low false alarm rate 2.63%, the detection rate is around 35%. To improve the performance, we propose a *layer-based fusion-based shape-independent* automatic pedestrian detection method with straightforward implementation. Our layer-based segmentation algorithm first horizontally divides a whole image into several vertical stripes (from top to bottom in the images) that might contain candidate pedestrians through *projection-based* horizontal segmentation, and then searches for pedestrians vertically within these stripes through *brightness/bodyline-based* vertical segmentation.

Our fusion-based classification algorithm defines multi-dimensional histogram-, inertia-, and contrast-based classification features. The novel classification features are shape-independent, complementary to one another, and characterize the statistical similarities of image patches containing pedestrians with different poses. Such feature vectors can also capture the statistical differences between pedestrian of different poses and non-pedestrian regions in infrared images.

Figure (4-3)(b) presents the major differences between our layer-based shape-independent methods and conventional shape-dependent methods. First, our segmentation algorithm does not make constraining assumptions about the background, for example, flat roads; thus our results are very general. The "horizontal-first, vertical-second" layer-based segmentation algorithm avoids brute force multi-scale searching by automatically determining horizontal locations or horizontal bounding regions for candidate pedestrians. Secondly, our multi-dimensional classification needs only one generic pedestrian template as shown in Figure (4-3)(c) with size 58×21 (details in Section 4.3). The similarity comparison is based on the unique statistical properties of far-infrared images that we discovered through investigating the differences between visible and infrared images [42], which is

shape-independent. Thirdly, the whole detection scheme focuses on improving combined segmentation/classification systems and balances the complexity and performance of two subsystems instead of maximizing one process while sacrificing the other. This is because accurate segmentation can ease the classification task and robust classification can tolerate segmentation errors.

The classification performance comparison is shown in Figure (4-3)(a). For pedestrian detection in winter, we achieve a higher detection rate when we set the false-alarm rate to be similar to other available published results. For summer, we achieve a lower false-alarm rate when we set the detection rate to be similar to other available published results. The technical details about our segmentation and classification algorithms are discussed in Sections 4.2 and 4.3 respectively.

4.2 Automatic Pedestrian Segmentation

As mentioned in 4.1.2, conventional template-shape-based segmentation involves searching with computational load that is proportional to image areas. We invented a new layered-based *horizontal-first, vertical-second* segmentation scheme involving only 1D searching in vertical direction with computational load that is proportional to image sides. The method first horizontally separates an entire image into several vertical stripes (from top to bottom in the images) which might contain candidate pedestrian regions, and then searches for pedestrian regions vertically within the corresponding image stripes. Thus search space and computational load are reduced significantly. In this section, we will respectively introduce our *horizontal segmentation* algorithm based on *bright-pixel-vertical-projection curves*, and *vertical segmentation* based on *brightness/bodylines*.

4.2.1 Horizontal Segmentation

In this subsection, we will first define the *bright-pixel-vertical-projection curve*, then explain how and why we can take advantage of the curve to split an image horizontally into several vertical stripes that help to determine pedestrians' horizontal locations or bounding regions.

Bright-pixel-vertical-projection Curves

For an infrared image, we define its **bright-pixel-vertical-projection curves** as the number of bright pixels in image columns versus their corresponding horizontal positions. To count *bright pixels,* an intensity threshold is used, which is adaptively defined as follows:

$$\text{Bright Pixel Threshold} = \max(\text{Image Intensity}) - \text{Intensity Margin} \tag{4.2}$$

where the variable *Intensity Margin* is a fixed constant for different video sequences. Typically the bright-pixel-vertical-projection curves can be divided into several *bumps* or *waves* with rising left curves and falling right curves, as well as flat regions with near zero height whose corresponding image stripes have no bright pixel, as shown in Figure (4-4)(b). Constant *intensity margin* used in Equation (4.2) is set to be large. Thus, the adaptive *bright pixel threshold* is small. This is to ensure that the image columns containing pedestrians have non-zero projection in the bright-pixel-vertical-projection curves.

The transitional locations of projection waves are insensitive to the choice of brightness thresholds or "intensity margin." Figure (4-4)(a) shows the variation of projection curves corresponding to two different brightness thresholds. Though the height and shape of the *waves* in the curves will change, the locations of transitional waves, steepness and relative height of transitional peak from background regions to pedestrian regions are not affected by different choices of brightness thresholds. We apply the same "intensity margin" for both winter and summer sequences.

Projection curves have the following special properties to help with pedestrian detection. Each pedestrian image will be captured in one image stripe corresponding to one wave in the projection curves. Such statements hold due to the following observations. First, for typical real-world far-infrared images, image columns passing through pedestrian regions tend to encounter many more bright pixels than neighbor columns passing through background regions. Secondly, columns passing pedestrians' head-areas might contain more bright pixels than their neighbor columns passing shoulder or limbs, which lead to the highest peak in bright-pixel-vertical-projections. Thirdly, the image stripe containing one

pedestrian is narrow, and the number of bright background pixels in each column can be treated as more or less constant.

Therefore, it is rare for one pedestrian image region to correspond to two different waves. The transition peaks in the projection curves between background regions and pedestrian regions are much higher and steeper than other waves in the projection curves. In most cases, the width of the pedestrian-image-region is equal to the width of the corresponding wave as shown in Figure (4-4)(a)(b), while the central horizontal locations for pedestrian regions normally correspond to wave peaks in the projection curves.

The above feature also holds when pedestrians occlude. Specifically, columns passing pedestrians' head-areas correspond to transitional peaks of for bright-pixel-vertical-projection curves. As shown in Figure (4-4)(a), for two most left pedestrians, one person partially occludes another. We still obtain two separate waves that can help us to differentiate two different pedestrians. If two people are too close to each other, we might treat both of them as one person. It would not lead to serious safety concerns as for applications' purposes.

The features of the defined curves are insensitive to the choice of brightness thresholds. As shown in Figure (4-4)(a), the horizontal locations and width of waves corresponding to pedestrians are robust to two different choices of brightness thresholds and the issues mentioned in Section 4.1.2.

Projection-based Horizontal Segmentation Algorithm

Based on the above properties of the bright-pixel-vertical-projection curves, we can segment an image horizontally into several image stripes (as shown in the bottom row of Figure (4-4)(b)), some of which contain individual pedestrians and roughly determine candidate pedestrians' horizontal locations or horizontal bounding regions. The procedure is as follows:

1. Use Equation (4.2) and choose an adaptive brightness threshold. Record the number of bright pixels in each column in the bright-pixel-vertical-projection curves.

2. Automatically search for the starting points of all rising curves (*wave-start points*)

and the ending points of all falling curves (*wave-end points*).

3. Separate the bright-pixel-vertical-projection curves into several waves by pairing *wave-start points* and *wave-end points*, and ignoring flat regions of zero height.

4. Record image stripes corresponding to these *waves*.

As mentioned in section 4.1.2, the hot objects in background regions in summer will produce many bright pixels in infrared images. Such background brightness "noise" leads to the following different properties of projection curves for winter and summer images, as shown in Figure (4-4)(a)(b). First, in summer, waves corresponding to pedestrians might not necessarily have higher peaks than background waves, and background "noise" may also produce additional individual high wave peaks in projection curves as shown in Figure (4-4)(b). For example, image regions containing light poles will bring in extra waves which appear narrow and high. These extra individual waves would add extra candidate locations for pedestrians that can be later removed during classification, but they would not affect segmentation results for real pedestrians. Secondly, under complicated urban driving scenarios in summer, as shown in Figure (4-4)(b), pedestrians and background brightness "noise" may be spatially proximate and their projection thus may merge into one wave, which is the case for the second pedestrian from the left in Figure (4-4)(d). For these cases, the additional brightness noise makes waves in projection curves wider than the actual pedestrian image width. Such situations would not happen for winter sequences (example: Seq1 shown in Figure (4-17)(a1)) and summer sequences in suburban areas (example: Seq2 in Figure (4-17)(b1)) with sparse foreground objects. In other words, pedestrian regions are less likely to be grouped with other "hot" foreground regions. However, in either cases, pedestrians will still be fully captured in individual horizontally separated stripes.

So far, we have presented a novel projection-based pedestrian pre-segmentation algorithm that horizontally separates infrared images into several image stripes that may contain pedestrians. In the next subsection, we will discuss how to search pedestrians' vertical locations in segmented image stripes.

4.2.2 Vertical Segmentation within Horizontally Segmented Image Stripes

Two vertical segmentation algorithms will be presented. The first is a *brightness-based* method (Section 4.2.2) that works best in winter and suburban situations where most segmented image stripes for pedestrians reflect the true width of pedestrian-image-regions. The second is a *bodyline-based* method (Section 4.2.2) for more complicated scenarios where the image stripes containing pedestrians might be wider than the pedestrian images' true width. These two methods provide complementary results that work best in different scenarios, and the results from both methods are input to the classification step to further improve reliability and accuracy.

Vertical Segmentation based on Brightness

After obtaining horizontally segmented image stripes from Section 4.2.1, the vertical positions of candidate pedestrian regions can be estimated by the highest and the lowest vertical locations of bright pixels within these stripes. This method is applicable when the estimate of the pedestrian region width is reasonably accurate. In this case, most brightness-based vertical segmentation results for both winter and summer data turn out correctly as shown in Figures (4-4)(c) and (d). Our classification algorithm has the ability to tolerate segmentation errors for pedestrian ROIs, such as the inclusion of extra background regions or conversely missed portions, as shown in the first and the fourth pedestrians from the left in Figure (4-4)(c). Non-pedestrian ROIs have bright pixels at the boundaries, which facilitates the inertia-based classification algorithm to be described later in Section 4.3.2. When segmentation stripes are much wider than the actual pedestrian image size, ROIs may be much larger than the true width, as occurs for the third pedestrian from the right in Figure (4-4)(d). A *bodyline-based* vertical segmentation algorithm (below) is proposed to improve segmentation performance for such difficult situations.

Vertical Segmentation based on Bodyline

In this method, we refine the pedestrian width estimation by detecting pedestrian regions' left and right boundary points within segmented image stripes. Thus we can further search

for pedestrians' vertical positions based on a geometric pedestrian-size model as described next.

For each row of image stripes, we define the portion of image rows within pedestrian regions as the **pedestrian-bodyline**, and define prominent feature points where image rows meet pedestrian boundaries as **pedestrian-bodyline terminals**. Figure (4-5)(a) presents one bodyline example in the waist area of a pedestrian image. Below we will describe in detail how to detect bodylines, and how to vertically segment pedestrians within image stripes.

Step 1 Pedestrian Horizontal Bodyline Detection.

Because of the features of infrared images, in each row within segmented image stripes, the *left* pedestrian-bodyline terminals are the points where image intensities change *from darkness to brightness* most rapidly. Similarly, at the *right* pedestrian-bodyline terminals, image intensities change *from brightness to darkness* most rapidly. To obtain pedestrian-bodyline terminals, we calculate intensity variation along the horizontal direction based on the modified Sobel method as below:

$$\begin{aligned} \Delta I(x,y) &= [I(x+1,y+1) - I(x-1,y+1) \\ &+ 2I(x+1,y) - 2I(x-1,y) \\ &+ I(x+1,y-1) - I(x-1,y-1)]/6 \end{aligned} \tag{4.3}$$

where (x,y) are pixel coordinates, $I(x,y)$ is image intensity, and $\Delta I(x,y)$ is **horizontal gradient**.

Within horizontal segmentation stripes, we first calculate horizontal-gradient for all pixels in each row, then we search in the left half of the row for a point with the *largest* horizontal-gradient as the candidate for the *left* bodyline-terminal points. We skip the row where horizontal-gradient for all pixels is smaller or equal to zero. Similarly, we determine the *right* bodyline terminal with the *most negative* horizontal-gradient in the right half of the row. Thus we obtain the two outmost boundaries and a bodyline for candidate pedestrians in each row within horizontal segmentation stripes.

For segmented image stripes shown in Figure (4-4)(b), Figure (4-5)(b) preserves all pixels within detected candidate bodylines, in which pedestrians stand out and the background pixels surrounding the pedestrian regions have been removed. It may happen that some boundary points belong to other "hot objects" next to the pedestrians, and we might not obtain a clear bodyline at every row of pedestrian regions. However, as long as we can obtain one bodyline, in the next step we can still estimate the candidate pedestrian's image location based on the bodyline information.

Step 2 Pedestrian Location Estimation based on Pedestrian-Bodyline Matching

In Figure (4-5)(a), we propose a geometric pedestrian-size model that defines one pedestrian's size and location based on the location and length of a waist-bodyline. The reason we use waist-bodylines is that the contrast between human waist areas and their local background neighborhoods tends to be robust to different poses of walking pedestrians. Horizontal waist-bodylines are more likely to be detected and are not easily missed under a variety of conditions. Using the size model, we can define multiple candidate pedestrian regions by assuming each detected bodyline to be the waist-bodyline of a pedestrian. Figure (4-5)(c) provides an example of bodyline-based pedestrian location estimation. A few estimated candidate pedestrian regions are marked.

Step 3 Histogram-based Bodyline/Pedestrian Searching

Among multiple candidate regions defined previously within a vertical image stripe, there is at most one actual pedestrian image region. Choosing one candidate pedestrian region is essentially a classification problem. We first use one histogram-based classification feature to search for the best candidate within each image stripe. After obtaining one candidate for each image stripe, we further determine whether it is an actual pedestrian image using multi-dimensional classification features explained in the next section. Details of the histogram-based feature and other classification features will be explained in Section 4.3. It is worth mentioning that we do not need to use a threshold in the searching process since we choose ROIs that are closest to our default pedestrian template (Figure (4-3)(c)) in histogram feature space.

For initial horizontally segmented image stripes in the bottom row of Figure (4-4)(b), the bodyline-based vertical segmentation result is shown in Figure (4-5)(d), which provides more accurate segmentation results than the brightness-based segmentation results shown in Figure (4-4)(d) where background noise causes segmentation errors.

As shown in Figure (4-3)(b), layer-based automatic pedestrian segmentation starts with projection-based horizontal segmentation. Within segmented image stripes, brightness-based vertical segmentation assumes that pedestrian pixels are brighter than the background pixels in the image stripes. The bodyline-based method assumes there exists clear brightness contrast between pedestrian image regions and their horizontal-neighbor-regions, and searches for the left-positive/right-negative edge-pairs with high horizontal-gradient in order to detect potential pedestrian bodylines and estimate candidate pedestrian positions. Both methods automatically estimate pedestrians' sizes and avoid multi-scale brute-force searching. The first method is straightforward and works reliably in suburban summer cases as well as winter cases. The second method works in complicated urban driving situations. Neither method needs to assume flat roads and both can work in a general driving situation. In real-world applications, both segmentation results will be fused in the classification step.

Conventional segmentation involves brute-force searching within an entire image based on multiple templates and produces multiple initial ROIs as in Equation (4.1). Instead, bodyline-based segmentation involves only searching among multiple bodylines within horizontally segmented image stripes and the number of produced initial ROIs is as follows:

$$n_{\text{bodyline ROI}} \approx n_{\text{image stripe}} n_{\text{bodyline}} \tag{4.4}$$

where $n_{image\ stripe}$ is the number of horizontally segmented image stripes and is usually less than 20 (much less than the number of image columns), and $n_{bodyline}$ is the largest number of bodylines in segmented image stripes and is much less than the number of image rows. Thus $n_{bodyline\ ROI}$ is significantly less than estimation n_{ROI} in Equation (4.1). The number of ROIs for brightness-based segmentation is equal to $n_{image\ stripe}$. In summary, our vertical

segmentation produces fewer candidate ROIs.

4.3 Classification

To recognize pedestrians, conventional classification is appearance-based and relies on the comparisons between ROIs and *multiple* templates. The meothod is shape-dependent, which is subject to segmentation errors and pose-changes as mentioned in Section 4.1.2. For robustness and reliability, we propose innovative classification that is based on comparing the similarity between multi-dimensional shape-independent feature vectors for ROIs and feature vectors for a single generic pedestrian template. All ROIs are normalized by the area of our template so that our classification features are size-independent. In this section, we will describe histogram-, inertia-, and contrast-based classification features individually, propose our multi-dimensional classification methods, and compare the classification ability of our defined shape-independent features with conventional shape-dependent features.

4.3.1 Histogram-based Classification

In this subsection we discuss the brightness histogram similarities among pedestrian regions with various poses, sizes and appearances, and introduce the histogram-feature's ability to separate pedestrian/non-pedestrian ROIs based on one generic pedestrian template.

Statistical Similarity of Brightness Histograms for Pedestrian ROIs

In Section 4.1.2, we have mentioned that pedestrian regions in infrared images are complex and not homogeneous. However, when pedestrians change pose, the intensity patterns should be consistent for similar body areas in different infrared images. Because of similar body temperatures and similar pedestrian surface properties, this observation applies not only for the same pedestrian in different poses, but also for different pedestrians with different gender, clothing, and in different seasons. Thus, there exists the similarity among

image-brightness-histogram curves for pedestrian patches containing different people, with different poses, and in different seasons. This property is demonstrated in the histogram curve comparison in Figure (4-6). Figure (4-6)(a0) is our default pedestrian template cut from a summer sequence. Figure (4-6)(a1) shows seven examples of pedestrian ROIs from four winter images, in which pedestrians of different gender have different poses. Figure (4-6)(b1) demonstrates the similarity among the brightness-histogram curves for the seven pedestrian regions. Figure (4-6)(c1) compares the average brightness-histogram curves of the above seven pedestrian regions from winter images (solid line) with the histogram curve for the pedestrian template from summer images (dashed line) in Figure (4-6)(a0).

We further demonstrate statistical histogram similarity for pedestrian regions through the variation of brightness-histogram curves from 911 rectangular pedestrian regions in seven different driving sequences. Figure (4-7)(a) shows examples of pedestrian appearances and sizes in two sample sequences. We normalize all pedestrian patches to a standard size $[58 \times 21]$ (1218 pixels) before calculating their smoothed brightness-histogram curves, i.e., $h^m_{\text{ROI}}(i)$, which is the number of pixels with brightness i for mth ROI patch. For a pair of histogram curves, $h^m(.)$ and $h^n(.)$, Figure (4-7)(b) defines the **histogram variation curve**, i.e., the distribution of histogram variation value $h^m(i) - h^n(i)$ for all brightness i. In this way, the variation of all 911 histogram curves h^m_{ROI} from their average histogram h_{mean} is presented as the collective *histogram variation curve* in Figure (4-7)(c), which resembles a Gaussian shape (of zero mean) with certain skewness. We can see that most histogram shape variation is within [-10, 10] pixels, which is only 8.2% of the largest variation (1218 pixels). This fact provides us statistical evidence that histogram curves for pedestrian regions are very similar.

The Classification Ability of Histogram Feature

Figure (4-6)(b2) shows the comparison among all histogram curves for non-pedestrian ROIs in Figure (4-6)(a2), and Figure (4-6)(c2) shows the comparison between the average histogram curves for these non-pedestrian ROIs and the brightness histogram of a summer pedestrian template (see Figure (4-6)(a0)). The results for non-pedestrian ROIs shown in Figure (4-6)(b2)(c2) are drawn in the same scale for pedestrian ROIs as shown in Fig-

ure (4-6)(b1)(c1). The comparison between Figure (4-6)(b) and Figure (4-6)(c) reveals that histogram features for pedestrian/non-pedestrian ROIs are different in most cases. Because of the histogram similarity for pedestrian regions, as well as histogram differences between pedestrian ROIs and non-pedestrian ROIs, pedestrians can be identified through histogram-similarity comparison between ROIs and one generic pedestrian template. Without losing generality, we choose Figure (4-6)(a0) as the generic pedestrian template. The **histogram difference** index is defined as the weighted summation for the square of brightness histogram difference at each brightness i as below:

$$\text{Histogram Difference} = \alpha \sum_{i=1}^{255} w(i) * [h_{\text{ROI}}(i) - h_{\text{template}}(i)]^2 \tag{4.5}$$

where h_{ROI} and h_{template} are histogram curves for ROIs and a template respectively, α is the normalization coefficient, $w(i)$ is weighting function that is fixed for all classification calculations. Typically segmentation errors might introduce extra dark background or bright regions, leading to higher histogram curve peaks at small/large brightness values. $w(i)$ is set to be small when brightness i is very dark or bright in order to reduce the impact of segmentation errors. The expected value of histogram difference for pedestrian ROIs is 0. The larger the histogram difference for an ROI, the less likely the ROI is to be a pedestrian.

4.3.2 Inertia-based Classification

The inertia-based classification feature is based on the similarity of inertia of pedestrian regions, and is not sensitive to the changes of poses, i.e., shape-independent. We define the relative inertia value for one image patch as:

$$\text{Relative Image Inertia} = \frac{\sum_{x,y} I(x,y)d(x,y)^2}{\sum_{x,y} I_{\text{template}}(x,y)d_{\text{template}}(x,y)^2} \tag{4.6}$$

where $I(x,y)$ is the pixel brightness values for image patches after size normalization, and $d(x,y)$ is the distance from a pixel to the image center as shown in Figure (4-8)(a). Image

inertia value is the summation of rotation momentum with respect to the image center for all pixels divided by a scaling factor. The scaling factor, the denominator in Figure (4-6)(a0)), is the summation of rotation momentum for all pixels in our generic pedestrian template patch. Relative inertia values for pedestrian patches with different poses should be close to 1. In the next two subsections, we will discuss the statistical similarity among all pedestrian ROI inertia values, and demonstrate the feature's classification ability.

The Statistical Similarity of Pedestrian ROI Inertia Feature

For the 911 pedestrian regions mentioned in Section 4.3.1(examples shown in Figure (4-7)(a)), the distribution of their inertia-values is plotted in Figure (4-8)(b), which resembles Rayleigh distribution and shows that inertia values are centered around the feature's expected value 1. Around 70% of pedestrian regions have inertia values between $0.8 - 1.2$, and around 94% of inertia values vary between $0.6 - 1.4$. The average inertia value is 1.03. Figure (4-8)(b) demonstrates inertia similarity for pedestrian regions in infrared images.

The Classification Ability of Inertia Feature

The inertia-based feature helps to remove classification ambiguity based on the histogram feature alone. When pedestrian/non-pedestrian ROIs produced by our segmentation algorithm have similar brightness histograms, ROIs have similar numbers of bright pixels, some of which must situate around image boundaries. For typical pedestrian ROIs, most bright pixels are found near the middle of the image patches and only a few pixels, at heads, hands, and feet areas, touch horizontal and vertical boundaries. For typical non-pedestrian ROIs, bright pixels are less centralized with more bright pixels near horizontal and vertical boundaries, leading to larger inertia values. As shown in Figure (4-8)(a) the inertia value for the right non-pedestrian patch is larger than for the left pedestrian patch despite their similar histogram feature.

4.3.3 Contrast-based Classification

In infrared images, there exists brightness contrast between pedestrian regions and their horizontal and vertical neighborhoods. The horizontal brightness contrast has been used in our segmentation algorithm to obtain pedestrians' left/right boundaries. The vertical brightness contrast is not directly used to identify pedestrians in segmentation. Instead, it is used to identify non-pedestrian as follows.

We evaluate the vertical brightness contrast for an ROI by comparing the *vertical edges* for an ROI region and for its *vertical neighborhood* regions. As shown in Figure (4-9) (a) (b), for a rectangular ROI region, its **upper/lower vertical neighborhood region** is defined as the rectangular region that is directly above/below the ROI with the same column width and half the ROI height. **Vertical edges** are defined as the image pixels with horizontal-gradient (defined in Equation (4.3), Section 4.2.2) larger than a constant threshold. The average number of vertical edge pixels in each row of a rectangular region is defined as the region's **row-edge index**. The row-edge indices for an ROI and its upper and lower vertical neighborhoods are, respectively, called **ROI row-edge index**, **upper row-edge index**, and **lower row-edge index**. These three variables are the components of our defined **ROI contrast-feature vectors**. Rich texture leads to a large row-edge index. For an ROI, the comparison between its ROI row-edge index and upper/lower row-edge index provides vertical texture contrast information between the ROI and its vertical neighborhoods.

For typical infrared images from real driving scenes, the vertical neighborhoods of pedestrian ROIs are narrow backgrounds since image stripes containing one pedestrian are narrow, and usually there is no pedestrian at the top of another pedestrian region within one segmented image stripe. The number of vertical edge pixels within narrow backgrounds is limited, and the upper/lower row-edge index for pedestrian ROIs should not be large.

Specifically, lower vertical neighborhoods beneath pedestrian ROIs contain road areas, in which we cannot find two long vertical lines or many vertical edge pixels. There is at most one vertical line produced by lane markers within narrow image stripes because of camera perspective, thus the lower row-edge index for pedestrian ROIs should not be larger than 1. If this is not the case, non-pedestrian ROIs can be identified since pedestrian ROIs

present vertical contrast between ROIs and their lower neighborhoods.

Similarly, in most cases, the upper row-edge indices for pedestrian ROIs should be smaller than 2 since their upper vertical neighborhoods generally contain sky, buildings, trees, etc., and produces less than two adjacent vertical long edges within the narrow stripes of infrared-images. The exception is when pedestrians stand right in front of "hot" light poles, which makes their upper row-edge indices close to 2. In this case, we check their ROI row-edge indices, which should be smaller than 2 for pedestrian ROIs, because some pedestrian image rows do not have any vertical edge pixel and other rows contain at most two vertical edge pixels, i.e., pedestrian-bodyline terminals. Thus, if both the upper row-edge indices and the ROI row-edge indices are large, there is no vertical contrast for ROIs, and non-pedestrian ROIs can be identified. The selected non-pedestrian ROIs are very likely to be in the middle sections of light poles, which is the case for all selected non-pedestrian ROIs shown in Figure (4-9)(a) that correspond to poles in Figure (4-5)(d). The ROI row-edge index is used to remove the ambiguity between non-pedestrian ROIs containing light poles and pedestrian ROIs in front of poles.

In summary, pedestrian ROIs and their vertical neighborhoods should present vertical contrast and lead to small upper/lower row-edge indices. Though we cannot identify pedestrian ROIs simply based on vertical contrast, a few non-pedestrian ROIs can be identified and removed when vertical contrast does not exist based on one of the two following conditions:

Case I:

Lower row-edge index is larger than 1.

Case II:

Both upper row-edge index and ROI row-edge index are close to or larger than 1.5.

The identification process is called **contrast-based non-pedestrian ROI-removal**. Figure (4-9) is an example of how we identify non-pedestrians among ten ROIs in Figure (4-5)(d) based on their vertical-neighborhood-contrast property. Rectangular regions for ROIs, their upper/lower neighborhood regions, and the corresponding image vertical edge pixels are plotted in Figure (4-9)(a)(b). Figure (4-9)(a) contains all selected non-pedestrian ROIs through contrast-based non-pedestrian ROI-removal. For each of them, the vertical-

neighborhood-contrast is vague since there are two clear vertical edges in either the upper or lower vertical neighborhoods, leading to large upper/lower row-edge indices. In this example, the identified non-pedestrian ROIs in Figure (4-9)(a) are all light pole regions as shown in Figure (4-5)(d). For remaining ROIs in Figure (4-9)(b), including all 3 pedestrian ROIs and 3 non-pedestrian ROIs, the upper/lower row-edge indices are small and we need histogram/inertia classification to separate them.

It is worth mentioning that for frames of the sequence we use one large constant threshold to determine vertical edges based on their horizontal-gradient. Usually the performance of contrast-based non-pedestrian ROI-removal is robust to threshold choices since the image contrast between ROIs and their neighborhoods is not sensitive to the threshold choices. In the case that a threshold is too large, both ROI row-edge indices and upper/lower row-edge indices for non-pedestrian ROIs are small, and the non-pedestrian ROIs cannot be removed based on the two above conditions. In this case, we can use other histogram/inertia-based classification to identify non-pedestrian ROIs.

Statistical Distributions of Contrast-based Classification Feature

To demonstrate the properties of ROI contrast-feature vectors, Figure (4-10)(a) and Figure (4-10)(b), respectively, plot the **upper-contrast-index**, i.e., ROI row-edge index (X axis) vs. upper row-edge index (Y axis), and the **lower-contrast-index**, i.e., ROI row-edge index (X axis) vs. lower row-edge index (Y axis), for the ROIs from Seq3 shown in Figure (4-17)(c1) (details in Section 4.4, Table 4.1). Feature points for pedestrian ROIs and non-pedestrian ROIs are labeled with circles and dots, respectively. As expected, the upper/lower row-edge indices for all pedestrian ROIs are not larger than 1, especially for the lower vertical neighbor regions. Among 248 pedestrian ROIs (circle-points) in Figure (4-10)(a) and (b), most have zero upper/lower row-edge indices. Only 5 pedestrian ROIs (2.02 %) have vertical edge information underneath that leads to non-zero lower row-edge indices (within $(0, 1]$), and only 13 pedestrian ROIs (5.24 %) have vertical edge information in the upper neighborhoods that leads to non-zero upper row-edge indices (within $(0, 1]$). The average upper/lower row-edge index is $0.0166/0.0083$. The largest ROI row-edge index is 1.667. On the other hand, we can see that points for non-pedestrian ROIs in

the upper-contrast-index and the lower-contrast-index 2D space are much more diversified. The statistical contrast property for pedestrian/non-pedestrian ROIs demonstrates that we can identify non-pedestrian ROIs by checking their contrast index based on the two given conditions, and our selected threshold is conservative.

4.3.4 Multi-dimensional Classification Feature

Among the three defined classification features, we can directly use 1D histogram-based or 1D inertia-based classification to determine pedestrians by measuring the similarity between ROIs and one generic pedestrian template. For pedestrian ROIs, the expected histogram feature index should be close to 0, and the inertia feature index should be close to 1. The farther the histogram or inertia feature of an ROI deviates from its expected value, the less likely the ROI is to be a pedestrian. Because contrast-based non-pedestrian ROI-removal is best at distinguishing non-pedestrian ROIs lacking in vertical contrast, the contrast-based feature should be combined with other classification features.

Classification results based on the 1D histogram feature alone can be very close to the ideal ROC boundary for winter sequences as shown in Figure (4-16)(a) (details in Section 4.4). To improve classification performance in complicated scenarios, we propose multi-dimensional classification methods. We first introduce 2D histogram/inertia-based classification in which the inertia feature helps to remove ambiguity introduced in 1D histogram-based classification as mentioned in Section 4.3.2. We then introduce 3D histogram/inertia/contrast-based classification The more classification feature-vectors to be fused, the higher the classification ability is. To demonstrate the efficiency of these fusion-based classification features, we have respectively computed the 1D histogram-based, 2D histogram/inertia-based, and 3D histogram/inertia/contrast-based, classification features for all pedestrian ROIs and non-pedestrian ROIs in three test sequences, which include winter driving (Seq1, Figure (4-17)(a1)), summer suburban driving (Seq2, Figure (4-17)(b1)), and summer urban driving (Seq3, Figure (4-17)(c1)). More details about three sequences are in Table 4.1, Section 4.4.

2D Histogram/Inertia-based Classification

For the 2D histogram/inertia-based classification method, the similarities between ROIs and our pedestrian template(see Figure (4-6)(a0)) are measured through 2D histogram/inertia feature vectors. The statistical distribution of 2D histogram/inertia feature vectors for all ROIs from the three sequences, Seq1, Seq2, and Seq3, are, respectively, presented in Figures (4-11)(a)(b), 4-12)(a)(b), and 4-13)(b)(c). Figures for both pedestrian ROIs and non-pedestrian ROIs in the same sequences are plotted using the same scale to demonstrate the distribution differences of their feature vectors. We can see that 2D feature values for all pedestrian ROIs are similar and close to their expected value $[1,\ 0]$ (X axis: inertia. Y axis: histogram.) as shown in Figure (4-11)(a), Figure (4-12)(a), and Figure (4-13)(b). Histogram/inertia feature vectors for non-pedestrian ROIs are far from $[1,\ 0]$ and much more diversified, as shown in Figure (4-11)(b), Figure (4-12)(b), and Figure (4-13)(c). Figure (4-13)(c) shows histogram/inertia feature vectors for remaining non-pedestrian ROIs after contrast-based non-pedestrian ROI-removal. The comparison confirms that 2D histogram/inertia-based features are efficient classification feature vectors.

3D Histogram/Inertia/Contrast-based Classification

Our 3D histogram/inertia-feature/contrast-based classification algorithm first calculates ROI contrast-feature vectors for each ROI, then removes partial non-pedestrian ROIs based on the two conditions given in Section 4.3.3, and finally identifies pedestrians among the remaining ROIs through 2D histogram/inertia-based classification. An example for Seq3 (shown in Figure (4-17)(c1)) is shown in Figures (4-10) and (4-13). After segmentation, there are a total of 248 pedestrian ROIs and 854 non-pedestrian ROIs whose contrast-feature vectors are plotted in Figure (4-10). In the process of contrast-based non-pedestrian ROI-removal, 284 non-pedestrian ROIs lacking in clear vertical contrast are identified and removed. The inertia vs. histogram 2D feature vectors for the 284 removed non-pedestrians, 248 segmented pedestrian ROIs, and 570 remaining non-pedestrian ROIs are plotted, respectively, in Figure (4-13)(a)(b)(c). The comparison between Figure (4-13)(a) and (b) shows that 76.76% of feature points in Figure (4-13)(a) for non-pedestrian ROIs

removed by contrast-based removal are within the data range for pedestrian ROIs as shown by the rectangular box. The contrast-based feature helps to remove potential ambiguity that occurred when using 2D histogram/inertia-based classification alone. Therefore, after contrast-based non-pedestrian ROI-removal, the percentage of segmented non-pedestrian ROIs, whose 2D feature vectors overlap with that of segmented pedestrian ROIs in 2D feature space, has dropped from 47.78% to 25.53% (as shown in Figures (4-13)(b)(c)). Thus when the detection rate is set as 100%, the false-alarm rate can drop from 47.78% to 25.53% as shown in Figures (4-16)(c1)(c2), improving classification performance.

4.3.5 Comparison with Conventional Classification Feature

In this section, we compare the classification ability of two shape-independent features — histogram-based and inertia-based — with that of a conventional pixel-comparison-based feature. We apply these three different 1D-classification features to measure the similarity between ROIs in Figure (4-6)(a1)(a2) and our default pedestrian template (as in Figure (4-6)(a0)). Specifically, the histogram feature and inertia feature are calculated according to the Equation (4.5) and (4.6), and the pixel-comparison-based feature is defined as the Frobenius norm of image pixel intensity differences between ROIs and the pedestrian template. The three 1D classification features for ROIs in Figure (4-6)(a1)(a2) are plotted in Figure (4-14)(a)(b)(c), where circles and crossed points, respectively, represent feature points for pedestrian and non-pedestrian ROIs. The inertia (X axis) vs. histogram (Y axis) feature vectors for the same ROIs are plotted in Figure (4-14)(d).

For ideal classification features, the feature points for multiple pedestrian ROIs are expected to be close to their expected values. The data ranges of feature values for pedestrian ROIs and for non-pedestrian ROIs should not overlap, and are expected to be separated as far as possible. We can see that the ratio of overlapped range over the data range for all non-pedestrian ROIs is, respectively, 0% for histogram-based method (Figure (4-14)(b)), 3.87% for inertia-based method (Figure (4-14)(c)), and 48.22% for conventional pixel-comparison-based method (Figure (4-14)(a)). In other words, to reach 100% pedestrian-detection rate, the false-alarm rate is 48.22% for conventional shape-dependent

pixel-comparison feature, while it is only 0% and 3.87% for 1D shape-independent histogram and inertia features respectively. We have demonstrated that histogram features can help identify pedestrian ROIs containing extra background regions as shown by the second pedestrian ROI in Figure (4-6)(a1). The above comparison also illustrates that conventional pixel-comparison-based features are sensitive to pose-changes in pedestrian ROIs, and the features have worse classification performance than 1D histogram-features or 1D inertia-features does when using only one pedestrian template. The comparison between Figure (4-14)(a)-(c) and Figure (4-14)(d) shows the advantages of 2D-based histogram/inertia classification over each 1D classification. To statistically demonstrate the above advantages, similar comparison is shown in Figure (4-11) for all ROIs from Seq1. Figure (4-11)(a)(b) respectively plots inertia feature (X axis) vs. histogram feature (Y axis) for pedestrian ROIs and non-pedestrian ROIs. Figure (4-11)(c)(d) respectively plots inertia feature (X axis) vs. pixel-comparison-based feature (Y axis) for pedestrian ROIs and non-pedestrian ROIs. In the vertical direction of Figure (4-11)(a) and (b), histogram feature points overlap for 19.64% of pedestrian ROIs and 16.13% of non-pedestrian ROIs in their data ranges. In the vertical direction of Figure (4-11)(c) and (d), pixel-comparison-based feature points overlap for all pedestrian ROIs and 85.33% of non-pedestrian ROIs. In 2D inertia vs. histogram space, the ratios of overlapped range over the data range for all pedestrian ROIs and for all non-pedestrian ROIs are, respectively, 12.13% and 16.31%. As expected, the histogram feature provides better classification performance than the shape-dependent pixel-comparison-based feature. Classification based on both histogram and inertia features further improves performance. More results will be shown in the next section.

4.4 Performance Evaluation

Up to now, we have presented our segmentation and classification algorithms. In real-world applications, both brightness-based and bodyline-based segmentation will be applied, and all segmented ROIs will be sent to multi-dimensional histogram-inertia-contrast-based classifiers for reliability. For the purpose of performance evaluation, we apply

different combinations of segmentation/classification algorithms to detect pedestrians in three typical scenarios: winter driving (Seq1, Figure (4-17)(a1)), summer suburban driving (Seq2, Figure (4-17)(b1)), and summer urban driving (Seq3, Figure (4-17)(c1)). From Seq1 to Seq3, driving complexity increases. The basic information for the three sequences as summarized in Table 4.1. For Seq1 and Seq2, even the simplified version of our pedestrian-detection (segmentation/classification) algorithm can improve the current detection performance as shown in Figure (4-3), which demonstrates the effectiveness of our algorithms.

We present segmentation results in Section 4.4.2 as summarized in Table 4.2, and classification results in Section 4.4.3 as summarized in Table 4.3. Pedestrian-detection examples for the three sequences are shown in Figure (4-17)(a), (b), and (c), in which the initial ROIs (after segmentation) and final detection results (after classification) are highlighted.

4.4.1 Test Sequences

The examples of pedestrian appearances for the three sequences can be seen, respectively, in Figure (4-6)(Seq1) and Figures (4-7)(a1) and (a2)(Seq2 and Seq3). All these video sequences were taken by Toyota R&D labs using a far-infrared camera with the wavelength band 8 to 14 um at a frame rate of 6 fps, i.e., 6 frames per second. The frame number and duration for Seq1, Seq2, and Seq3 are, respectively, 240 frames (40 seconds), 289 frames (48.1 seconds), and 248 frames (41.3 seconds). All three sequences recorded the whole process: pedestrians first appeared far away with small image patches (as in the first column in Figure (4-17)), then became closer and larger, until they finally disappeared from the roadside (as in the last column in Figure (4-17). The total number of pedestrians in the three sequences and the variation ranges of pedestrian sizes are listed in Table 4.1. Within these sequences, the sizes of pedestrian images change significantly from as small as 9×17 (in Seq3) to as large as 83×182 (in Seq1), 99 times larger in area. In the middle of Seq2, a pedestrian was obscured by a truck in 21 frames. Seq2 also recorded 92 additional frames after pedestrians disappeared. We expect no false alarm in these "empty" frames if our proposed shape-independent segmentation/classification works correctly.

Table 4.1: Sequence Information for three Examples

No.	Sequence Infor.	Image Figure	Frame #	Duration (sec)	Ped. #	Pedes Size Range	Size Change	Complexity
Seq1	Winter	Fig.4-17)(a1), 4-6)(a1)	240	40	331	[83 × 182, 36 × 96]	4.3	Low
Seq2	Summer, Suburban	Fig.4-17)(b1), 4-7)(a1)	289	48.1	176	[9 × 18, 30 × 68]	12.6	Medium
Seq3	Summer, Urban	Fig.4-17)(c1), 4-7)(a2)	248	41.3	248	[9 × 17, 33 × 67]	14.5	High

Table 4.2: Segmentation Algorithms & Performance for three Examples

No.	Segmentation Method	Image Figure	ROI # ped./non-ped.	missed-ped. #	Eval. Figure	Accuracy Avg. [range]	Efficiency Avg. [range]
1st	Brightness	Fig.4-17)(a)	[331, 750]	0	Fig.4-15)(a2)	95.23% [0.8058, 1]	85.84% [0.4972, 1]
2nd	Brightness	Fig.4-17)(b)	[176, 909]	0	Fig.4-15)(b2)	74.99% [0.4648 1]	89.36% [0.2375 1]
3rd	Bodyline	Fig.4-17)(c)	[248, 854]	0	Fig.4-15)(c2)	90.11% [0.5847, 1]	89.08% [0.5278, 1]

4.4.2 Segmentation Performance

To demonstrate the segmentation performance, we apply brightness-based segmentation to Seq1 and Seq2 (winter and summer suburban driving) and bodyline-based segmentation to Seq3 (summer urban driving). To evaluate segmentation quality based on our index proposed in Section 4.1.1, i.e., side accuracy and side efficiency, we have manually labeled true pedestrian's regions (in rectangular boxes) within all sequence frames. The closer the two segmentation indices are to 100%, the more accurate and efficient the performance is.

Some examples of initial segmented ROIs are highlighted in the second rows of Figures (4-17)(a), (b), and (c), which include both pedestrians and false alarms to be removed in classification procedures. Table 4.2 lists the number of segmented pedestrian/non-pedestrian ROIs and missed pedestrians, and summarizes the mean and range for both performance evaluation indices, segmentation side accuracy and side efficiency. Figure (4-15) plots segmentation side accuracy (X axis) vs. segmentation side efficiency (Y axis) for each frame as a point in 2D space in which 90.42% of Seq1 frames and 94.97% of Seq3 frames have both accuracy and efficiency indices larger than 70%. For Seq2, 93.18% of frames

Table 4.3: Classification Algorithms & Performance for three Examples

No.	Classification Method	Image Figure	Feature Vector Fig.	ROC Figure
1st	1D Inertia or Histogram	Fig.4-17)(a3)	Fig.4-11	Fig.4-16)(a), 4-3)(a)
2nd	2D Inertia/Histogram	Fig.4-17)(b3)	Fig.4-12	Fig.4-16)(b), 4-3)(a)
3rd	3D Inertia/Histogram/Contrast	Fig.4-17)(c3)	Fig.4-13, Fig.4-10	Fig.4-16)(c), 4-3)(a)

have accuracy and efficiency indices larger than 50% and 70% respectively. In a total of 777 frames from all three sequences, only 9 frames (1.16%) have segmentation side efficiency less than 50%, and only 4 frames (0.51%) have segmentation side accuracy less than 50%. Twelve out of these thirteen frames are from summer Seq2, as shown in Figure (4-15)(b). This is because brightness-based segmentation performance for summer data (case for Seq2) is less accurate than for winter data (case for Seq1), and also less accurate than using bodyline-based segmentation (case for Seq3). Full segmentation algorithms based on both brightness/bodyline will improve segmentation performance.

4.4.3 Classification Performance

The classification algorithms for the three sequences are, respectively, 1D histogram-based, 2D histogram/inertia-based, and 3D histogram/inertia/contrast-based. The classification performance indices for the three sequences — ROC boundary as defined in Section 4.1.1 — are, respectively, plotted with solid lines in Figures (4-16) (a), (b), and (c2). All ROC curves or ROC boundaries are close to the ideal ROC boundary shown in Figure (4-2)(b), which means high detection rate and small false-alarm rate.

Some examples of classification results are highlighted in the third rows of Figures (4-17)(a), (b), and (c), where some false alarms as shown in the second rows of Figures (4-17)(a), (b), and (c) are removed from initial segmentation results. Figure (4-3)(a) compares the classification results of marked points in Figures (4-16)(a), (b), and (c2), with other available published results in different seasons by plotting their frame false-alarm/detection rate index points in 2D space. For winter driving, we mark an ROI curve point in Figure (4-16)(a) whose false-alarm rate is similar to other published winter results [41], but our detection rate is higher. For summer driving, we mark ROI curve points in Figure (4-16)(b)(c2) whose detection rates are similar to other published summer results [40] and notice that our false-alarm rates are smaller. Figure (4-3)(a) shows that our classification index points are at the upper and left regions of other classification results, which means higher detection rates with fewer false alarms.

The performance of 1D histogram-based classification shown in Figure (4-16)(a) (Seq1)

is reliable for winter driving sequence, which partially benefits from accurate segmentation performance as shown in Figure (4-15). In general, 1D-feature-based classification performance is limited for summer driving as shown in Figure (4-16)(b) (Seq2) and Figure (4-16)(c) (Seq3) where dashed and dotted lines are, respectively, for 1D-histogram-based and 1D-inertia-based classification. This is because of more complex image properties and more image "noise" for summer images than those for winter images. In addition, brightness-based segmentation accuracy for summer suburban driving Seq2 is relatively less accurate than for winter driving, which adds to classification difficulties.

Fusing the histogram-based and inertia-based classification features substantially improves classification performance, as shown by the ROC curve comparison between solid lines (for 2D histogram/inertia-based classification) and dashed/dotted lines (for 1D histogram-based and 1D inertia-based classification) in Figure (4-16)(b) (Seq2) and Figure (4-16)(c) (Seq3). The performance of 2D histogram/inertia classification for Seq2 reflects its effectiveness for summer suburban driving.

The contrast classification feature helps to remove the ambiguity when using 2D histogram/inertia classification. Figure (4-16)(c1) and Figure (4-16)(c2), respectively, show the different classification results before and after contrast-based non-pedestrian ROI-removal. The advantage of 3D histogram/inertia/contrast-based classification over 2D histogram/inertia-based classification can be seen from the difference between solid lines in Figures (4-16)(c1) and (c2). The comparison between dashed (or dotted) lines in Figure (4-16)(c1) and (c2) shows the advantage of 2D histogram/contrast-based (or 2D inertia/contrast-based) classification over 1D histogram- (or 1D inertia-) based classification. For three sequences, Seq1, Seq2, and Seq3, we only apply 3D histogram/inertia/contrast-based classification to the most complicated sequence, Seq3, as shown in Figure (4-16)(c2), since 1D or 2D classification has already provided reliable results for the rest of sequences.

In summary, the segmentation performance illustrated in Figure (4-15) shows that our segmented pedestrian regions are relatively accurate and efficient. The classification performance illustrated in Figure (4-16) shows that many of the false alarms from segmentation process are removed.

4.5 Summary and Future work

In this chapter, we propose new fusion-based and layer-based methods for detecting pedestrians in far-infrared images in order to improve night driving safety. In summary, our method has the following properties or contributions.

1. We propose an original layer-based horizontal-first, vertical-second segmentation scheme, and convert a typical 2D search problem into 1D search problem.

In Chapters 2 and 3, our layer-based segmentation split one whole image into several distance-based edge layers which have much less noise and involves simpler obstacle segmentation than in original images. In this chapter, our segmentation is also layer-based. The second category of layers is several vertical stripes within infrared images in which we can detect potential pedestrians. The proposed strategy simplifies the original two-dimensional full-image segmentation into one-dimensional segmentation by splitting an image with a size of $n_{row} \times n_{col}$ into several vertical stripes with a size of $n_{row} \times n_i$, $n_i < n_{col}$. Within these vertical stripes, images of human beings can be further determined based on either *brightness* or *bodylines*. The bodyline-based vertical segmentation only needs to search among at most n_{col} number of candidate bodyline to determine the real location of pedestrians. This is called "Horizontal segmentation first, vertical segmentation second" scheme. The proposed segmentation algorithm automatically estimate the size of pedestrian regions based on the properties of bright-pixel vertical-projection curves and pedestrian horizontal contrast.

2. We have defined new shape-independent classification features and proposed fusion-based multi-dimensional classification scheme.

We have defined unique shape-independent classification features: histogram-, inertia-, and contrast-based features. We have demonstrated the similarities of these features among pedestrian image regions with different poses, as well as the differences of these features between pedestrian and non-pedestrian ROIs. The histogram variation curve for all pedestrian regions resembles a Gaussian shape of zero mean, while the distribution of inertia-features resembles a Rayleigh distribution with an expected value of 1. Contrast-features for pedestrian ROIs — the ROI row-edge indices, and the upper/lower row-edge indices —

fall within specific data range. We have shown that our proposed classification features are much less sensitive to variances of different pedestrian ROIs and have better classification ability to differentiate pedestrian ROIs and non-pedestrian ROIs. The fusion-based multi-dimensional classification scheme shows its advantages in differentiating pedestrian ROIs and non-pedestrian ROIs.

3. Our pedestrian-detection method is shape-independent.

Our horizontal segmentation algorithm is shape-independent. The projection-curves help to split original image into several layers which would not be affected by poses of different pedestrians. The segmentation procedure is robust to threshold choices.

Our algorithm uses only *one* pedestrian template — corresponding to a generic walking pose. The template is used in vertical segmentation to pick one candidate pedestrian ROI and used in classification phase to compare the similarity of multi-dimensional features derived from segmented ROIs with those of a pedestrian template. For vertical segmentation, bodyline-based template matching in given vertical stripes might provide false alarm, but it would not miss real pedestrians in given vertical stripes with given width information.

Figure (4-4)(c) and Figure (4-5)(d) show the detection results when pedestrians move in different directions, in different poses and in different seasons. Though there are partial occlusions of pedestrians in Figure (4-4)(c), individual pedestrians still correspond to single waves and are detected accurately. The performance for more cluttered scenes needs to be further investigated.

In contrast, traditional segmentation/classification algorithms are shape-dependent. Multiple pedestrian templates are necessary to deal with pedestrians in different poses. Traditional image pixel-comparison-based classification involves brightness-similarity comparisons between candidate image patches and a multiplicity of pedestrian templates at different scale. We have shown that shape-independent features are more robust with respect to pedestrian pose-changes than traditional shape-dependent features.

4. Our segmentation and classification processes collaborate with one another.

Our vertical segmentation step fuses the histogram classification features and segmentation features in order to pick one candidate within each separated vertical stripe. Initial horizontal segmentation and bodyline searching improves the segmentation accuracy and effi-

ciency, and fewer segmentation errors lead to fewer classification errors. Our proposed new statistical classification features can also be fused with other general pedestrian-detection features for multi-dimensional feature-based detection to further improve reliability and speed.

5. The algorithm has wide applicability.

Our pedestrian-detection system is not based on tracking, nor does it depend on camera calibration to determine the relationship between an object's height and its vertical image locations. Our segmentation algorithm only assumes that there is some local contrast between the image of a pedestrian and its surroundings, and does not make any other assumptions about the driving environment. Thus, it is less restricted in applicability.

6. The computational load is low.

Our segmentation process avoids brute-force searching over the whole image, and converts a typical 2D search problem into two 1D search problems. Our classification process avoids the need for comparison with multiple pedestrian templates. All these significantly decrease the computational load.

On the whole, though the proposed pedestrian detection methodology is by no means perfect, and much work is still needed to bridge the gap between present performance and the high reliability required for real-world applications, our straightforward pedestrian-detection system has made much progress and provides encouraging results in improving speed, reliability, and simplicity.

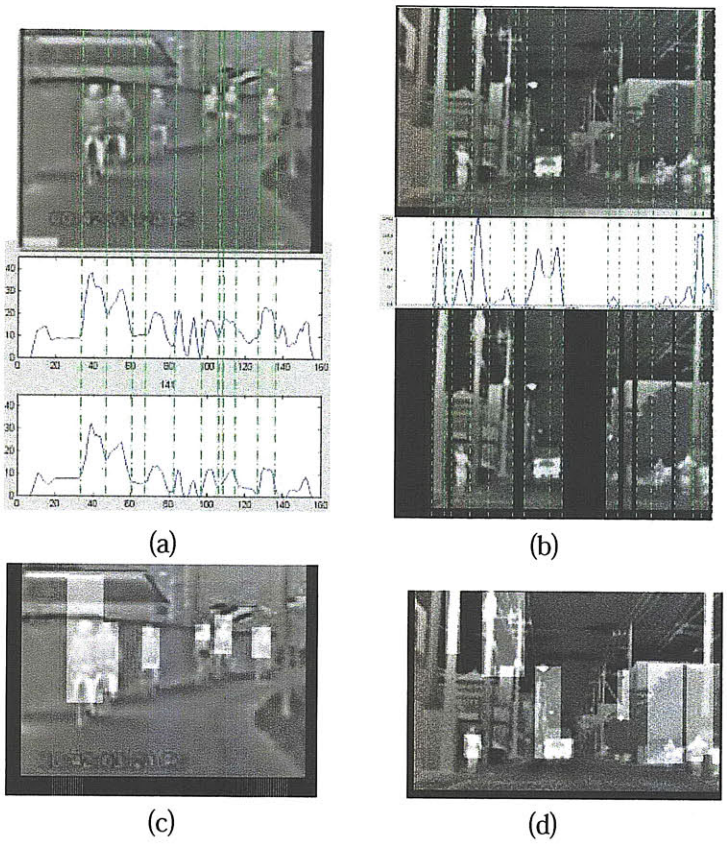


Figure 4-4: The Feature of Bright-Pixel-Vertical-Projection Curves for Infrared Images and Brightness-based Vertical Segmentation Results. For (a)(c): Winter results. For (b)(d): Summer results. (a): Top row: original infrared image in winter. Center row and Bottom row: bright-pixel-vertical-projection curves when using two different thresholds. (b): Top row: original infrared image in summer. Center row: bright-pixel-vertical-projection curve. Bottom row: horizontally segmented image stripes based on projection curve. Note that Several separated stripes shown in the center row seem to be connected. For (c)(d): Brightness-based vertical segmentation results. For all projection curves: X axis: Image column position. Y axis: Number of bright pixels in each column.

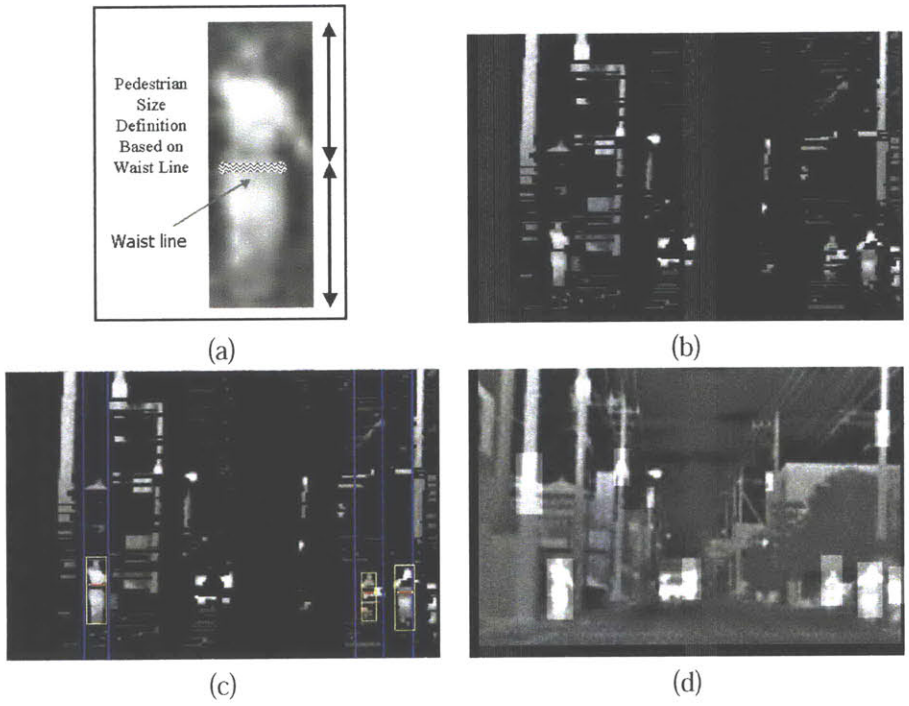


Figure 4-5: Pedestrian segmentation based on two different methods. (a): Bodyline-based geometric pedestrian-size model. (b): Bodyline image. (c): Candidate pedestrian region estimation based on (a) and (b). (d): Bodyline-based segmentation result.

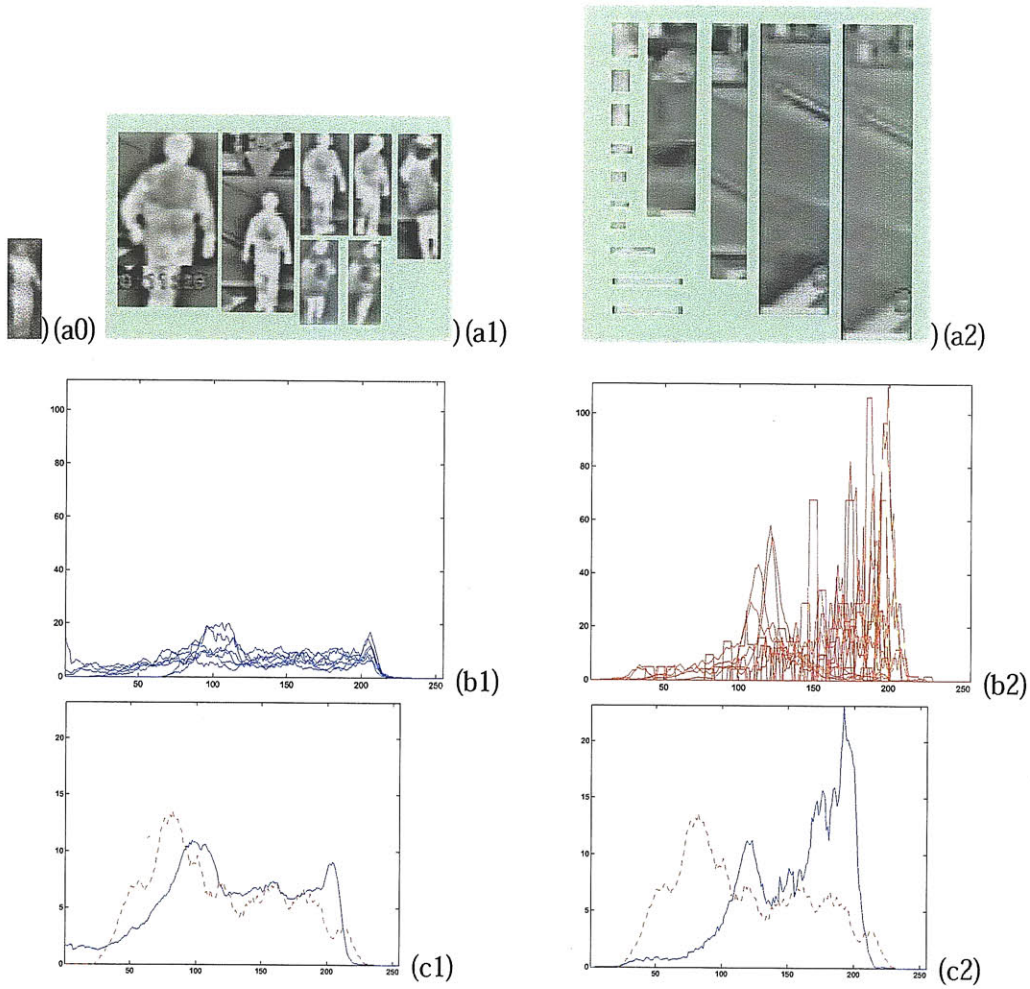


Figure 4-6: Properties of Brightness-histogram curves for Pedestrian/non-Pedestrian ROIs. (a0): Pedestrian from summer data. Used as default template in our algorithm. (a1): Pedestrian ROIs with different poses. (a2): Non-pedestrian ROIs. (a1)(a2) are segmentation results for winter data. For (b1)(b2): Brightness histograms for (a1)(a2). (b1): demonstrates histogram similarity among winter pedestrian ROIs. (b2): demonstrates the histogram variation among winter non-pedestrian ROIs. For (c1)(c2): Solid lines: Average brightness histogram for winter pedestrian ROIs (b1) and winter non-pedestrian ROIs (b2) respectively. Dashed lines: Histogram curve for summer pedestrian (a0). (c1): demonstrates the histogram similarity between winter pedestrians and summer pedestrian template. (c2): demonstrates the disparity between winter non-pedestrian ROIs and summer pedestrian template. For (b1)(b2)(c1)(c2): X axis: Image intensity range (0-255). Y axis: brightness histogram.

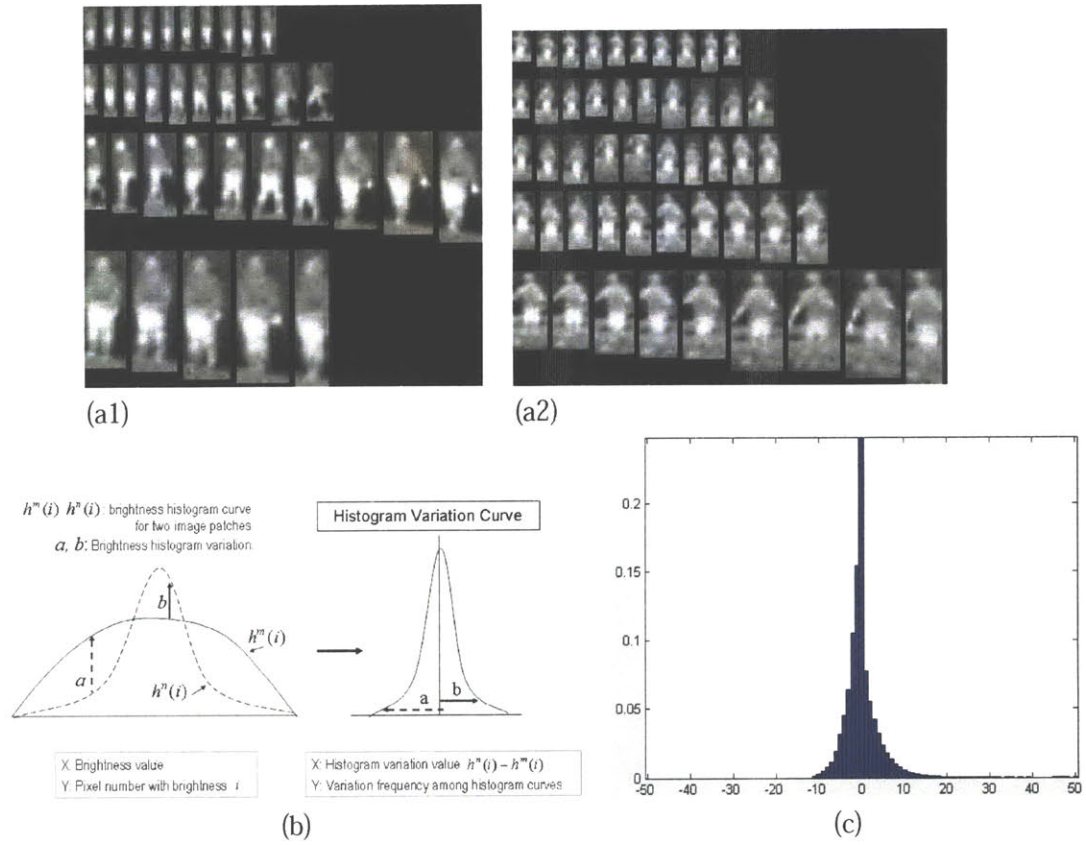


Figure 4-7: For (a1)(a2): Sample pedestrian regions from 2 sequences (every $five$ frames) to show the variation of pedestrian poses and sizes, which correspond to the Seq2 and Seq3 shown in Figure (4-17)(b1)(c1) and Table 4.1, two pedestrian-detection examples in Section 4.4. (b): Left: two brightness-histogram curves with brightness i (X axis) vs. $h^n(i)$ and $h^m(i)$ (Y axis) that are pixel numbers with brightness i from two image regions. Right: Definition for "histogram variation curve" with all possible histogram variance value $h^n(i) - h^m(i)$ (X axis) vs. variation frequency (Y axis). (c): Collective "histograms variation curve" for 911 pedestrian samples (in 7 sequences), with all possible histogram variation value (X axis) vs. the distribution of histogram variation value from all pedestrian histogram curves and their mean (Y axis).

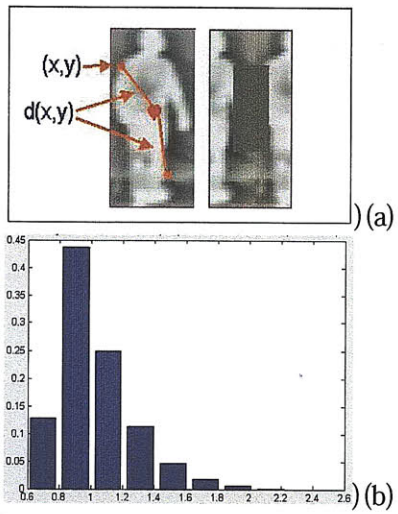


Figure 4-8: (a): ROI inertia definition. (b): Distribution of inertia values for all 911 pedestrian samples. X axis: Inertia value. Y axis: Distribution percentage.

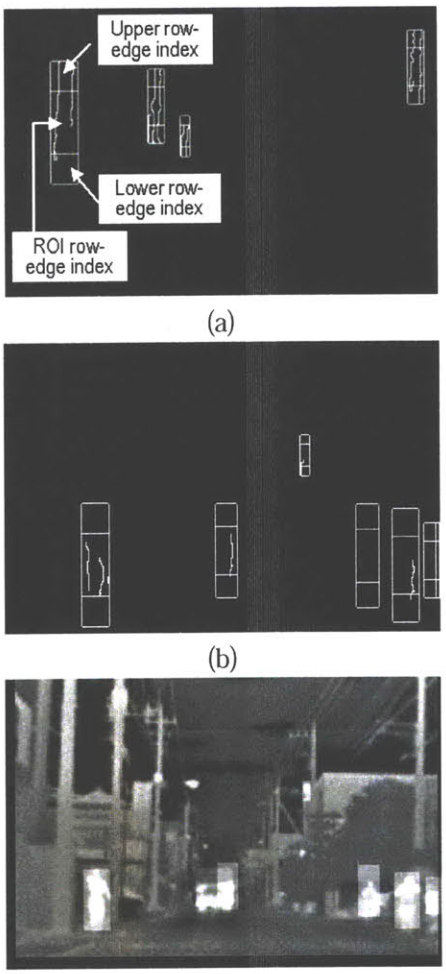


Figure 4-9: Contrast-based non-Pedestrian ROI Removal for ROIs in Figure (4-5)(d). For (a)(b): ROIs and their vertical neighborhood regions on edge map, i.e., vertical-neighborhood-contrast property. (a): For detected non-pedestrian ROIs. (b): For remaining ROIs. (c): Remaining ROIs on the original image.

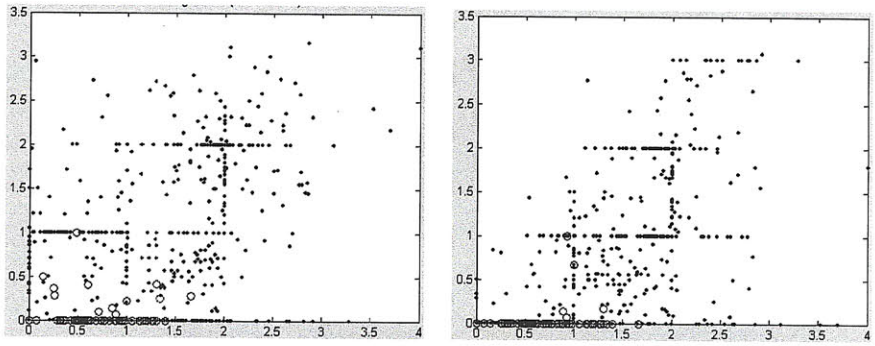
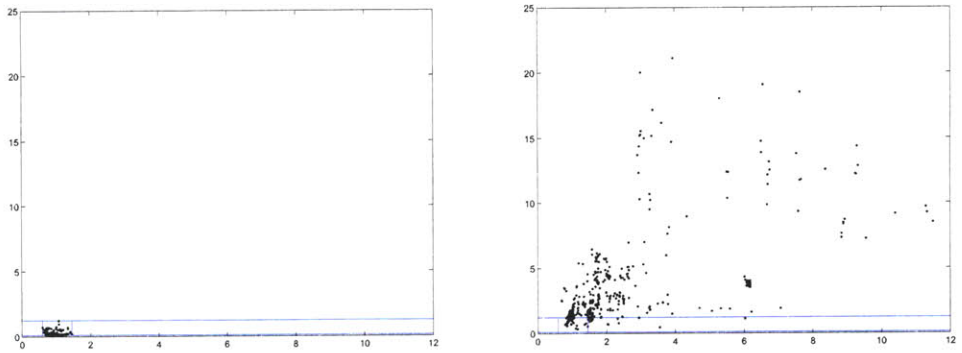
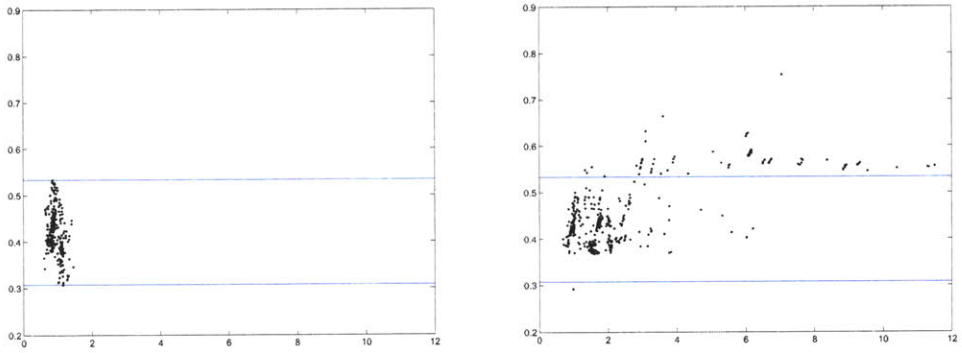


Figure 4-10: Contrast Feature Vectors for Pedestrian ROIs and non-Pedestrian ROIs from Seq3 shown in Figure (4-17)(c1) (details in Section 4.4, Table 4.1). Circles: pedestrian ROIs. Dots: non-pedestrian ROIs. (a): 2D "upper-contrast-index" for ROIs. X axis: ROI row-edge index. Y axis: upper row-edge index. (b): 2D "lower-contrast-index" for ROIs. X axis: ROI row-edge index. Y axis: lower row-edge index.

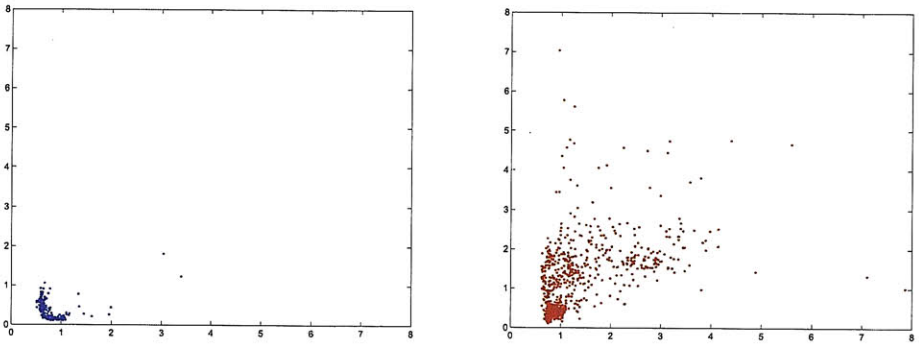


(a) pedestrian ROIs (b) non-pedestrian ROIs



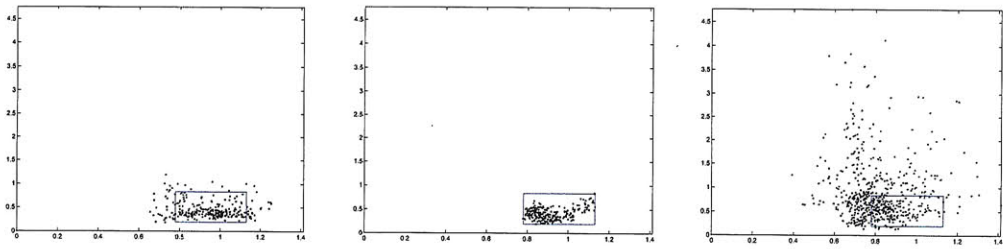
(c) pedestrian ROIs (d) non-pedestrian ROIs

Figure 4-11: 2D Feature Vectors for Pedestrian ROIs and non-Pedestrian ROIs from Seq1 shown in Figure (4-17)(a1) (details in Section 4.4, Table 4.1). For (a)(b): 2D inertia/histogram feature vectors for pedestrian ROIs and non-pedestrian ROIs respectively. X axis: Inertia feature. Y axis: "Histogram Difference," for ROIs and pedestrian template in Figure (4-6)(a0). For (c)(d): 2D inertia/pixel-comparison-based feature vectors for pedestrian ROIs and non-pedestrian ROIs respectively. X axis: Inertia feature. Y axis: "Image-Intensity-Difference" between ROIs and pedestrian template in Figure (4-6)(a0). In Figure (4-11)(a) and (b), histogram feature points for 19.64% of pedestrian ROIs and 16.13% of non-pedestrian ROIs overlap in their data ranges. In Figure (4-11)(c) and (d), pixel-comparison-based feature points for all pedestrian ROIs and 85.74% of non-pedestrian ROIs overlap in their data ranges. The comparison between (a)(b) and (c)(d) shows the advantages of the histogram feature over the shape-dependent pixel-comparison-based feature.



(a) pedestrian ROIs (b) non-pedestrian ROIs

Figure 4-12: 2D Inertia/Histogram Feature Vectors for Pedestrian ROIs and non-Pedestrian ROIs from Seq2 shown in Figure (4-17)(b1) (details in Section 4.4, Table 4.1). X axis: Inertia feature. Y axis: Histogram feature.



(a) (b) (c)

Figure 4-13: 2D Inertia/Histogram Feature Vectors for Pedestrian/non-pedestrian ROIs from Seq3 shown in Figure (4-17)(c1) (details in Section 4.4, Table 4.1) after "contrast-based non-pedestrian ROI-removal." Rectangular box: data range of feature vectors for pedestrian ROIs. (a): For removed non-pedestrian ROIs. (b): For original pedestrian ROIs. (c): For remaining non-pedestrian ROIs. X axis: Inertia feature. Y axis: Histogram feature.

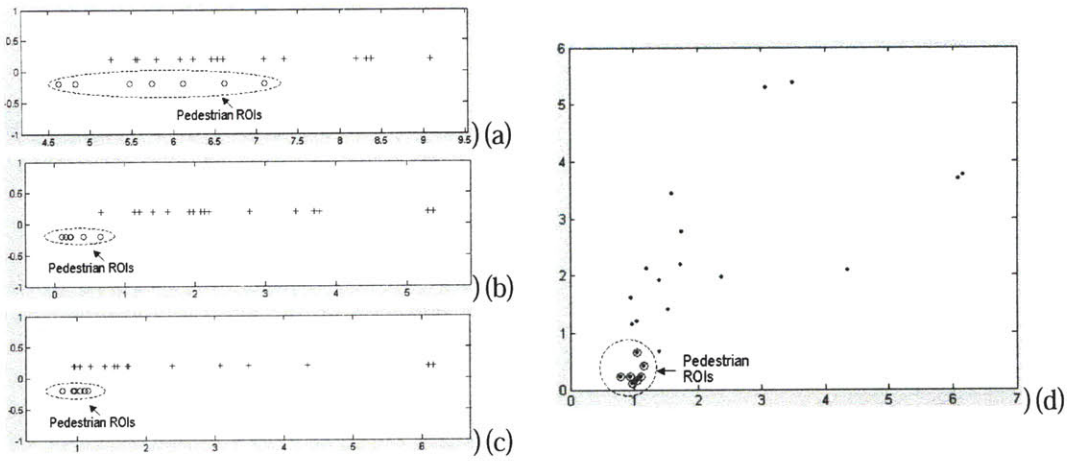


Figure 4-14: Classification Ability Comparison. For (a)-(d): feature points of different definitions that measure the similarity between ROIs in Figure (4-6)(a1)(a2) and the default pedestrian template shown in Figure (4-6)(a0). Circles and dots respectively denote pedestrian ROIs and non-pedestrian ROIs. (a): Conventional 1D Pixel-comparison-based Feature. (b): 1D Histogram-based Feature. (c): 1D Inertia-based Feature. (d): 2D Histogram/Inertia Feature. For (a)-(c): Bottom/Top lines: data ranges for pedestrian/non-pedestrian ROIs. X axis: feature value. The Y axis is only used to separate two lines. The overlap percentage between them over non-pedestrian data range: 48.22%(a), 0%(b), $3.87\%(c)$. For (d): X axis: inertia value. Y axis: histogram difference value.

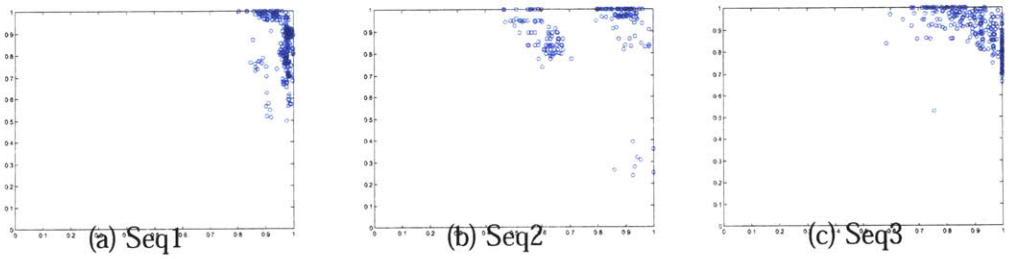


Figure 4-15: Segmentation Evaluation for 3 Sample Sequences. Detection Accuracy vs. Efficiency. X axis: frame segmentation side accuracy. Y axis: frame segmentation side efficiency.

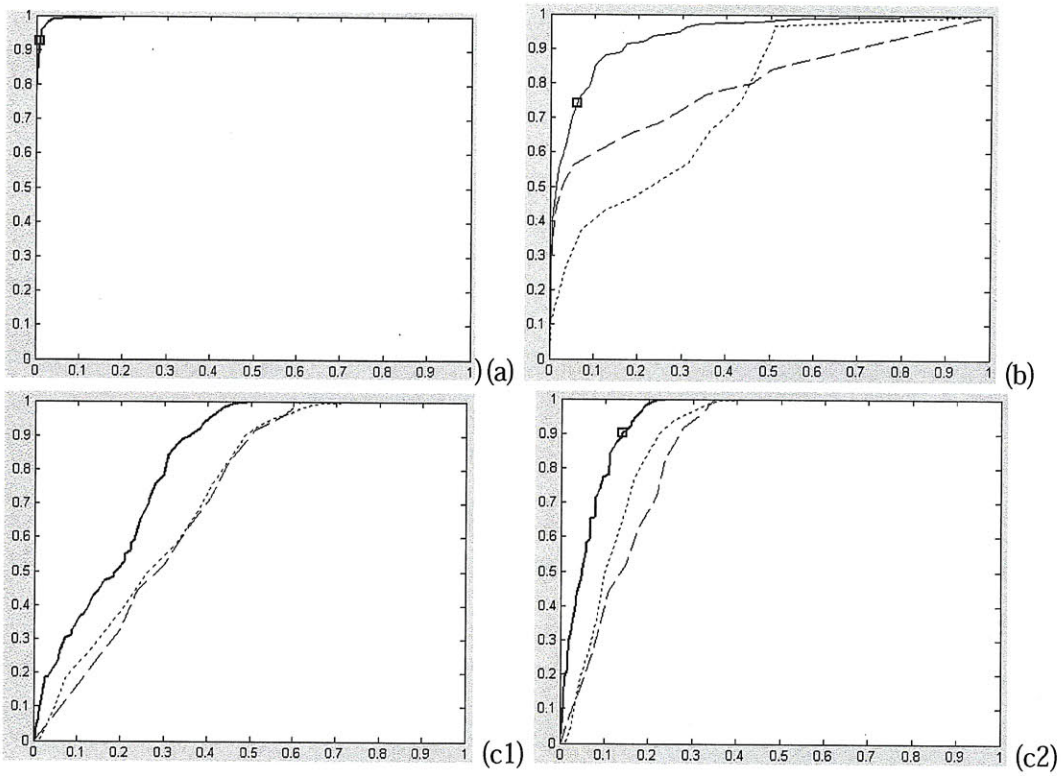
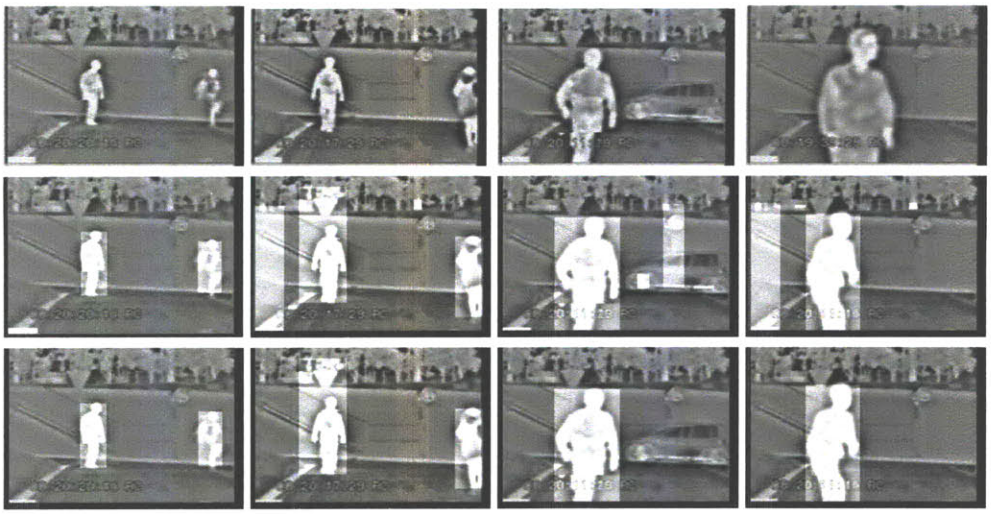
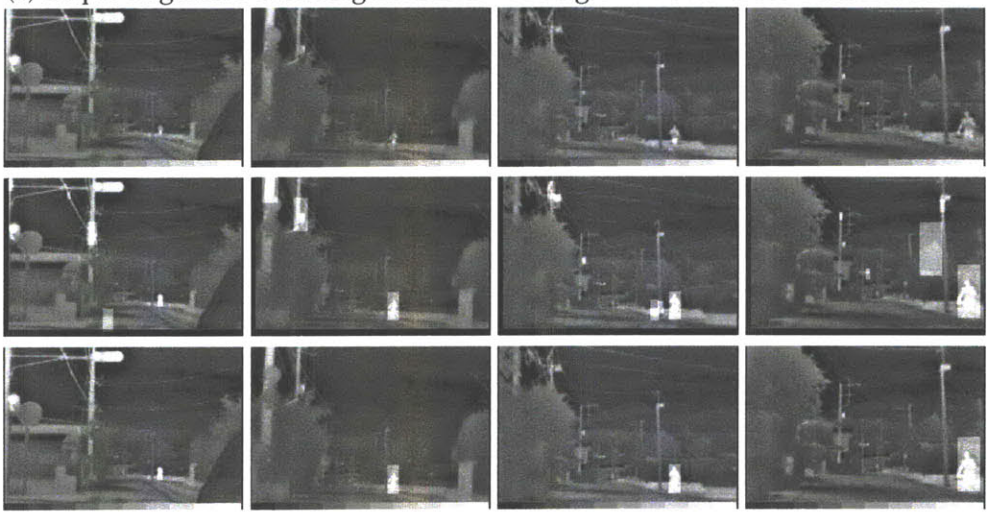


Figure 4-16: Classification Performance Evaluation for Three Sample Sequences. X axis: false_alarm_rate. Y axis: detection_rate. (a): Seq1: ROC for histogram-based classification. (b): Seq2: Histogram/inertia-based classification. Dashed line: ROC for inertia-based classification. Dotted line: ROC for histogram-based classification results. Block points: Histogram/inertia-based classification result. (c1) Seq3: ROC for histogram/inertia-based classification. Dashed line: ROC for inertia-based classification. Dotted line: ROC for histogram-based classification results. Star points: Histogram/inertia-based classification result. (c2) Seq3: ROC for histogram/inertia/contrast-based classification. Dashed line: ROC for inertia/contrast-based classification. Dotted line: ROC for histogram/contrast-based classification results. Star points: Histogram/inertia/contrast-based classification result.



(a) Seq 1: brightness-based segmentation + Histogram-based classification

(b) Seq 2: brightness-based segmentation + Histogram/Inertia-based classification



(c) Seq 3: bodyline-based segmentation + Histogram/inertia/contrast-based classification

Figure 4-17: Pedestrian-detection performance for Seq1, Seq2, and Seq3. (a1)(b1)(c1): original images. (a2)(b2)(c2): Segmentation results. (a3)(b3)(c3): Classification results.

Chapter 5

A Layered-based Fusion-based Approach to Detect and Track the Movements of Pedestrians through Partially Occluded Situations

To obtain perception abilities, conventional methods independently detect static and dynamic obstacles, and estimate their related information, which is not quite reliable and computationally heavy. We will use a fusion-based and layered-based approach to systematically detect dynamic obstacles and obtain their location and timing information.

5.1 Introduction

In order to enhance the safety of intelligent transportation systems, a large amount of surveillance cameras are mounted for the purpose of intelligent monitoring and automatic environment/event understanding. Recently intelligent surveillance for intersections and large supermarkets has attracted lots of attention since it is important to understand the human behavior of passing pedestrians. The intelligent surveillance involves automatic human detection, identification and activity understanding. Such information can help to provide early warning information for vehicles and for intelligent management of trans-

portation systems.

To understand environment automatically, we need to identify objects of interest and their static and dynamic information. Typically two steps are involved: *initial segmentation* in the first few frames, which is known very complicated, and *human tracking* in the subsequent frames to detect the identified foregrounds in initial segmentation. In general, human tracking is less complicated and more computationally efficient than initial segmentation.

5.1.1 Traditional Algorithms for Initial Segmentation and their Limitation

The initial segmentation runs into difficulties when the contrast between foreground and background is low, for examples, when people wear clothes close to background's color. It is also challenged by unpredictable background, specifically with non-rigid objects, such as trees, or constantly-changing environment. Traditional segmentation algorithms usually make special assumptions. Researchers often take advantage of visual features, such as obstacles' motion, the symmetry of obstacles' regions and asymmetry of background, etc., [5], to differentiate foregrounds from backgrounds. The traditional methods for initial segmentation include reference-image-based segmentation, feature-based segmentation, and geometric-model-based segmentation.

Reference-image-based segmentation is to detect moving objects based on the differential images between video frames and reference images. If cameras are fixed (surveillance situations), a reference can be background images estimated through statistical model [44] or acquired when no foreground shows up [58] [59]. Foregrounds are detected from differential images through k-means clustering based on motion information [54] [60]. Typically motion information estimation is noisy, especially at occlusion boundaries. Human body is not rigid and their motion is not uniform. Furthermore, segmentation accuracy and reliability depends on the quality of differential images, so they would be affected by changing lighting and shading situations, shape-changing non-rigid background objects such as trees branches, etc.

Feature-based segmentation detects humans by capturing their distinctive features, such as human faces, heads [50], skin color [49] [57], the rigidity of KLT interested corner feature points in head/header areas, depth information, or deviation from given human template, etc. These features might not exist during occlusion, which limits the application. It is difficult to find a general template for pedestrians with various poses.

Geometric-model-based segmentation generates a specific human geometric model that describes human beings, including three dimension human models /high DOF articulated models for bodies and/or limbs [51] [52], two dimension appearance-based model (blob-based / pixel-based texture model [53], a texture-free shape model based only on Mahalanobis distance map [56], statistical fore- and background models, illuminance based model [54]) , etc. These models build the connection between different poses and their corresponding images. Complicated human configuration model is usually needed for pose analysis, which usually involves high computational cost and may not work very well for occlusion situations [55].

5.1.2 Traditional Algorithms for Human Tracking and their Limitations

In general, tracking processes are challenged by the significant variation of image positions and sizes for objects between successive video frames, especially when there is an occlusion and when there are unexpected dynamic movements, etc. Complicated math models are needed to independently detect information for obstacles, and to estimate the dynamics of interested objects.

Mean Shift Iteration algorithm [65] tracks non-rigid objects based on visual features whose statistical distributions characterize the objects of interest. The technique works for objects with different color/texture patterns, and is robust to partial occlusions, clutter, rotation in depth, and changes in camera positions. However, mean shift iterations are needed to find target candidates that are most similar to a given target tracking model with specific similarity metric, for example, Bhattacharyya coefficient.

Geometric model based algorithm tracks human geometric models parameters, for example, human body contour curves [55], human body regions (blobs) [67], pose and location parameters of bodies and hands, etc. Such tracking models are usually not uni-modal, involve transformation, and usually are not Gaussian [55]. The complicated algorithms need a few frames' initialization time before tracking performance gradually stabilizes.

Particle filtering [63] and related algorithms require no model and support multi-modal tracking, and thus they can deal with nonlinear and non-Gaussian problems for stochastic dynamic estimation at the cost of higher computational load. [64] formulates human segmentation as a Bayesian MAP estimation problem whose solution is based on Markov chain Monte Carlo approach.

In summary, because of heavy computational load, most tracking techniques work better with larger and slower objects and are limited in their ability to track small fast objects. Traditional human detection and identification algorithms are computationally heavy while their accuracy and reliability are limited in difficult scenarios. We need to develop a new algorithm with a higher performance and fewer assumptions than traditional methods.

5.1.3 Layered-based and Fusion-based Dynamic Tracking of Pedestrians

We apply fusion-based and layered-based principle to detect and track the locations of pedestrians in visible images from a single camera. In the previous chapters, we have discussed methods to separate one image into several layers while the sequential segmentation of interested targets in multiple layers is much easier than in the original image. Two categories of layers and corresponding layer-separation methods are introduced. The first category of layers that we defined are distance-based image-layers for stereo visual images in Chapters 2 and 3. The second category of layers are several vertical stripes within monocular infrared images for pedestrian detection in Chapter 4. The above methods respectively take advantage of statistical distribution of distance features and brightness features, which cannot be used to applications with single visible camera. In this chapter, we will propose a new layer-based method for monocular visible images for pedestrian detection.

Similar to Chapter 4, we exploit layered-based segmentation that first searches pedestrians in horizontal dimension, and then searches in vertical dimension, which is introduced in Chapter 5.2. While applying initial segmentation to every frame, we also introduce fusion-based layered-based tracking models that combine initial segmentation information and dynamic track models in order to detect the dynamics of interested targets, which is introduced in Chapter 5.3. The results, discussion, and conclusion are respectively presented in Chapters 5.4, Chapters 5.5, and Chapter 5.6. Our detection method has the ability to deal with both non-occlusion and occlusion cases.

5.2 Layered-based Image Separation and Initial Segmentation

In this section, we will discuss how to extend the idea of "Horizontal segmentation first, vertical segmentation second" for pedestrian detection and tracking from monocular infrared images to monocular visible images. We respectively discuss the *horizontal segmentation* and *vertical segmentation* in Section 5.2.1 and 5.2.2.

5.2.1 Initial Horizontal Segmentation for Visible Images from Single Camera

In Chapter 4, for infrared images, the *bright-pixel-vertical-projection curves* are effective shape-factors that help to horizontally split an infrared image into several vertical stripes and determine pedestrians' horizontal locations [42][100]. Such method depends on the feature of bright pixels corresponding to pedestrians. For applications with single visible camera discussed in this chapter, we need to find a new feature to differentiate foregrounds and backgrounds. In this section, we define *Combined-Difference-Image*(CDI or CD-Image) and *Combined-Difference-Image based Vertical Projection-Curves*(CDI-VPC or CDI-VP curves) for visible images, and then explain how to take advantage of CDI-VPC to horizontally split an visible image into several vertical stripes and to determine pedestrians' horizontal locations or bounding regions. In the next section, the CD-Images,

their edge maps and their CDI-VP curves are fused together to further identify the vertical location of real pedestrians.

The Definition of Combined-Difference-Image and Combined-Difference-Image based Vertical Projection-Curves

In order to define *Combined Difference-Image* (CDI or CD-Images) and *Combined-Difference-Image based Vertical Projection Curves* (CDI-VPC or CDI-VP-Curves), we need three continuous monocular visible images with sequence numbers $(k-1)$, (k), $(k+1)$. We first compute the absolute value of the difference images between every two continuous frames, which are defined as **Backward Difference-Image (BDI)** and **Forward Differential-Image (FDI)** . BDI and FDI are labelled as (Back_Diff_Img) (Forward_Diff_Img) in Equations (5.1) and (5.2) as below.

$$\text{Backward\_Diff\_Img}(k) = |\text{img}(k) - \text{img}(k-1)| \tag{5.1}$$

$$\text{Forward\_Diff\_Img}(k) = |\text{img}(k) - \text{img}(k+1)| \tag{5.2}$$

We then compute the **Binary Backward Difference-Images (BBDI)** and **Binary Forward Difference-Images (BFDI)** based on the following equations:

$$\text{Binary\_Backward\_Diff\_Img}(k) = \begin{cases} 1 & \text{if Backward\_Diff\_Img}(k) > \texttt{thres\_backward\_DI} \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

$$\text{Binary\_Forward\_Diff\_Img}(k) = \begin{cases} 1 & \text{if Forward\_Diff\_Img}(k) > \texttt{thres\_forward\_DI} \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

where `thres_backward_DI` and `thres_forward_DI` are adaptive thresholds and defined as *one tenth* of the maximum value of difference-images as in Equation (5.5) and (5.6). The adaptive threshold is set to be very small in order to ensure that the combined-difference-

image have non-zero projections in CDI-VP-curves at the columns corresponding to pedestrians' locations.

$$\texttt{thres\_backward\_DI} = \frac{1}{10} max(\text{Backward\_Diff\_Img}(k)) \tag{5.5}$$

$$\texttt{thres\_forward\_DI} = \frac{1}{10} max(\text{Forward\_Diff\_Img}(k)) \tag{5.6}$$

We name the intersection region of the two binary images: binary backward and forward difference images to be *FDI-BDI Binary Intersection region*(FDI-BDI BI-region). Inside the region, there are large temporal intensity differences between the current image (k) and both the previous $(k-1)$ and the next frames $(k+1)$ as described below:

1) The absolute value of image pixel intensity-difference at coordinates (row, col) between Seq# (k) and Seq# $(k-1)$ is larger than `thres_backward_DI`

2) The absolute value of image pixel intensity-difference at coordinates (row, col) between Seq# (k) and Seq# $(k+1)$ is larger than `thres_forward_DI`

Within the region of FDI-BDI binary intersection (FDI-BDI BI-region), pixel intensities are defined based on two different criteria, leading to two versions of **combined difference-images** as below.

- Def_1:

  CDI intensities are defined as the minimum intensity values of backward difference-images and forward difference images at the FDI-BDI BI-region as shown in Equation (5.7).

$$\text{Combine\_Diff\_Img}(k) \text{ .....Def\_1} = \begin{cases} \min(\text{Backward\_Diff\_Img}(k), \text{Forward\_Diff\_Img}(k)) \\ \qquad \text{if Binary\_Backward\_Diff\_Img}(k) > 0 \\ \qquad \& \text{ Binary\_Forward\_Diff\_Img}(k) > 0 \\ 0 \qquad\qquad \text{otherwise} \end{cases} \tag{5.7}$$

- Def_2:

  CDI intensities are defined to be the same as the intensity values of backward difference-images within the FDI-BDI BI-region as shown in Equation (5.8).

$$\text{Combine\_Diff\_Img}(k) \text{ ..... Def\_2} = \begin{cases} \text{Backward\_Diff\_Img}(k) & \\ \qquad \text{if Binary\_Backward\_Diff\_Img}(k) > 0 & \\ \qquad \& \text{ Binary\_Forward\_Diff\_Img}(k) > 0 & \\ 0 \qquad \text{otherwise} & \end{cases} \tag{5.8}$$

Both definitions work well for the purpose of horizontal segmentation. The difference between these two definition will be discussed later in Section 5.2.1. **Combined-Difference-Image based Vertical Projection-Curves** are defined as the vertical projection of Combined-Difference-Images.

Properties of Combined-Difference-Image and Layer-based Segmentation

Figure (5-1) and Figure (5-2) present two examples of three synthetic continuous image frames, their backward and forward Difference-Images (BDI/FDI), and their Combined-Difference-images (CDI). Foregrounds move to the right at two different speeds in Figure (5-1) or at three different speeds in Figure (5-2). For case (1), (2), and (3) in Figure (5-2), the horizontal distances of pedestrian regions in continuous frames are respectively larger than one person's width (the left column), between half of one person's width and one person's width (the middle column), and smaller than half of one person's width (the right column). In Figures(5-1) and (5-2), row (a) shows the original three continuous images, Img$(k-1)$, Img(k), Img$(k+1)$, row (b) vertically displays the corresponding Backward-Difference-Image (BDI), Forward-Difference-Image (FDI), row (c) and (d) are the Combined-Difference-Image (CDI) based on two definitions.

For case (1) in Figure (5-2), the locations of the pedestrian in three frames do not overlap, BDI and FDI contain pedestrians with their original intensities at two different locations in two continuous frames. Thus CD-Images in Figure (5-2)(c) keep common

pixels of BDI and FDI whose intensity values are the same as from the middle of three continuous images. Thus the locations and boundaries of our interested foreground are identified.

For case(1) in Figure (5-1) and case(2) in Figure (5-2), foreground regions in three frames start to overlap in the horizontal direction and the widths of overlapped regions are less than half of original width of foreground. CD-Images in Figure (5-1)(b1) and Figure (5-2)(b2) keep all pixels of foregrounds' center axis. Therefore, if the original images at the center axis have largest vertical projections, CDI will inherit the peaks.

For case(2) in Figure (5-1) and case(3) in Figure (5-2), the widths of overlapped regions are more than half of original width of pedestrians. The overlapped regions contain center axis, and corresponding pixels in CDIs are the absolute differences between two continuous frames carried from overlapped areas in Backward-Difference-Image. CDIs in all these situations preserve the locations and rough shape of our interested foreground.

For simplicity of illustration, we ignore background information in two examples. The above observation holds true for general situations with fixed backgrounds. The impact of background pixels can be deducted in computation of BDI and FDI. and would not affect CDIs and their corresponding vertical projections.

An example for a real video sequence is shown in Figure (5-3), in which Figure (5-3)(a) shows three continuous image frames, and Figure (5-3)(b) sequentially shows an example of BDI, FDI, and two definition of CD-Images. Black regions correspond to 0 intensity. The brighter the region, the higher the intensity.

Three examples, simple synthetic ellipse foreground, synthetic human shape, and the human in real video sequences, demonstrate the following properties:

1. The horizontal and vertical boundary-locations for two defined CD-Images are the same as the boundary-locations of foregrounds in the middle frame Img(k) in three continuous frames.

2. The bright pixels remaining in Combined-Difference-Images roughly preserve the shape of our interested foregrounds in the middle frame Img(k) in three sequential images.

If there is no background, both statements hold under general situations assuming there is only one foreground in an image. Scanning each column from the left to the right in the BFDI, we can see the transitions from columns containing zero pixels to non-zero pixels happening at the *left* boundaries of foreground in the middle frame $\mathrm{Img}(k)$. Scanning each column from the right to the left in the BBDI, we can see the transitions from columns containing zero pixels to non-zero pixels happening at the *right* boundaries of foreground in the middle frame $\mathrm{Img}(k)$. Thus, the common non-zero area of the BFDI and BBDI, provides the left and right boundaries for our interested foreground. Similarly, the common non-zero area in BFDI and BBDI provides the top and bottom boundaries for our interested foreground.

If the impact of background is considered, the second property is more sensitive to the choices of thresholding parameters than the first. CDI areas corresponding to background regions might not be completely zero. However, the non-zero pixel due to noises would not affect the transitional positions of CDI-VP-Curves from background to foreground since the first property depends on the summation of column pixels. Thus, we can use the first property to determine foregrounds' locations.

If there are more than one foreground, we can divide the original image into multiple sub-regions each of which contains only one foreground. Then we can easily locate interested foregrounds in these separated sub-regions. Our algorithm takes advantage of the vertical projections of these CDIs, and horizontally separates an image into several vertical stripes containing possible foregrounds as shown in this section.Within each vertical stripe, the vertical edge maps of corresponding CDIs help to determine the vertical locations of interested objects as in Section 5.2.2 and 5.3. As shown in Figure (5-3)(c), two pedestrians can also be vertically determined in their individual vertical stripes.

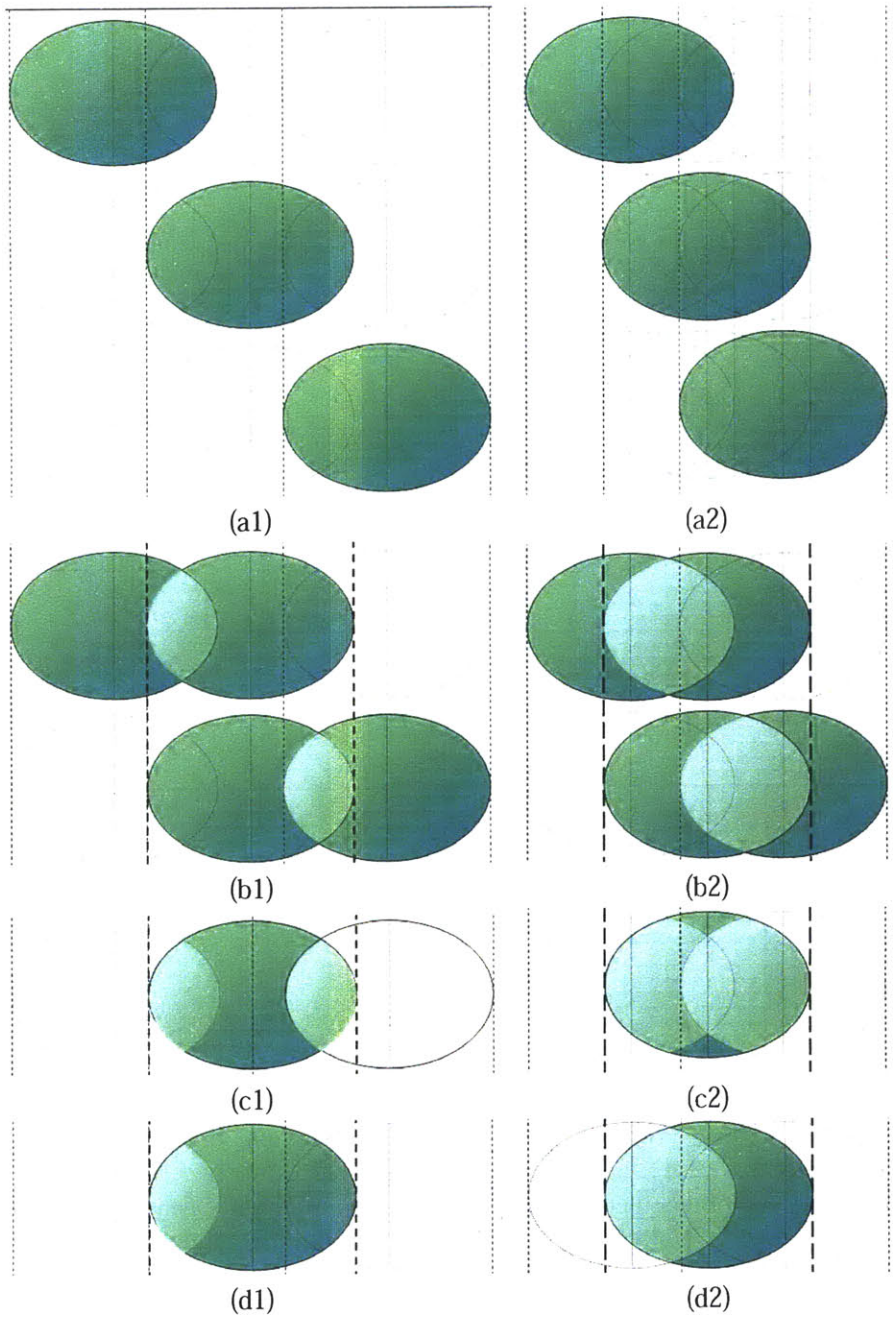


Figure 5-1: Three synthetic continuous image frames, their Backward Difference-Images(BDI), Forward Difference-Images(FDI), and Combined Difference-images(CDI) when their foregrounds (ellipses) move at two different speeds: case (1) and (2). (a) The top, middle and bottom rows are three continuous synthetic images frames: *previous, current and next.* Two columns (a1)(a2) respectively correspond to situations when foreground moves fast(a1) and slow(a2). (b) Top/bottom: BDI and FDI for (a). (c)(d) The 1st and 2nd definition of CDI for both cases. The shapes for CDI are equal to the shapes for the *current* frames.

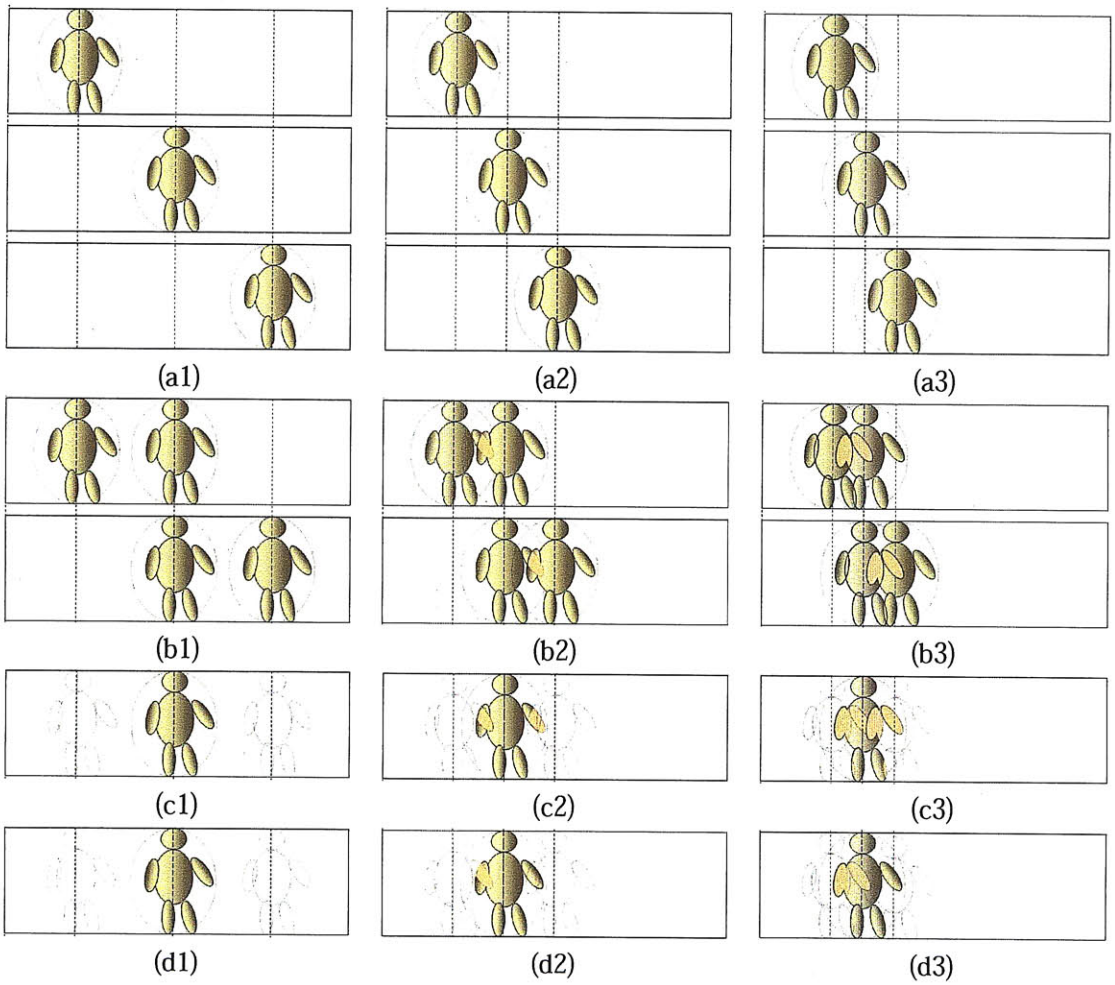
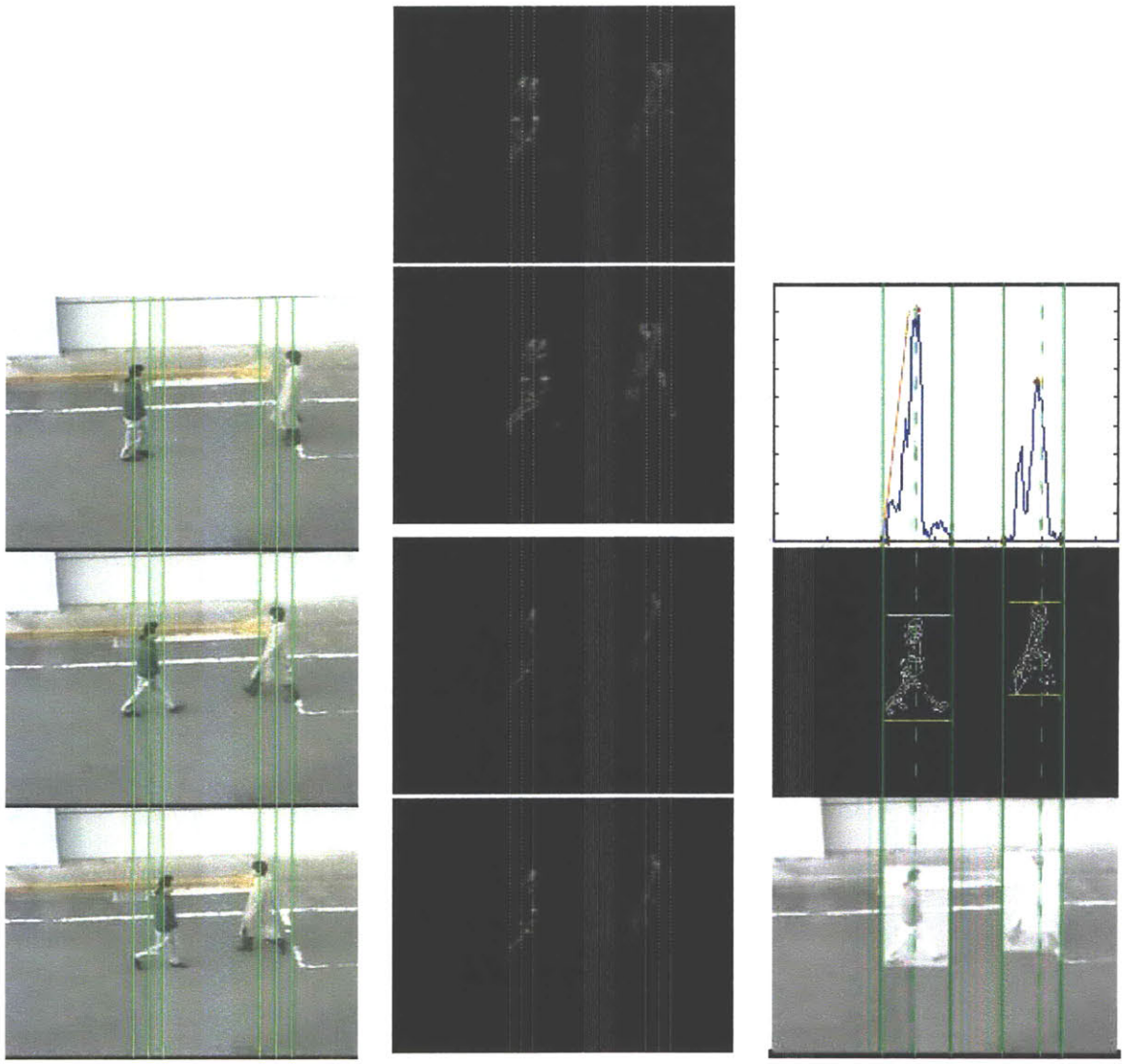


Figure 5-2: Three synthetic continuous image frames, their Backward Difference-Images(BDI), Forward Difference-Images(FDI), and Combined Difference-images(CDI) when their foregrounds (pedestrians) move at three different speeds: case (1), (2) and (3). (a) The top, middle and bottom rows are three continuous synthetic images frames: *previous, current and next.* Three columns (a1)(a2)(a3) respectively correspond to situations when people walk extremely fast(a1), normal(a2) and extremely slow(a3). For the top row, the horizontal regions of pedestrian images in three frames do not overlap. (b) Top/bottom: BDI and FDI for (a). (c)(d) The 1st and 2nd definition of CDI for three cases. The shapes for CDI are equal to the shapes for the *current* frames.



(a) (b) (c)

Figure 5-3: Example of horizontal segmentation based on the CD-images and CDI-VP-curves. (a) Three continuous sample images with pedestrians (b) Top two figures: BDI and FDI for (a). Bottom two figures: two definitions of CDIs for (a), Def_1, Def_2. (c) Horizontal segmentation based on the vertical projection of CDIs. Top row: CDI-VP curves. Middle row: Edge maps for CDI. Bottom row: Segmentation results.

Additional Properties of Combined-Difference-Image based Vertical Projection-Curves

Based on the properties of CD-Images discussed above, the vertical projection for CDI presents the following properties.

- Single foreground that moves in any three continuous image frames corresponds to one wave in CDI-PV-Curves.

  If we assume that each image has several foregrounds not occluding with each other in each frame and among any three continuous frames, the CDI-PVC is thus composed of multiple waves.

- The horizontal locations for waves in CDI-VP Curves correspond to the horizontal boundaries of related pedestrians.

Such properties are shown in Figure (5-3)(c) in which the top figure plots the vertical projection for the CDI image shown in the bottom row of Figure (5-3)(d). Based on the CDI-VP-Curve, horizontal segmentation is applied and then the object width and locations are recovered.

In Figure (5-3)(c), we observe that the dotted green vertical lines pass through the pedestrians' heads and projection peaks after aligning PVC curves and its corresponding middle frames Img(k). The defined "CDI-VP curves" provide not only the horizontal boundaries of interested foregrounds, but also their heads' locations. This observation can be extended to the general situations. We will discuss and demonstrate this conclusion in the following four situations.

Proof 1: When the width of overlap region between two continuous frames is less than half of the foreground's width. Cases (1)(2) in Figure (5-2), and Case (1) in Figure (5-1) fit into such scenario.

Proof 2A: When foregrounds are pedestrians and their *body-trunk* regions in two continuous frames do not overlap for more than half of body width. Cases (3) in Figure (5-2) fits into such scenario. The *body-trunk* region is defined as the image area corresponding to the head and torso part of a pedestrian.

Proof 2B: When foregrounds have uniform intensity. Adopt CDI with the first definition

Proof 2C: When foregrounds have randomly distributed intensity with same mean. Adopt CDI with the first definition

For Proof 2A, 2B, 2C, we only need to discuss the situations when the width of overlap region between two continuous frames is smaller than half of the foreground's width, and we also assume that vertical projection for foregrounds decreases monotonically from the VP-peak location to both ends.

For all these situations, we define three continuous images with sequence numbers $(k-1)$, (k), $(k+1)$ containing one foreground, and define x_{peak} to be the horizontal location where the vertical projection of the foreground peaks in the middle frame Img(k) as shown in Equation (5.9).

$$\text{VP}_{x_{peak}}\ (\text{Img}(k)) \geq \text{VP}_{col \neq x_{peak}}\ (\text{Img}(k)) \tag{5.9}$$

where VP_{col} (Img(k)) represents the vertical projection of Img(k) at column col. We name x_{peak} to be the *VP-peak location* for image Img(k), and name the column at x_{peak} to be the *VP-peak column*, which is normally the center axis.

Proof 1: When the width of overlap region between two continuous frames is less than half of the foreground's width.

For situations shown in Figure (5-1)(a1) and Figure (5-2)(a1)(a2), when we compute the absolute difference-image FDI and BDI based on Equation (5.1) and (5.2), the most right boundaries of foregrounds in the previous frame Img$(k-1)$ would not cross *VP-peak location* x_{peak}, and the most left boundaries of foregrounds in the next frame Img$(k+1)$ would not cross *VP-peak location* x_{peak}. The overlap regions between FDI and BDI do not contain *VP-peak column* for Img(k). Under such condition, the pixel columns at the location x_{peak} for CDI of both definitions are the same as *VP-peak column* for Img(k) as in Equation (5.10).

$$\text{VP}_{x_{peak}}\ (\text{CDI}(k)) = \text{VP}_{x_{peak}}\ (\text{Img}(k)) \tag{5.10}$$

Without losing generality, we can assume that the vertical projection of difference-values in the overlap-region for FDI and BDI are smaller than the vertical projection of the original values as in Equation (5.11).

$$\mathrm{VP}_{col \neq x_{peak}}\ (\mathrm{CDI}(k)) \leq\ \mathrm{VP}_{col \neq x_{peak}}\ (\mathrm{Img}(k)) \tag{5.11}$$

Such assumption holds since pixel values in foreground are quite close to each other and the pixel values in CDI are much smaller than in original images, which is also confirmed in the comparison of CDI and original images as in Figure (5-3). Combining Equation (5.9), (5.10) and (5.11) we have

$$\mathrm{VP}_{x_{peak}}\ (\mathrm{CDI}(k)) \geq\ \mathrm{VP}_{col \neq x_{peak}}\ (\mathrm{CDI}(k)) \tag{5.12}$$

The peaks of vertical projections for CDI(k) (for both definitions) and the ones for Img(k) are the same and happen at the same locations x_{peak} as long as Img($k-1$) and Img($k+1$) do not pass through the column at x_{peak}, which are the situations shown in Figure (5-1)(a1) and Figure (5-2)(a1)(a2). The above condition requires the relative motion of foregrounds between two continuous frames is equal or larger than half of foreground's width since x_{peak} is usually close to the center axis of foreground. In other words, the width of overlap region between two continuous frames should be less than half of the foreground's width.

If the relative movement is not fast enough and the overlap regions between two continuous frames pass through the center axis for arbitrary foregrounds, there still exists the correspondence relationship between head's location and VP-peak location for CDI (1st definition). We will first sequentially present the proof for the three situations discussed in Proof 2A, Proof 2B, Proof 2C using the first definition of CDI. In Section 5.2.1, we will discuss situations using the second definitation of CDI. For Proof 2A, Proof 2B, Proof 2C, without losing generality, we assume that vertical projection for foregrounds decreases monotonically from the VP-peak location to both left and right sides.

Proof 2A: When foregrounds are pedestrians and their body-trunk regions in two continuous frames do not overlap for more than half of body width.

In this situation, upper limbs occluded partial regions in the column at x_{peak} as well as the neighborhood columns around x_{peak}, which would not affect the VP-peak location for CDI(k) as demonstrated by the Figure (5-2), case (3). Thus, due to the special shape for pedestrians, VP-peak location for CDI(k) will be the same as for Img(k) as long as their body-trunk regions in two continuous frames do not overlap for more than half of body width. Figure (5-3) is an example that supports this observation. The horizontal movement (around 10 pixels) is larger than half of body-trunk width (less than 10 pixels) as shown by the long vertical dotted lines. The projection curves for both CDI definitions will have peak corresponding to the heads' location as demonstrated by green vertical lines. The relative movement in Figure (5-3) is very typical in most real time video sequences.

Proof 2B: When foregrounds have uniform intensity, we adopt CDI with the first definition

Let us consider Figure (5-1)(b). If the pixel intensities inside three ellipses are uniform, the overlap regions between FDI and BDI will be zero. CDI in Figure (5-1)(c2) will degenerate into the pattern shown in Figure (5-4)(a). We respectively define VP_{CO1} (Img(k)) and VP_{CO2} (Img(k)) to be the vertical projection of line segmentation $CO1$ and $CO2$ at image Img(k). Thus, we have

$$\text{VP}_{x_{peak}}(\text{CDI}(k)) = \text{VP}_{CO1}(\text{Img}(k)) + \text{VP}_{CO2}(\text{Img}(k)) \tag{5.13}$$

$$\text{VP}_{x_{col_1}}(\text{CDI}(k)) = \text{VP}_{LO1}(\text{Img}(k)) + \text{VP}_{LO2}(\text{Img}(k)) \tag{5.14}$$

For uniform pixel intensity, the projection of a column is proportional to the column length. As in Equation (5.9), the vertical projection for Img(k) decreases monotonically, thus we have

$$\text{Length(CO1 + CO2 +CI) } \geq \text{ Length(LO1 + LI + LO2)} \tag{5.15}$$

Since Column at col_1 is still at the right side of the axis in Img$(k-1)$ which is the left blue dashed-line as shown in Figure (5-4)(a), we have:

$$\text{Length(LI) } \geq \text{ Length(CI)} \tag{5.16}$$

Combining Equation (5.15) and Equation (5.16), we have

$$\text{Length(CO1+CO2) } \geq \text{ Length(LO1+LO2)} \tag{5.17}$$

Combining Equation (5.13), (5.14) and (5.17), we can show that CD-Images (Def_1) still peak at the VP-peak for original Img(k) as below:

$$\begin{aligned} \text{VP}_{x_{peak}}\text{ (CDI}(k)) &= \text{VP}_{CO1}\text{ (Img}(k)) + \text{ VP}_{CO2}\text{ (Img}(k)) \\ &\geq \text{VP}_{LO1}\text{ (Img}(k)) + \text{ VP}_{LO2}\text{ (Img}(k)) = \text{ VP}_{x_{col_1}}\text{ (CDI}(k)) \end{aligned} \tag{5.18}$$

Proof 2C: When foregrounds have randomly distributed intensity with same mean, we adopt CDI with the first definition

For pixel intensity randomly distributed around the same mean, the projection of an column is statistically proportional to the column length. Thus the column length would also decrease monotonically from the center to both ends, and we can make the similar observation shown in Equation (5.16) and (5.17) as long as col_1 is still at the right side of the axis in Img$(k-1)$, the left blue dashed-line as shown in Figure (5-4)(b).

From Figure (5-4)(b), we have

$$\text{VP}_{x_{peak}}\text{ (CDI}(k)) = \text{ VP}_{CO1}\text{ (Img}(k)) + \text{ VP}_{CO2}\text{ (Img}(k)) + \text{ VP}_{CI}\text{ (CDI}(k)) \tag{5.19}$$

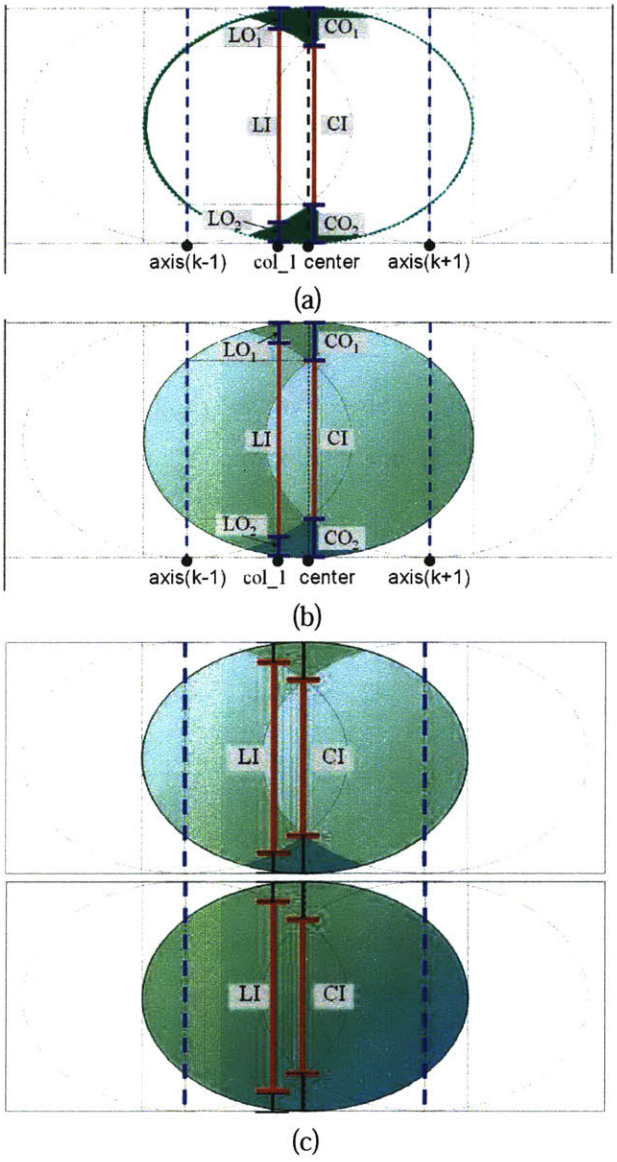


Figure 5-4: (a) CDI (1st Def) for uniform images (b) CDI (1st Def) for general cases (c) The comparison of CDI images and original frames.

$$\mathrm{VP}_{x_{col_1}}(\mathrm{CDI}(k)) = \mathrm{VP}_{LO1}(\mathrm{Img}(k)) + \mathrm{VP}_{LO2}(\mathrm{Img}(k)) + \mathrm{VP}_{LI}(\mathrm{CDI}(k)) \quad (5.20)$$

If we have

$$\mathrm{VP}_{CO1+CO2}(\mathrm{Img}(k)) - \mathrm{VP}_{LO1+LO2}(\mathrm{Img}(k)) \geq \mathrm{VP}_{LI}(\mathrm{CDI}(k)) - \mathrm{VP}_{CI}(\mathrm{CDI}(k)) \quad (5.21)$$

Then we can claim that

$$\mathrm{VP}_{x_{peak}}(\mathrm{CDI}(k)) \geq \mathrm{VP}_{x_{col_1}}(\mathrm{CDI}(k)) \quad (5.22)$$

The monotonic decreasing feature of vertical projection for original images shown in Equation (5.9) leads to the following relationship:

$$\mathrm{VP}_{CO1+CO2}(\mathrm{Img}(k)) + \mathrm{VP}_{CI}(\mathrm{Img}(k)) \geq \mathrm{VP}_{LO1+LO2}(\mathrm{Img}(k)) + \mathrm{VP}_{LI}(\mathrm{Img}(k)) \quad (5.23)$$

which can be also written as

$$\mathrm{VP}_{CO1+CO2}(\mathrm{Img}(k)) - \mathrm{VP}_{LO1+LO2}(\mathrm{Img}(k)) \geq \mathrm{VP}_{LI}(\mathrm{Img}(k)) - \mathrm{VP}_{CI}(\mathrm{Img}(k)) \quad (5.24)$$

Comparing the above equation and Equation (5.21), in order to make Equation (5.22) holds, the following relationship should meet:

$$\mathrm{VP}_{LI}(\mathrm{Img}(k)) - \mathrm{VP}_{CI}(\mathrm{Img}(k)) \geq \mathrm{VP}_{LI}(\mathrm{CDI}(k)) - \mathrm{VP}_{CI}(\mathrm{CDI}(k)) \quad (5.25)$$

In other words, the differences between vertical projection for line (LI) and line (CI) in CDI images, as demonstrated in the top part of Figure (5-4)(c), should be smaller than the differences between vertical projection for line (LI) and line (CI) in original images, as demonstrated in the bottom part of Figure (5-4)(c). When col_1 is at the right side of the

axis in Img$(k-1)$, the left blue dashed-line in Figure (5-4)(b), LI length is larger than CI length as shown in Equation (5.16).Most foreground' column pixels have similar intensities values as their neighborhood, so the intensity values in difference images FDI/BDI/CDI are much smaller than in the original images for most general situations. Thus, Equation (5.25) will hold for the general situations, which leads to Equation (5.22). When col_1 moves to the left side of the axis for Img$(k-1)$, LI length is smaller than CI length, Equation (5.21) will be easily met, which also leads to Equation (5.22).

In summary, when we do not have information about the foreground pixel, we can only prove these above inequalities in the statistical sense. Due to the similarity of foreground pixels with their neighborhood, most situations are similar to the case shown in Figure (5-4)(a), for which we have rigorous proof. Thus, we can claim an additional property for CDI-VP-Curves as followed:

- For non-occluded foregrounds, the horizontal locations of VP-peak for the original images Img(k) correspond to VP-peak for CD-Images.

We run a simulation experiment to demonstrate the above relationship numerically. We test three continuous images of ellipse foreground with two different image patterns and twelve different movement speeds as shown in Figure (5-5). Three original frames are labeled as Img$(k-1)$, Img(k), and Img$(k+1)$. In all images, black regions correspond to 0 intensity, and the brighter the region, the higher the intensity. For the cases (a1)-(a12) in two most left columns, pixel intensities inside foregrounds with ellipse-shape are random variables with uniform distribution. For the cases (b1)-(b12) in two most right columns, pixel intensities at every column of ellipse-foregrounds are random variables with uniform distribution while the vertical means of pixel intensities for each column decrease monotonically from the center of an ellipse to both ends. From case (a1) to case (a12), and from case (b1) to case (b12), the movement speed for an ellipse changes from slow to fast, and the relative movement of foregrounds between two continuous frames becomes larger and larger.

Figure (5-6) shows the changing trend of CD-Images and CDI-VP-curves (Def_1) from case (a1) to case (a6) and from case (b1) to case (b6). For each subfigure, CD Images

are shown in the top part, and the corresponding CDI-VP curves are the red curves in the bottom part for the sub-figure. The curves in blue, cyan, and green colors respectively correspond to the vertical projections for three original continuous frames, BDI and FDI. For all cases, CDI-VP-Curves have peaks at the center axis of the ellipse in the middle frame Img(k). When the relative motion is very small, CDI-VP-Curves are flat. When foregrounds move faster and faster, peaks in the CDI-VP-Curves become sharper and sharper, and higher and higher until the heights of the VP-peak for both definitions of CDI-VP-Curves are the same as the ones for the VP-peak for Img(k) as shown for cases (a6) and (b6), which is consistent with our proof 2B and proof 2C.

We also plot the two definitions of CDI-VP-curves for cases (a6)-(a12), (b6)-(b12) in Figure (5-8) when the relative motion is equal or larger than half of foreground's width. The results are consistent with our proof 1. For both definitions of CDI-VP-curves, Def_1 and Def_2, the VP-peak columns for CDI are the same as for the middle frame Img(k), and the heights of the VP-peak for CDI are the same as the one for Img(k). Therefore, we need not plot CD-Images for those cases.

Discussion on Two Definitions of CDI and CDI-VPC

Figure (5-7) and Figure (5-8) respectively compare two definitions of CDI-VP-curves for cases (a1)(b1)-(a6)(b6) and (a6)(b6)-(a12)(b12). The comparison between two definitions shows that the transitions from zero to non-zero projection on CDI-VPC for Def_2 are crisper than for Def_1 as shown in Figure (5-7) and Figure (5-8). The second definition of CDI and CDI-VPC is more robust to the choices of brightness thresholds, and has more advantages than the first definition as for the robustness of horizontal segmentation.

Besides, CDIs in Def_2 have richer texture information than the ones in Def_1. Two CD-Images of these two given definitions are shown in the bottom two rows of Figure (5-3)(b). The second definition contains brighter pixels than the first definition based on minimization. Such feature helps to improve the performance of future vertical segmentation.

One possible disadvantage for Def_2 is that CDI-VP-curves of Def_2 would not preserve the correspondence relationship between VP-peaks for CDI and Img(k) when the relative motion of foregrounds between two continuous frames is smaller than half of fore-

ground's width. The deviations of VP-peaks for CDI in Def_2 from the center axis of Img(k) are shown in Figure (5-7) for cases (a1)-(a5), (b1)-(b5). But such deviations are not very large since the relative motion is small.

Our results show that CDI-VPC in Def_1 works in all situations discussed in Proof 1, 2A, 2B, and 2C, while CDI-VPC in Def_2 only works in situations discussed in Proof 1 and 2A. CDI-VPC in Def_1 serves in more situations than CDI-VPC in Def_2 does. However, for situations when both Def_1 and Def_2 work, horizontal segmentation based on CDI-VPC in Def_2 is more robust than the one based on CDI-VPC in Def_1. Proof 2A requires that the relative motion between pedestrians in two continuous frames is larger than half of body-trunk, which is met for most real time monitoring videos. Therefore, we use CDI-VPC in Def_2 for our pedestrian tracking.

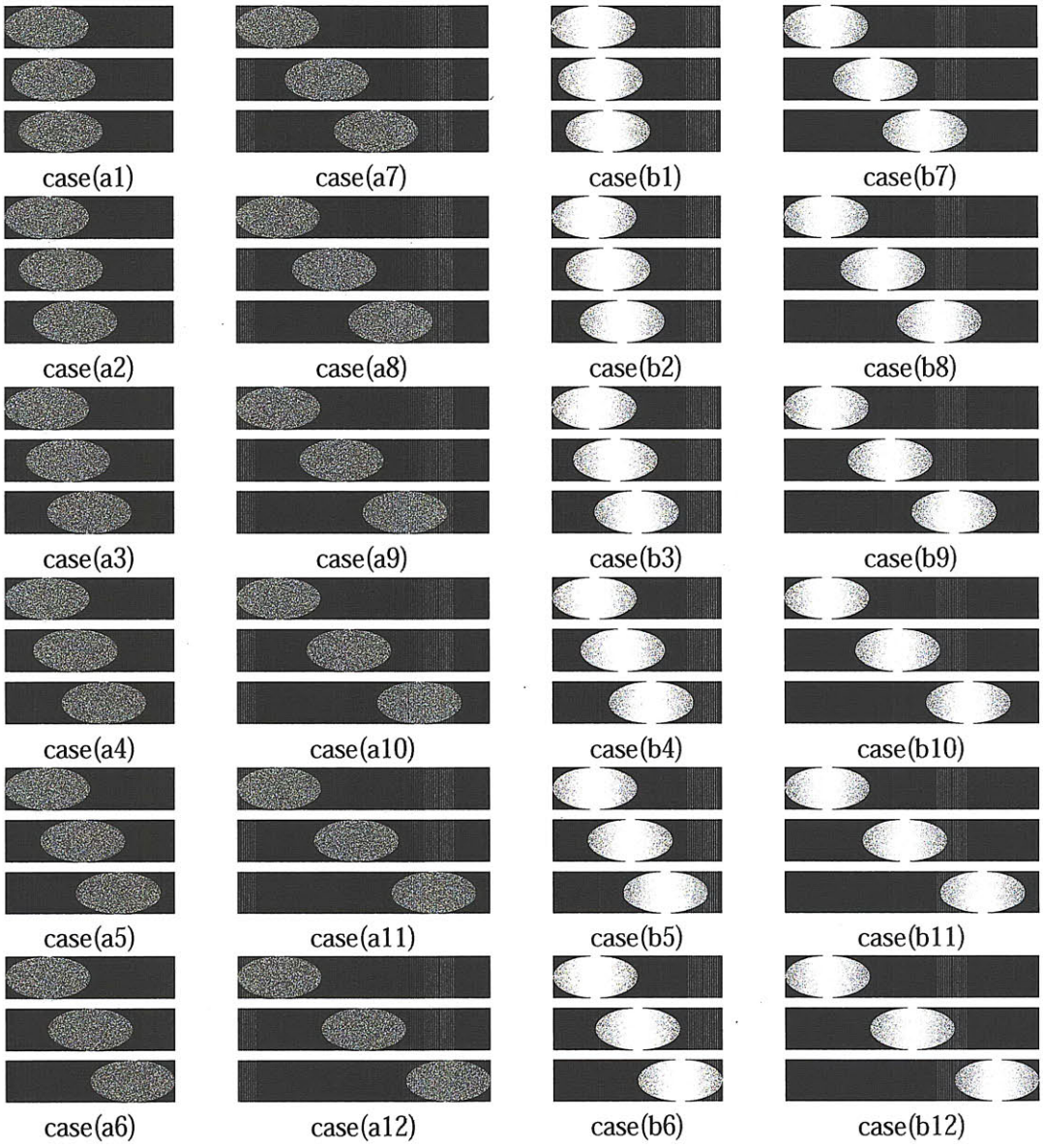


Figure 5-5: The three continuous video frames with two different image patterns(a/b) and 12 different movement speeds. From (1) to (12), the relative motion increases.

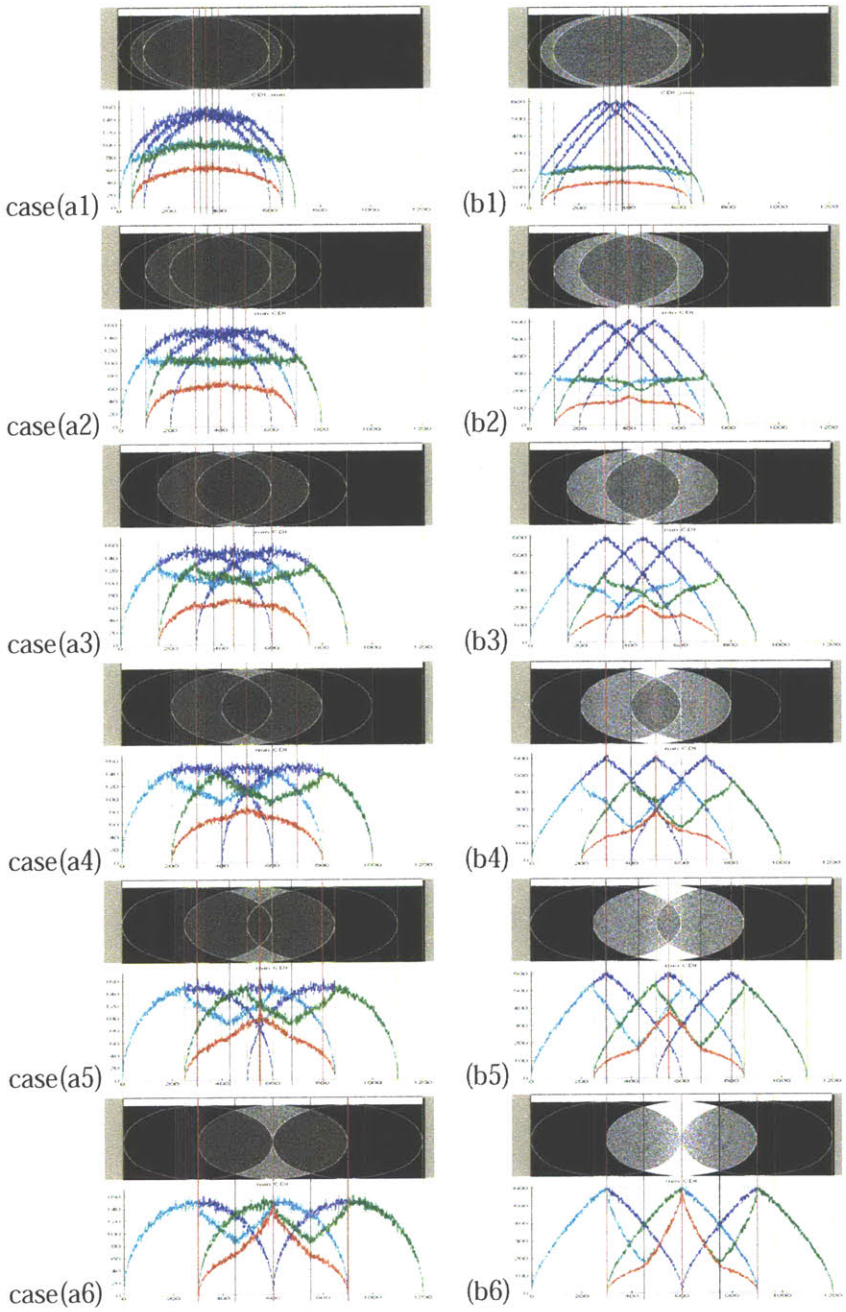


Figure 5-6: The changing trend of CD-Images and CDI-VP-curves(Def_1) for three continuous video frames defined in cases (a1)-(a6), (b1)-(b6) of Figure (5-5). CD-Images are plotted in the top row of each figure. CDI-VP curves are plotted in the red curve is the bottom row for each figure. In the bottom rows, the curves in blue, cyan, and green colors respectively correspond to the vertical projections for three original continuous frames, for BDI, FDI.

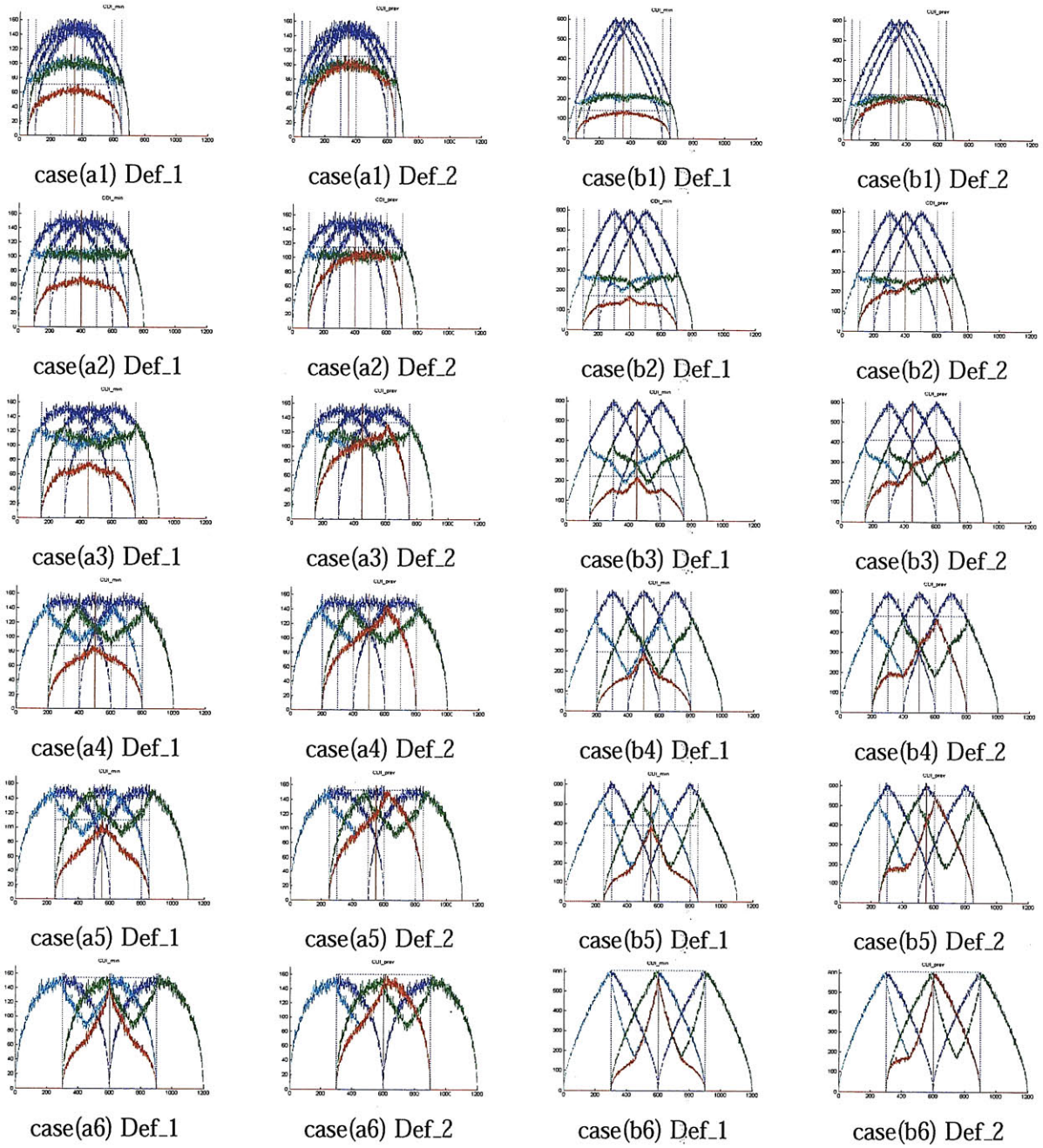


Figure 5-7: The comparison of two definition of CDI-VP-curves for three continuous video frames defined in cases (a1)-(a6), (b1)-(b6) of Figure (5-5).

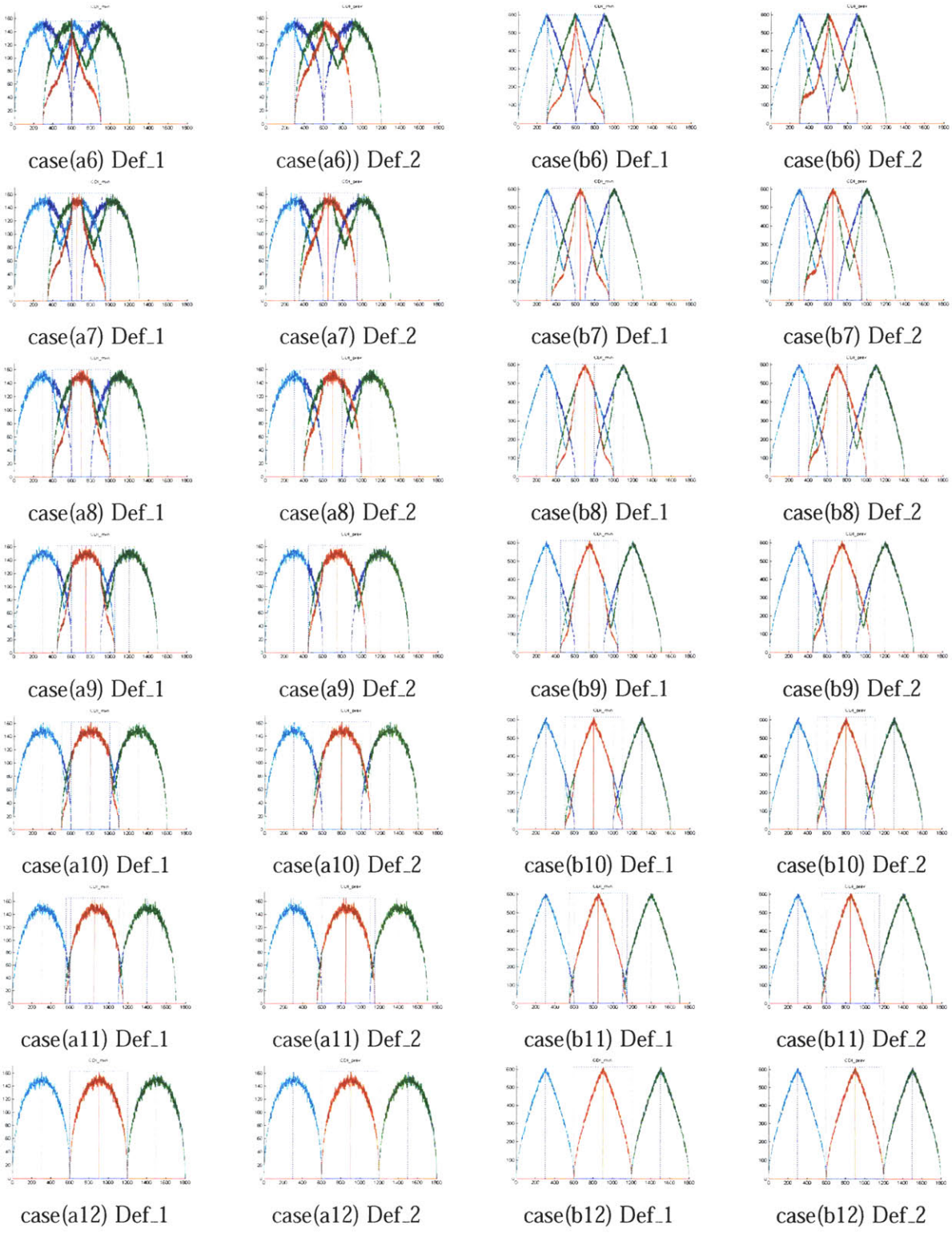


Figure 5-8: The comparison of two definition of CDI-VP-curves for three continuous video frames defined in cases (a6)-(a12), (b6)-(b12) of Figure (5-5)

The Properties of CDI-VPC during Occlusion

When occlusion happens, there still exists correspondence relationship between CDI-VPC and foregrounds as shown in Figure (5-9) in which the top row contains original image Img$(k-1)$, Img(k), and Img$(k+1)$ and bottom row plots their corresponding projection curves, CDI-VPC$(k-1)$, CDI-VPC(k), and CDI-VPC$(k+1)$. (We do not include the two other frames Img$(k-2)$ and Img$(k+2)$ that are respectively needed to produce CDI-VPC$(k-1)$ CDI-VPC$(k+1)$.) Similar to non-occlusion case, the transition from zero to non-zero projection curves happens at the *left* boundary of the occlusion-grouped pedestrians. The transition from non-zero to zero projection curves happens at the *right* boundary of the occlusion-grouped pedestrians. The horizontal locations of pedestrians' heads still correspond to the local maximum or transitional peaks even though they do not necessarily correspond to global peaks. Such correspondence also helps to narrow the searching area of interested objects. Thus, for occluded situations, CDI-VP-Curves still provide us information about the horizontal locations of pedestrians. More examples about the correspondence between pedestrians' locations and their corresponding CDI-VPCs before, during and after occlusion can be seen in Figure (5-10). Figure (5-10) shows the changing trends of vertical projection-curves of CD-Images before and after two people intersect. Two individual peaks in CDI-VP-Curves correspond to two pedestrians. As pedestrians walk close to each other, they occlude and finally pass each other. The corresponding two CDI-VPC waves gradually move close, merge, and finally separate again. The peaks correlate to the horizontal locations of pedestrians during the whole process. The series of vertical projection-curves not only provide a good foundation for initial segmentation, but also provide important dynamic information for tracking.

Projection-based Horizontal Segmentation Algorithm

In summary, our "projection-based, horizontal first, vertical next" segmentation algorithm separates an image into several vertical stripes in the following steps as shown in Figure (5-11)(a) and (b1):

1) Compute the absolute difference-value between two continuous frames and obtain

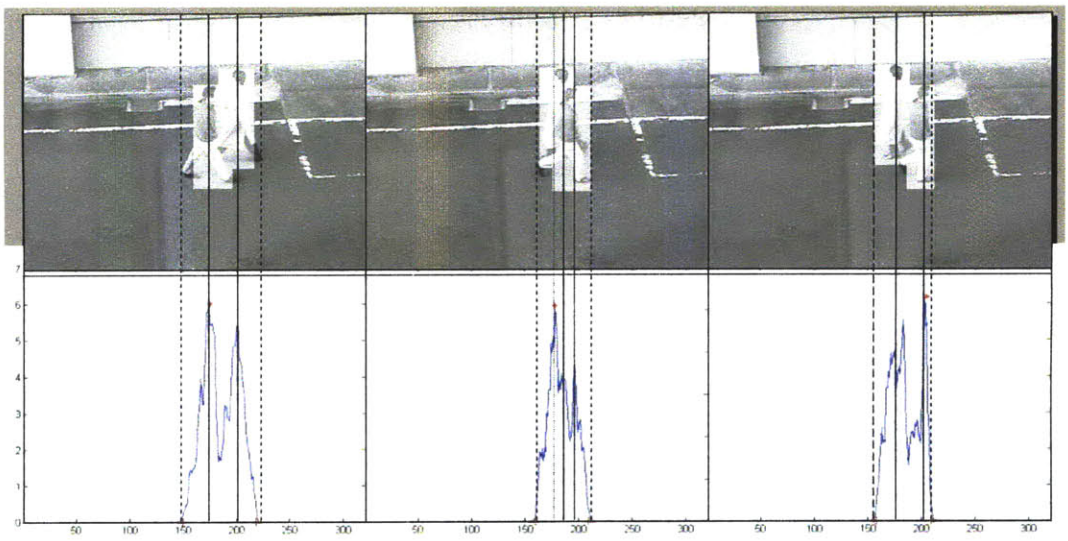


Figure 5-9: Examples that the horizontal locations of pedestrians correspond to transition peaks in projection-curves. Dotted lines correspond to the boundaries of intersection regions. Solid lines correspond to the locations of pedestrians' heads. We notice that the horizontal locations of people's head are exactly at the transitional peaks of the projection-curves, and the horizontal locations of merged regions at accurately determined by the peak boundaries of projection-curves.

BDI and FDI. Adaptively apply thresholds to BDI and FDI to obtain binary backward/forward difference images.

2) Compute CDI based on BDI, FDI and their binary versions. Compute the vertical projections of CDI to obtain CDI-VPI.

3) Detect triangle spikes (local maxima) to capture the transitions in CDI-VPI. Merge neighbor spikes when necessary. Adaptively adjust minimum wave-height thresholds based on wave-heights in previous frames in order not to miss objects.

4) Separate an original image into several vertical stripes and then obtain the initial horizontal segmentation.

For non-occlusion cases, the horizontal locations of the triangle spikes provide the ideal widths and horizontal locations of individual foregrounds. The horizontal locations of the triangle peaks on CDI-VPCs provide the horizontal locations of potential moving pedestrians' heads.

For occlusion cases, the horizontal locations of the triangle spikes still provide the hor-

izontal boundaries of grouped foregrounds. The horizontal locations of local transition peaks on CDI-VPCs provide the horizontal locations of candidate pedestrians' heads as shown in Figure (5-9).

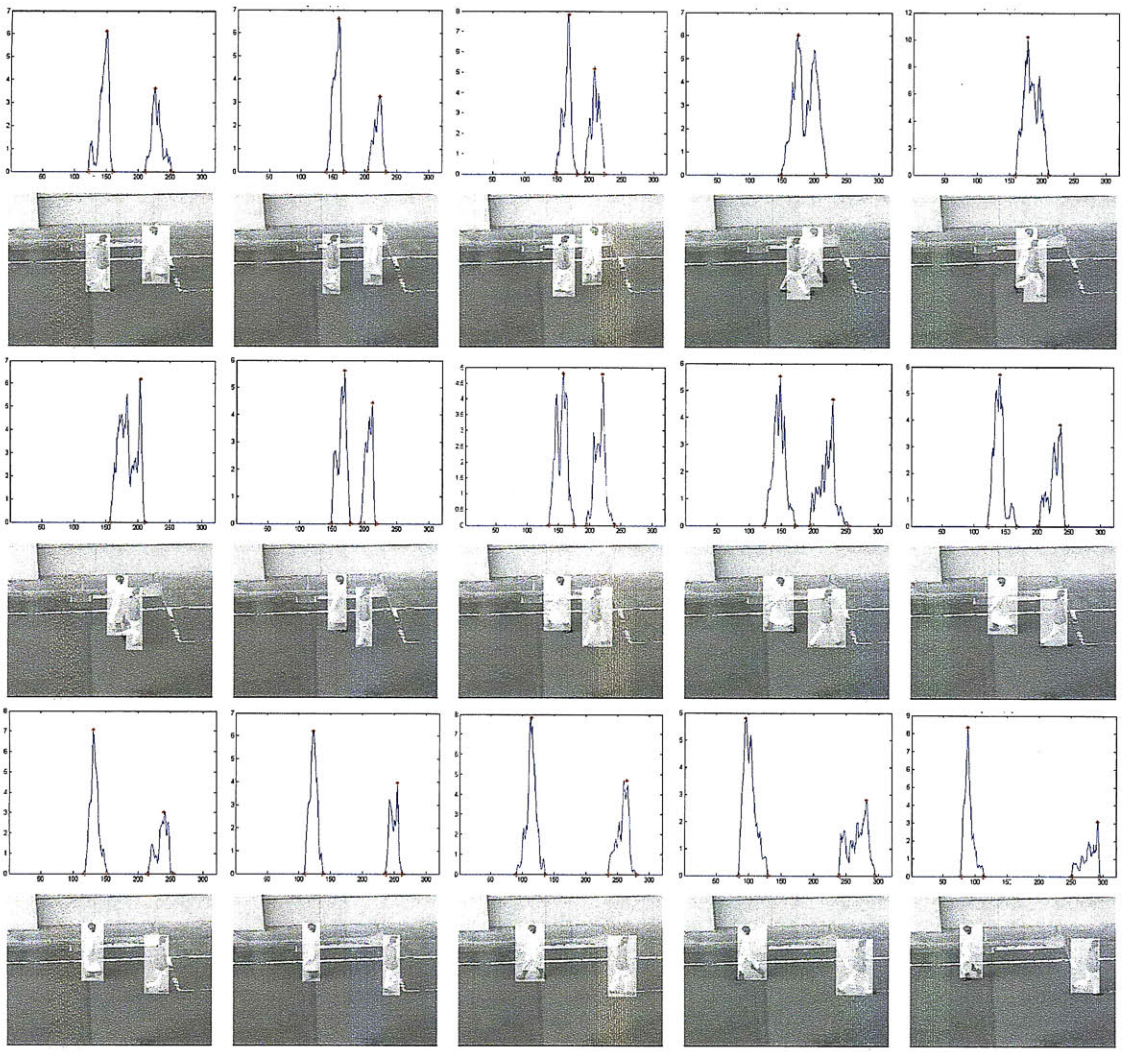


Figure 5-10: The changing trend of CDI-VP-Curves and the results of segmentation and tracking before and after two people intersected for Sequence VID_1.

5.2.2 Initial Vertical Segmentation within Horizontal Segmented Stripes

After horizontal layer separation, the vertical positions of human beings can be directly obtained by finding the highest and lowest positions of all edge pixels in CD-Images within

horizontally segmented stripes as shown in Figure (5-3)(c). For non-occlusion cases, our initial segmentation algorithm provides the vertical locations of pedestrians. Normally the highest points in the segmented regions correspond to the heads' locations and should be at the same column where CDI-VPC peaks.

For occlusion cases, the results of vertical segmentation are the vertical locations of pedestrians' group. The detail segmentation for individual pedestrians within grouped regions will be identified during fusion-based searching and tracking discussed in Section 5.3. Normally there is no occlusion when pedestrians first show up so that we have the correct segmentation information for each pedestrian. If there is serious occlusion when several pedestrians first show up, i.e., they walk in a group, the initial segment algorithm will treat them as one group. When pedestrians later separate, the individual pedestrian can be later identified and added when we update segmentation in tracking process. If pedestrians are together all the time, they will be treated as one moving group during tracking.

5.3 Fusion-based Layered-based Tracking

Initial segmentation provides possible locations of interested foregrounds or the grouped regions for occluded foregrounds. Because of light computational load, we can apply "horizontal first, vertical second" initial segmentation to every frame so that our detection system can quickly respond to fast changing environments.

During the tracking procedure, newly identified pedestrians should be matched to previous detected results to determine the dynamics of interested foregrounds. If occlusion happens, the detail locations of interested foregrounds in the grouped regions should be further identified. In the tracking process, we fuse the following information from multiple resources as shown in Figure (5-11)(a). The steps are:

1. Conduct initial segmentation for the current frame
2. Obtain CD-Images, the edge of CD-Images, and the vertical-projection-curves for CD-Images(CDI-VPC)
3. Estimate horizontal locations and velocity of pedestrians from KF prediction step

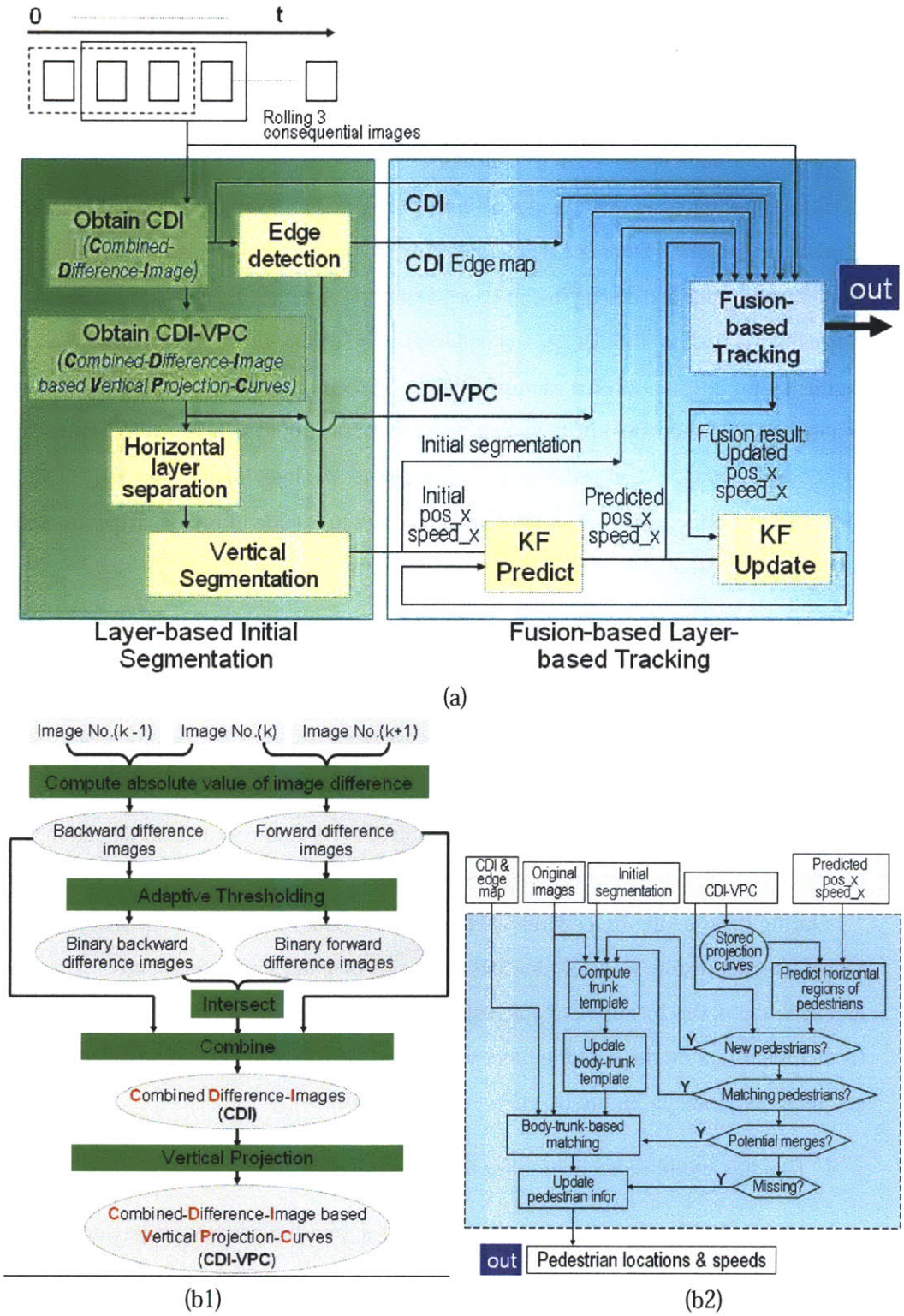


Figure 5-11: (a) Algorithm flowchart. (b1) Flowchart for CDI, CDI-VPC computation. (b2) Flowchart for fusion-based tracking.

Before explaining our fusion-based tracking algorithm, we first define several features or elements involved in the tracking process.

5.3.1 Template with different poses

One key point to track moving pedestrians with various poses is to identify the common signatures among pedestrians in different frames. Though people change their poses during walking, their body-trunks regions change little between two continuous frames and can be used to match pedestrians among different frames. The similarity of body-trunk is shown in Figure (5-12) in which the left column shows the detected pedestrian regions during initial segmentation, and the right column shows the body-trunk regions for these people. In four continuous frames, the pedestrian in the left side of the images have very similar body-trunks.

Given initial segmentation, corresponding body-trunk regions are defined in the following steps:

1) We first obtain the heads' locations which are the top row of initial segmentation and at the column where CDI-VPC peaks. When there is no occlusion, there is very little noise in the combined-difference-images and the heads' location can be accurately determined. When there is occlusion where heads' location might not be accurately detected, we do not update the template. So the negative impact would not be accumulated during the tracking process.

2) In areas surrounding heads' locations, we define a portion of initial segmented regions whose width and height are respectively fifty percent (50%) and seventy percent(70%) of initial segmented regions.

For the purpose of pose-independent body-trunk based matching, two body-trunk regions are first aligned based on their VP-peaks. Then, the similarity comparison between two pedestrians are computed based on the cross correlation for the common area of two corresponding body-trunk regions. The human body-trunk templates will be used as one input for fusion-based tracking in the next step.

We only apply pose-independent body-trunk based matching when pedestrians com-

pletely show up in both frames. For the pedestrian who is walking into the image from the right side in Figure (5-12), their initial segmentation regions are not related based on body-trunk. Instead, we can recognize the action of having a new foreground based on the increasing area of initial segmentation at the right boundary of the image.

5.3.2 Layered-based Tracking

Similar to the layer-based initial segmentation, our tracking mechanism is also layer-based. While traditional tracking algorithms search for previous detected foregrounds in a whole image, our algorithm limits the searching within the areas defined through initial "horizontal first, vertical second" segmentation for each individual frame. Figure (5-13) shows the tracking and matching process between two continuous frames, Img$(k-1)$ and Img(k). The top and bottom rows of Figure (5-13) respectively correspond to Img$(k-1)$ and Img(k). For both Figure (5-13)(a) and (b), the right columns show the original image texture, and the left columns show the CDI-VP-curves for two continuous frames. In order to track pedestrians in Img$(k-1)$, we only search among possible regions of body-trunk regions in Img(k) as indicated by blue areas and compare the similarity of the corresponding regions with previous pedestrian body-trunk templates. The detail process of determining these candidate searching regions is explained below.

For non-occluded cases as in Figure (5-13)(a), searching regions are the body-trunk regions corresponding to initial segmentation regions. We first determine heads' locations, the highest points in the columns where isolated CDI-VPC for Img(k) peaks. We then define a small portion of initial segmentation regions around heads' locations to be the candidate body-trunk regions according to Section 5.3.1.

For occluded cases as in Figure (5-13)(b), the horizontal locations for two pedestrians' heads are at transitional peaks of the vertical projections according to the properties of CDI-VPCs described in Figure (5-9) (see Section 5.2.1). Again, the highest points in the columns corresponding to these CDI-VPC local transitional peaks are treated as heads' locations. Different from non-occluded regions, we cannot use the newly identified initial segmentation regions to define candidate body-trunks. We reuse the size from prior

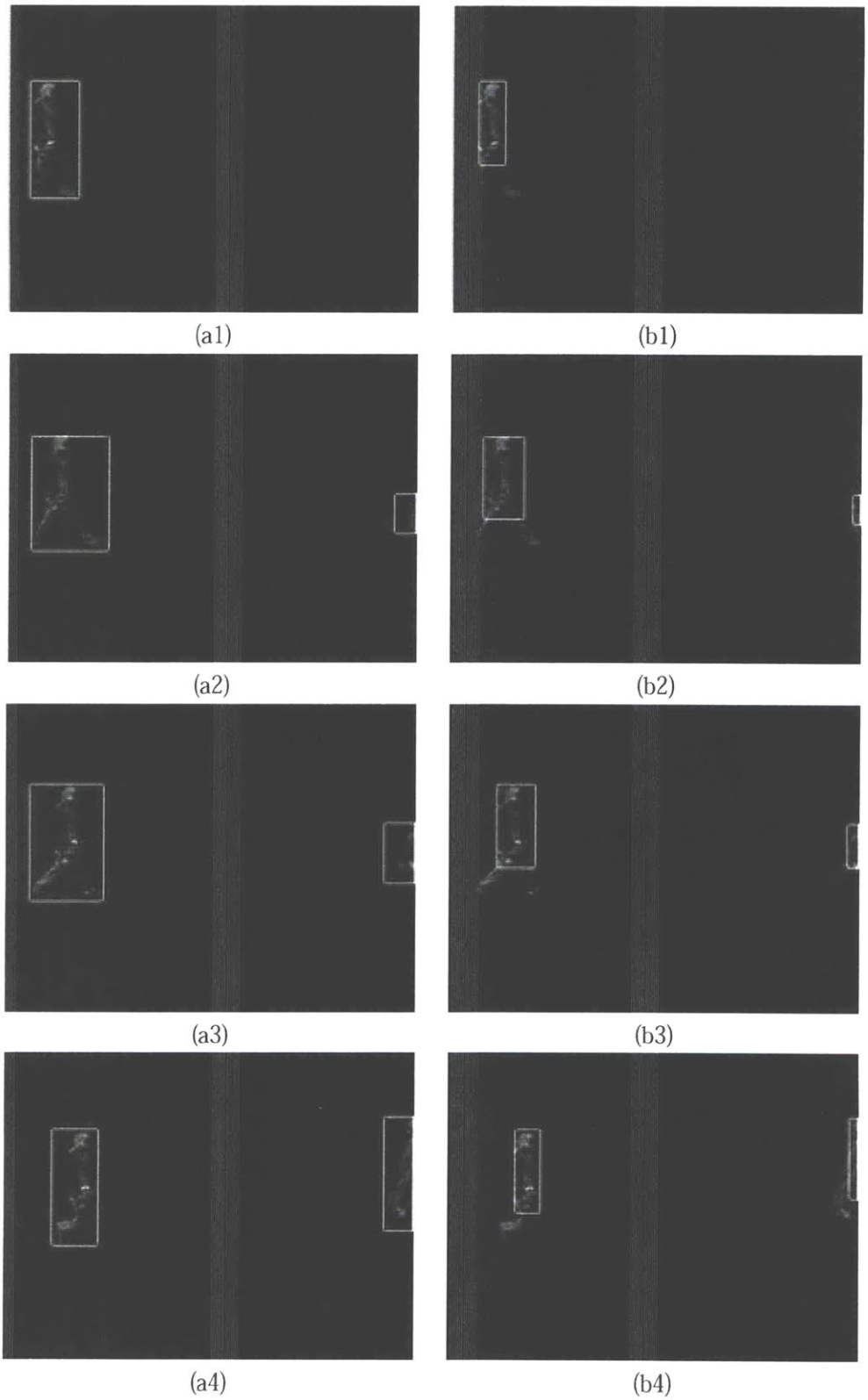


Figure 5-12: The similarity of body-trunk signature for pedestrians with different poses. (a) Segmentation for whole pedestrians. (b) Body-trunk regions.

body-trunk templates and define regions with the same template size surrounding heads' locations as candidate body-trunk during intersection(occlusion).

We further consider the motion and speed information from the predicted tracking models and limit the search ranges to neighboring peaks as marked. Such process and more details for non-occluded and occluded situations will be respectively discussed later in Section 5.3.4 and Section 5.3.4.

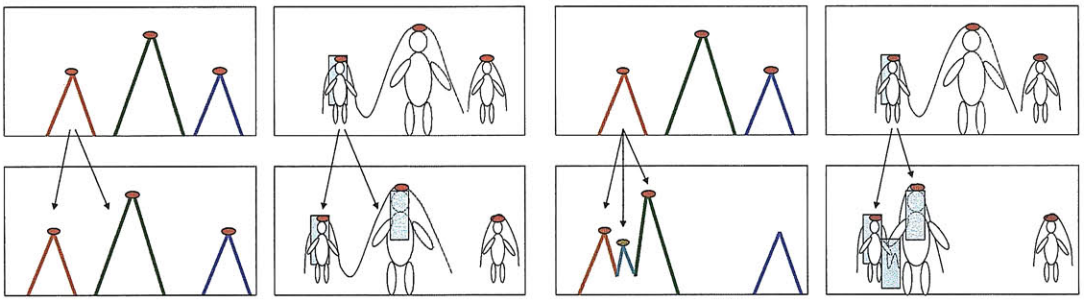


Figure 5-13: The process of pedestrian tracking (a) For non-occluded situations where CDI-VPC curves are separated waves. (b) For non-occluded situations where CDI-VPC curves are merged waves. Top row: For Img$(k-1)$. Bottom row: For Img(k).Left part of (a) and (b): CDI-VPC. Right part of (a) and (b): Two continuous image frames.

5.3.3 Prediction Models

To keep track of the dynamics of N pedestrians' movement, our tracking algorithm defines N Kalman filter (KF) models whose states include the horizontal position and horizontal velocity for each pedestrian. Traditional tracking algorithms pick both horizontal and vertical positions and velocities as tracking parameters. We focus only on the horizontal positions and velocities due to the following three reasons. First, our segmentation and detection algorithms are layered-based and follow the principles of "horizontal first, vertical second." As long as the horizontal locations can be determined, the vertical positions can be easily detected in non-occluded situations. Even if for occluded situations, we still can determine several candidate horizontal locations. Secondly, the correspondence relationship between CDI-VPC peaks and heads' horizontal locations makes it much easier and more accurate to identify the horizontal locations than the vertical locations, especially

during the occlusion situations. Thirdly, when people walk, the vertical positions change much rapidly than horizontal positions. In other words, they would be very "bumpy." It is much easier to accurately track the horizontal locations than the vertical locations. Thus, it is computational efficient for our mathematical tracking models to only track the horizontal locations and velocities.

The detailed models are described in Equation (5.26). The ith KF model states $\mathbf{x}$ include the horizontal position (pos_x) and horizontal velocity (v_x) for ith pedestrian, i.e, $\mathbf{x} = [pos_x \ \ v_x]$. For the model defined below, u is input variable, z is the measurement, w and v are respectively process noise and measurement noise, Q and R are respectively the covariance of process noise and measurement noise.

$$\begin{aligned} \mathbf{x}_k &= A\mathbf{x}_{k-1} + Bu_{k-1} + w_{k-1} \\ z_k &= H\mathbf{x}_k + v_k \\ p(w) &- N(0, Q) \\ p(v) &- N(0, R) \end{aligned} \tag{5.26}$$

Since we have the relationship $pos_x[k] = pos_x[k-1] + v_x[k-1]$ and $v_x[k] = v_x[k-1]$, we set $B = [0; 0]$, $H = [1 \ \ 0]$, $A = [1 \ \ 1; 0 \ \ 1]$.

Kalman filtering involves two steps: *estimation* and *update*, which are respectively described in Equation (5.27) and Equation (5.28). In the *estimation* step, we estimate horizontal locations and velocities for pedestrians $\hat{\mathbf{x}}_k^-$, and the error covariance P_k^- in the next frame based on historical data from previous frame.

$$\begin{aligned} \hat{\mathbf{x}}_k^- &= A\hat{\mathbf{x}}_{k-1} + Bu_{k-1} \\ P_k^- &= AP_{k-1}A^T + Q \end{aligned} \tag{5.27}$$

In the *update* step, the measured horizontal locations and velocity of previously identified pedestrians are used to update the gain K_k, measurement $\hat{\mathbf{x}}_k$, and error covariance P_k

of KF models as shown in Equation (5.28).

$$\begin{aligned} K_k &= P_k^- H^T (H P_k^- H^T + R)^{-1} \\ \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + K_k(z_k - H\hat{\mathbf{x}}_k^-) \\ P_k &= (I - K_k H) P_k^- \end{aligned} \tag{5.28}$$

The initial segmentation results, human locations, are used to initialize dynamic tracking models for human positions.

5.3.4 Fusion-based Tracking

As shown in the flowchart in Figure (5-11)(a), fusion-based tracking-block receives multiple information, CDI, CDI-VPC, initial segmentation and Kalman filter prediction results, in order to detect the new locations of monitored pedestrians and potential new pedestrians. Based on the horizontal locations of the triangle spikes from VP-Curves for in Img$(k-1)$, KF prediction model can predict the new horizontal regions for all pedestrians. Unlike traditional Kalman filtering where predicted results from Equation (5.27) are only intermediate results, the predicted results are used in fusion-based tracking block.

In this subsection, we first discuss how to detect the dynamics of tracking and to differentiate occluded and non-occluded situations, then propose how to track pedestrians in two situations.

Detect Possible Dynamics

By comparing the predicted locations and the horizontal locations from CDI-VP-Curves for Img(k), we can identify whether there exist independent sharp triangle spikes at the predicted regions, and detect possible intersection or new independent waves for new incoming pedestrians.

If estimated regions for pedestrians at Img(k) do not overlap, and the initial segmentation results appear at the estimated regions, we can conclude that there are no occlusion.

If the predicted regions for two or more pedestrians start to overlap, triangle spikes fail to show up at estimated regions, or the number of independent non-overlapped CDI-VPC waves in that neighborhood decreases, we can conclude that the corresponding vertical projection waves might start to merge and the corresponding pedestrians might start to occlude partially.

If the number of independent non-overlapped CDI-VPC waves in that neighborhood decreases and the predicted regions start to reach the left/right boundaries, corresponding pedestrians might start to walk out monitoring regions and even disappear.

Pedestrian Tracking during Non-Occluded Situations

Where there is no occlusion, we can directly apply body-trunk-based matching to associate the previous pedestrian with newly segmented pedestrians as discussed in Section 5.3.2.

As shown in Figure (5-13)(a), three separated pedestrians in the right column lead to three isolated peaks as shown in the left column for both Img$(k-1)$ and Img(k), which will provide three corresponding search ranges in Img(k) in order to track the most left pedestrian in Img$(k-1)$. The location and velocity information from predicted tracking models help to limit the search ranges to two neighboring peaks as marked. If there are new additional triangle peaks that do not correspond to any previous pedestrians, the newly segmented pedestrians are recorded as new foregrounds. Both cases do not involve occlusion, and the initial segmentation is reliable. The new information of the tracked pedestrians is used to update dynamic KF models and stored body-trunk template.

Pedestrian Detection during Occluded Situations

When two originally independent triangle spikes start to merge, it means that two pedestrians move close and start to intersect (occlude). As shown in Figure (5-13)(b), two originally separated pedestrians in Img$(k-1)$ start to occlude. In order to track the most left pedestrian in Img$(k-1)$, body-trunk template in Img$(k-1)$ should be compared with candidate body-trunk regions in Img(k) which are defined at three local transitional peaks within interested section of CDI-VP-Curves as discussed in Section 5.3.2. Such three regions in Img(k) are marked in Figure (5-13)(b). The occluded cases in Figure (5-13)(b) have one

extra candidate region than in Figure (5-13)(a), which is caused by the extra peaks due to occlusion.

Our method might have problems when two people's heads are completely at the same horizontal locations, i.e., when one person is completely occluded by another. However, it is impossible for one person to be completely occluded by another in several frames unless they are static or both walk together. In that case, they are treated as one group in our method. In most occlusion situations, the horizontal locations for two pedestrians are at different transitional peaks during the whole merging process even though the images of two people might merge in several continuous frames.

If occlusion happens, we do not update KF dynamic models and pedestrian trunk templates as shown in Figure (5-11)(b2). This is because that segmentation reliability may degrade because of matching ambiguity when searching for head area among wave peaks. Interested pedestrians might not completely show up and human body-trunk area might include texture information from another person. During the occlusion process, we reuse the same templates before occlusion for matching and tracking to avoid the impact of incorrect measurement data on detection and on the performance of Kalman filter based dynamic models.

Our tracking algorithms keep track of transient process, such as, whether people are about to enter and to leave scene boundaries, and update body-trunk templates and tracking models accordingly.

5.4 Results

In this section, we will discuss tracking results for four picked sequence out of 26 test video sequences. As shown in Table 5.1, in Sequences VID_1, two people walk toward to each other, intersect, and then depart. In Sequences VID_2, one pedestrian gradually changes from standing poses to bowing poses. In Sequences VID_3, one pedestrian first walks to the right side of the image, then turns 90 degrees to left and walks toward the back. In Sequences VID_4, one pedestrian first walks backward and then forward, while the other one walks forward and pass another pedestrian. The intersection happens when the first

Table 5.1: Summarization of test video sequences and results

Seq num	Sequence situations	Result Figures
VID_1	Two walked toward each other	Fig. (5-9), Fig. (5-10), Fig. (5-17)
VID_2	Two walked toward each other. One later bowed.	Fig. (5-14), Fig. (5-18)
VID_3	Two walked toward each other. One later turned 90 degree to the left and changed the direction of walking.	Fig. (5-15), Fig. (5-19)
VID_4	One walked to the left. Another first walked backwar and then forward.	Fig. (5-16), Fig. (5-20)

pedestrian walks backward and the second one walks forward.

Fig. (5-10), Fig. (5-14), Fig. (5-15) and Fig. (5-16) respectively show the results of segmentation and tracking before and after two people intersected for Sequence VID_1, VID_2, VID_3, and VID_4. Our algorithm accurately captures two people even during the intersection/occlusion period. Figure (5-10) also shows the correspondence relationship between segmentation/tracking results and CDI-VP-Curves for sequence VID_1. Figure (5-9) shows the human detection details before and after two people intersect. Specifically, these two figures demonstrate that the horizontal locations of people's heads are exactly at the transitional peaks of the projection-curves, and the horizontal locations of merged regions are accurately determined by the boundaries of CDI-VPC waves, which is the foundation of our algorithms.

Figure (5-17), Figure (5-18), Figure (5-19), and Figure (5-20) respectively plot walking paths of pedestrians for the whole sequences and show the integrated tracking results for Sequence VID_1, VID_2, VID_3, and VID_4 . In these figures, square black dots present detail locations of each detected person's head in different image frames. One person's movement in a video sequence is then represented by a dotted line, which describes the moving paths of the pedestrians and demonstrates the performance of our segmenting/tracking method in a video sequence. The fixed background image is the average between the first and the last image of a video sequence. The image contains the initial and final appearance of pedes-

trians, which correspond to the starting and ending positions of the movement path for the corresponding pedestrians.

Our method can produce reliable tracking results even when people change poses or walking directions. For all these sequences, two people walk toward different directions, meet at some point in the middle of video sequences, intersect and occlude, and finally depart again. People change their directions from forward to backward (VID_4), or from forward to the left (VID_3). People changed their poses from up-straight (side-look) to bowing (VID_2), from the side appearance to back appearance (VID_3). Our algorithm accurately captures the changes in the detect pathway. In each sequence, the number of dotted lines is equal to the number of pedestrians, and the number of dot is equal to the number of frames in which pedestrians show up. The smooth flat dotted line reflects the smoothness of human movement. Figure (5-17), Figure (5-18), Figure (5-19), and Figure (5-20) demonstrate that our algorithms provide reliable and correct tracking results.

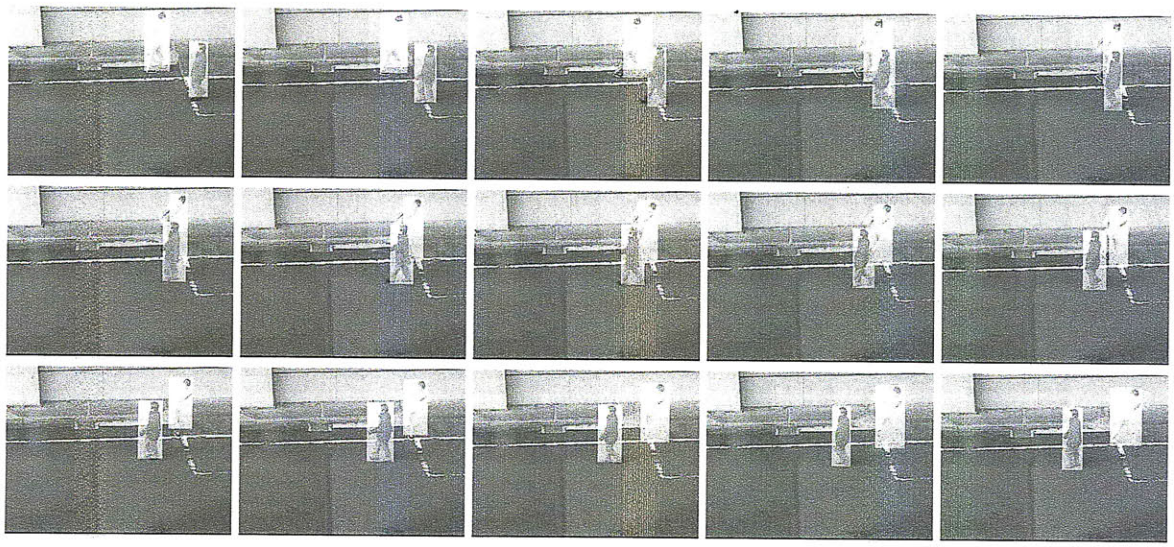


Figure 5-14: The segmentation/tracking results before and after two people intersected for Sequence VID_2. One person started to bow during walking.

5.4.1 Performance Evaluation

We have tested a total of 26 available video sequences in which there are two or more people walking toward different directions with different poses. Our goal is to track the

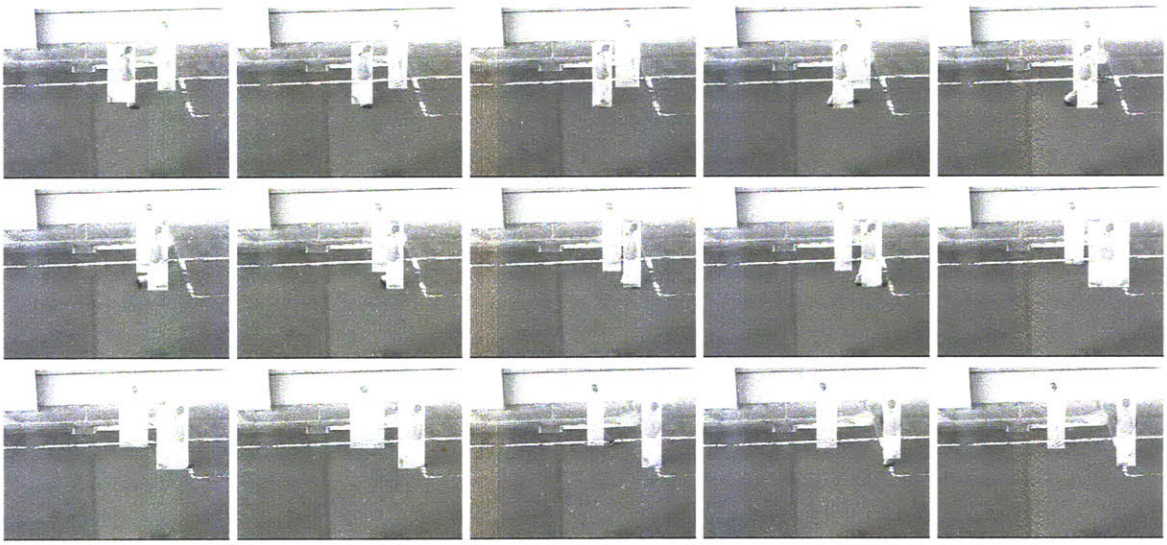


Figure 5-15: The segmentation and tracking results before and after two people intersected Sequence VID_3. One person turned 90 degree and changed the direction of walking.

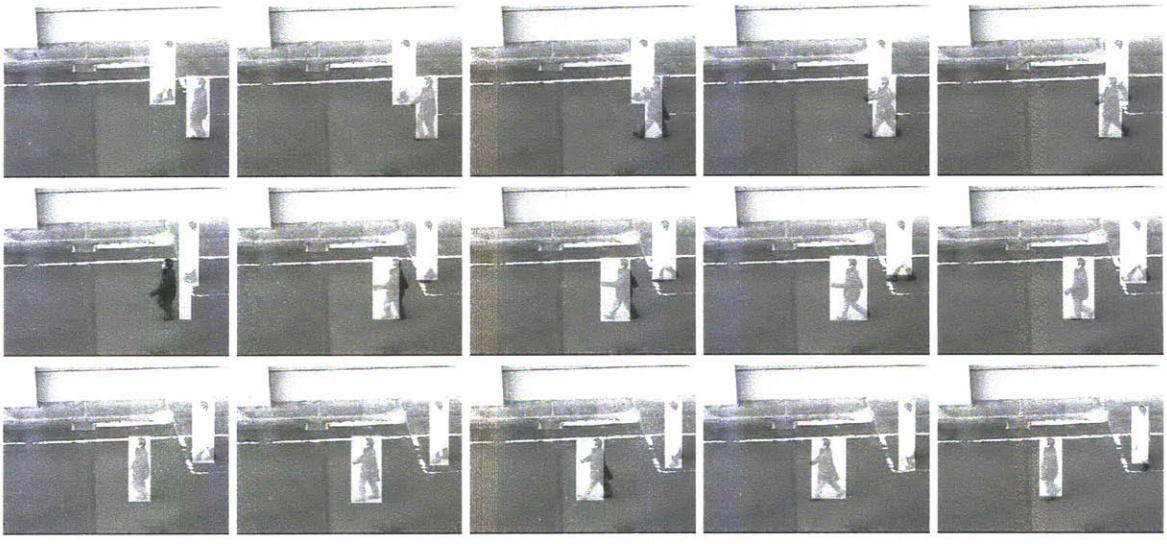


Figure 5-16: The segmentation and tracking results before and after two people intersected Sequence VID_4. One person first walked backward and then forward.

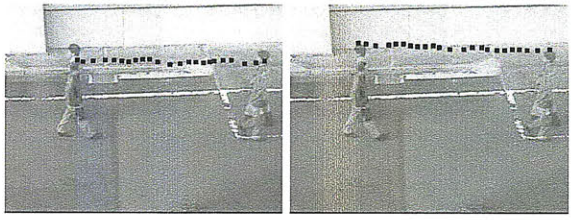


Figure 5-17: Tracking results for Sequence VID_1 in Figure (5-10). Left: Pedestrian 1. Right: Pedestrian 2.

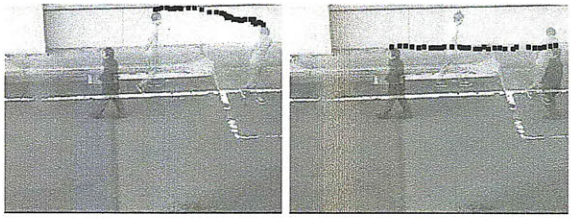


Figure 5-18: Tracking results Sequence VID_2 in Figure (5-14). Left: Pedestrian 1. Right: Pedestrian 2.

Figure 5-19: Tracking results Sequence VID_3 in Figure (5-15). Left: Pedestrian 1. Right: Pedestrian 2.

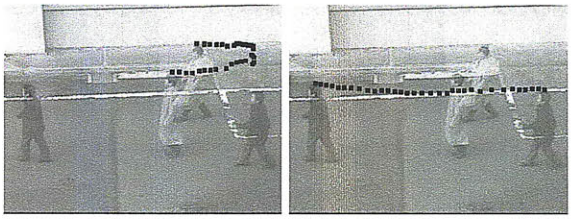


Figure 5-20: Tracking results Sequence VID_4 in Figure (5-16). Left: Pedestrian 1. Right: Pedestrian 2.

same person successfully and accurately all the time. A person can be mistakenly tracked or even lost after intersection. If a person is lost during the tracking process, and is later detected as a new person in the sequence, the total number of detected people will be larger than the actual number of people, and the motion path for a person is incorrect. If this happens, the motion path for related pedestrians would not be incorrect and a "failure" is assigned to the sequence. We claim "success" only when the motion path for a person is detected accurately with minor segmentation errors.

After applying our evaluation standard to all 26 test sequences, our test results indicate that 3 test sequence results are "failure" (around 12%) and there are 23 "success" test results (around 88%).

To compare the performance of our proposed algorithms with conventional methods, we applied conventional background-subtraction-based segmentation and mean-shift tracking algorithms to track pedestrians' behavior in the same 26 sequences. The rate of "success" is only 40%. Furthermore, the results based on conventional methods have to rely on the color information while our algorithm only requires grey level images for the same set of sequences.

Our detection results show that our algorithm has the capability to track humans with changing walking-poses, different walking directions, and to deal with segmentation with partial occlusions and scale variations of targets. Humans are accurately tracked even if intersecting or merging happens. Our proposed algorithm has significantly improved the detection reliability.

5.5 Discussion: the Features & Advantages of Proposed Algorithms

Our proposed layered-based and fusion-based algorithm has the following features and advantages.

5.5.1 The Advantage of Layered-based Processing

Both our segmentation and tracking algorithms follow the layered-based principle.

The Advantage of Layered-based Processing for Segmentation

We calculate CD-Images and CDI-VP-Curves to detect multiple irregular subjects. If there is only one object in an image, there exists unique correspondence relationship between the object's location and CD-Image's boundaries which helps to accurately locate the interested object. In order to detect multiple objects in an image, we divide the image into multiple regions using CDI-VPCs and then search for individual foreground in each region. Typically in an image one pedestrian would not directly at the top of another. Thus, if we can horizontally separate an image into several vertical stripes, we can uniquely determine the corresponding pedestrian's locations.

Therefore, we take advantage of defined CD-Images and CDI-VP-Curves to separate an image into several vertical stripes, and each vertical stripe contains one pedestrian. Then we search for the vertical locations of potential pedestrians. Our "CDI-VPC-based, horizontal first, vertical next" segmentation changes the original two-dimensional search of pedestrians within a whole image into two one-dimensional searching. Compared with traditional segmentation algorithms for 2D searching, our layered-based method significantly improves segmentation reliability and decreases the computational load.

The Advantage of Layered-based Processing for Tracking

After the "CDI-VPC-based, horizontal first, vertical next" segmentation, our dynamic models only need to track horizontal locations and velocities of pedestrians. Thus, the dynamic tracking model is simplified and but is still reliable enough to provide good estimation for future horizontal locations of interested pedestrians. In contrast, traditional complicated tracking models have to track shape, contour, and location information. The traditional methods have to track both horizontal and vertical locations of interested objects because of limited reliability and accuracy of vertical detection during occlusion situations.

Our tracking algorithm is also layered-based as discussed in Section 5.3.2. We divide

the original whole image into several candidate layers/regions and limit tracking/searching within these regions for candidate matching.

Besides, the CDI-VPC-based algorithm does not depend on human features, such as faces, skins, etc., ellipse shape, etc. Thus our algorithm can have more general applications than detection algorithms relying on special human features.

5.5.2 The Advantage of Fusion-based Principle

In order to enhance tracking accuracy, our fusion-based scheme takes advantages of multiple information, including, initial horizontal segmentation, predicted human locations, CD-Images and CDI-VP-Curves, and body-trunk templates.

Tracking based on Fusion of Initial Segmentation and Dynamic Tracking

Traditionally, pedestrian detection involves initial detection based on human's features from one frame and tracking for several frames afterward due to the heavy computational load associated with 2D segmentation and searching. Instead, our segmentation only does two 1-D searching whose computational load is very light so that we can afford to implement initial segmentation and to update body-trunk templates for every frame, which allows for more adaptability in situations with rapid pose changes.

We take advantage of the similarity of pedestrian trunk template to associate the detected pedestrians among different frames. Our initial segmentation provides satisfying segmentation for non-occluded pedestrians, which significantly decreases the matching ambiguity during tracking. The combination of segmentation and dynamic tracking does not need long initialization time, does not depend heavily on tracking, and can quickly respond if people dramatically change their poses. Our detection system is simple but effective in reducing the segmentation/tracking ambiguities when human intersects and occludes. Compared with traditional segmentation/tracking algorithms purely based on template matching, our proposed algorithm is simple, accurate and effective.

Tracking based on Fusion of CDI-VP-Curves, Historical Information and Tracking Models

The defined CDI-VP-Curves are not only useful when searching for horizontal locations of potential pedestrians during initial segmentation, but also helpful to decrease the searching regions when matching pedestrians among different frames. The candidate searching regions are determined by two factors, predicted regions from KF prediction, and isolated waves or the transitional local peaks in CDI-VPC curves.

We only need to compare human trunk template from previous frame with very few candidate regions corresponding to the triangular CDI-VPC waves within estimated location regions. For merging cases, the search regions for candidate pedestrians can be limited to transitional peaks within the merged CDI-VPC waves. Thus, the fusion-based tracking is very useful to identify individual pedestrians with various poses, including the cases when people bow (which is very rare in most monitoring situation) as shown in Figures (5-14) and (5-18).

The predicted information from KF models helps to identify the potential merging cases. When the reliability is low, tracking processes stop updating the template and dynamic models with new measurement data, which prevents the ambiguity and failure during occlusion to pollute the dynamic model and main or even improve estimation performance for unexpected situations. The mechanisms to update tracking information based on the reliability of segmentation and tracking improves the robustness of our systems, especially when dealing with serious tracking failure.

5.6 Conclusion

We have proposed a layered-based and fusion-based human detection and tracking system that can handle human intersection and conclusion in complicated environment. Our proposed algorithm applies layered-based initial segmentation for every frame based on "projection-based, horizontal first, vertical next" segmentation scheme. When occlusion occurs, our mechanism chooses potential head locations at local projection peaks within predicted regions. Converting a typical 2D search problem into two 1D search problems

significantly decreases computational loads.

Our fusion-based obstacle detection algorithm fuses the information from initial segmentation and dynamic tracking model to avoid complicated tracking schemes. Kalman-filter-based dynamic models are used to predict pedestrians' horizontal location and speed. The tracking process identifies and keeps track of detected pedestrians through fusing the information from the initial segmentation, CD-Images, CDI-VP-Curves, prediction results from KF filters, and historical information of pedestrians' body-trunk templates. The layered-based search mechanism tracks the movements of pedestrians at the candidate regions defined by transitional peaks in CDI-VP-Curves. The final detection results are used to update human locations in Kalman filters and human matching templates. The tracking mechanism stops updating in case of poor segmentation reliability.

In summary, our methodology takes advantage of connection between different information. Compared with other human detection methods, our layered-based fusion-based algorithm reduces both the computational load and matching-ambiguity. The simple segmentation and tracking algorithms are very effective and reliable in detecting and tracking the movements of pedestrians in complicated environments such as partially occluded situations when human intersects.

Chapter 6

Fusion-based hierarchical method for direct gradient-based time-to-contact estimation

The time-to-contact (TTC) estimation is a method to analyze surrounding environment in order to detect approaching objects, and potential danger. TTC can be estimated directly from a single camera though neither distance nor speed information can be estimated based on a single camera. Traditional TTC estimation depends on "interesting feature points" or object boundaries, which is noisy and time consuming. We are proposing a hierarchical method, direct gradient-based estimation, to compute TTC from time varying images using sums of suitable products of image brightness derivatives. The method determines TTC based on constant brightness assumption and the analysis of the motion field associated with rigid body motion under perspective projection. Our method is called "direct" method since it can take advantages of all related pixels for better computation, and avoid any "higher level" processing, such as feature/object detection, tracking of features, estimation of optical flow, etc. Such method essentially has no latency, since it can be based on analysis of just two frames of a video sequence, and it does not require a calibrated camera. The new method enhances accuracy, robustness and is computationally efficient, which is important to provide fast response for vehicle applications. An implementation of the method is demonstrated on synthetic image sequences and stop-motion sequences

– where the ground truth is known – as well as on video sequences taken in outdoor driving environment. The proposed hierarchical fusion framework for direct gradient-based time-to-contact estimation enhances accuracy, robustness and is computationally efficient, which is important to provide fast response for vehicle applications.

6.1 Introduction

6.1.1 Definition

The time-to-contact (TTC) is defined as the time that would elapse before the center of projection (COP) reaches the surface being viewed if the current relative motion between the COP and the surface was to continue without change as shown in Figure (6-1).

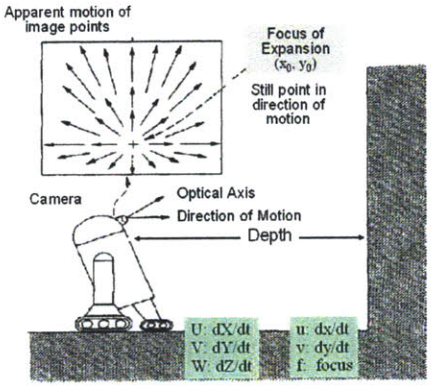


Figure 6-1: Time-to-contact.

Such a parameter can be used for intelligent vehicles, such as, obstacle avoidance, automated parallel parking, braking & steering systems, etc., as well as for advanced automation, automated assembly and robotics, where parts need to be moved rapidly into close physical alignment while at the same time avoiding damage due to high speed impact.

To define TTC mathematically, we first establish a camera-oriented coordinate system as shown in Figure (6-1), in which the origin at the COP, the Z axis along the optical axis, and the X and Y axes parallel to axes of the image sensor. Z is actually the distance from the center of projection (COP) to an object.

The TTC is the ratio of distance to velocity:

$$\text{TTC} = -Z/\frac{dZ}{dt} = -1/\frac{d}{dt}log_e(Z) \tag{6.1}$$

While distance and velocity can not be recovered from images taken with a single camera without additional information, such as the principal distance and the size of the object, the ratio of distance to velocity can be recovered directly, even with an uncalibrated sensor.

Here below we first define perspective projection model and relative motion parameters, then introduce the relationship between motion field and time-to-contact/focus-of-expansion.

Perspective projection model and Motion Parameters

Image coordinates x and y are measured from the principal point (foot of the perpendicular dropped from the COP). The units of measurement for x and y are the same as that used for the principal distance, f (e.g. the inter-pixel spacing). The vector format for an object point and its image projection are respectively described as $\vec{R}$ and $\vec{r}$.

The perspective projection equations of image formation can then be written in the simple form:

$$\frac{x}{f} = \frac{X}{Z} \qquad \text{and} \qquad \frac{y}{f} = \frac{Y}{Z} \tag{6.2}$$

As shown in Figure (6-1), $(u, v) = (\dot{x}, \dot{y})$ is the motion field and $(U, V, W) = (\dot{X}, \dot{Y}, \dot{Z})$ is the velocity of a point on the object relative to the sensor (which is opposite to the motion of the sensor relative to the object). The velocity $W = dZ/dt$ is negative if the object is approaching the camera.

By differentiating the perspective projection Equations (6.2) with respect to time, we obtain

$$\frac{u}{f} = \frac{U}{Z} - \frac{X}{Z}\frac{W}{Z} \qquad \text{and} \qquad \frac{v}{f} = \frac{V}{Z} - \frac{Y}{Z}\frac{W}{Z} \tag{6.3}$$

The motion field is subject to the scale factor ambiguity, since multiplying the coordinates X, Y, and Z and the velocity components U, V, and W by the same factor does not change u or v. The motion field depends on the rigid body motion between the camera and the surface being viewed, as well as the shape of the surface.

Using the perspective projection Equations (6.2) we can rewrite the above in the form

$$\frac{u}{f} = \frac{U}{Z} - \frac{x}{f}\frac{W}{Z} \qquad \text{and} \qquad \frac{v}{f} = \frac{V}{Z} - \frac{y}{f}\frac{W}{Z} \tag{6.4}$$

or

$$u = \frac{1}{Z}(fU - xW) \qquad \text{and} \qquad v = \frac{1}{Z}(fV - yW) \tag{6.5}$$

Equation (6.5) can also be written as

$$u = -\frac{W}{Z}(x - x_0) \qquad \text{and} \qquad v = -\frac{W}{Z}(y - y_0) \tag{6.6}$$

where $x_0 = f(U/W)$ and $y_0 = f(V/W)$, which is the "focus-of-expansion" (FOE) at which point the motion information is zero ($u = 0$ and $v = 0$) as shown in Figure (6-1) and (6-2). When the object point corresponding to FOE moves along the direction of (U, V, W), its image projection remains the same. The motion directions at other image points are determined by vectors from the FOE to other image points.

When the object rotates relative to the sensor, the general expression of object motion in 3D is expressed in $\dot{\vec{R}} = \vec{t} + \vec{\omega} \times \vec{R}$ in which $\vec{t}$ and $\vec{\omega}$ respectively denotes the translational component of the motion and the angular velocity, specifically, $\vec{t} = (U, V, W)$ and $\vec{\omega} =$

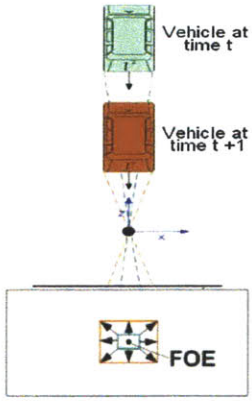


Figure 6-2: Example of FOE(focus-of-expansion).

$(\omega_x, \omega_y, \omega_z)$. Then the motion field of image points is as followed:

$$\begin{aligned} u &= -\frac{W}{Z}(x - x_0) - f\left[\omega_x \frac{xy}{f^2} - \omega_y(1 + \frac{x^2}{f^2}) + \omega_z \frac{y}{f}\right] \\ v &= -\frac{W}{Z}(y - y_0) - f\left[-\omega_y \frac{xy}{f^2} + \omega_x(1 + \frac{y^2}{f^2}) - \omega_z \frac{x}{f}\right] \end{aligned} \tag{6.7}$$

In summary, the motion field is related to the time-to-contact, focus of expansion, rotation parameters, as well as the shape of the surface.

The relationship between motion field and time-to-contact/focus-of-expansion

First, we rewrite Equation (6.1) as:

$$\text{TTC} = -Z/W \tag{6.8}$$

Then we can respectively rewrite Equation (6.6) and Equation (6.7) as followed:

$$u = \frac{(x - x_0)}{\text{TTC}} \quad \text{and} \quad v = \frac{(y - y_0)}{\text{TTC}} \tag{6.9}$$

$$
\begin{aligned}
u &= \frac{(x - x_0)}{\text{TTC}} - f\left[\omega_x \frac{xy}{f^2} - \omega_y(1 + \frac{x^2}{f^2}) + \omega_z \frac{y}{f}\right] \\
v &= \frac{(y - y_0)}{\text{TTC}} - f\left[-\omega_y \frac{xy}{f^2} + \omega_x(1 + \frac{y^2}{f^2}) - \omega_z \frac{x}{f}\right]
\end{aligned} \tag{6.10}
$$

6.1.2 Traditional methods and the limitation

There are two traditional methods to estimate TTC. The first method is optical-flow based, and the second one is size-based.

Optical-flow based method

This method takes advantage of the relationship between time-to-contact and optical flow [79] [80] [81] [82]

According to Equation (6.9), we have

$$
\begin{aligned}
u_x &= \frac{1}{\text{TTC}} \qquad u_y = 0 \\
v_x &= 0 \qquad v_y = \frac{1}{\text{TTC}}
\end{aligned} \tag{6.11}
$$

For general motion with translation and rotation as described in Equation (6.10), we have:

$$
\begin{aligned}
u_x &= \text{TTC} - f\left[\omega_x \frac{y}{f^2} - \omega_y(\frac{2x}{f^2})\right] \\
u_y &= -f\left[\omega_x \frac{x}{f^2} + \omega_z \frac{1}{f}\right] \\
v_x &= f\left[\omega_y \frac{y}{f^2} + \omega_z \frac{1}{f}\right] \\
v_y &= \text{TTC} + f\left[\omega_y \frac{x}{f^2} - \omega_x(\frac{2y}{f^2})\right]
\end{aligned} \tag{6.12}
$$

Thus, people can compute time-to-contact based on the following methods [79] [80] [81] [82]:

$$\text{TTC} = \left[\frac{u_x + v_y}{2}\right]^{-1} \quad \text{or}$$

$$\text{TTC} = \left[\frac{u_x + v_y}{2} \pm \frac{\sqrt{(u_y + v_x)^2 + (u_x - v_y)^2}}{2}\right]^{-1} \tag{6.13}$$

People can also recover TTC through the expected form of the flow field based on rigid body motion relative to a known shape [84], [90], [91], [92], [93].

Such methods need to estimate optical flow iteratively at multiple scales, tend to be computationally expensive and require a significant effort to implement properly.

Size-based method

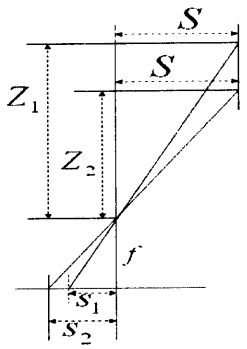


Figure 6-3: Projection relationship for a linear object model. S: Linear object size. f: Principal distance. When a linear object is at two different distances Z_1 and Z_2, the image size corresponding to the linear object is respectively s_1 and s_2.

The size-based method is based on the following principle. Consider a simple situation as shown in Figure (6-3) where a camera is approaching a linear object lying perpendicular to the optical axis, with the direction of translational motion along the optical axis. If the (linear) sizes of the object and its image is respectively defined as S and s, then, from the perspective projection equation, we have $(s/f) = (S/Z)$, that is, $fS = sZ$. (In Figure (6-3), we have $s_1 Z_1 = s_2 Z_2 = Sf$.) Differentiating w.r.t time yields

$$s\frac{dZ}{dt} + Z\frac{ds}{dt} = 0 \tag{6.14}$$

Together with Equation (6.1), TTC is equal to the ratio of the size s of the image of the object to the rate of change of the size, that is

$$T = s/\frac{ds}{dt} = 1/\frac{d}{dt}log_e(s) \tag{6.15}$$

It is convenient to use inter-frame interval as the unit of time and express the TTC as a multiple of the interval.

Equation (6.15) calculates TTC based on the size of the image of an object and the change in that size over time [94] which requires to extract features and track features from frame to frame. The time varying image is sampled at regular intervals and the time derivative of size is estimated using the difference between sizes of the images of the object in two frames. High accuracy is needed in measuring image sizes of targets in order to obtain accurate estimates of the TTC when it is large compared to the inter-frame interval.

For example, when the size of the image of a van is about 100 pixels, and the estimated time-to-contact is 100 frames, then the image of size 100 pixels changes by only 1 pixel from frame to frame. To achieve even 10% error in the TTC one would have to measure the size of the image with an accuracy of better than $1/10$ of a pixel. The tolerance for measurement error becomes even smaller when the object is further away and the TTC larger. When the estimated TTC is 500 frames, the frame to frame change in size of the image is only 0.2 pixel and the motion of either end of the van is about 0.1 pixel. Measuring the TTC to an accuracy of say 10% requires, in effect, measuring image positions with an accuracy better than 0.01 pixel. Obviously, it is very difficult. Thus the accuracy of TTC computation based on the size-based method is far from satisfying.

Furthermore, it is difficult to generalize Equation (6.15) to translational motions that are not along the optical axis – or to objects other than planar ones that lie at right angles to the optical axis.

6.2 Direct method for time-to-contact

Instead of relying on feature detecting, feature tracking, or estimation of the optical flow, we propose a method directly based on the derivatives of image brightness by exploiting constraints between the brightness gradient (spatial derivatives of brightness) and the time derivative of brightness. The constraints is called "constant brightness assumption" as shown in Equation (6.16).

$$\frac{d}{dt}E(x,y,t) = 0 \qquad \rightarrow \qquad uE_x + vE_y + E_t = 0 \tag{6.16}$$

where $E(x,y,t)$, E_x, E_y, E_t are respectively the image brightness, and the partial derivatives of brightness w.r.t. x, y, and t. The equation defines the relationship between the brightness derivatives and the motion field based on the observation that in many situations the brightness of the image of a point in the scene does not change significantly as it moves in the image [83]. The constant brightness assumption holds for many practical situations when the light sources do not move relative to the scene and related surface are not specular surfaces which will have different brightness when viewed from different directions.

In order to estimate time-to-contact, we first introduce time-to-contact information into Equation (6.16) by substituting Equations (6.5) for the motion field components u and v, then we formulate a solution based on least-squares-based optimization.

Since all objects are composed of several planar areas, we limit our discussion to planar objects which is applicable to most of general situations. Our results as discussed in Section 6.5 show that our proposed method based on planar objects works fine for real objects. Thus, we consider the general case when there is $arbitrary$ translational motion between a camera and a planar surface with $arbitrary$ orientation as shown in Figure (6-4).

The plane with arbitrary orientation can be described by:

$$Z = Z_0 + pX + qY \tag{6.17}$$

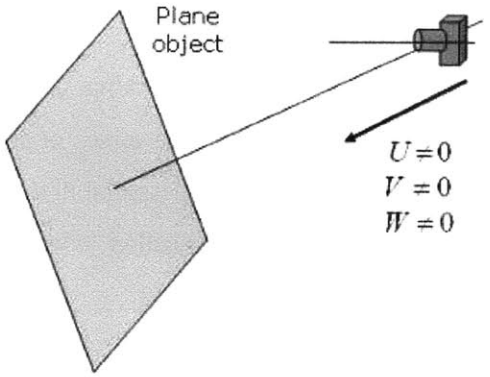


Figure 6-4: The situation when there is arbitrary translational motion between camera and an plane with arbitrary orientation.

where p and q be the slopes of the planar surface in the X and Y directions. Substituting the perspective projection equation $X = (x/f)Z$ and $Y = (y/f)Z$ into Equation (6.17) yields:

$$Z\left(1 - p\frac{x}{f} - q\frac{y}{f}\right) = Z_0 \tag{6.18}$$

Substituting the expression for Z given by Equation (6.18) in expressions for u and v given by Equation (6.6) and then inserting (u, v) into the brightness change constraint Equation (6.16) leads to

$$\begin{aligned} \frac{1}{Z}(fU - xW)E_x + \frac{1}{Z}(fV - yW)E_y + E_t = 0 \\ \frac{W}{Z_0}\left(1 - p\frac{x}{f} - q\frac{y}{f}\right)[(fU/W - x)E_x + (fV/W - y)E_y] + E_t = 0 \end{aligned} \tag{6.19}$$

If we define

$$
\begin{aligned}
P &= (p/f)(W/Z_0) \qquad Q = (q/f)(W/Z_0) \\
A &= f(U/Z_0) \qquad B = f(V/Z_0) \qquad C = -W/Z_0 \\
G &= xE_x + yE_y
\end{aligned} \tag{6.20}
$$

Equation (6.19) can be rewritten as:

$$
C\left(1 + x\frac{P}{C} + y\frac{Q}{C}\right)\left[E_x\frac{A}{C} + E_y\frac{B}{C} + G\right] + E_t = 0 \tag{6.21}
$$

We notice that $C = -W/Z_0$ is the inverse of the TTC, and G is a short-hand for the "radial gradient" $(xE_x + yE_y)$, A and B are proportional to the coordinates of FOE as shown below:

$$
\begin{aligned}
A &= f(U/Z_0) = -(-W/Z_0)(fU/W) = -Cx_0 \\
B &= f(V/Z_0) = -(-W/Z_0)(fV/W) = -Cy_0
\end{aligned} \tag{6.22}
$$

We formulate a least-squares method to find the five unknown parameters A, B, C, P, and Q that minimize the following error integral or sum over all pixels of a region of interest (which could be full image):

$$
\sum\left\{C\left(1 + x\frac{P}{C} + y\frac{Q}{C}\right)\left[G + E_x\frac{A}{C} + E_y\frac{B}{C}\right] + E_t\right\}^2 \tag{6.23}
$$

If the five unknown parameters can be solved, then we can compute time-to-contact which is $1/C$. If desired, we can also find the focus-of-expansion using

$$
x_0 = -A/C \qquad \text{and} \qquad y_0 = -B/C \tag{6.24}
$$

The principal distance, f, needs not be known in order to compute the TTC or the FOE. If f is known, the actual direction of translational motion and the actual surface orientation can respectively be determined as below:

$$\frac{U}{W} = -\frac{1}{f}\frac{A}{C} \quad \text{and} \quad \frac{V}{W} = -\frac{1}{f}\frac{B}{C} \tag{6.25}$$

$$p = -f\frac{P}{C} \quad \text{and} \quad q = -f\frac{Q}{C} \tag{6.26}$$

In summary, the least-squares-based optimization provides the time-to-contact(TTC), focus-of-expansion(FOE), and surface parameter $p/f, q/f$. If f is known, the motion parameters $(U/W, V/W)$ and surface parameters (p, q) can be determined.

If the surface plane is perpendicular to the optical axis, we should have $p = 0$ and $q = 0$. If the translational movement is along the optical axis, the FOE should be at the origin and we should have $U/W = 0$ and $V/W = 0$.

6.2.1 Linear and nonlinear solutions for different cases

So far our discussion focuses on the most general case shown in Figure (6-4) where the planar surface can be oriented in an arbitrary way, and the translational motion can be in an arbitrary direction. For the most general situation, the minimization of cost function in Equation (6.23) is a nonlinear problem, and we do not have closed-form solutions for five unknown parameters A, B, C, P, Q. However, we do have linear solutions for three special cases as shown in Figure (6-5)(a)(b)(c). So we have the following four cases:

- Case (I)

  Translational motion along the optical axis towards a planar surface perpendicular to the optical axis. This case assumes that motion $U = V = 0$ and slope $p = q = 0$.

- Case (II)

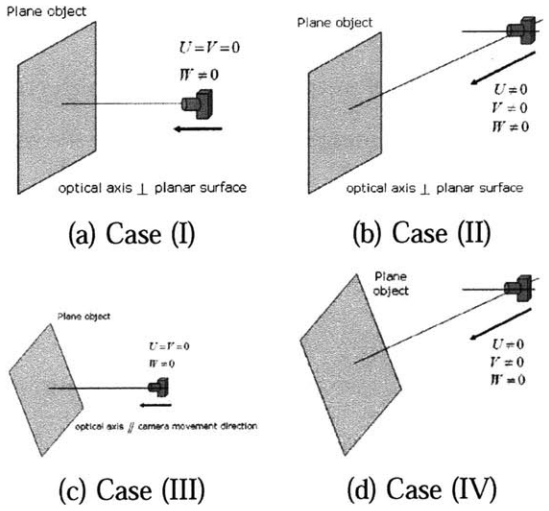
Translational motion in an *arbitrary* direction relative to a planar surface that is perpendicular to the optical axis; This case assumes that motion $U \neq 0, V \neq 0$ and slope $p = q = 0$.

- Case (III)

  Translational motion along the optical axis relative to a planar surface of *arbitrary* orientation; This case assumes that $U = V = 0$, and slope $p \neq 0, q \neq 0$.

- Case (IV)

  The *arbitrary* translational motion relative to a planar surface of *arbitrary* orientation. This case assumes that motion $U \neq 0, V \neq 0$ and slope $p \neq 0, q \neq 0$



(a) Case (I) (b) Case (II)

(c) Case (III) (d) Case (IV)

Figure 6-5: Four cases of relative motion. (a) Case (I): Translational motion along the optical axis towards a planar surface perpendicular to the optical axis. (b) Case (II) Translational motion in an arbitrary direction, with the optical axis perpendicular to a planar surface. (c) Case (III) Translational motion along the optical axis relative to an arbitrary plane. (d) Case (IV) Arbitrary translational motion relative to an arbitrary plane.

Case (IV) is the most general case which requires non-linear optimization techniques. Cases (I)(II)(III) are the special cases for Case (IV) and have closed form solutions to minimize the cost function described in Equation (6.23). We respectively discuss the close

form solutions for Cases (I)(II)(III) in Section 6.2.2, 6.2.3, 6.2.4. In Section 6.2.5, we will discuss the iterative method to solve the nonlinear problem for Case (IV)

6.2.2 Case (I): Translational motion along the optical axis towards a plane perpendicular to the optical axis

As shown in Figure (6-5)(a), this case assumes that the surface plane is perpendicular to the optical axis and the translational motion is in the direction of the optical axis of the imaging system, i.e., motion $U = V = 0$ and slope $p = q = 0$. According to Equation (6.20), we have $A = B = 0$ and $P = Q = 0$, and the cost function in Equation (6.23) becomes:

$$\sum (CG + E_t)^2 \tag{6.27}$$

where the sum is over all pixels of a region of interest (which could be full image). To find the best fit values of C, differentiating w.r.t. C and setting the result equal to zero yields

$$\sum (CG + E_t)G = 0 \quad \Longrightarrow \quad C = -\sum GE_t / \sum G^2 \tag{6.28}$$

One we have the value of C, we can compute the time-to-contact $1/C$. The computation requires only accumulation of products of "radial gradient," and time derivatives of brightness.

6.2.3 Case (II): arbitrary translational motion relative to a plane perpendicular to the optical axis

As shown in Figure (6-5)(b), this case assumes that the surface plane is perpendicular to the optical axis, i.e., slope $p = q = 0$. But the translational motion is not necessarily in the direction of the optical axis of the imaging system (nor perpendicular to the surface), which is different from case (I). According to Equation (6.20), we have $P = Q = 0$, and

the cost function in Equation (6.23) becomes:

$$\sum (AE_x + BE_y + CG + E_t)^2 \tag{6.29}$$

where the sum is over all pixels of a region of interest (which could be full image). To find the best fit values of A, B, and C, differentiating with respect to A, B, and C and setting the results equal to zero yields

$$\begin{aligned} \sum (AE_x + BE_y + CG + E_t)E_x &= 0, \\ \sum (AE_x + BE_y + CG + E_t)E_y &= 0, \\ \sum (AE_x + BE_y + CG + E_t)G &= 0. \end{aligned} \tag{6.30}$$

i.e.,

$$\begin{bmatrix} \sum E_x^2 & \sum E_x E_y & \sum G E_x \\ \sum E_x E_y & \sum E_y^2 & \sum G E_y \\ \sum G E_x & \sum G E_y & \sum G^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} -\sum E_x E_t \\ -\sum E_y E_t \\ -\sum G E_t \end{bmatrix} \tag{6.31}$$

Using Equation (6.31), we can compute the unknowns A, B, and C, and the corresponding TTC and FOE(x_0, y_0). If f is known, the actual direction of translational motion $(U/W, V/W)$ can be computed using Equation (6.25).

Note that the coefficients of the symmetric 3×3 matrix are all sums of products of components of the radial gradient and the brightness gradient, while the quantities on the right-hand sides of the equations are sums of products of components of the brightness gradient, and the time derivative of brightness.

If $U = V = 0$, that is, if the translational motion happens to be along the optical axis, then the least squares problem is simplified, leading to the single equation for C only, discussed above in Case (I).

6.2.4 Case (III): translational motion along the optical axis relative to an arbitrary plane

As shown in Figure (6-5)(c), this case assumes that the translational motion is in the direction of the optical axis of the imaging system, i.e., motion $U = V = 0$. But the planar surface is not necessarily oriented perpendicular to the optical axis (or the direction of the translational motion), which is different from case (I). According to Equation (6.20), we have $A = B = 0$ and the cost function in Equation (6.23) becomes:

$$\sum [G(C + Px + Qy) + E_t]^2 \tag{6.32}$$

where the sum is over all pixels of a region of interest (which could be full image). To find the best fit values of P, Q, and C, we differentiate with respect to P, Q, and C and set the three results equal to zero:

Since the translational motion here is along the optical axis, the contact point on the plane is $(0, 0, Z_0)^T$, and so the TTC is again just the inverse of C.

$$\begin{aligned} \sum [G(C + Px + Qy) + E_t]\, xG &= 0, \\ \sum [G(C + Px + Qy) + E_t]\, yG &= 0, \\ \sum [G(C + Px + Qy) + E_t]\, G &= 0. \end{aligned} \tag{6.33}$$

i.e.,

$$\begin{bmatrix} \sum G^2x2 & \sum G^2xy & \sum G^2x \\ \sum G^2xy & \sum G^2y2 & \sum G^2y \\ \sum G^2x & \sum G^2y & \sum G^2 \end{bmatrix} \begin{bmatrix} P \\ Q \\ C \end{bmatrix} = \begin{bmatrix} -\sum GxE_t \\ -\sum GyE_t \\ -\sum GE_t \end{bmatrix} \tag{6.34}$$

Using Equation (6.34), we can compute the unknowns P, Q, and C, and the corresponding TTC and plane orientation(P, Q). If f is known, then the actual surface orientation can

also be determined using Equation (6.26).

Note that the coefficients of the symmetric 3×3 matrix are all sums of products of components of the radial gradient and image coordinates, while the quantities on the right-hand sides of the equation are sums of products of components of the brightness gradient, image coordinates, and the time derivative of brightness.

If $p = q = 0$, that is, if the planar surface happens to lie perpendicular to the optical axis, then the least squares problem is simplified, again leading to the single equation for C only, discussed above in Case (I).

6.2.5 Case (IV): arbitrary translational motion relative to an arbitrary plane

In general, to find the best fit values of the five unknown parameters we can differentiate either of the two sums with respect to the five parameters and set the results equal to zero. This leads to five equations for five unknowns. The equations are nonlinear and need to be solved numerically.

However, if we define

$$F = 1 + xP/C + yQ/C \tag{6.35}$$

$$D = G + E_x A/C + E_y B/C \tag{6.36}$$

and rewrite the cost function in Equation (6.23) as:

$$\sum [CFD + E_t]^2 \tag{6.37}$$

We observe that if P/C and Q/C are given, then $F = 1 + xP/C + yQ/C$ is known and the cost function Equation (6.23) are linear in the remaining unknowns A, B, and C as followed:

$$\sum [F * (CG + E_x A + E_y B) + E_t]^2 \tag{6.38}$$

Parameters A, B, and C can be solved based on the following linear equations:

$$\begin{bmatrix} \sum F^2 E_x^2 & \sum F^2 E_x E_y & \sum F^2 G E_x \\ \sum F^2 E_x E_y & \sum F^2 E_y^2 & \sum F^2 G E_y \\ \sum F^2 G E_x & \sum F^2 G E_y & \sum F^2 G^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} -\sum F^2 E_x E_t \\ -\sum F^2 E_y E_t \\ -\sum F^2 G E_t \end{bmatrix} \tag{6.39}$$

Conversely, if A/C and B/C are given, then $D = G + E_x A/C + E_y B/C$ is known and the cost function Equation (6.23) are linear in the remaining unknowns P, Q, and C.

$$\sum [(C + xP + yQ) * D + E_t]^2 \tag{6.40}$$

Parameters P, Q, and C can be solved based on the following linear equations:

$$\begin{bmatrix} \sum D^2 x2 & \sum D^2 xy & \sum D^2 x \\ \sum D^2 xy & \sum D^2 y2 & \sum D^2 y \\ \sum D^2 x & \sum D^2 y & \sum D^2 \end{bmatrix} \begin{bmatrix} P \\ Q \\ C \end{bmatrix} = \begin{bmatrix} -\sum x D E_t \\ -\sum y D E_t \\ -\sum D E_t \end{bmatrix} \tag{6.41}$$

Note that the coefficients of two symmetric 3×3 matrix in Equation (6.39) and (6.41) are the function of the radial gradient, brightness gradient, and image coordinates, while the quantities on the right-hand sides of the equations are sums of products of components of the brightness gradient, image coordinates, and the time derivative of brightness.

Given an initial guess P/C and Q/C, one can alternately solve for A, B, and C based on initial estimate of P/C and Q/C using Equation (6.39). Then, given the new estimates of A/C and B/C, we can update P, Q and C with Equation (6.41). A few iterations of this pair of steps typically yield a close enough approximation to an exact solution.

The time-to-contact is the inverse of the parameter C. If desired, the direction of translational motion, given by $(U/W)(= (A/C)/f)$ and $(V/W)(= (B/C)/f)$ can also be

Table 6.1: Summarization of computational models and relative assumptions

Cases	Figure	Assumption	Unknown parameters	Solution
(I)	6-5(a)	$U = V = 0, p = q = 0$ or $A = B = 0, P = Q = 0$	C	Linear Equation Eqn (6.28)
(II)	6-5(b)	$p = q = 0$ or $P = Q = 0$	A, B, C	Linear Equation Eqn (6.31)
(III)	6-5(c)	$U = V = 0$ or $A = B = 0$	P, Q, C	Linear Equation Eqn (6.34)
(IV)	6-5(d)	None	A, B, P, Q, C	Non-linear, Iteration Eqn (6.39), (6.41)

calculated, as can the orientation of the surface specified by $f(P/C)$ and $f(Q/C)$.

6.2.6 Summary: comparison between different computational models

Table (6.1) summarizes all four cases of computational models.

The parameters involved in LSQ optimization problems are:

$$(A = fU/Z_0, B = fV/Z_0, C = -W/Z_0, P = (p/f)(W/Z_0), Q = (q/f)(W/Z_0)).$$

The first three, (A, B, C), are the scaled velocity of translational motion relative to the sensor, $(U, V, W) = (\dot{X}, \dot{Y}, \dot{Z})$, and the last two, (P, Q), are relative to object-shape parameters, p and q, the slopes for the planar surface in the X and Y directions.

For Case (IV) in Figure (6-5)(d), the most general case, the translational motion is not in the direction of the optical axis of the imaging system, and the planar surface is not be oriented perpendicular to the optical axis. No assumptions are made for five unknown parameters: (A, B, C, P, Q).

For Case (II) in Figure (6-5)(b), the planar surface is assumed to be perpendicular to the optical axis while the translational motion is in an *arbitrary* direction. It is assumed $p = q = 0$, (i.e., $P = Q = 0$). Three unknown parameters, (A, B, C) for optimization, are related to motion parameters (U, V, W). Case (II) is a special case of Case (IV).

For Case (III) in Figure (6-5)(c), the translational motion is assumed to be along the optical axis while the planar surface is of *arbitrary* orientation. It is assumed $U = V = 0$,

(i.e., $A = B = 0$). Three unknown parameters, (P, Q, C), are related to object orientation parameters (p, q) and time-to-contact information. Case (III) is a special case of Case (IV).

For Case (I) in Figure (6-5)(a), the planar surface is assumed to be perpendicular to the optical axis while the translational motion is assumed to be along the optical axis. It is assumed $U = V = 0$ and $p = q = 0$, (i.e., $A = B = 0$ and $P = Q = 0$). There is only one unknown parameter, C. Clearly Case (I) is a special case of all Cases (II)(III), and (IV).

Based on computed parameters (A, B, P, Q, C) from Equations (6.39) and (6.41), we can calculate the time-to-contact information $TTC = 1/C$, the movement information (as well as the focus-of-expansion) $x_0 = -A/C = fU/W$ and $y_0 = -B/C = fV/W$ according to Equations (6.24) and (6.25), and obtain planar parameters $p/f = -P/C$ and $q/f = -Q/C$ according to Equations (6.26).

6.2.7 Identification of different setups in real applications

In applications where we do not have any extra information about the orientation of objects and the relative motion between objects and cameras, we need to apply the computational methods based on Case (IV) to compute (A, B, P, Q, C).

If we have $P \approx 0, Q \approx 0$ in applications, then we can identify the Case (II) where the object plane surface is perpendicular to the optical axis, i.e., $p/f = -P/C = 0 \approx 0$ and $q/f = -Q/C = 0 \approx 0$.

If we have $A \approx 0, B \approx 0$ in applications, then we can identify the Case (III) where the translational motion between camera and object surface is along the optical axis, i.e., $U = 0$ and $V = 0$. For such case, FOEs should be also around the origin, i.e., $x_0 \approx 0, y_0 \approx 0$ based on Equation (6.24).

If we have $A \approx 0$, $B \approx 0$ and $P \approx 0$, $Q \approx 0$ in applications, we can identify the special Case (I) where the translational motion is along the optical axis and object plane surface is perpendicular to the optical axis, i.e., $U \approx 0, V \approx 0$, $p/f = 0 \approx 0$ and $q/f = 0 \approx 0$.

The physical model of Case (I) is not only a special situation of cases (IV), but also a special situation of cases (II) and (III). We can also identify Case (I) in computation with video sequences if we apply the computational method for Case (II) and obtain $A \approx 0$

and $B \approx 0$, or if we apply the computational method for Case (III) and obtain $P \approx 0$ and $Q \approx 0$.

6.3 Factors affecting computational results

After presenting the theoretical solution of TTC estimation, we would like to discuss the following factors that would have impact on the accuracy and reliability of the computational results.

1) Numerical method for partial derivatives

2) Computational areas: full images vs. segmented regions

3) Threshold for time derivative of brightness

4) Four various computational models

5) Different subsampling rates

6) Fusion of four computational models with multi-scale subsampling

The discussion on the impact of the first five factors leads to the necessity of the sixth factor, which inspired us to proposed the hierarchical time-to-contact estimation based on fusion of computational models and multi-scale subsampling presented in Section 6.4. More in-depth discussions on some factors are also presented in Section 6.5, the section for experiment results.

6.3.1 Numerical method for partial derivatives

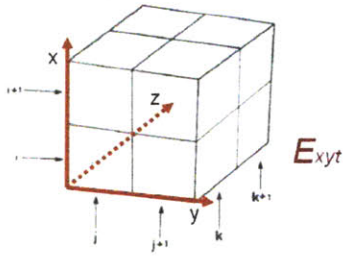


Figure 6-6: Computing cell. The involved cubics E_{xyt} are the elements involved in Equation (6.43). (x, y) corresponds to the location. The z is along the time dimension, t.

In our implementation, the partial derivatives of image brightness are estimated using a $2 \times 2 \times 2$ cube of pixel values from two continuous images as illustrated in Figure (6-6) and in the following Equation:

$$\begin{aligned} E_x &= (E_{100} - E_{000}) + (E_{101} - E_{001}) + (E_{110} - E_{010}) + (E_{111} - E_{011}) \\ E_y &= (E_{010} - E_{000}) + (E_{011} - E_{001}) + (E_{110} - E_{100}) + (E_{111} - E_{101}) \\ E_t &= (E_{001} - E_{000}) + (E_{011} - E_{010}) + (E_{101} - E_{100}) + (E_{111} - E_{110}) \end{aligned} \tag{6.42}$$

where E_{xy1} and E_{xy0} respectively correspond to pixels at location (x, y) in the current and the previous frames. To compute E_x (or E_y), the equation averages the horizontal (or vertical) differences of two neighbor columns (or rows) from both current and previous frames. E_t is computed by averaging the differences of four neighbor pixels from both the current and the previous frames. Averaging along the additional time/position dimension helps to improve the accuracy. The above equation is actually the scaled version of partial derivatives. The extra bits in the sums of blocks of pixel values are retained rather than being discarded in the division by the number of picture cells in a block. Such operation can help to improve the computational accuracy. For all equations for TTC computation, Equaions (6.28) for Case(I), (6.31) for Case(II), (6.34) for Case(III), and Equations (6.39) and (6.41) for Case(IV), both sides of equations are scaled. Thus the final TTC results are not affected by the scaling.

6.3.2 Computational areas: full images vs. segmented regions

The variables in computational equations for Cases(I)-(IV) all involve the integration (or summation) of contribution from pixels in a specified region, which can be full image or segmented regions containing interested foregrounds.

TTC computation based on full images is close to the weighted average of TTCs for foregrounds and backgrounds. When background objects are static and far away, TTC estimation based on background regions is much larger than the one based on foreground regions, meaning that collision would not happen for a long time regarding static background regions. Thus, the integrated TTC estimation based on the mixture of foreground

and static background regions will be larger than using pure foreground regions. The farther away is the foreground, the larger is the background proportion, the larger TTC estimation based on full images is.

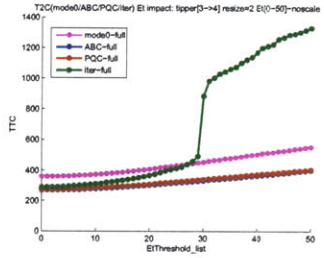
In order to reduce the impact of static background pixels on TTC computation, one method is to segment foregrounds of our interest and to remove most background pixels from integration areas. As long as most background pixels are removed, TTC results based on different sets of foreground pixels should be close to each other. Therefore, TTC estimation is relatively robust to the choices of segmentation, and it does not require accurate segmentation. Segmented areas capturing partial foregrounds can also provide satisfying TTC estimation. More examples and discussion for our test cases in Section 6.5.4 and 6.5.5 will illustrate how much segmentation helps to improve the accuracy of TTC results. The impact of segmentation on TTC and FOE will be summarized in Section 6.6.1.

6.3.3 Threshold for time derivative of brightness

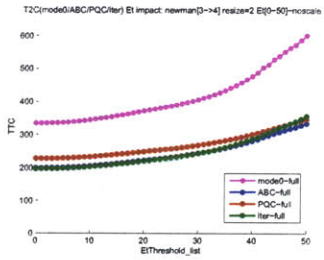
Another way to reduce the impact of static background pixels is to differentiate background pixels from foreground pixels when collecting their contributions in integration computation for TTC Equations (6.28) for Case(I), (6.31) for Case(II), (6.34) for Case(III), and Equations (6.39) and (6.41) for Case(IV). The differentiation is done by setting thresholds on the time derivative of brightness, $E_t\_Threshold$. The contributions of image pixels with minor intensity changes are blocked or masked. The blocked pixels are normally from static backgrounds that are far away and correspond to large TTC. While segmentation is the "hard rejection" of background pixels, thresholding is the "soft rejection."

The threshold for E_t cannot be larger than most pixel differences between two images. Otherwise pixel changes from frame to frame are ignored and TTC results become really large leading to wrong impression that foregrounds are far away. Figure (6-7) shows that how TTCs are affected by different values of $E_t\_Threshold$ values which change from 0 to 50. TTC are computed using two continuous image frames (No.3 and No.4 frames) from each of 6 test sequences SEQ_1 - SEQ_6, respectively in Figure (6-9), (6-10), (6-24), (6-25), (6-32), and Figure (6-35). Details about these sequences are introduced in Sec-

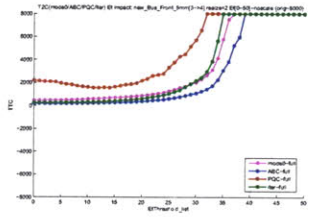
tion 6.5.1. In Figure (6-7), TTCs remain relatively consistent unless E_t thresholds become too large. We can conclude that TTC estimation is robust to the choices of segmentation and $E_t\_Threshold$ since TTCs based on different sets of foreground pixels are the similar as long as the impact of background pixels are removed.



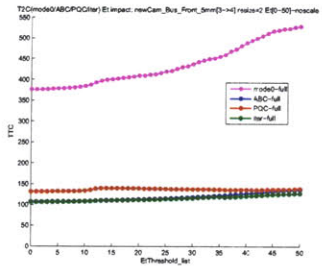
TTC vs. $E_t\_Threshold$ for (a) SEQ_1



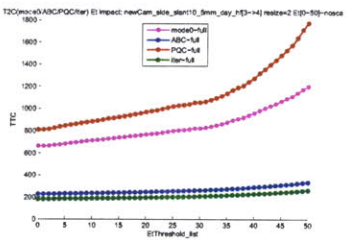
(b) SEQ_2



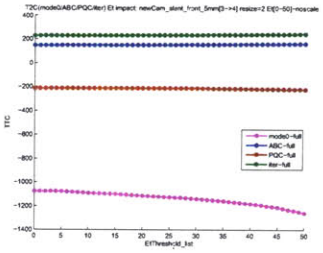
(c) SEQ_3



TTC vs. $E_t\_Threshold$ for (d) SEQ_4



(e) SEQ_5



(f) SEQ_6

Figure 6-7: The impact of different thresholds of brightness derivative ($E_t\_Threshold$) on TTC computation for four computational models and different sequences. X axis: subsampling parameters which change from 1 to 120. Y axis: TTC values computed based on two continuous image frames (No.3 and No.4 frames). Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. Figure (a)(b): TTC for synthetic sequences SEQ_1 in Figure (6-9) and SEQ_2 in Figure (6-10). (c)(d): TTC for stop-motion sequences with relative motion along optical axis, SEQ_3 in Figure (6-24) and SEQ_4 in Figure (6-25). (e)(f): TTC for stop-motion sequences with relative motion off optical axis, SEQ_5 in Figure (6-32) and SEQ_6 in Figure (6-35).

6.3.4 Four various computational models

In Section 6.2.7, we have discussed four different computational models for different situations. Normally the computational model for Case (IV), the most general form, is the best choice when we do not know any prior information about the relative motion. However, the computational models for cases (I)(II) and (III) have their own merit. The computa-

tional method for Case (IV) depends on iteration which involves heavier computational load than other discussed cases with fast close-form solutions. Indeed we need not limit our computational methods for specific situations only. We can apply all four methods to estimate TTC results in any situations. For example, though the simple solution in Equation (6.28) only applies to the simplest Case (I) theoretically, applying the equation for more complicated situations can still provide us useful and relatively accurate TTC information, especially when objects are very close. This is very important to help providing useful warning for urgent situations. In some situations, simpler methods provide better results than complicated methods.

In Figure (6-8) we compare the TTC computation results with different subsampling parameters based on four computational models for synthetic sequences SEQ_1 in Figure (6-9) and SEQ_2 in Figure (6-10). As shown in Figure (6-8), when subsampling rates are small, TTC computation based on Case (IV) is less sensitive to the changes of subsampling rates than based on Case (I)(II)(III). However, for large subsampling rates, Case (IV) does not necessarily provide better results than the other cases. The green line in Figure (6-7(a)) also shows that TTC computation based on Case (IV) is more sensitive to large derivative of brightness, $E_t\_Threshold$ than other methods. The estimation accuracy will be improved if we fuse the results from four computational models.

More details about the impact of different computational models on TTC and FOE/orientation results are discussed for different experimental cases in Sections 6.5.

6.3.5 Different subsampling rates

In our implementation, we use block averaging as a computationally cheap approximation to low pass filtering. The size of block averaging is defined as the subsampling rate. When there is no subsampling, the value of the subsampling rate is 1. Figure (6-8) shows TTC computation results with different values of subsampling rates. TTC computational results without subsampling are significantly larger than the results with subsampling, which explains the necessity of applying subsampling. For small values of subsampling rate, TTC results are relatively insensitive to the changes of subsampling rates. When subsampling

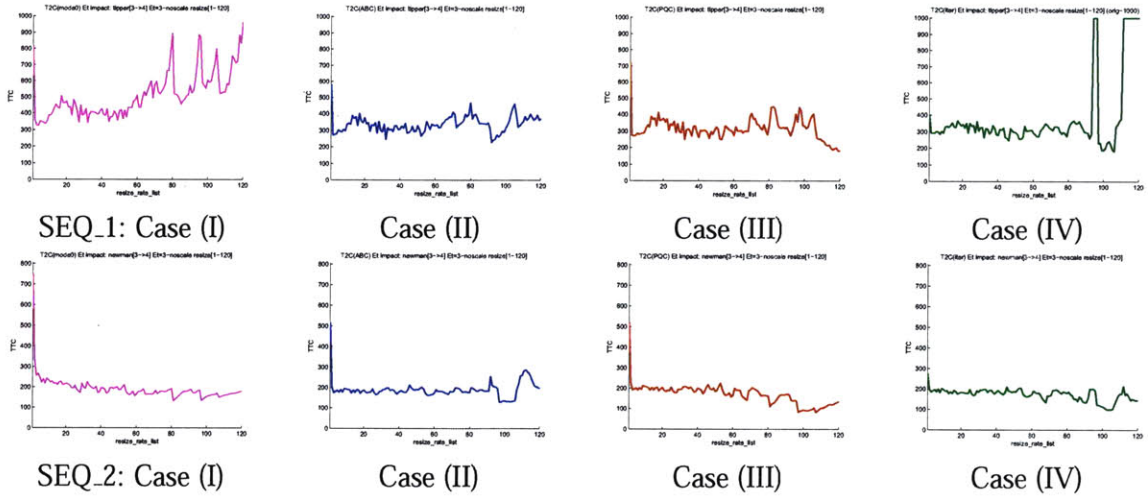


Figure 6-8: The impact of different subsampling rates on TTC computation for four computational models and two synthetic sequences, SEQ_1 in Figure (6-9) and SEQ_2 in Figure (6-10). X axis: subsampling parameters which change from 1 to 120. Y axis: TTC values computed based on two continuous image frames (No.3 and No.4 frames). Top row: for SEQ_1 in Figure (6-9). Bottom row: for SEQ_2 in Figure (6-10). Each column, from the left to the right, respectively corresponds to computational case (I), (II), (III), (IV).

rates are large, TTC results change dramatically, especially for Case (I).

The dependency of TTC estimation accuracy on subsampling rates is also affected by TTC values. As shown in Figure (6-34) and Figure (6-38), the estimation errors for small values of TTCs show the tendency of going up at the very end of the sequence when subsampling rates are small because of de-focus and the large motions between frames. When subsampling rates are very large(> 16), the estimation results for large TTCs are also not reliable because of limited pixel input for small foreground areas. More discussion on how subsampling rates affect the accuracy of TTC estimation for small and large TTCs, and how they will affect the accuracy of FOE computation, and the accuracy of orientation (p, q) parameters will be discussed in Section 6.5 when we analyze TTC estimation results for test sequences.

In summary, we have discussed five factors affecting TTC/FOE/(p, q) results. Since TTC computation is robust to its choice E_t threshold, E_t threshold can be fixed for all test experiments. Numerical method for partial derivatives is also fixed. From now on, we can focus on further understanding the impact of computational areas (full image vs.

segmentation scheme), computational models, subsampling rates. In order to increase the robustness of our algorithm, a fusion scheme is necessary to decrease the sensitivity of TTC to the computational models, subsampling rates. We discuss our fusion schemes in a separate section 6.4 due to its significance.

6.4 Hierarchical Time-to-Contact estimation based on fusion of multi-scale subsampling

Because of the sensitivity of TTC estimation on subsampling rates and computational models, we propose two fusion-based TTC estimation methods to combine TTC results from hierarchical multi-scale subsampling for different computational models in order to improve the robustness and reliability.

The first fusion method is to find the minimum value of all TTC results. Since the purpose of our TTC evaluation is for warning, choosing the minimum TTC is the most conservative strategy to avoid any dangerous situations. The method is very simple but effective. In Sections 6.5.5 and 6.5.6, we can see that minimization fusion scheme significantly improves the estimation accuracy and reliability as shown in Figure (6-34), (6-38), and Figure (6-41).

The second fusion scheme is based on the reliability of each TTC estimation and then fuse TTC data, which is introduced below in Section 6.4.1.

6.4.1 Multi-scale TTC fusion based on condition numbers

Our proposed TTC computation algorithms depend on matrix inversion to solve for unknown parameters as shown in Equation (6.31) for Case (II), Equation (6.34) for Case (III), Equation (6.39) and (6.41) for Case (IV). If any matrix in these equations is close to singular, their numerical results would not be very reliable, which in part leads to the huge peak noises when subsampling rates are too high or too low as shown in Figure (6-14), (6-15), (6-16), (6-17), (6-34), and (6-38).

Therefore, we can use condition numbers of these corresponding computational matri-

ces as the index of TTC estimation reliability. The condition number of a matrix is the ratio of between the largest singular value over the smallest singular value. If a matrix is close to singular, the condition number would be very large. Thus we can identify the reliability of TTC results based on their corresponding condition numbers of matrices involved in Equation (6.31), (6.34), (6.39) and (6.41). When fusing TTC results from multiple case models with multi-scale subsampling, condition-number based fusion method ignores the contribution of TTCs whose corresponding condition numbers are too large. The threshold of condition-number for acceptable TTCs is adaptively determined by an upper bound and the relative order of all corresponding condition numbers for different TTCs. At each frame, for N subsampling rates, we have total N TTC results and N corresponding condition numbers. We first sort these numbers, then compute the gap between every two consecutive numbers. We then sort the gap array and search from the largest gap to the smallest gap. If there are several continuous gaps larger than average gap, their corresponding data are discarded. More details about the condition-number based fusion are discussed in Section 6.6.3 when we present the experimental results.

6.5 Experiments

In order to evaluate the computation accuracy of TTC and FOE due to different factors, we apply four computational models at different subsampling rates with/without TTC fusion to a total of nine image sequences which can be divided into three categories.Test sequences will be presented in Section 6.5.1. Evaluation method will be in Section 6.5.2. The actual results will be presented in Section 6.5.3 to Section 6.5.6.

6.5.1 Test image sequences

Table (6.2) summarizes the information of these test sequences, including their length, image sizes, tested impact factors, figure number for original sequence image and TTC/FOE/(p, q) results. As can be seen from the table, for the purpose of evaluation, test sequences from Category 1, Category 2(a) and 2(b) are generated under the controlled environment so that true TTCs are known. Test sequences from Category 3 are obtained in an outdoor driving

situation for which a size-based method is applied to evaluate our algorithm. Details of different categories of test sequences are presented below.

- Category 1 – Synthetic Sequences: SEQ_1, SEQ_2

  Test sequences in the first category are synthetic image sequences generated with arbitrary known "ground truth" in the first test experiment. They are based on a single image which is shifted and magnified so as to simulate a specified translational motion relative to a planar surface. The sequence is generated assuming constant forward motion with various positions for the FOE. Results for this Category are discussed in Section 6.5.3.

- Category 2 – Stop-Motion Sequences: SEQ_3 - SEQ_8

  Test sequences in this category are constructed using a stop-motion technique. In all six sequences, images are taken when the camera – or the object – is moved by a controlled amount between exposures. When taking images for one sequence, camera parameters remain the same. Sequences involving camera motion are subject to small camera rotations which are hard to avoid and cause significant apparent lateral image motions. To avoid such impact caused by small camera rotations and to create reasonable stop-motion sequences, we choose to move objects instead of moving a camera. To achieve accurate increments in position, we put a toy car on the platform of a scaled optical bench with length 550mm and moved it along the optical bench by rotating a knob. The initial position of the platform was at the far end of the bench. For each step we slowly moved the sliding platform toward the camera by 5mm and took a picture until the object could not be further moved because of mechanical limitation. We can take at most 110 frames. The motion increment is 5 mm in the depth for each frame. The accuracy of each movement is 0.1mm. The optical bench ensures the $2\% = (0.1/5)$ accuracy for TTC "ground truth."

  We took images in two circumstances. The first circumstance is when the relative translational motion is along the optical axis. There are two video sequences, called Category 2(a), whose results are discussed in Section 6.5.4. The second circumstance

is when the relative motion is off the optical axis. There are 4 video sequences, called Category 2(b), whose results are discussed in Section 6.5.5.

The two video sequences in Category 2(a) are taken using two different cameras. The first video sequence was to use the image function of a Sony Camcorder. We had to physically press the button on the camcorder to take the video sequence. Besides the introduced mechanical vibrations, the stop-motion sequences captured with Sony Camcorder suffer from the effects of automatic focus and automatic exposure adjustments, as well as artifact introduced by image compression. The second video sequence was to use Canon Digital Rebel xTi (EOS-400D), SLR Digital Camera Kit w/ Canon 18-55mm EF-S Lens, in order to avoid the noises with the Sony Camcorder. We took advantage of camera's remote control function and controlled the camera by computer operation instead of physically pressing a button on the camera to take pictures, which ensures the stability of the camera. We chose manual-focus option and all parameters were manually controlled and fixed during the complete imaging process. We set small aperture and long focal length to ensure large depth of field.

Figure (6-24) and Figure (6-25) respectively show the sequence captured with the Sony camcorder, SEQ_3, and the sequence captured with the Canon camera, SEQ_4.

For Category 2(b), there are four video sequences. We use the Canon camera to produce four image sequences, one side-view sequence, SEQ_5, as shown in Figure (6-32), and three front-view sequences, SEQ_6, SEQ_7 and SEQ_8, in Figures (6-35), (6-36) and (6-37). To take different sequences, the camera was rotated by different angles along y axis in order to produce different yaw angles between the optical axis and the relative motion.

Results for Category 2(a) are discussed in Section 6.5.4, and results for Category 2(b) are discussed in Section 6.5.5.

- Category 3 – Video from a camera mounted on an automobile: SEQ_9

  Sequence SEQ_9 was taken with a camera mounted on an automobile driven around Cambridge, Massachusetts. The data captured in outdoor driving environment have

been provided by test vehicles of "DARPA challenges" competition. Results for this Category are discussed in Section 6.5.6.

6.5.2 TTC "ground-truth" and evaluation/test scheme

Our first task is to evaluate the accuracy of our proposed algorithm by comparing our TTC estimation with their ground-truths for different scenarios, including synthetic planar images, stop-motion sequences for non-planar objects in controlled environments with motion close to optical axis and in arbitrary directions, and sequences for non-planar objects in outdoor driving environments. For synthetic image sequences and stop-motion sequences, we have pre-defined TTC ground-truths to be compared with our computed ones so we can evaluate the system performance.In order to evaluate our system's performance in the outdoor driving environment, we estimated TTC "ground-truth" based on the traditional size-based method introduced in Section 6.1.2, and manually measured the sizes of objects in the images and differences between sizes observed in different frames. We computed TTC using Equation (6.15), and then compared two TTC results based on our method and size-based method as in Section 6.5.6. We also compared the size-based TTC estimation results for Sequence SEQ_8 with given TTC ground-truths as well as our TTC estimation in Section 6.6.2, which clearly shows the accuracy and reliability of our methods.

Our second task is to discuss the impact of the different computational models, subsampling factors, computational areas (full images or segmentation regions), and/or TTC fusion schemes. We use synthetic sequences from Category 1 to analyze the impact of four computational models and different subsampling rates on the estimation of TTC, focus-of-expansion (FOE), and orientation parameters.

We used sequences from Category 2(a) to discuss the impact of segmentation on TTC estimation. We calculated TTC by applying four different computational models to both full images and segmented regions, and we analyzed their impact on TTC/FOE/(p, q).

We used sequences from Category 2(b) and Category 3 to discuss the performance of TTC fusion scheme based on four computational models and seven different subsampling rates. We also apply two different segmentation schemes to SEQ_5 from Category 2(b) and

discuss their impacts on TTC fusion performance.

The test sequences are in color, but we use grey level pixel-intensity in TTC computation. Based on the discussion in Section 6.3.3, TTC computation is robust to the choices of E_t thresholds. So we use fixed E_t thresholds during all computations. For in Case (IV) it took about one to five iterations before converging. In the following sections, for most TTC figures, the dashed black line shows the theoretical ground-truth, and TTC results for four cases (I) (II) (III) (IV) are respectively plotted in four different colors, magenta, blue, red and green lines.

Table 6.2: Summarization of computational sequences and related tests

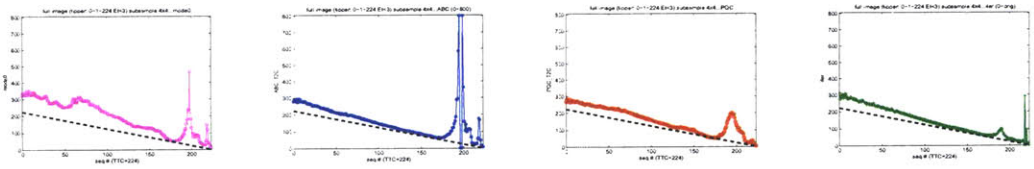
Category	Seq.[len] & Fig.#	Tested impact factors & Result figures
Category 1: Synthetic sequences		
(640× 480)	SEQ_1 [225] Fig. (6-9)	Case models on TTCs: Fig. (6-11) Case models on FOEs & P/Qs: Fig. (6-12), (6-13) Case models & Subsampling rates on TTCs: Fig. (6-14), (6-15) Case models & Subsampling rates on FOEs: Fig. (6-18), (6-19) Case models & Subsampling rates on P/Qs: Fig. (6-22)
	SEQ_2 [150] Fig. (6-10)	Case models & Subsampling rates on TTCs: Fig. (6-16), (6-17) Case models & Subsampling rates on FOEs: Fig. (6-20), (6-21) Case models & Subsampling ratess on P/Qs: Fig. (6-23)
Category 2(a): stop-motion seqs: motion along optical axis Sony camcorder & Canon Camera		
Camcorder (640× 480)	SEQ_3 [103] Fig. (6-24)	Case models & Segmentation on TTCs: Fig. (6-26)(top row), (6-27)(top row) Case models & Segmentation on FOEs: Fig. (6-29)(top 2 rows), (6-30)(top 2 rows) Case models & Segmentation on P/Qs: Fig. (6-31)(top 2 rows)
Canon (968× 644)	SEQ_4 [81] Fig. (6-25)	Case models & Segmentation on TTCs: Fig. (6-26)(bottom row), (6-27)(bottom row) Case models & Segmentation on FOEs: Fig. (6-29)(bottom 2 rows), (6-30)(bottom 2 rows) Case models & Segmentation on P/Qs: Fig. (6-31)(bottom 2 rows)
Category 2(b): stop-motion seqs: motion off optical axis, Canon Camera		
Side view (1944× 1296)	SEQ_5 [103] Fig. (6-32)	Case models & Segmentation on TTCs: Fig. (6-33) Segmentation & Fusion on TTCs: Fig. (6-34) (Case models & Subsampling rates on TTCs)
Front view (1936× 1288)	SEQ_6 [106] Fig. (6-35)	Segmentation & Fusion on TTCs: Fig. (6-38) (Case models & Subsampling rates on TTCs)
	SEQ_7 [107] Fig. (6-36)	Segmentation & Fusion on TTCs: Fig. (6-38) (Case models & Subsampling rates on TTCs)
	SEQ_8 [107] Fig. (6-37)	Segmentation & Fusion on TTCs: Fig. (6-38), (6-43), (6-42) Comparison with size-based method: Fig. (6-43), (6-42)
Category 3: outdoor video seqs taken with a moving camera		
(400× 300)	SEQ_9 [299] Fig. (6-39)	Segmentation & Fusion on TTCs: Fig. (6-41) Comparison with size-based method: Fig. (6-40)

6.5.3 Synthetic image sequences: time-to-contact computation based on different computational models and different subsampling rates

Figure 6-9: Sample frames of Sequence SEQ_1, synthetic image sequence "tipper" $(0-224$ frames). Frame seq: 0, 40, 80, 120, 160, 200.

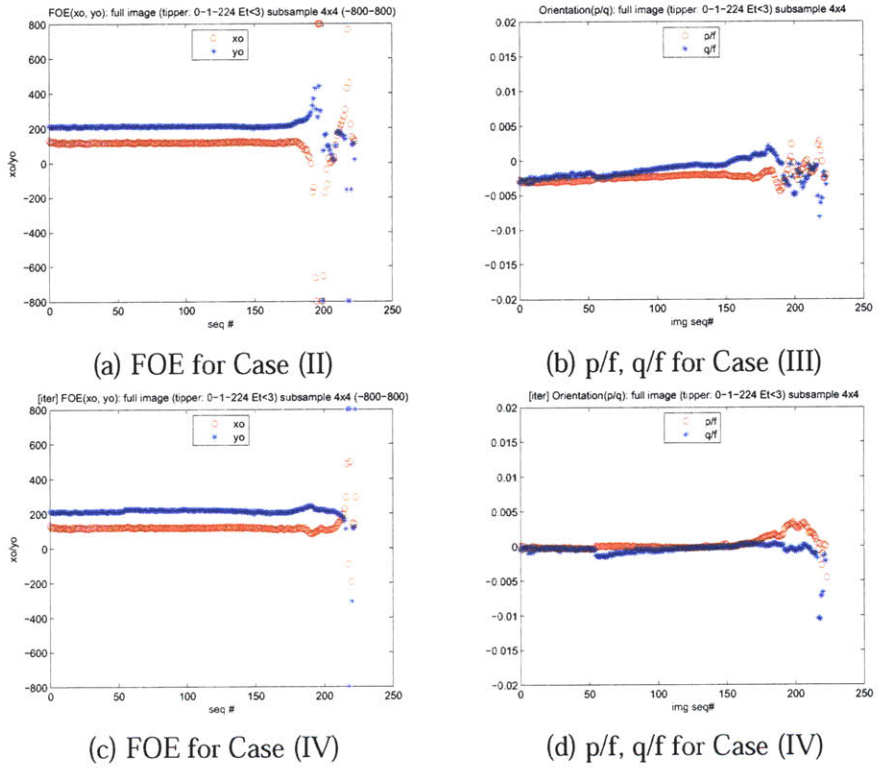
Figure 6-10: Sample frames of Sequence SEQ_2, synthetic image sequence "newman" $(0-149$ frames). Frame seq: 0, 25, 50, 75, 100, 125.

Our synthetic image sequence, SEQ_1 and SEQ_2, were generated from the side views of two different large trucks which were shifted and magnified. Thus objects are strictly planar objects. We first compare the impact of four computational models (at subsampling rate 4×4) on TTC, FOE and object-shape parameters (p, q) for Sequence SEQ_1 as presented respectively in Figures (6-11), (6-12) and (6-13). Then we compare TTC results at multiple subsampling rates. Figures (6-14) and (6-16) collectively plot TTC from four computational models in one sub-figure for each subsampling rate. Figure (6-15) and (6-17) collectively plot TTC at different subsampling rates in one sub-figure for each computational model. The instabilities are due to the impact of big subsampling rates (64×64) which decrease the number of pixel inputs and make matrix singular during TTC computation, which is discussed in details in Section 6.5.3 and Section 6.4.1. We choose such large subsampling rates for discussion purpose, which can be avoided in real applications.



Case (I) Case (II) Case (III) Case (IV)

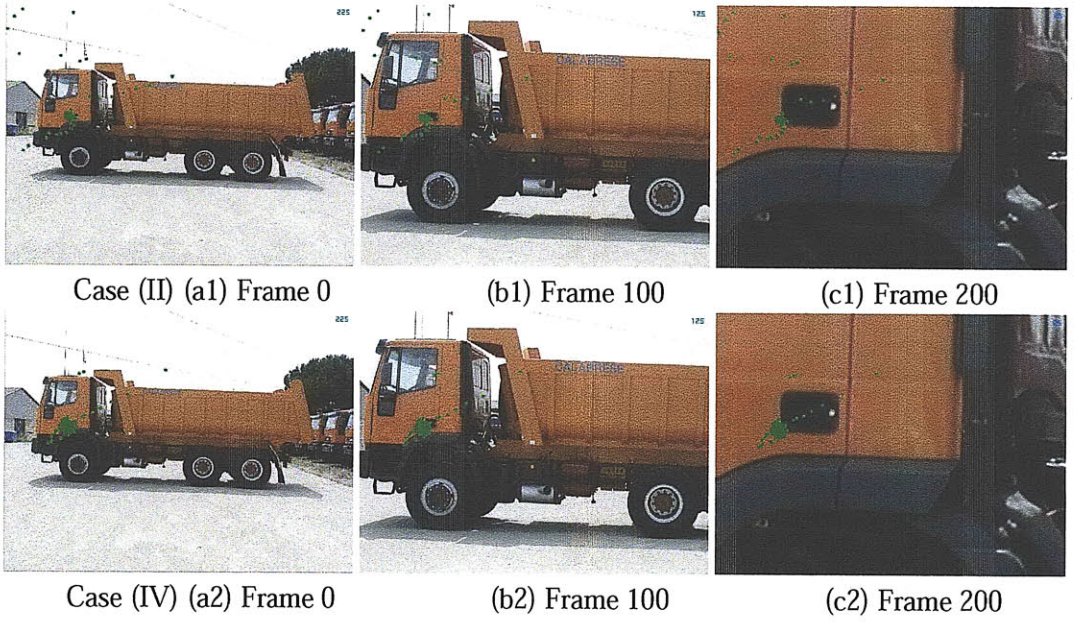
Figure 6-11: TTC calculated based on 4 assumptions at subsampling rate 4×4 for Sequence SEQ_1, synthetic image sequence "tipper" compared to true TTC (dashed black line). x axis: frame number; y axis: TTC. All figures are plotted with the same scales. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line.



(a) FOE for Case (II) (b) p/f, q/f for Case (III)

(c) FOE for Case (IV) (d) p/f, q/f for Case (IV)

Figure 6-12: Parameter computation for object motion and object orientation based on Case (II), (III) and (IV) at subsampling rate 4×4 for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: FOE and (p, q) parameters. The FOE coordinates are determinded in a coordinate system with its origin at the left/top corner in images. Figures in the same column are plotted with the same scales. (a)/(c): FOE for case (II)/(IV). FOE coordinates x_0 and y_0 are respectively represented by red and blue lines. (b)/(d): object orientation parameters for case (II)/(IV). Parameters p/f and q/f are respectively represented by red and blue lines.



Case (II) (a1) Frame 0 (b1) Frame 100 (c1) Frame 200

Case (IV) (a2) Frame 0 (b2) Frame 100 (c2) Frame 200

Figure 6-13: FOE results plotted on different image frames at subsampling rate 4×4 for Sequence SEQ_1, synthetic image sequence "tipper." (a1)(b1)(c1): FOE results at frame 0, 100, 200 from Case (II). (a2)(b2)(c2): FOE results at frame 0, 100, 200 from Case (IV). This Figure helps to identify the FOE directly by observing image pixels at the same location corresponding to the same object points.

Comparison of TTC results based on different computational models for synthetic images

Figure (6-11) shows calculated TTC values for four models. The overestimate of TTCs in all cases are mainly due to the temporal aliasing and the intrinsic mechanism how synthetic images are generated, which will be explained in more details in Section 6.5.3.

If we ignore the overestimate when TTC is very small, the TTC values drop linearly as expected. The results for Case (IV) fit the ground-truth-line the best while the results for Case (II) and (III) also closely approximate the linear ground-truth-line. The latter is because the process that synthetic images were produced roughly simulated the physical model described in Case (II) or Case (IV). In Figure (6-12), the results of orientation parameters p/f and q/f for Case (III) model are very close to zero, which confirms that the arbitrary planar object in the model is actually perpendicular to the optical axis.

In Figure (6-12), FOE results are at the left side of the image center instead of the center of an image, which shows that the relative movements for synthetic sequences are not exactly along the optical axis. Though the assumption for Case (III) does not hold, Case (III) model still yields acceptable TTC results, which is mainly due to pure planar objects. Further discussions on different computational models in Section 6.5.4 and Section 6.5.5 shows that Case (III) models normally would not yield ideal results if the direction of relative movements are not exactly along the optical axis.

Since the direction of relative motion is not too far away from along the optical axis, the image generating setup for Category 1 is also close to Case (I) model. Therefore, TTC estimation based on Case (I) in Figure (6-11)(a) is also acceptable.

We observe the similar trend when different subsampling rates are used in TTC computation as shown in Figures (6-14) and (6-15) for Sequence SEQ_1 and Figures (6-16), (6-17) for Sequence SEQ_2.

Comparison of FOE/orientation results based on different computational models for synthetic sequences

For SEQ_1, the results of FOE points for all image frames are plotted together in three different images as shown in Figure (6-13). The group of FOE points from both Case (II) and (IV) are at the top and left region of images. The computed FOE coordinates are around position $(119, 216)$ in a coordinate system with its origin at the left/top corner in images as shown in Figure (6-12). In a coordination system with origin at image centers as used in our initial computational models, we have $x_0 = f(U/W) < 0$ and $y_0 = f(V/W) < 0$. It shows that the relative motion of object w.r.t to cameras are $U > 0$, $V > 0$, and $W < 0$, which corresponds to the moving trend of the whole sequence for SEQ_1. Specifically, we observe that computed FOE positions do correspond to the same object feature points at three image frames, the left/bottom corner of a black rectangular.

Since the motion direction is fixed, FOE results are supposed to be constant when being computed at different image frames or using different computational models. As expected, the TTC curves in Figure (6-12) are roughly straight and FOE points in Figure (6-13) are quite close to each other except for a few outliers.

In Figure (6-12) and Figure (6-13), FOE points are more spread out and have more outliers for Case (II) than for Case (IV). The area of clustered FOE points for Case (IV) is much larger than for Case (II). Thus, the Case (IV) provides better FOE estimation than Case (II).

We can observe the similar trend for FOE results with different subsampling rates shown in Figures (6-18) and (6-19) for Sequence SEQ_1, Figures (6-20) and (6-21) for Sequence SEQ_2, and Figure (6-29) for stop-motion sequences SEQ_3 and SEQ_4.

There exists similar observation for orientation parameters. Since the object orientation is fixed and perpendicular to the optical axis, $(p/f, q/f)$ results are supposed to be close to zero and to remain constant when being computed at different image frames, with different parameters or different computational models. As expected, the computation results of parameters p/f and q/f for Cases (III) and (IV) as shown in Figure (6-12)(c) and (d) are all close to zero. The $p/f, q/f$ curves in Figure (6-12) are quite constant for the first 180

frames.

The computed results of $p/f, q/f$ for Case (IV) are better than for Case (III). There are less outliers and less variations for computed data for Case (IV) than for Case (III). Case (IV) provides better orientation estimation than Case (III).

We can observe the similar trend for the computation of object-shape parameters with different subsampling rates shown in Figure (6-22) for Sequence SEQ_1, Figure (6-23) for Sequence SEQ_2, and Figure (6-31) for stop-motion sequences SEQ_3 and SEQ_4.

However, we do notice that FOE curves are much smoother than (p, q) curves, which means that the estimation reliability for FOE points is higher than the orientation parameters.

Necessity of subsampling

After examining the data, we conclude that the outliers at the end of TTC, FOE, (p, q) curves for SEQ_1 and SEQ_2, are caused by two reasons. The first reason is due to the intrinsic mechanism how synthetic images are generated. At the end of sequences, the original image is greatly magnified and so appears "out of focus" or blurry, lacking high frequency content, and may have artifacts resulting from the interpolation method. The second reason is due to temporal aliasing, or the very large frame to frame change near the end of the sequence. For example, when the TTC is 10, the size of the image of an object changes by $1/10 = 10\%$. However, for our gradient-based methods, frame to frame image movement should not be greater than about half the size of typical "texture elements" in order to produce reliable results. If the expected size of texture elements is not known, motion should typically be less than about a pixel from frame to frame in most parts of the image region used in order to obtain reasonable results [68]. Therefore, in order to produce reliable TTC/FOE/(p, q) estimation, subsampling should be applied in order to increase the effective "pixel" size. TTC estimates with subsampling are more reasonable and closer to the actual time-to-contact than the ones without subsampling.

Comparison of TTC results based on different subsampling rates

TTC comparisons at different subsampling rates, as shown in Figures (6-14) and (6-15) for SEQ_1 and Figures (6-16) and (6-17) for SEQ_2, imply that spatial averaging and subsampling can change the range in which TTC estimates are reliable.

When there is no subsampling, i.e., when images are not spatially averaged and subsampled, TTC results from four cases significantly differentiated from the truth line though they still reflect the fast-approaching trend of that object. There are systematic overestimating TTC values. The bias drops significantly with 2×2 block averaging and sub-sampling. The time-to-contact curves are quite close to each other for different subsampling rates at 2×2, 4×4, 8×8, 16×16, 32×32. In Figures (6-16) and (6-17) for Sequence SEQ_2, sub-sampling using 4×4 block averages yielded reliable TTC estimates down to a TTC of about 60 frames. With 16×16 block averaging and subsampling, good results are obtained down to a TTC of less than 15 frames. TTC computation results might be still unreliable for small subsampling rates near the end of the sequence.

Increasing spatial averaging and subsampling, however, comes with a price. In Figure (6-14), as for the estimation of *large* TTCs at the beginning of the sequence, subsampling using 16×16 block averages yields less accurate TTC estimates than using 8×8 block averaging and subsampling. Large TTCs are difficult to be estimated accurately because the frame to frame motion of image patterns is only a small fraction of a pixel. This problem becomes more difficult with spatial subsampling, especially when foreground objects are far away and have small image sizes. Big subsampling rates decrease the number of pixel inputs during TTC computation. Thus there are large oscillations on TTC curves at the beginning of the sequence when TTCs are still large. *Temporal* averaging and subsampling can extend the range of TTC estimates to larger values.

The above observations suggest that a wide range of TTCs can be estimated reliably if spatial averaging and subsampling is used for small values of TTCs and temporal averaging and subsampling for large values of TTCs. The results are not ideal if the subsampling rates are too big at the beginning of sequences or if the subsampling rates are too small at the end of sequences.

Comparison of TTC results based on different subsampling rates and computational models for synthetic sequences

For synthetic image sequences, the methods for Cases (II),(III), and (IV) all provide acceptable results with respect to different input parameters, especially for results from Case (IV) as shown in the bottom row of Figure (6-15). TTC results based on Cases (II)(III)(IV) are more robust to subsampling rates than for Case (I). However, with suitable subsampling rates, simple model for Case (I) can also produce acceptable TTC results.

Comparison of FOE/orientation results based on different subsampling rates and computational models for synthetic sequences

The subsampling rates have low influence on the computation of motion information FOE (x_0, y_0) and plane orientation $(p/f, q/f)$. The computation of FOEs and object-shape parameters are a little robust to the choices of subsampling rates except for the very small subsampling rates (1×1), (2×2) or the very large subsampling rate (64×64). Similar to the previous observation made for different subsampling rates, most data curves for both FOE and $(p/f, q/f)$ are roughly straight as shown in Figures (6-18), (6-20) and Figures (6-22), (6-23). Most FOE points in Figure (6-19) and (6-21) are close to each other. As expected, FOE results from Case (IV) are better than from Case (II) and the results of object orientation from Case (IV) are better than from Case (III). In summary, iteration based methods yield better results for both TTC and other shape/motion parameters.

6.5.4 Stop-Motion image sequences with translational motion along the optical axis

In this section, we test our algorithms for two stop-motion sequences from Category 2(a), SEQ_3 and SEQ_4, with relative motion along the optical axis. Note that SEQ_3 has mechanical vibrations when image frames were taken. For both sequences, TTC results are computed at subsampling rate 2×2 along with four different computational models based on full images and the segmented/labeled rectangular regions, which are marked in red rectangular boxes in Figure (6-24), (6-25).

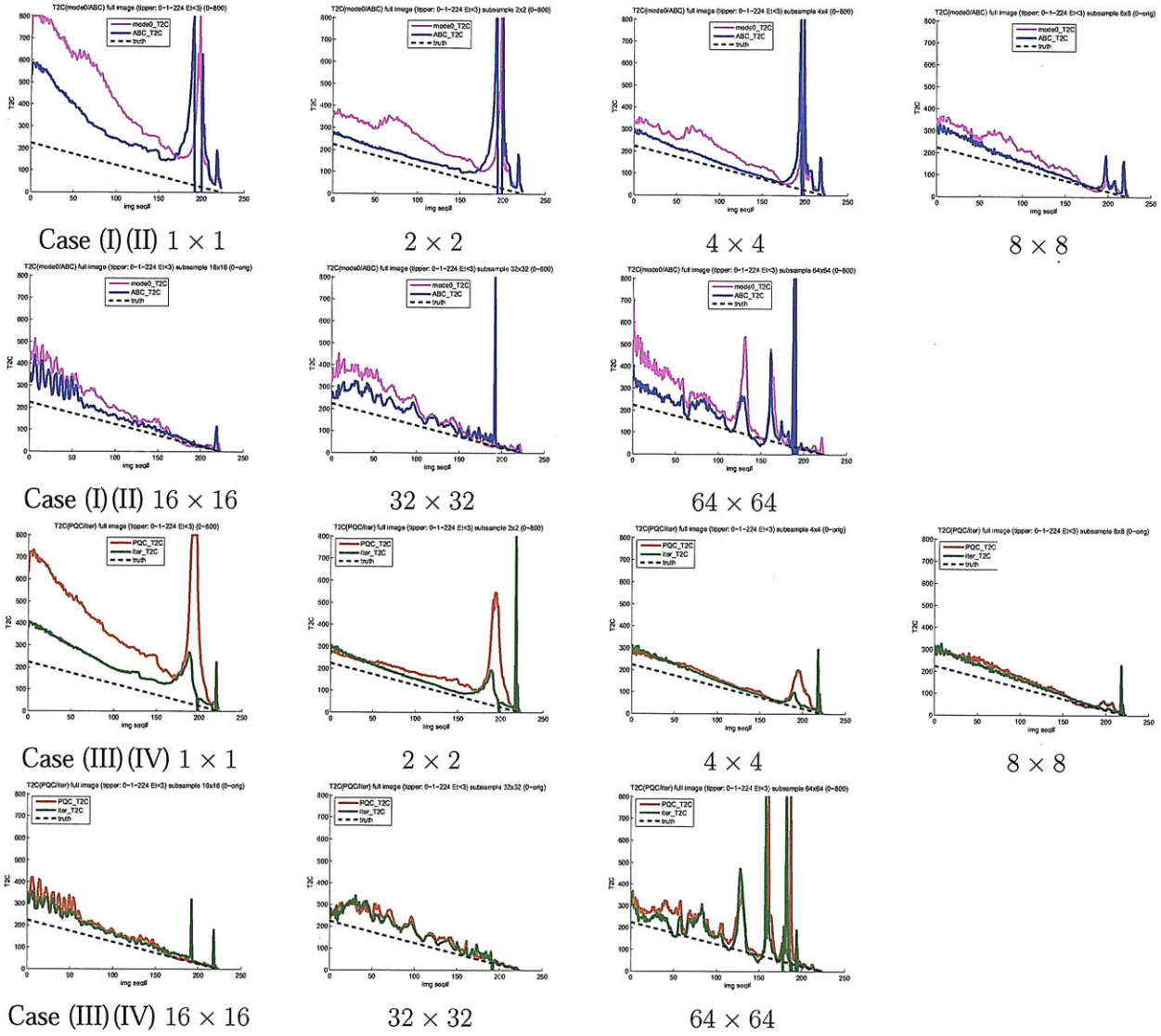


Figure 6-14: TTC computation (for Case (I) (II) and Case (III) (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: TTC. Dashed black line: TTC ground-truths. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. All figures are plotted with the same scales.

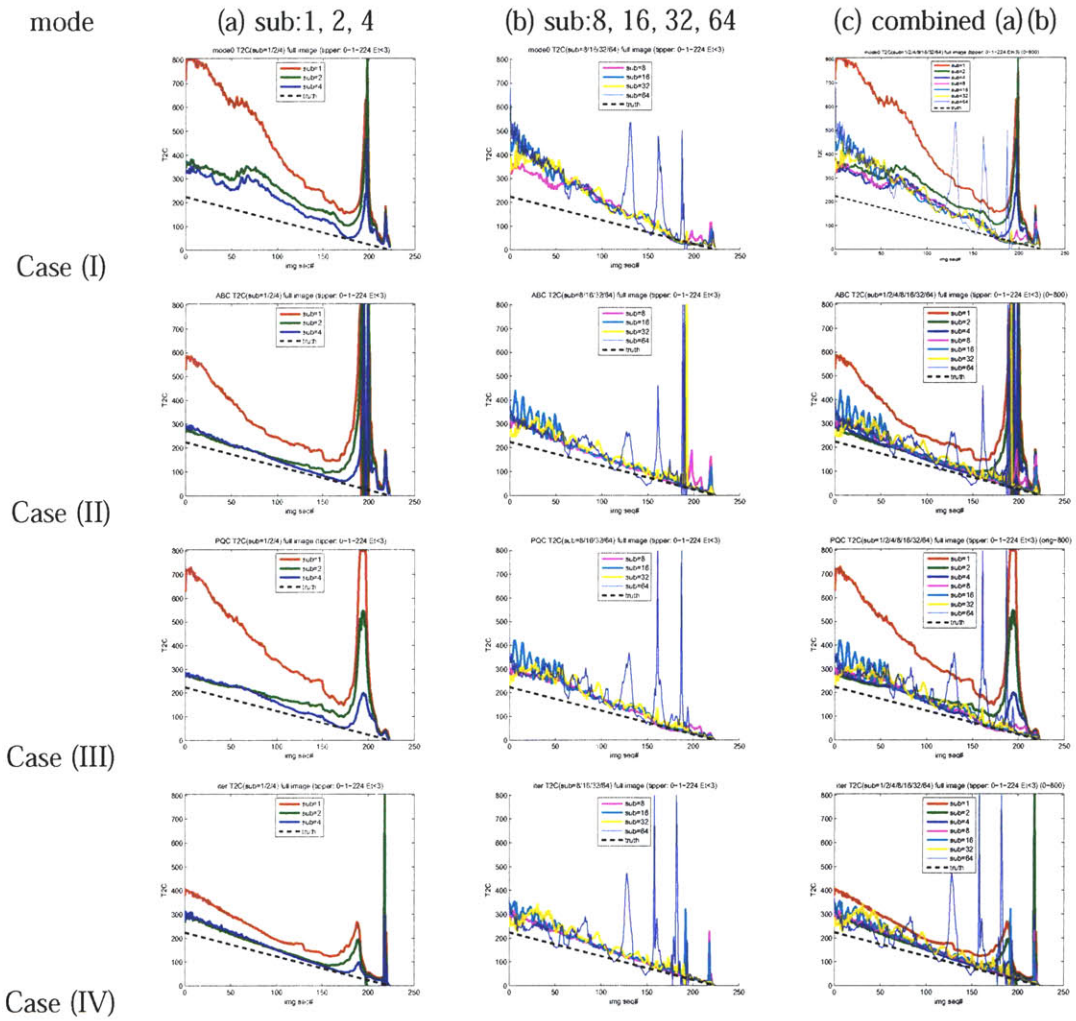


Figure 6-15: TTC comparison at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for four cases (Sequences: tipper). Horizontal axis: frame number. Vertical axis: TTC. Dashed black line: TTC ground-truths. 1×1: Red thick lines. 2×2: Green thick lines. 4×4: Blue thick lines. 8×8: Magenta thick lines. 16×16: Cyan thick lines. 32×32: Yellow thick lines. 64×64: Blue thin lines. All figures are plotted with the same scales.

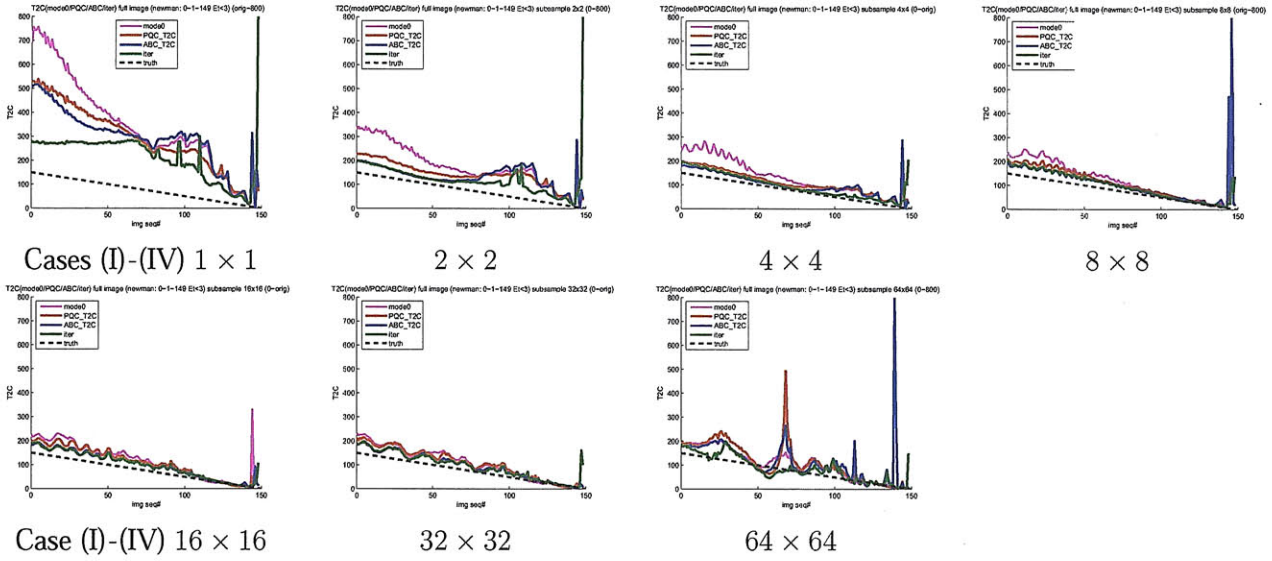


Figure 6-16: TTC computation (for all cases) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_2, synthetic image sequence "newman." x axis: frame number; y axis: TTC. Dashed black line: TTC ground-truths. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. All figures are plotted with the same scales.

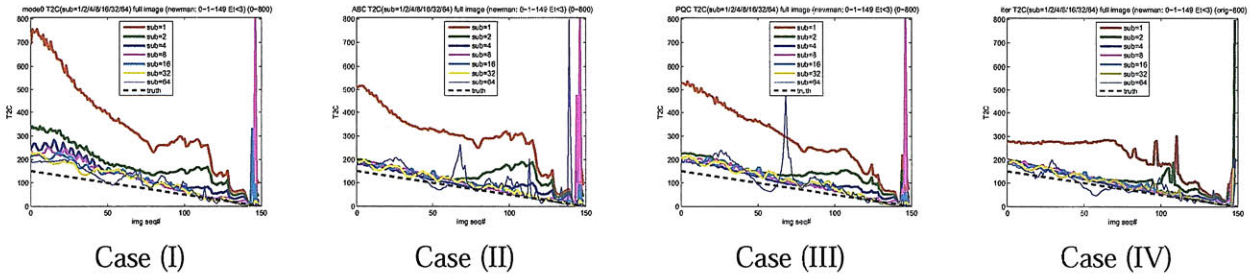


Figure 6-17: TTC comparison at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for all cases (Sequences: newman). Arbitrary planar orientation and arbitrary translational motion. Horizontal axis: frame number. Vertical axis: TTC. Dashed black line: TTC ground-truths. 1×1: Red thick lines. 2×2: Green thick lines. 4×4: Blue thick lines. 8×8: Magenta thick lines. 16×16: Cyan thick lines. 32×32: Yellow thick lines. 64×64: Blue thin lines. All figures are plotted with the same scales.

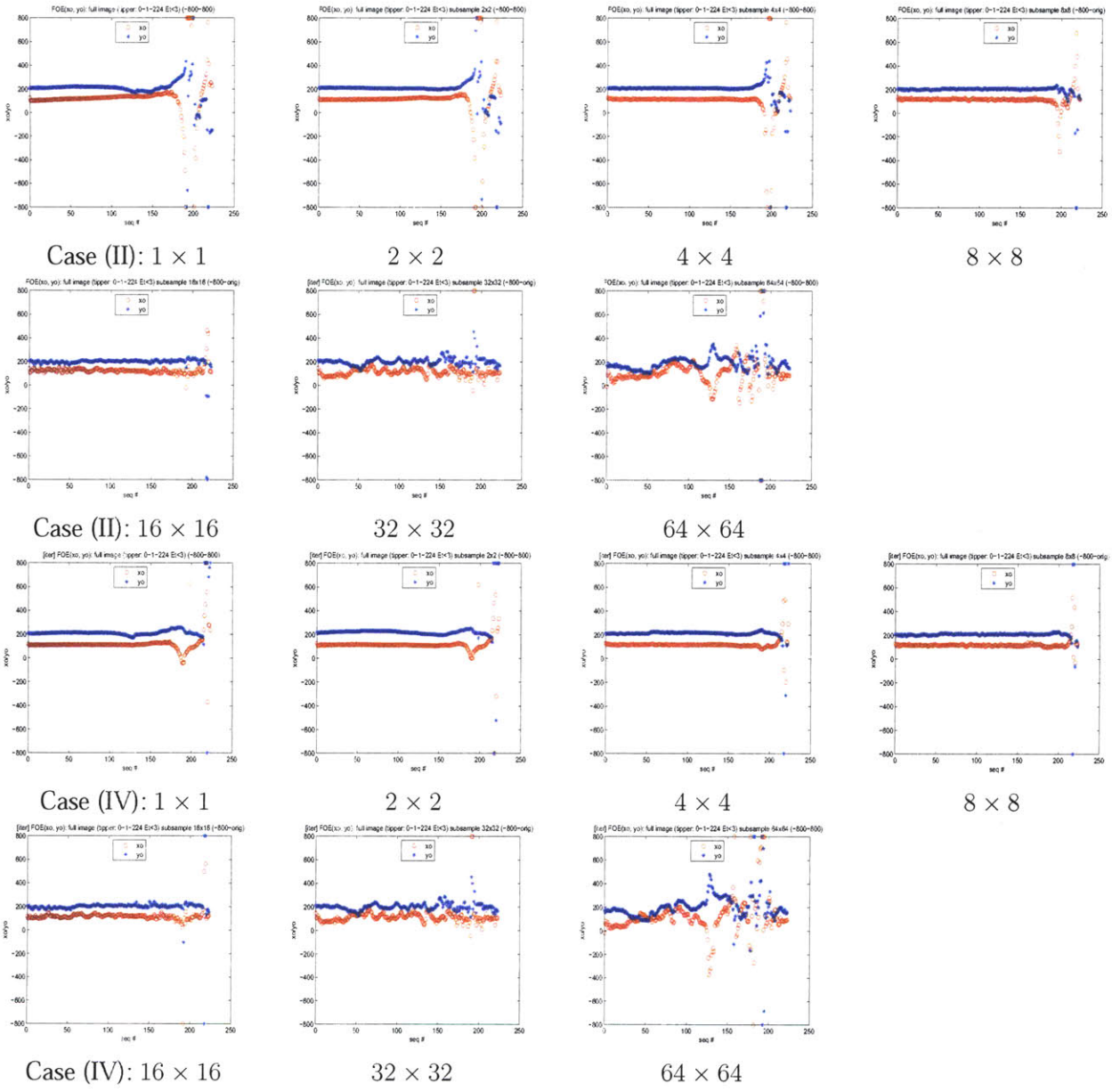


Figure 6-18: FOE computation (for Case (II) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: FOE coordinates x_0 and y_0 respectively represented by red and blue lines. All figures are plotted with the same scales.

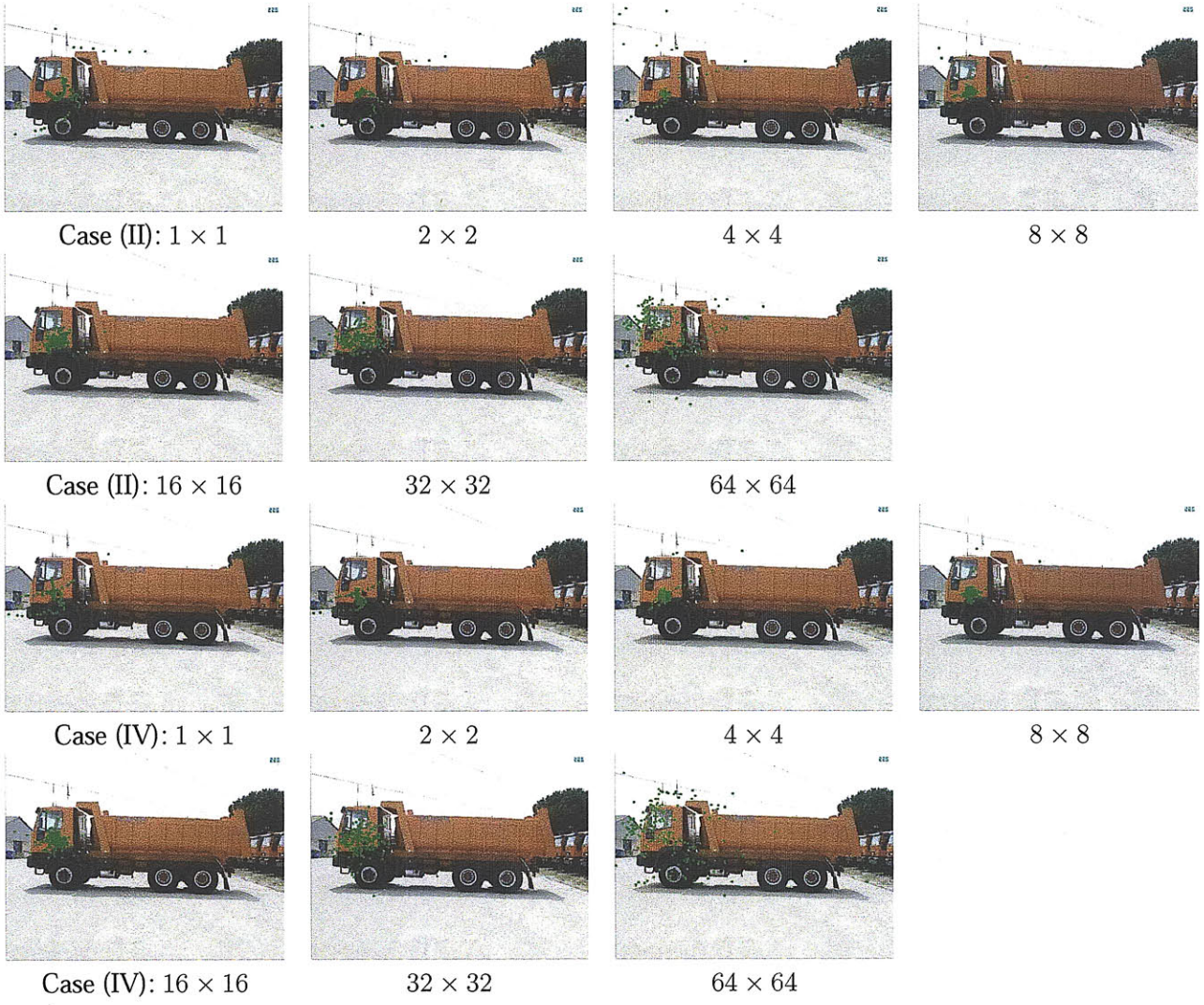


Figure 6-19: FOE comparison for Case (II) and Case (IV) at different subsampling rates(1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64). FOE points are plotted on the first image of synthetic image sequence(tipper). Top two rows: Case (II). Bottom two rows: Case (IV)

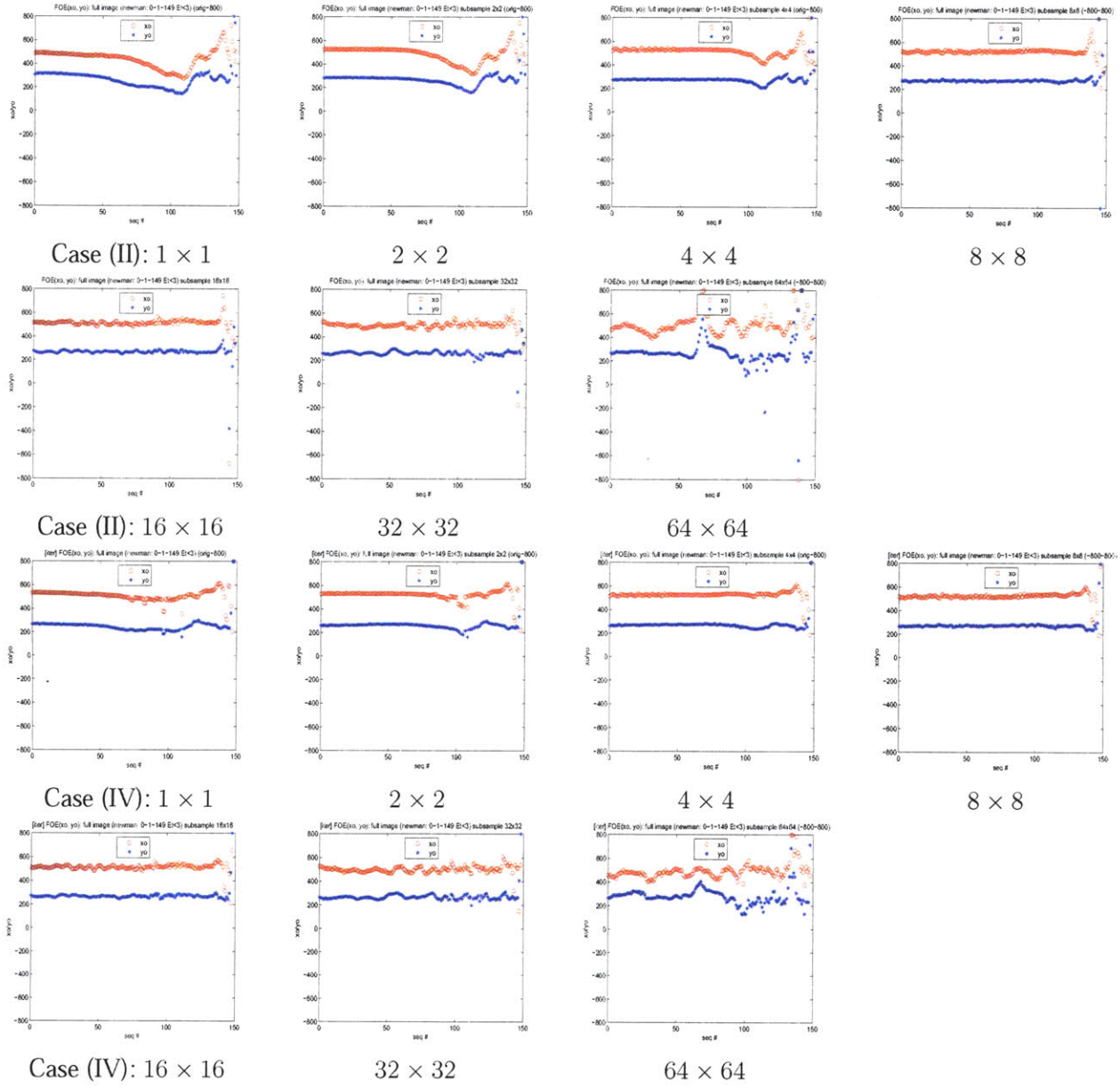


Figure 6-20: FOE computation (for Case (II) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_2, synthetic image sequence "newman." x axis: frame number; y axis: FOE coordinates x_0 and y_0 respectively represented by red and blue lines. All figures are plotted with the same scales.

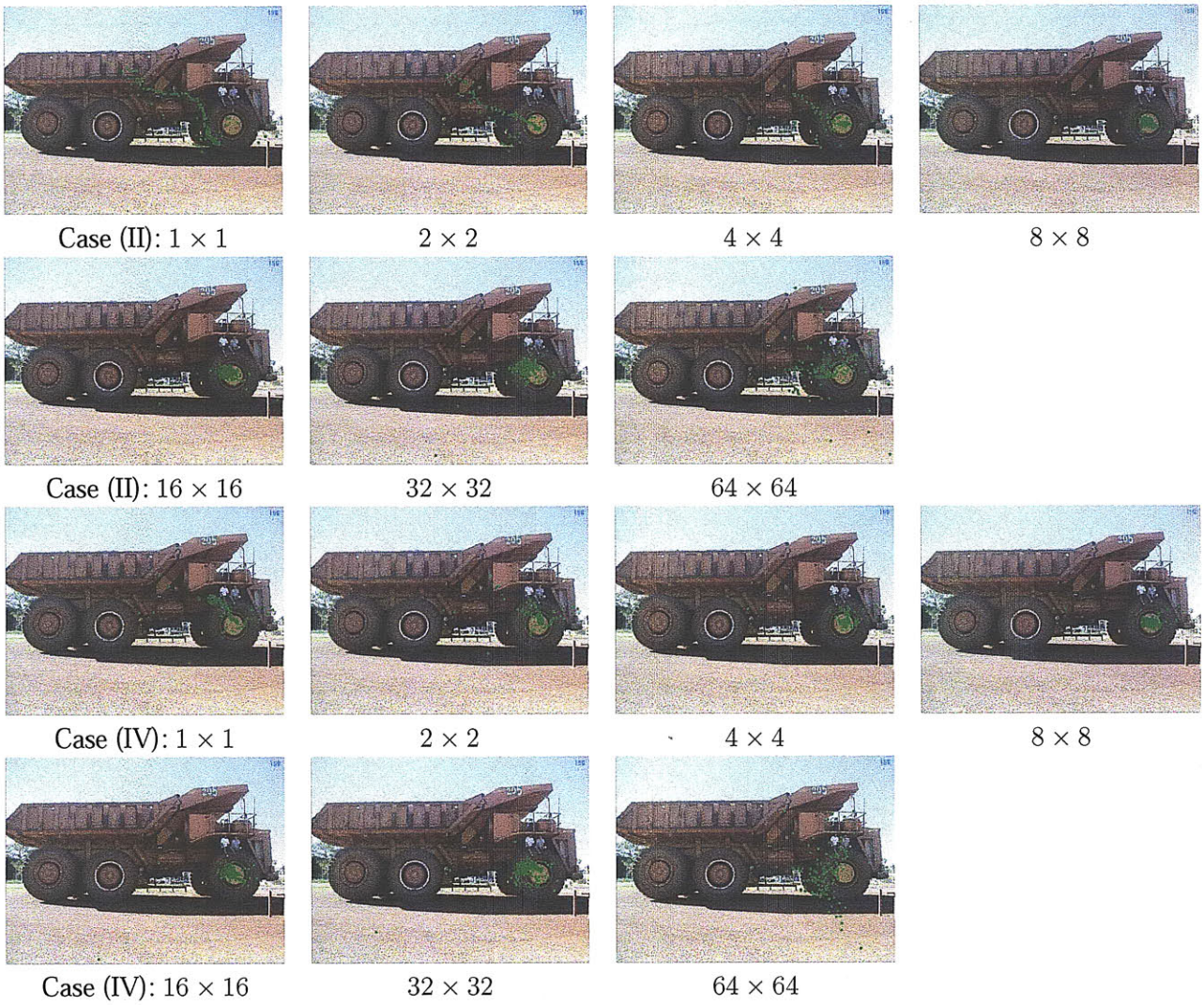


Figure 6-21: FOE comparison for Case (II) and Case (IV) at different subsampling rates(1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64). FOE points are plotted on the first image of Sequence SEQ_2, synthetic image sequence "newman." Left column: Case (II). Right column: Case (VI)

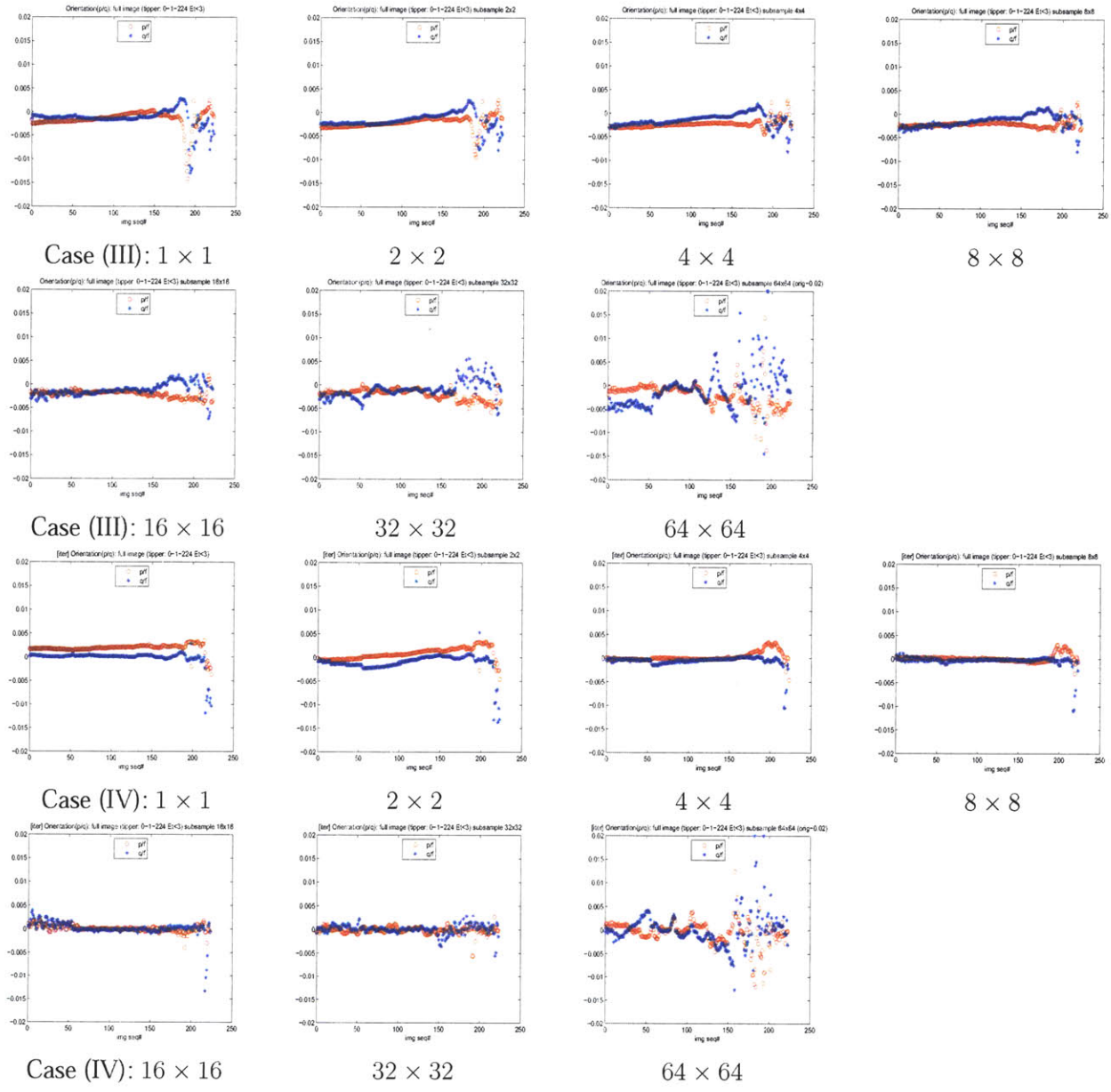


Figure 6-22: Estimated plane orientation parameters (for Case (III) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_1, synthetic image sequence "tipper." x axis: frame number; y axis: parameter $(p/f, q/f)$ respectively represented by red and blue lines. All figures are plotted with the same scales.

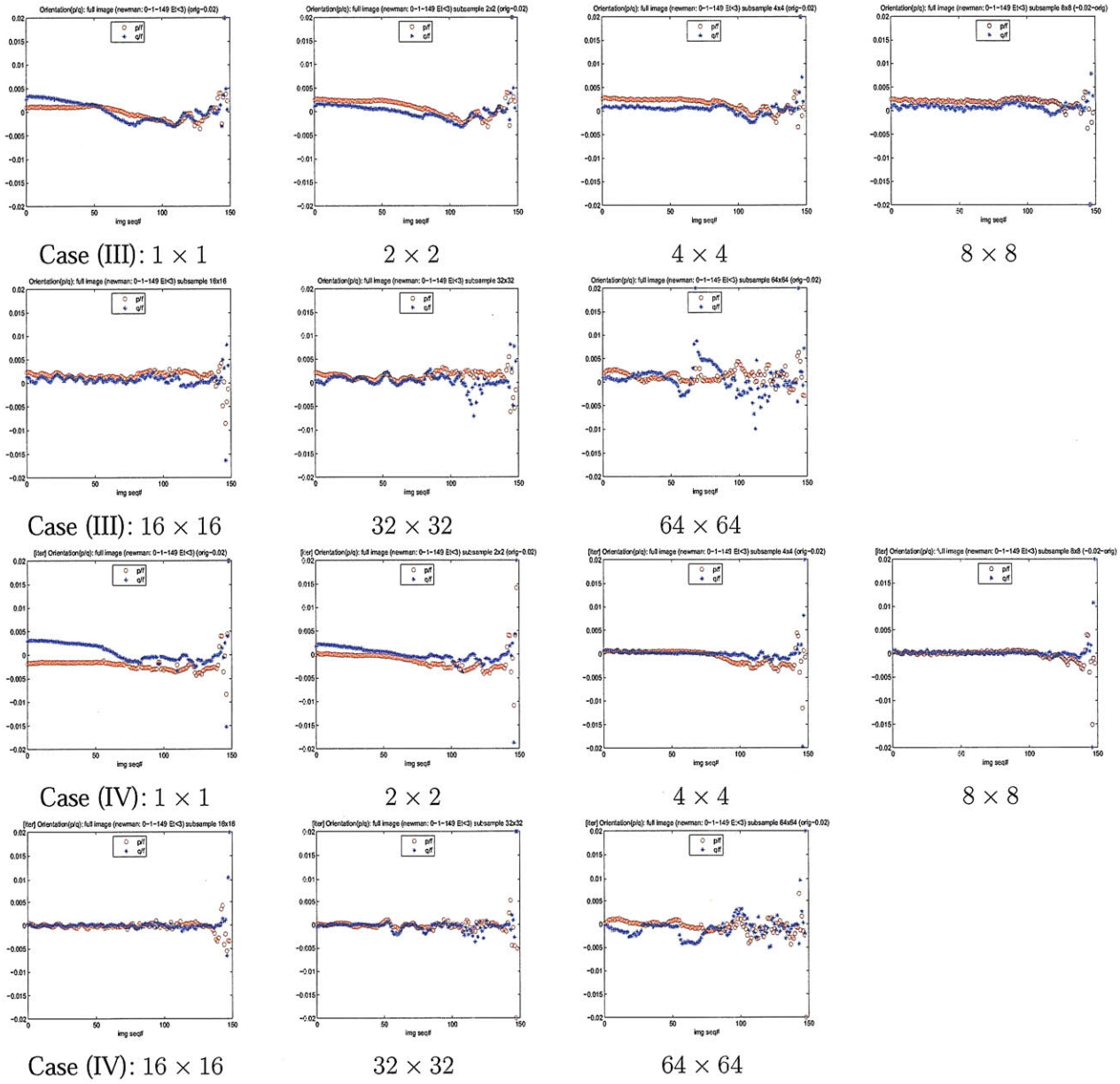


Figure 6-23: Estimated plane orientation parameters (for Case (III) and (IV)) at different subsampling rates (1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64) for Sequence SEQ_2, synthetic image sequence "newman." x axis: frame number; y axis: $(p/f, q/f)$ values respectively represented by red and blue lines. All figures are plotted with the same scales.

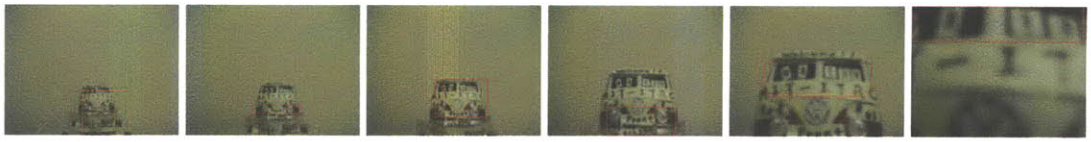


Figure 6-24: Sample frames and their segmentation for Sequence SEQ_3, stop-motion image sequence "new_Bus_Front_5mm" (104 frames). Frame seq: 1, 21, 41, 61, 81, 101.

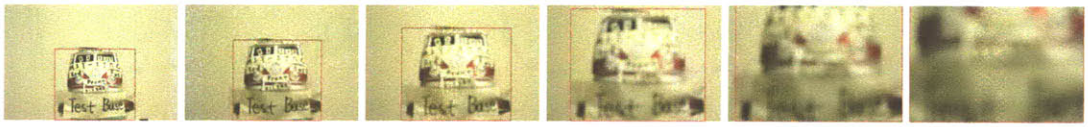


Figure 6-25: Sample frames and their segmentation for Sequence SEQ_4, stop-motion image sequence "newCam_Bus_Front_5mm" (81 frames). Frame seq: 1, 16, 31, 46, 61, 76.

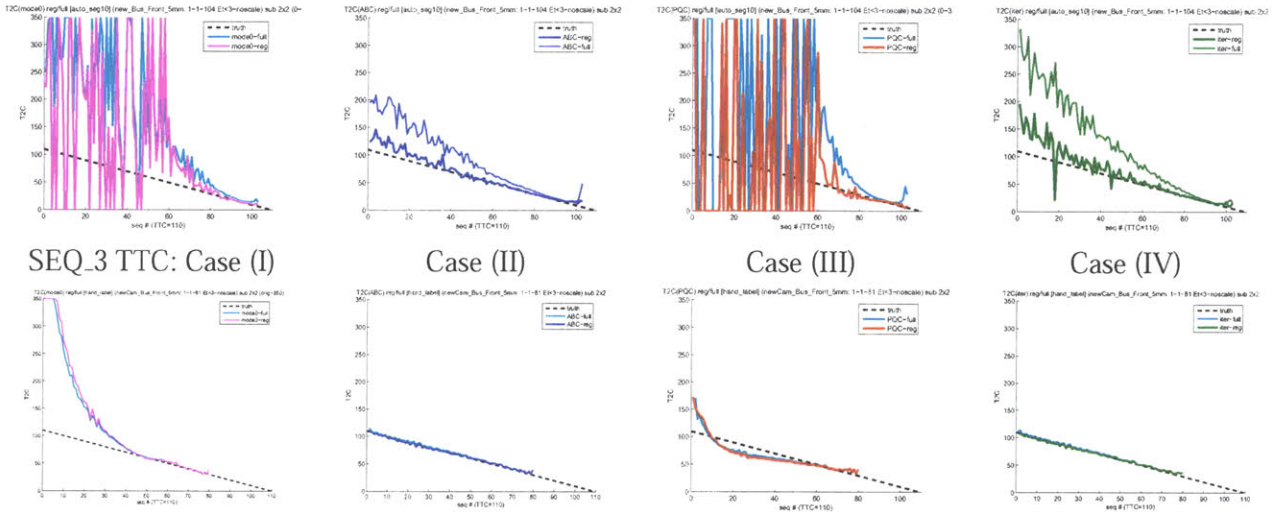


Figure 6-26: The comparison of TTC computation based on full image and the segmented area for different cases at subsampling rate 2×2 for Sequence SEQ_3, stop-motion sequence "new_Bus_Front_5mm," and Sequence SEQ_4, stop-motion sequence "newCam_Bus_Front_5mm." x axis: frame number. y axis: TTC. Dashed black line: TTC ground-truths. TTC thin/thick lines: based on full image or segmented areas. TTC results based on full images/segmented regions: Case (I): Magenta/Cyan lines, Case (II): Blue thin/thick lines, Case (III): Red/Cyan lines, Case (IV): Green thin/thick lines. All figures are plotted with the same scales.

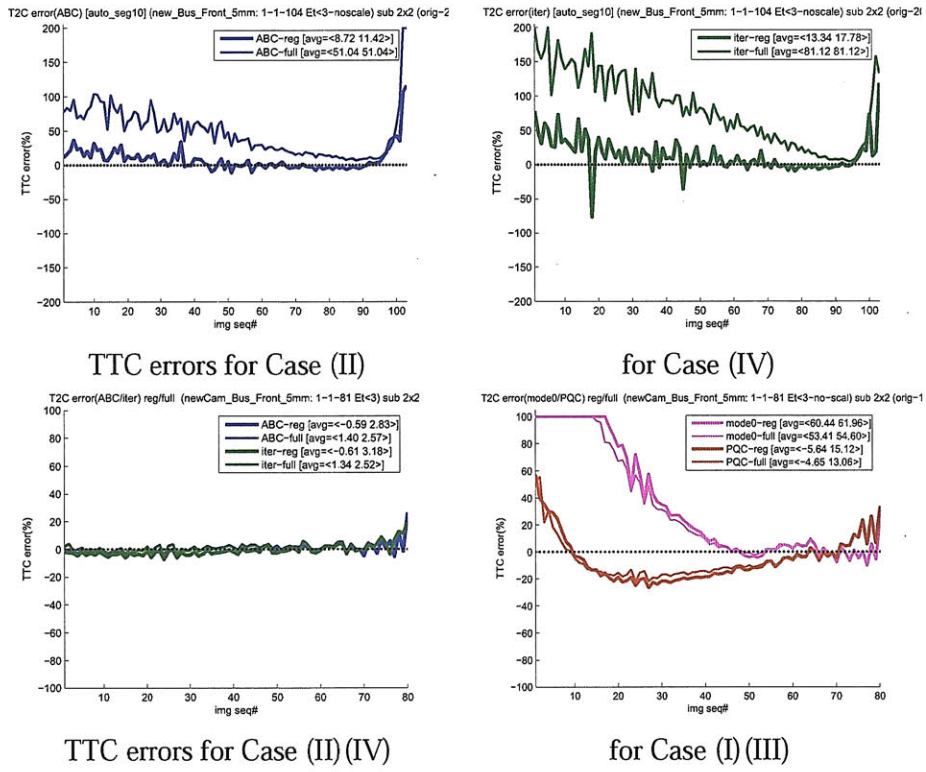


Figure 6-27: TTC deviation rates based on full image and the segmented area with different cases at subsampling rate 2×2 for stop-motion Sequence SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm." Top row: SEQ_3. Bottom row: SEQ_4. x axis: frame number. y axis: TTC deviation rate (TTC-truth)/truth. TTC thin lines: based on full image. TTC thick lines: based on segmented areas. Case (I): Magenta line. Case (II): Blue line. Case (III): Red line. Case (IV): Green line. All figures are plotted with the same scales.

Comparison of TTC results and ground-truths

As shown in Figures (6-26), (6-27) and Table (6.3), (6.4), for both sequences SEQ_3 and SEQ_4, the calculated TTC values based on both full image and segmented regions are quite close to the ground-truths which decrease linearly, especially for Sequence SEQ_4. For Sequence SEQ_4, the best result is from Case (II) with segmented region. As shown in Table (6.4), the average TTC estimation errors based on manually labeled region is 0.59% and 2.83% for Cases (II) and 0.61% and 3.18% for Cases (IV). The two error index are the average of error itself and the average of absolute error values. The results show that our computational models can provide ideal results for the test sequences from ideal imaging process.

Though Sequence SEQ_3 is produced by a camcorder with low image qualities and vibration noises, TTC results in the top row of Figure (6-26) are also close to ground-truths. As shown in Table (6.4), the average TTC estimation errors based on segmented region for Case (II) are 8.72% and 13.34% for Case (IV). If we ignore the frames from 95 to 103 (which are affected by defocus and large motion), the average TTC estimation error based on segmented region for Case (II) is 5.21% and 11.58% for Case (IV). As shown in Figure (6-28), image frame at 95 is affected by defocus, but it has captured all foreground elements.

When relative distances between objects and cameras are small, i.e., when TTCs are small, TTC estimation results from most computational models are all very close to ground-truth. For SEQ_3, the average TTC estimation errors for frames within range 61-103 in segmented regions is 23.08% for Case (I) and 5.2% for Case (III) as in the top row of Figure (6-26). For frame range 71-95, the average error of TTCs with segmentation for Case (I) is 8.19% and -10.49% for Case (III). At image frame 61, TTC results for Cases (I) and (III) stop oscillation. At image frame 71, TTC results for Cases (I) and (III) start to fit ground truths very well. Figure (6-28) shows the three transition frames at 61, 71, and 95.

At the very end of Sequence SEQ_3, the estimation errors for small values of TTCs show the tendency of going up. As seen in the top row of Figure (6-26), the error rates start

Table 6.3: The error rate (percent) of TTC estimation in the bottom row of Figure (6-26) for Sequence SEQ_3.

method	reg/case(II)	reg/case(IV)	full/case(II)	full/case(IV)
avg	8.72	13.34	51.04	81.12
avg(abs)	11.42	17.78	51.04	81.12
avg([1:94])	5.21	11.58	45.02	83.07
avg(abs[1:94])	8.17	16.44	45.02	83.07

Table 6.4: The error rate (percent) of TTC estimation in the bottom row of Figure (6-26) for Sequence SEQ_4.

method	full/case(I)	full/case(II)	full/case(III)	full/case(IV)
avg	53.41	1.40	-4.65	1.34
avg(abs)	54.60	2.57	13.06	2.52
method	reg/case(I)	reg/case(II)	reg/case(III)	reg/case(IV)
avg	60.44	-0.59	-5.64	-.61
avg(abs)	61.96	2.83	15.12	3.18

to increase significantly starting from frame No.95. The inaccuracy is due to large motions of patterns from frame to frame and de-focus, as explained in Section 6.5.3.

Figure 6-28: The transition image frames for Sequence SEQ_3, stop-motion sequence "new_Bus_Front_5mm(104 frames)." Frame seq: 61, 71, 95.

Comparison of TTCs from different computational models for stop-motion sequences

For both sequences SEQ_3 and SEQ_4, Case (II) and Case (IV) produce more accurate results than Case (I) and Case (III) do as shown in Figure (6-26). Case (IV) provides the most satisfactory performance. Even if the front view of the toy bus is not completely planar and perpendicular to the optical axis (assumption for Case (II) and (IV)), the planar-object-based computation models yield satisfactory estimation results.

The setup for Category 2(a) is supposed to correspond to Case (I) and Case (III) where

approaching objects move along the optical axis. However the results for Case (I) and (III) are not as good as for Case (II) and (IV). This is because we do not have a systematic method to ensure the alignment between the optical axis of camera and the relative motion along optical bench though we try to. The computation results of FOE for both SEQ_3 and SEQ_4 in Figure (6-29) indicate that FOEs are right below the image centers. The departure of FOEs from image centers is due to the tiny rotation of camera around pitch axis. In other words, the camera somehow looked down at the moving object. The estimated orientation-parameters in Figure (6-31) are close to zero, which confirms that the object surface is actually perpendicular to the optical axis. Thus, the imaging model is actually close to Case (II). Not surprisingly, TTC results from Cases (II) and (IV) both yield very satisfactory estimation results.

Since the camera rotation around pitch axis is very small, the set up is not too far away from Case (I)(III). Thus TTC results for the second half of Sequences SEQ_4 and SEQ_3 based on Cases (I) and (III) with both full region and segmented area are not too far away from the ground-truth.

When the object is getting close to the camera, even the simplest model for Case (I) provides acceptable results for both sequences. The ability to quickly estimate TTC when TTC becomes small is very important in order to identify incoming danger.

In summary, TTC estimation is more robust to object orientation than to moving direction. Computational method for Case (II) normally yields better results than for Case (III). But computational model Case (IV) is always the best choice among all four cases.

TTC computation using full images vs. segmented regions

For SEQ_3, TTC results based on segmented regions are closer to ground-truths than based on full images as shown in the top row of Figure (6-26), even if segmentation results as shown in Figure (6-24) are not in high qualities. As explained in Section 6.3.2, results based on segmentation are generally more accurate than results based on full images. Our TTC computation does not depend heavily on the foreground boundaries and only needs suitable amount of foreground pixels as input data.

For SEQ_4, TTC results using both full images and segmentation areas are very similar

for all four computational models. This is because the "soft rejection" works as well as "hard rejection" for SEQ_4 as explained in Section 6.3.3 when pixels Background pixels in Sequence SEQ_4 have similar intensities. The threshold of time derivative effectively reduces the impact of background. When the object is getting close to the camera, TTC results with both full region and segmented area are not too far away from the ground-truth. In summary, the algorithm is robust to segmentation errors and does not require accurate segmentation.

FOE/orientation computation using full images vs. segmented regions for different computational models

For both SEQ_3 and SEQ_4, the computation of FOE is not sensitive to the choices of using full images or segmented regions for both Case (II) and Case (IV) as shown in Figures (6-29) and (6-30). FOE results are very robust to impact factors: computational models and computational regions.

However, the computation of orientation parameters is sensitive to impact factors. As shown in Figure (6-31), orientation parameters from Case (IV) is significantly better than those from Case (III). For computational model Case (IV), orientation parameters based on segmented regions are all better than those based on full images. The observation agrees with the discussion for synthetic sequences in Section 6.5.3.

Impact of mechanical vibrations on TTC/FOE/(p, q)

We notice that TTC results for SEQ_3 based on Cases (I) and (III) oscillate significantly in the first half of sequence as shown in Figure (6-26), and FOE points for SEQ_3 are widely spread especially along the vertical axis as shown in Figure (6-30)(top two rows). This is due to the nodding vibrations of Sony Camcorder each time when we press the button to take each image in SEQ_3, which introduced additional rotation along pitch axis and/or the additional translational motion in y dimension. The frequent changes of relative motion also contributes to the drifting of FOE points in y dimension for SEQ_4. In contrast, that FOE points for SEQ_4 are clustered together. The large oscillations in TTC for SEQ_3 show that TTC estimation is very sensitive to the direction of relative motion, which is also

consistent with our previous observation.

6.5.5 Stop-Motion image sequences with translational motion off the optical axis

In this section, we test our algorithms on stop-motion sequences from Category 2(b) in which a toy car moves forward along four different directions with yaw angles relative to the optical axis. The sequences correspond to the most general situations defined in computational model Case (IV). We want to evaluate how our algorithm performance in the most general situations and to demonstrate the advantages of TTC fusion scheme for side-view sequence, SEQ_5 using two different segmentation schemes, and for front-view sequences, SEQ_6, SEQ_7, SEQ_8.

Later in the discussion Section, 6.5.6 and 6.6.3, we have also computed the size-based TTC results and TTC fusion results for SEQ_8 based on condition-number in order to demonstrate the advantages of our proposed gradient-based and fusion-based algorithms over the traditional size-based algorithm.

Comparison of TTC results based on different computational models and segmentation schemes

We first discuss TTC results for side-view Sequence SEQ_5 from four computational models with both full images and segmented/labeled regions (marked in red rectangular boxes) at subsampling rate 2×2. The comparison results are shown in Figure (6-33) and in Table 6.5. When stop-motion sequences are produced, the relative motion between objects and cameras are arbitrary which in general corresponds to Case (IV) shown in Figure (6-5). Due to the robustness of TTC computation to surface orientation, even Case (II) provides satisfying results. Similar to conclusion in Section 6.5.4, Cases (II) and (IV) produce better results than Cases (I) and (III) do, and TTC results from Case (IV) are still the best among the four cases. For Case (II) and (IV), results based on segmentation are quiet close to ground-truths. The average error and average absolute error for TTC estimation based on segmented region for Case (IV) is respectively 11.84% and 11.96% as shown in Table 6.5.

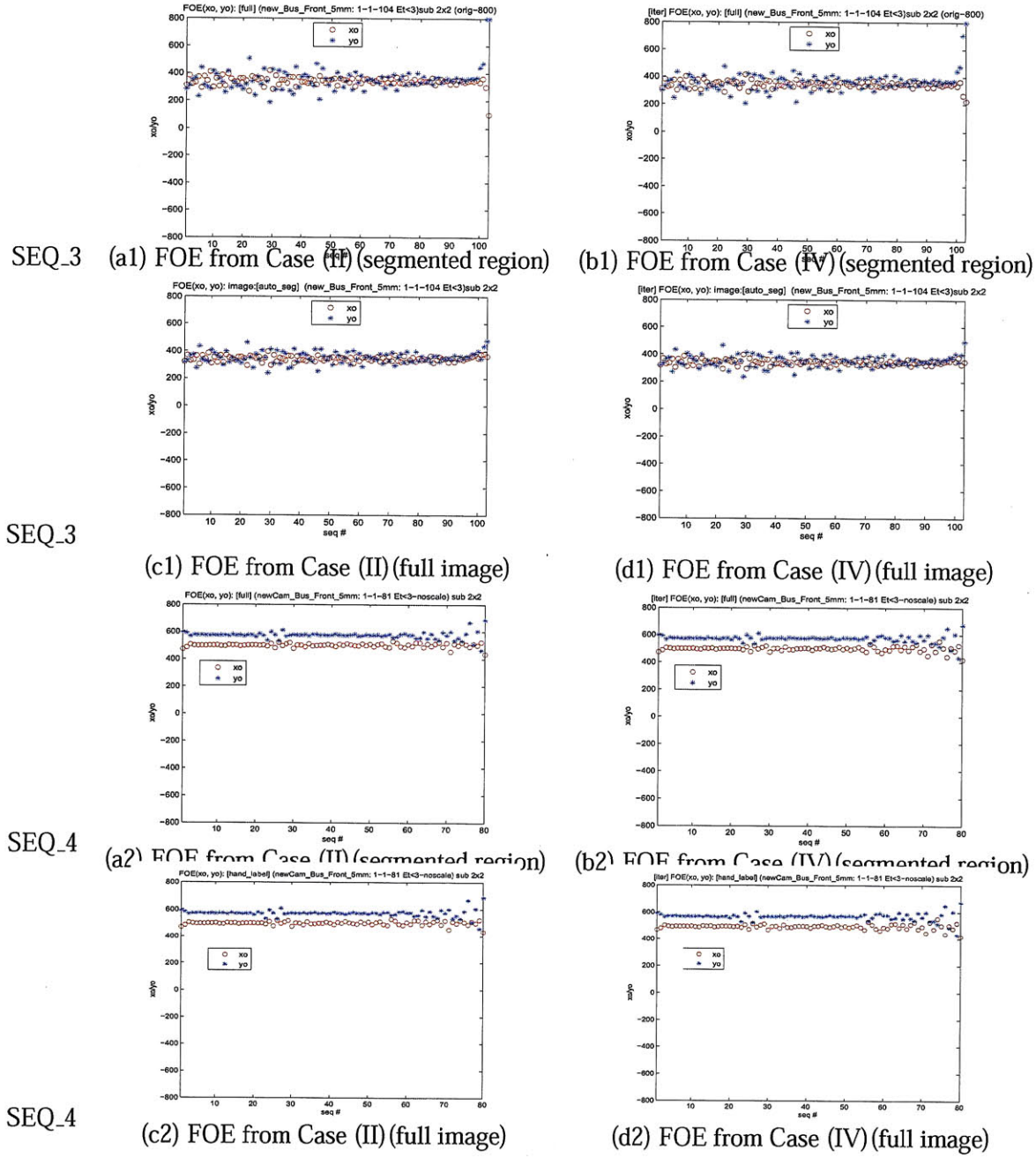
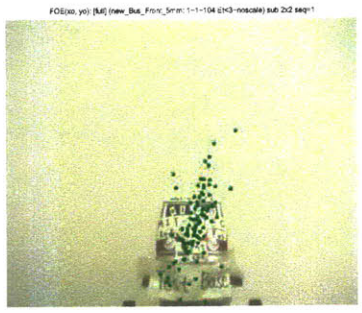
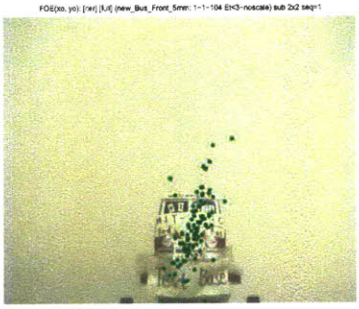
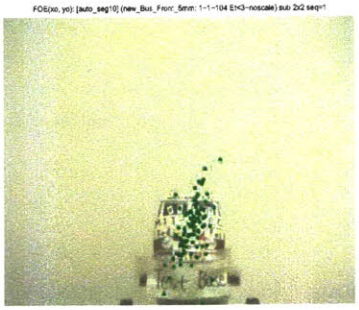


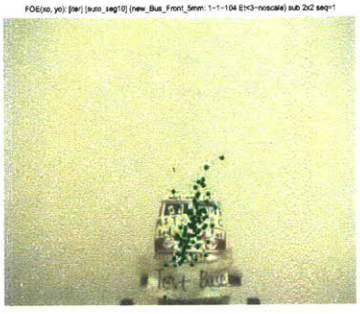
Figure 6-29: FOE based on full images and segmented areas for Case (II) and (IV) at subsampling rate 2×2 for stop-motion sequences SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm." x axis: frame number; y axis: FOE coordinates x_0 and y_0 respectively represented by red and blue lines. (a1)(b1)(a2)(b2): based on segmented areas. (c1)(d1)(c2)(d2): based on full images. Left column: Case (II). Right column: Case (IV). All figures are plotted with the same scales.





SEQ_3





SEQ_3 (c1 on) (d1 on)

SEQ_4

SEQ_4 (c2) FOE from Case (II) (segmented region) (d2) FOE from Case (IV) (segmented region)

Figure 6-30: FOE results plotted on different image frames for stop-motion sequences SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm." The computation is based on full image and the segmented area for Case (II) and (IV) at subsampling rate 2×2. (a1)(b1)(a2)(b2): based on full image. (c1)(d1)(c2)(d2): based on the segmented area. Left column: Case (II). Right column: Case (IV).

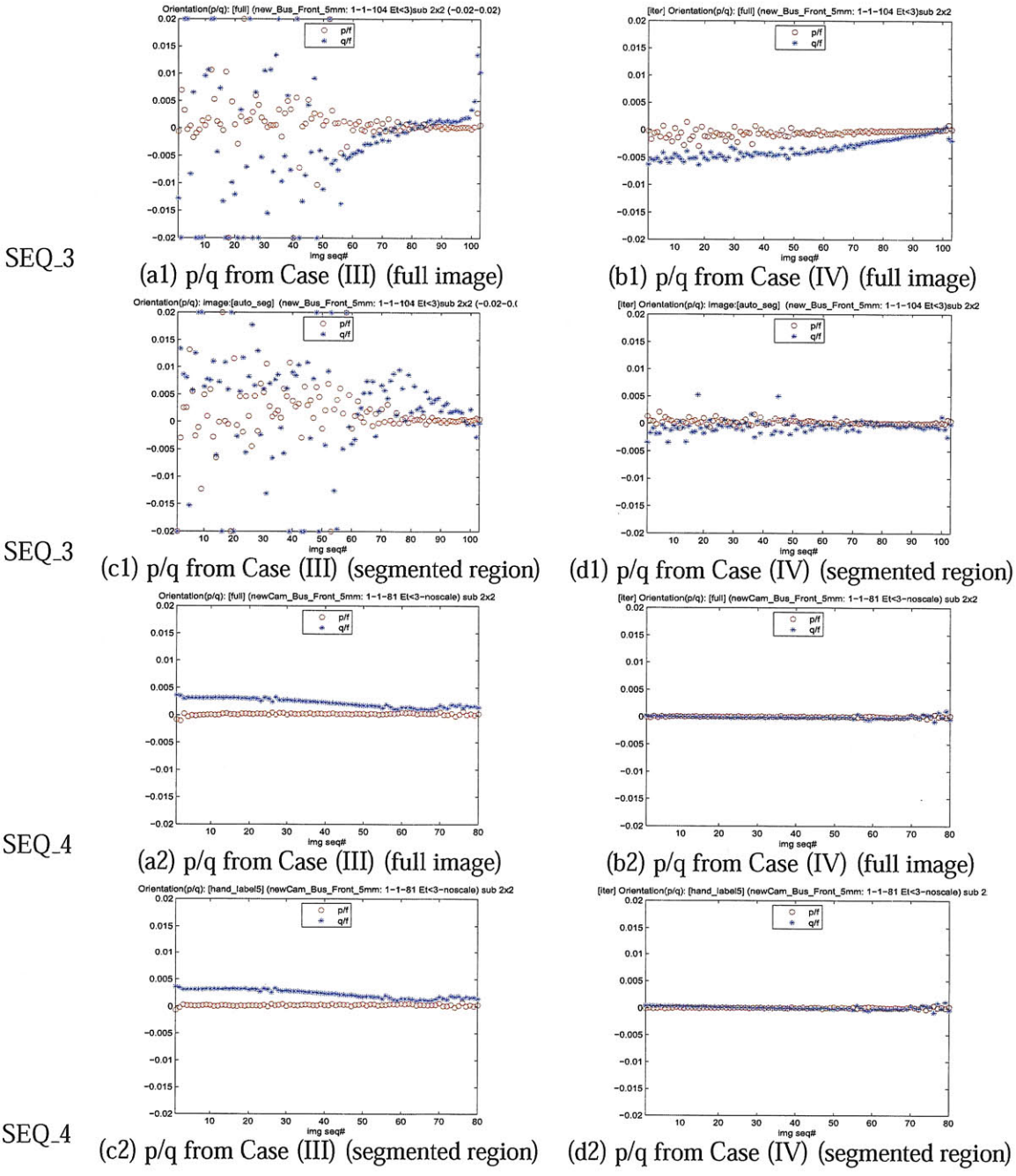


Figure 6-31: Orientation parameters (p, q) for stop-motion sequences SEQ_3, "new_Bus_Front_5mm," and SEQ_4, "newCam_Bus_Front_5mm," based on full image and segmented area for Case (II) and (IV) at subsampling rate 2×2. (a1)(b1)(a2)(b2): based on full image. (c1)(d1)(c2)(d2): based on the segmented area. x axis: frame number; y axis: $(p/f, q/f)$ values respectively represented by red and blue lines. Left column: Case (III). Right column: Case (IV). All figures are plotted with the same scales.

Table 6.5: TTC estimation error (percent) in Figure (6-34) for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf."

Error	case(II) sub2 × 2	case(IV) sub2 × 2	reg1/fusion	reg2/fusion
Average	16.59	11.84	4.88	3.24
Average absolute	16.59	11.96	5.21	3.96

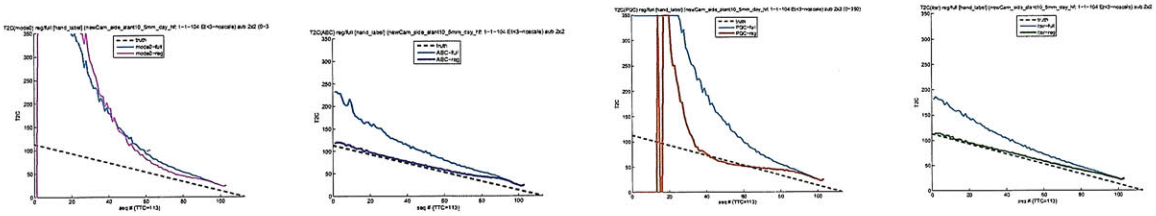
In Figure (6-33), for SEQ_5 from Category 2(b), TTC results based on segmentation schemes are better than that based on full images. The observation is consistent with the comparison of TTCs at multiple subsampling rates and their fusion results based on two segmentation schemes, hand labeling and auto segmentation, as shown in Figure (6-34).

In summary we claim that segmentation helps to improve TTC, and TTC computations are robust to segmentation errors and different segmentation schemes. We have the similar statement for other sequences from Category 2(a), though the differences of TTC based on full images and segmentation for Sequence SEQ_4 are not that significant.

TTC fusion based on multiple computational models and multi-scale subsampling

We then compare TTC estimation data using different subsampling rates and/or different computational models, and evaluate the performance of fusion scheme discussed in Section 6.4 to improve robustness and higher reliability. Figure (6-34) and Figure (6-38) present the fusion results based on TTCs from case (IV) at different subsampling rates, 1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64, and their fusion results based on minimization. TTC fusion results fit the ground-truth lines very well, showing the efficiency of our algorithm for different setup and different lighting situations. Specifically, the top and lower rows in figure (6-34) respectively correspond to two different segmentation schemes, hand labeling and auto segmentation. For hand-labeling segmentation scheme, the deviation of TTC fusion is 4.88% for average errors and 5.21% for the average absolute errors as shown in Table 6.5. For auto segmentation scheme, the deviation of TTC fusion is 3.24% for average errors and 3.96% for the average absolute errors. Our results clearly show that the proposed TTC fusion based on minimization is effective and computationally robust.

Figure 6-32: Sample frames and their segmentation for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf" (103 frames). Frame seq: 1, 21, 41, 61, 81, 101.



TTC Case (I) Case (II) Case (III) Case (IV)

Figure 6-33: The comparison of TTC computation based on full image and the segmented area for different cases at subsampling rate 2×2 for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf." x axis: frame number. y axis: TTC. Dashed black line: TTC ground-truths. TTC thin/thick lines: based on full image or segmented areas. TTC results based on full images/segmented regions: Case (I): Magenta/Cyan lines, Case (II): Blue thin/thick lines, Case (III): Red/Cyan lines, Case (IV): Green thin/thick lines. All figures are plotted with the same scales.

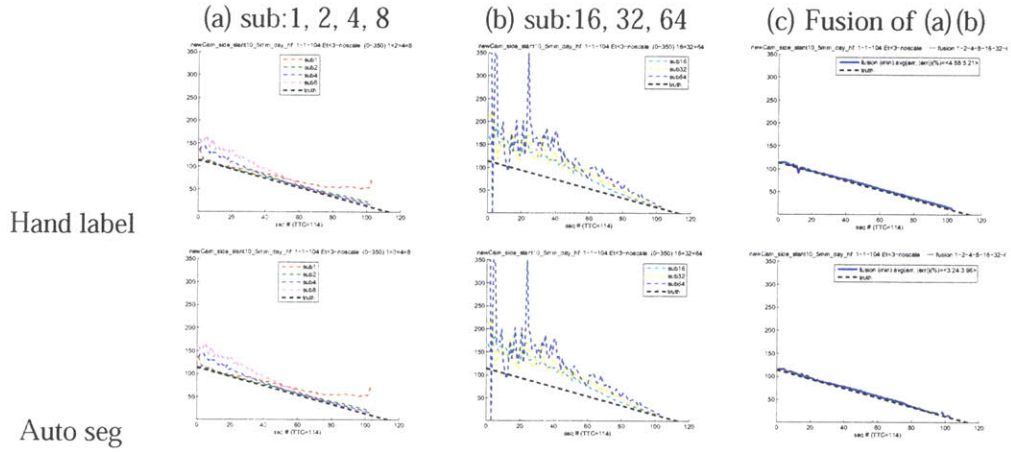


Figure 6-34: The comparison of individual TTCs at different subsampling rates and related TTC fusion results based on given segmentation, hand labeling and auto segmentation, for Sequence SEQ_5, stop-motion image sequence "newCam_side_slant10_5mm_day_hf." x axis: frame number; y axis: TTC. Top row: results for segmentation based on hand labeling. Bottom row: results for segmentation based on auto segmentation. (a)(b): TTC results computed at different subsampling rates for segmented areas. (c) Solid lines: Fusion results based on (a) and (b). Dashed black lines in (a)(b)(c): TTC ground-truths. (a) TTC dotted lines: 1×1 (Red), 2×2 (Green), 4×4 (Blue), 8×8 (Magenta). (b) TTC dotted lines: 16×16 (Cyan), 32×32 (Yellow), 64×64 (Blue). All figures are plotted with the same scales.

Figure 6-35: Sample frames and their segmentation for Sequence SEQ_6, stop-motion image sequence "newCam_slant_front_5mm" (106 frames). Frame seq: 1, 21, 41, 61, 81, 101.

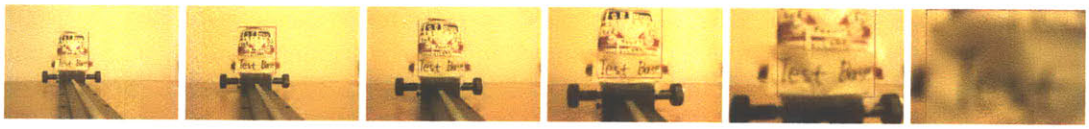


Figure 6-36: Sample frames and their segmentation for Sequence SEQ_7, stop-motion image sequence "newCam_slant15_front_5mm" (107 frames). Frame seq: 1, 21, 41, 61, 81, 101.

Figure 6-37: Sample frames and their segmentation for Sequence SEQ_8, stop-motion image sequence, "newCam_slant25_front_5mm"(107 frames). Frame seq: 1, 21, 41, 61, 81, 101.

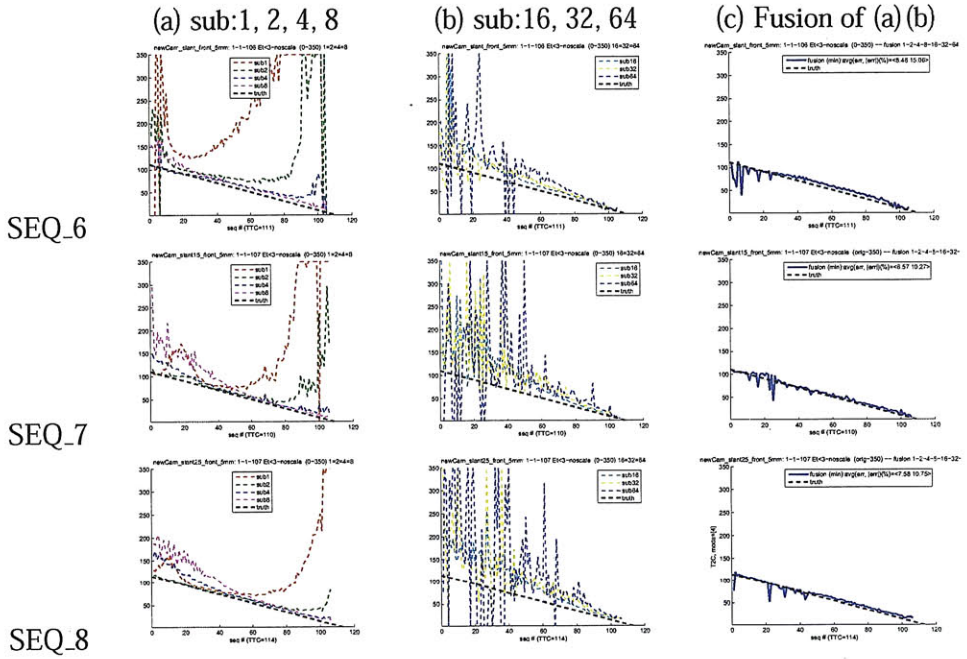


Figure 6-38: The comparison of individual TTCs at different subsampling rates and related multi-scale-based TTC fusion results based on given segmentation for stop-motion image sequences, SEQ_6, "newCam_slant_front_5mm," SEQ_7, "newCam_slant15_front_5mm," and SEQ_8, "newCam_slant25_front_5mm." x axis: frame number; y axis: TTC. (a)(b) TTC results computed at different subsampling rates for segmented areas. (c) Solid lines: Fusion results based on (a) and (b). Dashed black lines in (a)(b)(c): TTC ground-truths. (a) TTC dotted lines: 1×1 (Red), 2×2 (Green), 4×4 (Blue), 8×8 (Magenta). (b) TTC dotted lines: 16×16 (Cyan), 32×32 (Yellow), 64×64 (Blue). All figures are plotted with the same scales.

6.5.6 Video from a camera mounted on a car

In this section, we will apply our methods to a video sequence, SEQ_9, in outdoor driving environment as shown in Figure (6-39). The sequence is extracted from the log file in the form of 300×400 images in normal RGB color format.

Size-based TTC estimation

Since we do not have "ground-truth" TTC values for the video sequence, we have to compute the true TTC value using traditional size-based method. The detail steps are as followed:

1. Label manually the interested foreground regions for all video frames, compute the square root of region areas for size information and smooth the size results with low pass filter. The results for SEQ_9 are show in Figure (6-40(a)).

2. Compute the logarithm of smoothed size, log(size), and their changes between every N frames. Figures (6-40) (b) and (c) respectively shows the log(size) information and its difference between every N frames.

3. Use Equation (6.15) to compute the size-based TTCs by N/[log(size(i+N))-log(size(i))]. The TTC estimation is plotted in Figure (6-40(d)).

Figure (6-40(a)) shows that the measured size of the target van for the first few frames are less than 80 pixels while the time-to-contact is around 200-500 frames. The corresponding changes of image sizes will be less than $0.16 - 0.4$ pixel between two consequential frames. Measuring TTCs to an accuracy of 10% requires, in effect, measuring image positions with an accuracy better than $0.016 - 0.04$ pixel.

Since the size-changes between two consequential frames – being small fractions of a pixel – are too small to be measured, we use changes in target size over every N frames. However, serious quantization occurs, because image sizes could be estimated only to about a pixel accuracy. We set $N = 16$ for the test. However, the size changes are still very small, around 2 to 6 pixels. Thus quantization errors would cause serious TTC measurement errors.

For Equation (6.15), if we take its derivative over image size s on both sides, we have:

$$\frac{dT}{ds} = \frac{1}{s'} - \frac{ss''}{(s')^2} \tag{6.43}$$

where $s' = ds/dt$ and $s'' = (d^2)/(ds^2)$. When s' is extremely small, the sensitivity of TTC to size-measurement noise is extremely high. The above equation indicates that size-based TTC estimation is very sensitive to measurement noises of object sizes. As shown in Figure (6-41)(d), size-based TTC estimation is very noisy, especially for the first 100 frames among total 400 frames, i.e., frame number 100-200.

Comparison of TTC results based on size-based method and our method

Figure (6-41) shows the TTC fusion results for SEQ_9 based on multi-scale subsampling for model Case (II) discussed in Section 6.4. The left column corresponds to TTC results at different subsampling rates 1×1, 2×2, and 4×4. The right column corresponds fusion results (solid blue line) versus size-based TTC measurements (green circles). TTC results in the top and bottom rows in Figure (6-41) are respectively computed based on full images and segmented regions.

Our TTC results for SEQ_9 agree with visual estimates of vehicle motion and distances. It is indicated that the driver seems to apply brake initially, which explains why TTC more or less constant. The vehicle was then brought to a complete stop so TTC computation at the end of SEQ_9 becomes unstable since motion parameters $C = -W/Z = 1/TTC$ approaches zero. Unfortunately, the actual "ground truth" is not known in this case, and we only have the size-based TTC estimation. TTCs from our algorithm and the manually estimated TTC generally agree with each other, although detailed comparison is not possible because of the coarse quantization of our manual estimated TTCs.

For SEQ_9, TTC estimation based on full images by our algorithm generally tends to be lower than the one estimated manually. This is because the image areas involved in TTC computation are dominated by fast moving parts of an image corresponding to nearby objects. This effect could be reduced by masking or segmentation of an image. As shown in

the comparison between Figure (6-41)(b2) and Figure (6-41)(b1), TTC estimation based on segmentation are larger than that based on full image and are close to manually estimated results.

Figure (6-41) shows that the size-based TTC estimation results are much more noisy than our calculated TTC results. The size-based TTC estimation between frames 100-200 vary significantly from our results. The difficulty with manual estimation of the TTC once again illustrates that TTC algorithm presented here works well for small image motions.

Figure 6-39: Sample frames and their segmentation for Sequence SEQ_9, stop-motion image sequence "camera3CUT1-horn"(299 frames). Frame seq: 100, 159, 218, 277, 336, 395.

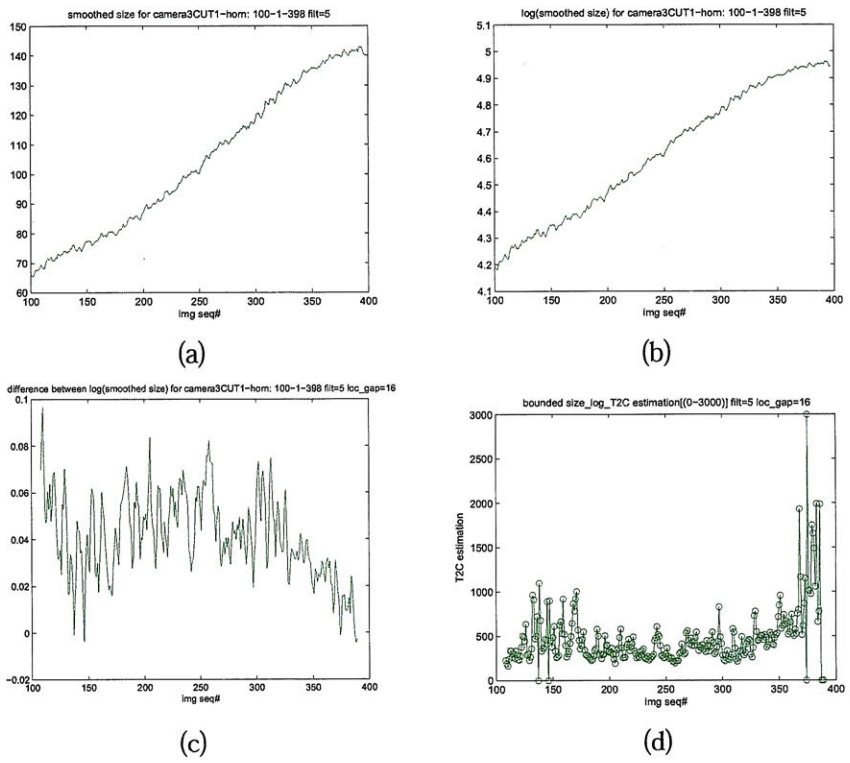
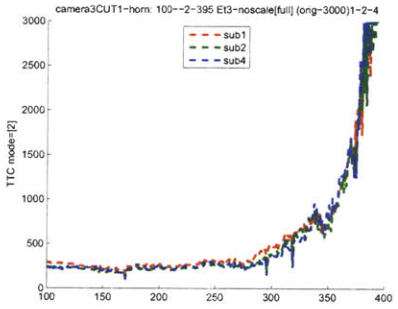
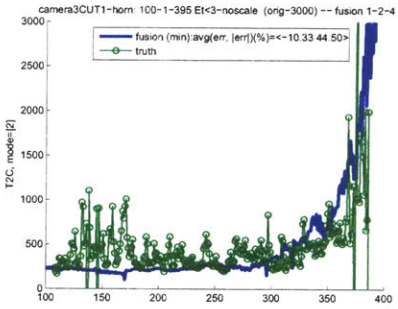


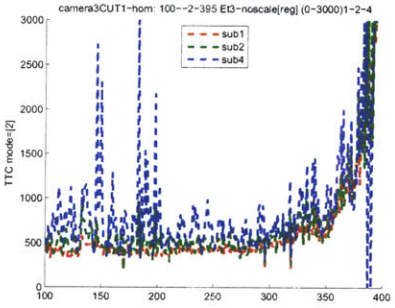
Figure 6-40: Size-based TTC estimation for Sequence SEQ_9, stop-motion image sequence "camera3CUT1-horn." x axis: frame number. y axis: (a) Smoothed size information based on labeled foreground regions. (b) Log of smoothed size. (c) Difference between the log of smoothed size between every 16 frames. (d) Size-based TTC estimation results.



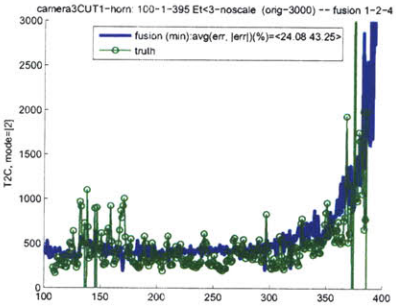
(a1) TTC @ multi-subsampling w. full img



(b1) TTC fusion based on (a1)



(a2) TTC @ multi-subsampling w. segmentation



(b2) TTC fusion based on (a2)

Figure 6-41: The comparison of TTC fusion results and traditional size-based TTC estimation for Sequence SEQ_9, stop-motion image sequence "camera3CUT1-horn." x axis: frame number; y axis: TTC. TTC dotted lines: results at different subsampling rates for segmented areas. TTC thick line: fusion results. Green circle: size-based TTC measurement. (a1)(a2) TTC results computed at different subsampling rates. TTC dotted lines: 1×1 (Red), 2×2 (Green), 4×4 (Blue). (b1)/(b2) Fusion results based on (a1)/(a2) respectively. (a1)/(a2) Results based on full images vs. labeled regions. All figures are plotted with the same scales.

Table 6.6: Performance Evaluation for TTC with different case models

Category	SEQ_#	full/seg.	Fig.#	Case (I)	Case (II)	Case (III)	Case (IV)
1	SEQ_1	full	Fig. (6-11), (6-15)	OK	Good	OK	Good
	SEQ_2	full	Fig. (6-17)	OK	Good	OK	good
2(a)	SEQ_3	full	Fig. (6-26), (6-27)	Bad	OK	Bad	OK
		seg		Bad	Good	Bad	Good
	SEQ_4	full		OK	Great	Good	Great
		seg		OK	Great	Good	Great
2(b)	SEQ_5	full	Fig. (6-33)	OK	OK	Bad	OK
		seg		OK	Great	OK	Great

Table 6.7: The different factors's impact on TTC

Category	Case models	Subsampling rates ($N \times N$)	Segmentation
1	High influence Best: Case (IV) Good results for small TTC Sensitive to movement direction	High influence Spatial averaging/subsampling for small TTCs Temporal averaging/subsampling for large TTCs. Small N for large TTC, large N for small TTC	N/A
2(a)	High influence. Best: Case (II) (IV) Nice results for small TTC for all cases	N/A	High influence TTC w. segmentation is smaller.
2(b)	High influence. Best: Case (IV)	High influence Small N for large TTC, large N for small TTC Best: Fusion resulst	High influence Not sensitive to segmentation error.
3	High influence Best: Case (II) (IV)	High influence Best: Fusion results	Medium influence TTC w. segmentation is smaller.

6.6 Discussion

6.6.1 The impact of different parameters on TTC computation

In this section, we will summarize the impact of four computational models, different subsampling rates, various image regions on TTC, FOE, and orientation parameters (p, q).

Table 6.8: The different factors's impact on FOE

Category	Case models	Subsampling rates ($N \times N$)	Segmentation
1	Medium influence Case (IV) is better than (II) More clustered. Less outliers. $x_0 < 0, y_0 < 0$	Medium influence unless N is too large	N/A
2(a)	Low influence Clustered. $x_0 \approx 0, y_0 > 0$	N/A	Low influence
2(b)	N/A	N/A	N/A
3	N/A	N/A	N/A

Table 6.9: The different factors's impact on (p, q)

Category	Case models	Subsampling rates	Segmentation
1	Medium influence Case (IV) is better than (III) More clustered. Less outliers. $(p, q) \approx 0$	Medium influence unless N is too large	N/A
2(a)	High influence Case (IV) is better than (III) More clustered. Less outliers. $(p, q) \approx 0$	N/A	Medium influence
2(b)	N/A	N/A	N/A
3	N/A	N/A	N/A

Impact of different parameters on time-to-contact

The choices of computation models would have large impact on TTC results as shown in Figures (6-11), (6-14) (6-15), (6-16), (6-17), (6-26), (6-33). In Table 6.6, we summarize the TTC performance based on different computational models. We find out that it is normally hard to meet the assumptions for Case (I) and Case (III) which require the relative motion between objects and cameras is along camera's optical axis. Even if we try to align the direction of relative motion along the camera's optical axis, TTC results based on Case models (I) and (III) are not very ideal due to the sensitivity to small misalignment. Compared to Case (I) and Case (III), Case (II) can fit in more situations including non-planar objects and objects which are not exactly perpendicular to optical axis. Our calculations indicate that computation based on Case (IV) provides the best results among the four computational models.

For some situations, however, other case models may produce better estimation than Case (IV) does. For example, as shown in Table (6.4) and Figure (6-26), TTC results for SEQ_4 from Case (II), have smaller error deviations from their ground truths than from Case (IV). Another example, as shown in Figure (6-26) and Figure (6-33), when objects are very close to camera, TTC results for most sequences based on four computational models provide quite satisfying estimations. This may be because when objects are getting closer, more and more foreground pixels contribute to TTC computation, and corresponding regions of non-planar objects can better fit planar-object based model than before. The last example is shown in Figure (6-8), when subsampling rates are large, iteration-based TTC results for Case (IV) are less reliable than those for other cases. It would make sense

to apply fusion-based methods in order to combine multiple TTCs and to take advantages of four different computational models.

TTC computation using segmented regions is more reliable than that using full images as shown in the top row of Figure (6-26) and Figure (6-33), (6-34), (6-41). But TTC estimation based on full images can still provide acceptable TTC results when obstacle is very close. For example, there is no clear differences between TTC estimation based on segmentation and full images for SEQ_4 in the bottom row of Figure (6-26).

Besides, TTC estimation algorithm is robust to the segmentation errors. As shown in Figure (6-34), TTC results based on two different segmentation schemes, hand labeling and auto segmentation, are quite close.

The choices of subsampling rates have large impact on the final TTC results as shown in Figures (6-8), (6-14), (6-15)(SEQ_1), (6-16), (6-17)(SEQ_2), (6-34)(SEQ_5), and (6-38)(SEQ_6, SEQ_7, SEQ_8) . Small subsampling rates provide better performance for early warning, while large subsampling rates provide better performance when relative distance between obstacle and camera is very small. Thus we propose a fusion-based method to take advantages of strength when using different subsampling rates.

Impact of different parameters on focus-of-expansion and orientation parameters

Our results show that the results of focus-of-expansion and orientation parameters provide us information about the relative motion and object orientation. If FOE points are close to the center of images, the relative motion between targets and cameras are along the optical axis. If orientation parameters (p, q) are very small, the surfaces of target objects are perpendicular to the optical axis.

Impact factors, including computational models, subsampling rates, and segmentation, have less influence on the computation of FOE and (p, q) than on TTC computation.

Among all impact factors, the choices of computation models have the largest impact on the computation of FOE and orientation parameters (p, q). The computational results based on Case (IV) are more reliable than the ones based on other models. The computational results of FOE for sequences in Table 6.2 based on Case (IV) are more clustered than the ones based on Case (II) as shown in Figures (6-12), (6-13), (6-18), (6-19), (6-20), (6-

21). The computational results of (p, q) for sequences in Table 6.2 based on Case (IV) are more clustered than the ones based on Case (III) as shown in Figures (6-12), (6-22), (6-23), and (6-31). The computation models have less impact on focus-of-expansion (FOE) than on orientation parameters (p, q). As shown in Figures (6-29) and (6-30), computational models do not result in clear differences on FOE computation.

The subsampling rates have low influence on the reliability of FOE and (p, q) only except when the subsampling rates are too small or too large as shown in Figures (6-18), (6-20) and Figures (6-22), (6-23). The curves for FOE and (p, q) versus frame number are quite flat and clustered, and their values do not change too much among most image frames for medium subsampling rates 4×4, 8×8, 16×16, 32×32.

Normally, the estimation of orientation parameters based on segmented regions is better than the one based on full images as shown by the comparison among different rows in Figure (6-31). But the improvement due to segmentation is not as significant as the one due to different computational models as shown by the difference between the left and the right column in Figure (6-31).

But segmentation has very little impact on FOE computation. As shown in Figures (6-29) and (6-30), the computed FOE values remain constant whether using full images or the segmented images. Actually for Sequence SEQ_3 and SEQ_4, there is no significant difference of FOE computation between using Case (IV) or Case (II) and between using segmented regions or full images.

In general, the computation of FOE and orientation parameters (p, q) are most sensitive to computational modes, and least sensitive to computing image areas (full vs. segmented areas). Also, compared with the computation of orientation parameters (p, q), the computation of FOE parameters are much less affected by computing models, image areas and reasonable subsampling rates.

Sensitivity of computation models to their assumptions and the reliability of FOE/(p, q) computational results

Up to now, we have found that Case (II) model can be applied to multiple situations while Case (III) is not suitable in most circumstances. Case (II) assumes that objects are planar

and perpendicular to optical axis, while Case (III) assumes that relative motions are aligned with the direction of optical axis. Thus for Case model (III), TTC results are very sensitive to the direction of relative motions, and for Case model (II), TTC results are very insensitive to the surface shape and the orientation of objects.

On the other hand, we notice that FOE curves are much smoother than (p, q) curves as shown in Figure (6-18), (6-20), (6-29) and Figure (6-22), (6-23), (6-31). As shown in Figure (6-29) and (6-31), the differences of FOE curves between Case (II) and Case (IV) are much less than the one for (p, q) curves, and the differences of FOE curves based on full images vs. segmented regions are also much less than the ones for (p, q) curves. Such observation shows that the estimation reliability for FOE points is higher than the orientation parameters.

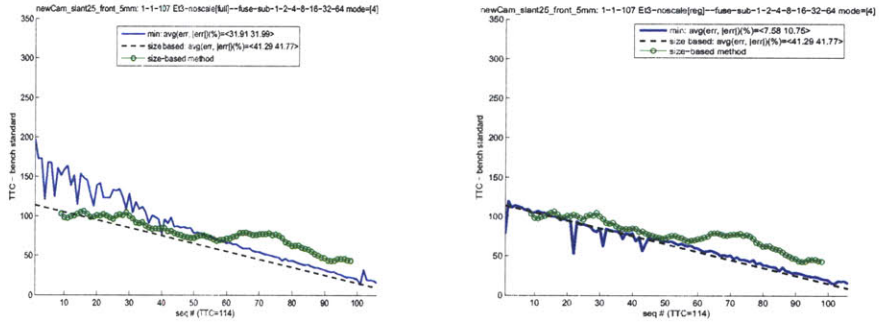
There exists interesting connection between the two above observations. Since the computation of TTC results is more sensitive to the relative motion direction than plane orientation, estimating motion information is more reliable than estimating orientation parameters. FOE points $x_0 = f(U/W)$ and $y_0 = f(V/W)$ are related to the motion information. Thus, the curves for FOE points are much smoother than the ones for orientation parameters.

6.6.2 Comparison of proposed algorithm and traditional size-based TTC estimation

In Section 6.5.6 and Section 6.5.6, we have compared TTC estimation for SEQ_9 based on our proposed gradient-based method and traditional size-based method. However, since we do not have ground-truth for SEQ_9, we cannot make statement based only on this comparison. In order to evaluate the efficiency of our proposed algorithms and size-based algorithms, we also compute the size-based TTC results for one of stop-motion sequence, SEQ_8, whose TTC ground-truths are given.

For Sequence SEQ_8 shown in Figure (6-37), the comparison of size-based TTC results and fusion-based TTC results is shown in Figure (6-42), in which blue lines in the left and right figures respectively correspond to the TTC fusion results based on multi-sampling for both full images and segmented regions. The deviation of size-based TTC results from

ground-truths is significantly larger than that for TTC fusion results. The error rates of TTC fusion results based on segmented regions are $7.58\%(avg)/10.75\%(avg\_abs)$, which are significantly smaller than the error rate of size-based estimation $41.29\%(avg)/41.77\%(avg\_abs)$. Even TTC results based on full images (no segmentation) are very close to the ground-truths. The error rates of TTC fusion results with full images are $31.91\%(avg)/31.99\%(avg\_abs)$, which is still less than error rates for size-based estimation $41.29\%(avg)/41.77\%(avg\_abs)$. Our results indicate that size-based TTC results are less accurate than TTC results based on our fusion method.



(a) (b)

Figure 6-42: The comparison of size-based TTC measurement and proposed gradient-based algorithms. x axis: frame number; y axis: TTC. TTC solid lines: TTC fusion results based on multi-scale subsampling at 1×1, 2×2, 4×4, 8×8. (b) 16×16, 32×32, 64×64. Dashed black line: TTC ground-truth. Green circle: size-based TTC measurement. Blue lines: (a) Fusion results based on full images. (b) Fusion results based on segmented regions. All figures are plotted with the same scales.

6.6.3 Multiple-scaled TTC fusion based on condition number

In Section 6.4.1, we proposed to use the condition number as the reliability index of TTC estimation and fuse TTC results only from reliable results with small condition number. In order to evaluate the performance of condition-number based fusion, Sequence SEQ_8 is chosen to because of the three following reasons. The first, SEQ_8 is a stop-motion sequence whose TTC ground truth is known and can be used for reliable evaluation. Secondly, SEQ_8 is one of the most general stop-motion sequences which simulates the situation of Case (IV), the general scenarios where both object surfaces and relative motion

are arbitrary. Thirdly, TTC estimation for SEQ_8 is the worst among TTC results for all stop-motion sequences in Table 6.2.

For SEQ_8 from Category 2(b), TTC estimation at different subsampling rates based on segmented regions are plotted in the last row of Figure (6-38). The condition numbers for the above TTC results are plotted in Figure (6-43)(d). Figure (6-38) shows that at the first half of SEQ_8, i.e., when TTCs are large, TTC estimation at large subsampling rates, such as, 32×32 and 64×64, are not reliable. There are large TTC oscillations involved. Figure (6-43)(d) shows that the corresponding condition numbers for large subsampling rate 32×32 and 64×64 at the first half of SEQ_8 are also very large. At the second half of SEQ_8, when targets are close and TTCs are small, TTC estimation at large subsampling rate are very accurate and close to ground truth. Simultaneously, corresponding condition-numbers present the trend of decreasing. When TTCs are very small, TTC estimation at small subsampling rates such as 1×1 and 2×2 are not reliable. As TTC further decreases, condition-number stops the trend of decreasing, and starts to go up again at the end of SEQ_8. This obvious correlation suggests condition number may be used as reliability index. With the introduction of such reliability index, the TTC fusion based on multi-scale subsampling is further improved. At each frame, we have total seven TTC contribution and seven condition numbers corresponding to seven subsampling rates. Figure (6-43)(c) and (d) respectively correspond to the curves of condition-numbers corresponding to TTC computed based on full images and segmented regions. Applying the condition-number based fusion scheme proposed in Section 6.4.1, we first set the thresholds of condition-numbers for SEQ_8 as plotted in red curve in Figure (6-43)(c) and (d). After rejecting TTC values with condition numbers higher than the threshold lines, we obtain TTC fusion results as plotted in blue curves in Figure (6-43), in which Figures (6-43)(a) and (b) respectively correspond to results based on full images and segmented regions. In Figures (6-43)(a) and (b), both fusion results plotted in blue lines present better performance than size-based TTC estimation plotted in green lines, especially when targets are getting closer. The error rates of the original TTC fusion with segmented regions are $7.58\%(avg)/10.75\%(avg\_abs)$. The error rates of improved TTC fusion with segmented regions are $9.23\%(avg)/9.76\%(avg\_abs)$. Both of them are significantly smaller than the

error rate of size-based estimation $41.29\%(avg)/41.77\%(avg\_abs)$.

The comparison between blue lines in Figure (6-43)(c)(d) and blue lines in Figure (6-42)(a)(b) shows that our TTC fusion results based on condition-number are much smoother than the ones based on simple minimization scheme. The spikes in TTC results in Figure (6-42)(b) do not show up in Figure (6-43)(d). The removed oscillation and peak errors in minimization-based curves are actually caused by those unreliable TTC estimation.

Our proposed fusion algorithms work well for the most complicated stop-motion sequence, SEQ_8. For the other stop-motion sequences which already have very ideal TTC estimation results using simpler method, the fusion-based methods will only further improve its current good performance. Then we can claim that the condition-number based multi-scale subsampling is very promising.

6.7 Conclusions

We proposed a new "direct," gradient-based method for estimating the time-to-contact, and analyzed how impact factors affect the performances of our algorithms.

The "direct method" operates directly on the spatial and temporal derivatives of brightness, and does not depend on the computation of the optical flow as an intermediate result. The method also does not require feature detection, feature tracking over continued images, thus having low latency and avoiding the computational load of calibration. Spatial averaging and sub-sampling extend the range of operation to small values of TTCs, while temporal averaging and sub-sampling extend the range to large TTCs. Some form of image segmentation is useful in suppressing contributions from image regions moving in ways different from those of the object of interest. For applications with general relative motion, the final performance is also very robust to segmentation errors. Tables 6.7, 6.8 and 6.9 respectively summarize the influences of these factors on computation for our discussed categories. The multi-scale-fusion based direct method further enhances TTC estimation accuracy. Condition numbers of TTC computation are good index for reliability which can be used to block the contribution of unreliable TTC results to fusion. We have tested our algorithms on multiple situations, including synthetic images, stop motion sequences in

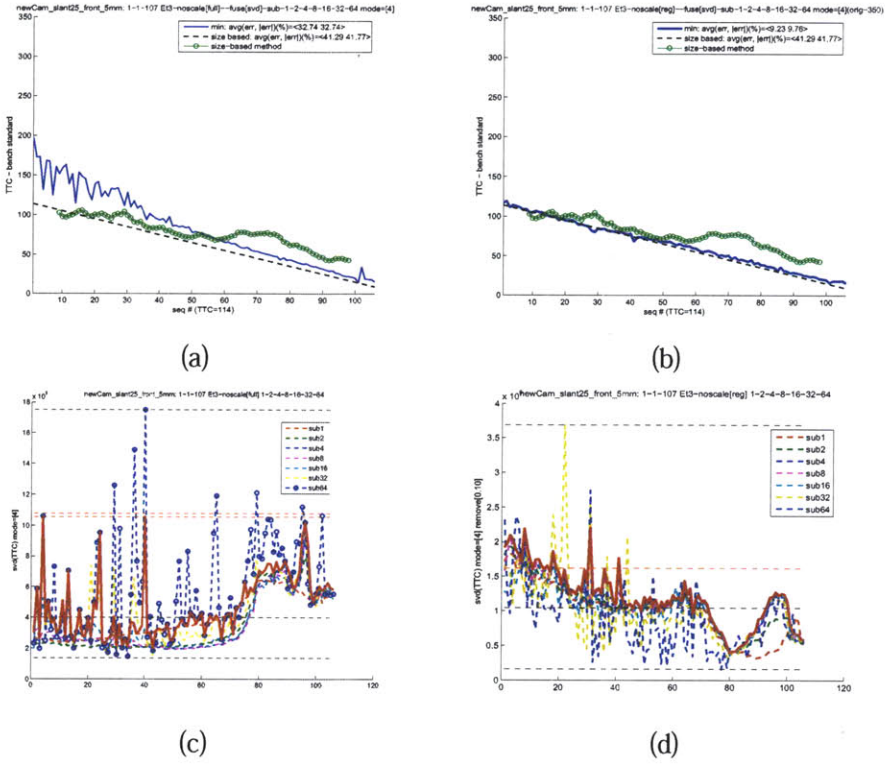


Figure 6-43: The condition-number-based TTC fusion for SEQ_8. (a) Condition-number-based fusion results based on full images. (b) Condition-number-based fusion results based on segmented regions. x axis: frame number; y axis: TTC. TTC solid lines: TTC fusion results based on multi-scale subsampling at 1×1, 2×2, 4×4, 8×8, 16×16, 32×32, 64×64. Dashed black line: TTC ground-truth. Green circle: size-based TTC measurement. (c)/(d) condition numbers for TTCs at different subsampling rates using full images/segmented regions. Dotted lines: condition numbers at different subsampling rates. Red solid line: threshold for condition numbers. If condition numbers are larger than threshold, their corresponding TTC results will be ignored in TTC fusion.

arbitrary relative motion, as well as the outdoor driving environment. Our results show the efficiency of our algorithm for different setup and different indoor/outdoor lighting situations.

6.8 Acknowledgments

Log files from MIT DARPA Urban Challenge vehicle were kindly made available by David Moore.

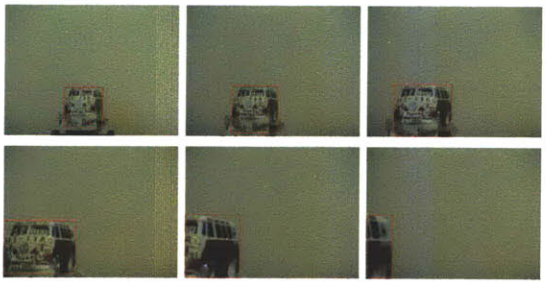


Figure 6-44: Sample frames and their segmentation for Sequence SEQ_9, stop-motion image sequence, slanted_Bus_5mm(93 frames). Frame seq: 1, 18, 35, 52, 69, 86.

Chapter 7

Discussion and Summary: the principle of fusion-based and layered-based schemes

In this thesis, we have proposed a general framework of the fusion-and-layer based methodology. The framework is shown in Figure 1-2 in Chapter 1. The major principle of our framework takes advantage of connection among multiple pieces of information to improve obstacle detection and perception abilities. The feature-level fusion converts a complex segmentation task into several simple ones to avoid time-consuming full-image search.

We sequentially introduce several systems based on feature-level fusion to obtain foreground information in different scenarios. In Chapter 2 and 3, we identify 3D information of targets without and with background noises from a pair of visible image sequences taken by two stereo cameras. In Chapter 4, we detect and classify pedestrians from infrared image sequences taken by single infrared camera. In Chapter 5, we detect and track the dynamic movement of our interested foregrounds from visible image sequences from single visible camera. In Chapter 6, we obtain time-to-contact information for obstacles based on visible image sequences from single visible camera.

The applications discussed in this thesis are examples that how we can apply the principle of fusion-based and layer-based to detect dynamic obstacles and obtain their dynamic information systematically from visible and infrared sequences in various typical situa-

tions. The performance of the discussed systems demonstrated that the proposed systems significantly improve the accuracy, reliability and robustness. In this chapter, we will summarize several important features about how the fusion-based and layer-based principles are used in our different applications.

7.1 The principle of layered technology: Divide and conquer

In our feature-level fusion systems, layer-based separation is the key component that separates an original image into several layers on which segmentation is simpler than in the original image. In Chapter 2 and Chapter 3, we discuss how to separate an image into several distance-based image-layers while each layer contains potential obstacles at a distance-range. In Chapter 4 and Chapter 5, we discuss how to separate an image into several vertical stripes that contain potential pedestrian obstacles in two different methods. In total, we have discussed three different methods to separate one image into two different categories of layers in which segmenting interested targets is much easier than in original images.

All our layer-based separation is based on statistical properties of corresponding features. Typical detection and segmentation methods differentiate foregrounds and backgrounds based on the local feature vectors of their pixels, including motion distances, vectors, edge features, etc. Feature computation might be noisy, and segmentation schemes might be unreliable due to the sensitivity to separation thresholds and outliers. Instead of directly setting thresholds for feature vectors, our layer-based separation methods first compute the statistical properties of these features, and use them to separate image pixels into several layers/groups that contain individual foreground as well as other noises. Within each group, we further locate interested foregrounds. Here below we briefly summarize the different statistical properties of different features used in our layer-based separation.

7.1.1 Disparity-Range-based Edge Layer Separation and Background Removal

The first category of layers is distance-based image-layers that have the same sizes as that of the original maps. Distance-Range-based edge-layer separation introduced in Chapters 2, 3 is based on the statistical distribution of distance or disparity features. Conventional segmentation based only on depth is very sensitive to correspondence errors. Instead, we first produce a binocular disparity-histogram as shown in Figure (2-9). The statistical distribution of disparity values provides the disparity ranges for potential foregrounds. The extra knowledge of disparity ranges make it possible to split an edge map into several edge layers at different distance-ranges, including a background range. Therefore, we can locate pixels that have correspondence pairs matching the given disparity-range and obtain the "distance-based edge-layers," containing obstacles within that range. In Chapter 2, the example of corresponding separated edge-layers is shown in Figure (2-11) for stereo image pairs shown in Figure (2-7)(a), while the disparity-histogram has three peaks representing three nearest vehicles as shown in Figure (2-9).

In Chapter 3, we identify the disparity ranges for background objects that are farther than our interested distances in the disparity histogram as shown in Figure (3-2) for stereo frames shown in Figure (3-1)(a)-(d). The separated backgrounds pixels are shown in Figure (3-3)(a), which helps to clean up noises in edge-layers corresponding to foregrounds. The comparison between Figure (3-3)(b) and (c) indicates that the distance-based background-layer-removal eliminated most tree pixels in the background while preserving all pixels of target objects including the ball.

In summary, after distance-based edge-layer separation, it becomes relatively easy to identify foregrounds from background pixels and to detect individual objects. Our method is very robust to the different choices of picking disparity ranges for different foregrounds in histogram figure since our segmentation algorithms in individual layers can discriminate noises from other layers.

7.1.2 Vertical-Projection based Layer Separation

The second category of layers is several vertical stripes in which we can detect potential pedestrians. Our method simplifies the original two-dimensional full-image segmentation into two one-dimensional segmentation by splitting an image with a size of $n_{row} \times n_{col}$ into several vertical stripes with a size of $n_{row} \times n_i$, $n_i < n_{col}$. Within these vertical stripes, images of human beings can be further segmented, which is called "Horizontal segmentation first, vertical segmentation second" scheme. In Chapter 4 and Chapter 5, we have introduced two different ways to horizontally segment an image.

Layer Separation relying on Vertical-Projection of Pixel Brightness for Infrared Images

For infrared images from night vision, the horizontal segmentation is based on vertical projection curves of pixel brightness, i.e., the summation of pixel brightness in image columns versus their corresponding horizontal positions, as introduced in Chapter 4. Instead of directly utilizing pixel brightness to segregate pedestrian foregrounds from background, we compute the vertical projection of image brightness, which represents the statistical means of pixel brightness in each column as shown in Figure (4-4)(a)(b). The variation of the statistical values provides the horizontal boundaries between foregrounds and backgrounds. As shown in Figure (4-4), the horizontal locations and width of projection waves correspond to pedestrians and are robust to parameter choices This corresponding relationship helps to divide night-time infrared images as shown in Figure (4-4)(a)(b) into different vertical stripes within which images of human beings can be further segmented based on either *brightness* or *bodylines*. For night vision, splitting infrared image horizontally is the key to our automatic pedestrian detection.

Layer Separation relying on Combined-Difference-Image based Vertical Projection-Curves for Visible Images (CDI-VPC)

For visible images in monitoring situations, the horizontal segmentation is based on vertical projection curves of Combined-Difference-Images (CDI-VPC), i.e., the summation of

pixel brightness of CD-Image for each column versus their corresponding horizontal positions [100], as introduced in Chapter 5. Instead of directly utilizing pixels from different images between two frames to segregate moving foregrounds from static background, we compute the vertical projection of CD-Images as shown in Figure (5-3), which represents the statistical means of brightness variation in each column. In the CDI-VP-Curves, the horizontal locations and width of projection waves correspond to pedestrians and are robust to parameter choices. The variation of the statistical values provides the horizontal boundaries between foregrounds and backgrounds. The vertical image-stripes corresponding to sharp triangular spikes in vertical projections may contain pedestrian candidates. Thus, we can also take advantage of CDI-VP-curves to detect pedestrians from visible image sequences using "horizontal segmentation first, vertical segmentation second" scheme. Within these vertical stripes, images of human beings can be further segmented. For occlusion situations, we can search for the potential head-locations at the peaks of merged waves.

Choises of Thresholds for Both Situations

The two horizontal segmentation algorithms, for video and infrared sequences, use adaptive thresholds respectively defined in Equations (4.2) and Equations (5.6)(5.5) in order to compute corresponding vertical projection curves directly from infrared images or combined-difference-images for visible images. These thresholds are all set to be very small in order to obtain non-zero projections in projection curves for infrared images or combined-difference-images at the columns corresponding to pedestrians' locations. For infrared images, the thresholding removes the contributions of very dark pixels. For visible images, the thresholding removes the contributions of pixels with little intensity variation between two frames. Since the intensity values for most bright pixels in infrared images are above a fixed threshold, Equations (4.2) defines threshold by subtracting a fixed constant from the largest brightness value. The scale for intensity differences for pixels at two continuous frames vary significantly, thus, we define the threshold to be a ratio of largest variation.

Layer-based Vertical Segmentation: Head Detection

For pedestrian detection in infrared and visible sequences, horizontal segmentation provides the ranges of horizontal locations. Detail 2D locations are obtained through vertical segmentation. For infrared sequences in the winter where there are very limited noises from background bright pixels, pedestrian detection within given vertical stripes can be directly determined based on the vertical locations of bright pixels. For infrared sequences in the summer, many bright pixels are from background regions, such as hot engines, electric pole, etc., and produce lots of noises in horizontally segmented regions. To determine pedestrians' locations in these vertical stripes, *one* pedestrian template is needed to search for pedestrians at all possible vertical locations. We decrease the number of candidate locations by searching for positive/negative edge pair in every row within vertical stripes.

For visible sequences where there is no occlusion, pedestrian detection in horizontally separated regions can be directly determined based on the vertical locations of pixels with large value in combined-difference-images at given vertical stripes. For visible sequences where pedestrians occlude, *one adaptive* pedestrian template is needed to search among multiple candidate locations corresponding to local transitional peaks within a big wave of of CDI-VP-curves. The noises here are mainly due to impact of occlusion on CDI-VP-curves. The matching template for each pedestrian is updated in very frame when there is no occlusion. For occluded situations, we use the matching template before occlusion happens.

Compared to traditional template-based 2D searching, our "horizontal first, vertical second" methods significantly decrease the number of candidates and hence increase the reliability. However, for both infrared and visible sequences, it is possible that template-based matching failed to find the best match. For infrared images, we apply initial segmentation to every frame and the wrong segmentation in the previous frame would not have impact on the next frame. For visible images, when there is occlusion where heads' location might not be accurately detected, we do not update templates. Besides, we fuse the process of initial segmentation and tracking. Thus the negative impact of the wrong detection in the previous frame would not be accumulated during the tracking process, and we can still

obtain the accurate segmentation in the next frame, especially when there is no occlusion.

7.2 Other Layer-based Processing

In previous section, we divide a whole image into several layers for the purpose of segmentation for one whole image, i.e., "divide and conquer." In this section, we summarize other layer-related techniques.

7.2.1 Similarity of Motion Vectors in Separated Layers

Because motion vectors are very noisy, segmentation based on them alone is not very reliable. As shown in Figure (3-6)(b), it is hard to discriminate between motion vectors for foregrounds and for backgrounds because of the impacts of large amount of background noises. However, as shown in Figure (3-6)(a), in our separated edge layers corresponding to foregrounds, motion-vectors of edge-pixels stand out much better in the obstacle-distance-layer than in the original edge maps. The comparison between Figure (3-7)(a) and Figure (3-7)(b) demonstrates that introducing motion information into distance-based layers helps to differentiate motion vectors for foreground from those for background and to improve segmentation accuracy. The fusion-based segmentation is discussed in Section 7.4.1.

7.2.2 Layer-based Dynamic Tracking Model

In Chapter 5, CDI-VP-Curves not only help with initial horizontal segmentation, but also provide the dynamics of our interested targets based on the changes of their corresponding CDI-VP-Curves. When two previously independent triangular spikes merge, corresponding foregrounds start to intersect. When there are no independent sharp triangular spikes at estimated regions, we identify the circumstances of occlusion.

With our layer-based segmentation techniques, "horizontal segmentation first, vertical segmentation second," our tracking process can exploit simple Kalman-filter based dynamic models that only track pedestrians' horizontal locations instead of both horizontal and vertical locations as traditional dynamic tracking algorithms do. Our tracking scheme

avoids the impact of vertical vibrations noises during walking, and improves the segmentation robustness during occlusion situations.

Different from conventional methods that track the movement of multiple feature pixels or only monitor the movement of foreground objects only, our methods follow the changes of the distribution of our interested features, and the changes of CDI-VPC while maintaining a simple dynamic tracking model.

7.3 Structural Fusion among Different Operations

Our framework shown in Figure (1-2) is based on feature-level fusion in which each process utilizes the extra information from other processes or sub-systems to simplify the original difficult tasks. In this section, we summarize three structural fusion blocks that fuse two typically independent operations. These three blocks are the fusion between segmentation and classification, the fusion between detection and tracking, and the fusion between the prediction step and update step for Kalman Filtering.

7.3.1 Segmentation based on Fusion of Segmentation and Classification Features

In Chapter 4, our fusion scheme shown in Figure (1-2)(c) takes advantage of classification features to enhance segmentation accuracy for infrared images. After initial horizontal segmentation, our bodyline-based vertical segmentation automatically identifies the sizes of potential pedestrians based on bodylines at each vertical location. We apply histogram-based classification features to search for the best candidate among multiple candidates within each vertical stripe as explained in Chaper 4.2.2.

Normally it is hard to directly apply classification features during segmentation procedure. Segmentation is to find the locations of all possible candidates, while classification process computes multiple features for these candidate ROIs, and differentiates foreground objects from noises. Typical brute-force based segmentation has to search among multiple candidates at different locations with various sizes. Due to large amount of candidates,

people normally would not compute classification features at segmentation stage. Besides, classification needs to set thresholds based on the distribution of all feature vectors, which is a compromised process between detection rates and false-alarm rates. Thus, classification features are not typically used in segmentation procedure.

On the contrary, for our layer-based segmentation on infrared images, the number of ROI candidates is very small due to our "horizontal segmentation first, vertical segmentation second" scheme. Thus we can afford to introduce some classification features into segmentation step to choose among multiple candidate regions within each stripe. Since there is at most one pedestrian within one stripe, we do not need to set any threshold. Instead, we simply pick the one whose histogram-feature is the closest to our template. And there will be at most one false alarm within each stripe. Our method balances the complexity and performance of two subsystems: segmentation and classification. The method focuses on improving the performance of combined segmentation/classification systems instead of maximizing one process while sacrificing the other. Segmentation with high quality can ease the classification task, while robust classification can tolerate segmentation errors.

7.3.2 Detection based on Fusion of Initial Segmentation and Dynamic Tracking

In Chapter 5, our fusion-based detection scheme actively fuses information of initial segmentation and dynamic tracking as shown in Figure (1-2)(d) of Chapter 1.

Our scheme takes advantage of combined information, including initial horizontal segmentation, CDI-VPC, previous detections, and estimated human locations from dynamic tracking models, as shown in Figure (5-11) of Chapter 5. Many algorithms cannot afford to implement initial segmentation for every frame due to its heavy computational load. Instead, tracking procedures are used to detect humans to take advantage of the connection between location similarities of the same obstacles. Each time, new segmentation results are used to update the dynamic model. Such type of tracking system would not respond fast enough in rapidly changing environment.

In contrast, our initial segmentation is based on "Horizontal segmentation first, vertical

segmentation second" scheme, whose computational load is very light, thus we can afford to implement segmentation algorithms for every frame. Segmentation for continuous image frames is independent. The combination of current segmentation and previous prediction results from dynamic tracking helps to improve segmentation reliability and reduce segmentation/tracking ambiguities, especially when humans intersect and occlude. Through combining the information from the initial segmentation and dynamic model, we can identify potential intersection of pedestrians, and search for individual pedestrians in very limited number of candidate position inside the initial segmentation. Thus, our algorithms can track humans with changing poses and deal with partial occlusions and scale-variations of targets.

7.3.3 Detection based on Fusion of Estimation/Update Step for Kalman Filtering

As discussed above, dynamic tracking models are heavily involved in fusion-based detection and help to improve segmentation accuracy, especially when humans dramatically change their poses. Conventionally, estimation results from traditional KF filters are only intermediate results. In our algorithm, both estimation and update steps are actively involved in fusion-based detection and play significant roles. Estimated results for pedestrian's future locations help to understand the dynamics of foregrounds, and determine the search range and candidate locations of pedestrians during merging process. Our KF update step accepts information not only from KF estimation and target locations, but also from the reliability of detection results. Our fusion-based tracking stops updating for occlusion situation, which is also different from traditional KF filters.

7.4 Other Related Fusion technologies

Besides the fusion across different computational blocks discussed in Section 7.3, there are three other major fusion-related techniques, fusion in time dimension, fusion in spatial dimension, and fusion in physical connection among obstacle features.

7.4.1 Fusion in Time Dimension

The information fusion in time domain provides extra information for foreground detection. One example is to identify possible intersection based on comparison of CDI-VP-Curves. Another example is the dynamic tracking models based on estimation and updating. Both of them are discussed in Section 7.2.2. Here we summarize fusion-based applications involved with motion information.

Layer-based Motion-based Clustering

As discussed in Section 7.2.1, there exists strong similarity of motion information and depth information for edge-pixels belong to same foregrounds. Such similarity serves two purposes. First, as shown in Figure (3-4)(c)(d) of Chapter 3, we can eliminate false segmentation blocks in background regions by removing static blocks based on motion information.Second, as shown in Figure (3-5)(d1)(d2) of Chapter 3, we can expand the initial segmentation (seed boxes), by absorbing the surrounding target pixels that have similar motion features as initial chosen pixels have but are lost in the process of background-removal. As a result, the initial segmentation can be enlarged to obstacles' real sizes as shown in Figure (3-7) and Figure (3-9)(b) of Chapter 3.

Motion-based Correspondence-Matching Criteria

In Chapter 2, we adopt motion information as one of correspondence-matching criteria in order to remove matching ambiguities associated with stereo vision. Because of the objects' rigidity, there exist similarities between the motion information calculated from the left and right frames corresponding to the same target feature-points. As in Figure (2-8)(b) of Chapter 2, motion-vectors for both stereo video frames show similar patterns in spite of the noises during motion-vector detection. As shown by the differences between Figure (2-9)(a) and Figure (2-9)(b), fusing both traditional epipolar-line constraint and motion constraint provides better correspondence reliability and more accurate histogram distribution than not using motion constraints.

7.4.2 Fusion in Spatial Dimension

Many machine vision algorithms, for examples, segmentation and TTC computation, depend only on the boundary information of interested targets. In this thesis, we have proposed new algorithms to take advantage of contribution from both boundary and interior pixels.

TTC Estimation based on the Fusion of Boundary and Interior Image Pixels

Traditional TTC computation relies on the information of location and motion for the boundary points or feature-points in interested obstacles. The performance of such method is limited by sub-pixel accuracy of line measurement. In Chapter 6, we have proposed a direct TTC computation method using both boundary pixels and interior pixels by accumulating suitable products of image brightness derivatives. Our method works directly with the derivatives of image brightness and does not require detecting objects, tracking features, estimating optical-flow, or any other "higher level" processing.

Integrating the contribution of both boundary pixels and interior pixels, our method greatly increases the calculation speed and accuracy, and can deal with situations when obstacles' sizes change by sub-pixel. When objects are very close and image sequences only capture the interior part of foregrounds, our method still provides accurate results. Traditional boundary-based methods would have inaccurate results since images are defocused and object boundaries fall outside of images.

Segmentation based on the Fusion of Boundary and Interior Image Pixels

Traditional segmentation algorithms are mainly based on identifying boundary pixels. The interior pixels within objects are not effectively used except for the template matching, edge detection or motion vector computation for boundary pixels. Instead, we take advantage of the fusion between boundary and interior pixels in segmentation.

In Chapter 2, both boundary and interior pixels contribute to producing binocular disparity-histograms. In Chapter 3, interior pixels also contribute to the motion-based expansion. Given initial segmentation, in order to decide whether to further absorb new surrounding

pixels or not, average motion/depth information of interior pixels is compared with ones of surrounding pixels. The similarities between interior and neighborhood pixels help to expand initial segmentation regions.

In Chapter 4, for infrared-based pedestrian detection, interior pixels within boundaries are included to compute the vertical projection of pixel brightness at each column. In Chapter 5, for pedestrian monitoring based on single visible camera, interior pixels are used to obtain CDI-VP-Curves. The contributions of these interior pixels result in distinct horizontal transition boundaries in the curves of vertical projection, which makes the performance of horizontal segmentation robust. Thus, it is feasible to search targets in the corresponding vertical stripes, which greatly simplifies traditionally difficult tasks and improves segmentation accuracy.

7.4.3 Fusion in Physical Connection among Obstacle Features

To understand driving situations, many conventional algorithms independently compute relevant static and dynamic information of targets, including distance ranges, segmentation, motion, TTC, etc. Information is independently detected or estimated and the mutual connection between these features is not efficiently used. Our scheme takes advantages of physical connections among these features in order to improve the accuracy of parameter estimation. The extra information, such as distance, motion, and historical dynamic information, can all be fused to enhance the detection accuracy.

The information fusion does not require high quality data. For example, though TTC results at each individual subsampling rate are not very ideal, TTC fusion results based on four computational models at different subsampling rates present much better performance than individual estimation. Furthermore, the extra distance information can be rough distance ranges with large error variance. Motion information can be noisy optical flows. The combination of these extra information helps to provide ideal segmentation accuracy. On the other hand, rough segmentation information helps to improve the detection accuracy of TTC estimation. The parameter estimation of TTC, focus of expansion and orientation are not sensitive to segmentation errors.

7.5 Conclusion

For different task requirements and environments, we have proposed various sensor fusion schemes to fuse data among various sensors. The fusion-based and layered-based principle is applied to detect and track the locations of obstacles in daytime visible images from stereo camera in Chapters 2 and 3, to identify pedestrians in nighttime infrared images from single camera in Chapter 4, to detect and track the locations of pedestrians in visible images from a single camera in Chapter 5, and to fuse TTC results from four gradient-based computational models with different subsampling rates in Chapter 6.

Our scheme is *task-oriented* and *object-oriented*. Instead of paying attention to all frame pixels and their complete 3D reconstruction, we focus on target objects and their relevant information. We split the original image map either into several distance-based layers, including background-layers, or projection-curve based vertical candidate stripes. The additional information helps to divide a complicated segmentation task into several simpler tasks for better performance than the original one. The combination of several simple segmentation operators is better than one complicated segmentation operator. By communicating information among sensors ahead of the final processing period and making use of their physical relationship, better environmental interpretation is achieved.

Chapter 8

Future Work

Here we will discuss several topics that might be interesting to further investigate, including the further investigation on previous research topics and the possible fusion scheme among different research topics.

8.1 Possible in-depth research topics for discussed systems

For the different systems mentioned in previous, there are the following possible topics to be investigated.

8.1.1 Application of "fusion-based layer-based 3D segmentation and detection" for tilted obstacles

The test examples used in Chapter 2 and Chapter 3 are mainly the video sequences captured on moving vehicles. The images of interested obstacles are mainly the frontal/back view, thus the depth ranges for obstacle pixels are wider than the situation when obstacles are tilted and both the frontal and rear ends of vehicles are visible. It would be interesting to investigate how algorithm will work in such situations.

8.1.2 Research on pedestrian detection in very cluttered scenes for both visible and infrared sequences

For infrared sequences in Chapter 4, we have not investigated the tracking performance before and after occlusion as we did for Chapter 5 except for the situation of partial occlusions in Figure (4-4)(c) in which pedestrians are detected accurately. We need to test more situations of partial occlusion to check whether the horizontal locations of pedestrians' heads still correspond to the transitional peaks or not, and to observe how the projection wave will change for fully occluded situation, i.e., when the horizontal locations of multiple heads overlap.

It would also be interesting to evaluate the performance of pedestrian detection in very cluttered scenes such as in a crosswalk where occlusion constantly happen for both visible and infrared sequences.

8.1.3 Possible research on "time-to-contact estimation"

In Chapter 6, we have developed fusion-based hierarchical method for direct gradient-based time-to-contact estimation. We can further pursue singular-value based fusion systems to improve its performance.

While our previous time-to-contact estimation is mainly used for visible video sequences, it is worth trying to apply the TTC estimation to infrared video sequences and to evaluate its performance.

While we compute TTC estimation, we do not have high requirement on pre-segmentation. TTC computation is reliable as long as majority regions are included. On the other hand, with newly computed TTC estimation, the possible sizes of interested obstacles are available. The additional information might help to improve segmentation performance. It might be feasible to develop an iterative scheme to accurately estimate the target sizes and TTC information.

8.2 Additional fusion schemes

While the projects described in previous chapters are independently implemented for different applications, there are lots of interesting connections among them that worth further investigation.

8.2.1 Application of "horizontal first, vertical scheme" scheme for general obstacles

So far, we can applied the detection scheme of "horizontal first, vertical segment" to pedestrian detection. It would be interesting to apply such schemes to detect non-pedestrian obstacles.

As discussed in Chapter 2 and Chapter 3, in the separated edge layers, the edge-line based morphological operation elongates the vertical edges at horizontal boundaries of interested targets and make them longer than at other horizontal locations of background regions. Thus the vertical projection curves of these pre-processed edge-maps of separated edge layers can provide useful sharp transitions. Boundary locations of interested targets can be determined by searching for horizontal locations of the left most and right most long edge-lines in separated edge maps. After some modification, the "horizontal segmentation" described in Chapter 4 and Chapter 5 might help to enhance segmentation accuracy.

8.2.2 Application of "fusion-layer-based dynamic tracking" scheme for infrared sequences and general obstacles

As discussed in Chapter 5, our layer-based tracking tracks only the horizontal locations and speed of obstacles, and actively combines the initial segmentation and tracking together to improve the understanding of obstacle's dynamics. The fusion-layer-based dynamic tracking can also used to track pedestrians in night vision as well as the general obstacles mentioned in Chapter 2 and Chapter 3. It is much more reliable to track the left and right boundary locations of interested targets and then apply "horizontal first, vertical segment" scheme to improve static segmentation for each individual frame.

8.3 Other possible applications and summary

Though our algorithms are developed for intelligent vehicles, they can be easily used in many new application area. In microbiology, to understand "chromosome congression in oocytes," biologist wants to track the movement of chromosome from three-dimensional time-lapse confocal fluorescence microscopy data of this active cytoskeletal process. Instead of segmenting chromosome separately in two continuous frames and associating them later, the fusion of initial segmentation and tracking mechanism based on Kalman filter would be helpful to better track its movement of chromosome.

In summary, we have proved that fusion-layer-based methodology is very useful for environment understanding. By fusing information among different sensors before their isolated operations and by decomposing original complicated task into several layer-based subtask, we have effectively improved the detection reliability and decreased the computational loads.

Bibliography

[1] M.Bertozzi, A.Broggi, A.Fascioli, S.Nichele. *Stereo Vision-based Vehicle Detection,* Proc. of IEEE Conf. on Intelligent Transportation System, 1997,pp.717-722.

[2] Bergendahl Jason. *A computationally efficient stereo vision algorithm for adaptive cruise control,* MIT Master thesis, 1997.

[3] Thomas Meier, King N. Ngan. *Automatic Segmentation of Moving Objects for Video Object Plane Generation,* IEEE Transactions on Circuits and Systems For Video Technology, Vol.8, No.5, September, 1998, pp.525-538.

[4] Rafael C.Gonzalez, Richard E.Woods. *Digital Image Processing,* Addison-Wesley Publishing Company, 1993.

[5] D. Willersinn, W.Enkelmann. *Robust Obstacle Detection and Tracking by Motion Analysis,* Proc. of IEEE Conf. on Intelligent Transportation System, 1997,pp.717-722.

[6] Charles XiaoJian Yan. *Planet Photo-Topography Using Shading and Stereo,* master thesis, MIT, 1993.

[7] Yajun Fang, Ichiro Masaki, Berthold Horn. *Motion-based Binocular Stereo: Fusion of Motion Stereo and Binocular Stereo,* AI Memo, MIT, 2001.

[8] J. S.Lim. *Two-Dimensional Signal & Image Processing,* Prentice Hall, 1990.

[9] D. Pelleg and A. Moore, *Accelerating Exact k-means Algorithms with Geometric Reasoning,* Proc. of the 5th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, August, 1999, pp.277-281.

[10] J. Shi, C. Tomasi. *Good Features to Track,* IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 1994, pp.593-600.

[11] M. Abdel-MOttaleb, R. Chellappa, A. Rosenfeld. *Binocular Motion Stereo using MAP Estimation,* Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1993, pp.321-327.

[12] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. *A Real-time Computer Vision System for Measuring Traffic Parameters,* IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 1997, pp.495-501.

[13] Mark Fiala, Anup Basu. *Hardware Design and Implementation for Underwater Surface Integration,* Proc. of IEEE International Conf. on Multisensor Fusion and Integration for Intelligent Systems, 1996,pp.815-822.

[14] Berthold Horn. *Robot Vision,* MIT Press, 1986.

[15] Tomohiko Ishikawa, Shigeyuki Sakane. *Estimation of Contact Position between a Grasped Objectand the Environment Based on Sensor Fusion of Vision and Force,* Proc. of IEEE International Conf. on Multisensor Fusion and Integration for Intelligent Systems, 1996,pp.116-123.

[16] Jung-Jae chao, Kuo-Chih Shao and Lain-Wen Jang. *Uncertain Information Fusion Using Belief Measure and Its Application to Signal Classification,* Proc. of IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent System, 1996, pp.151-157.

[17] A.Mirabadi, N.Mort, F.Schmid. *Application of Sensor Fusion to Railway Systems,* Proc. of IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent System, 1996, pp.185-192.

[18] N.S.Love, I.Masaki. *Three-Camera Stereo Vision for Intelligent Transportation Systems,* Proc. of SPIE's Photonic East 96 Symposium, Nov.1996

[19] N.S.Love, I.Masaki. *Recognition of 3D Compressed Images and Its Traffic Monitoring Applications,* Proc. of IEEE Intelligent Vehicles Symposium, pp.463-467.

[20] Peter Wide, Dimiter Driankov. *A Fuzzy Approach to Multi-Sensor Data Fusion for Quality Profile,* Proc. of IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent System, 1996, pp.215-221.

[21] Prasad Palacharla, Peter C.Nelson, Virginia P.Sisiopiku. *Data Fusion Using Fuzzy-Valued Logic,* Proc. of IEEE Intelligent Vehicle 94, Oct.24-26, 1994, pp.115-119.

[22] Kazunori Umeda, Jun Ota, Hisayuki Kimura. *Fusion of Multiple Ultrasonic Sensor Data and Imagery Data for Measuring Moving Obstacle's Motion,* Proc. of IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent System, 1996, pp.742-748

[23] Xiao-Gang Wang, Wen-Han Qian, Enrico Pagello, Ren-Qing Pei. *On the Uncertainty and Ignorance of Statistical Decision and Evidence Combination,* Proc. of IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems, 1996, pp.166-173.

[24] Jonathan S. Yedidia, William T. Freeman, Yair Weiss. *Characterization of belief propagation and its generalizations,* Mitsubishi Electric Research Lab Report, TR2001-15.

[25] Aki Lumiaho, "eEurope and the Information Society Technologies Programs to Boost the Intelligent Infrastructure and Intelligent Vehicle Deployment and Objectives in Europe," http: //www.netpark.or.jp/ ahs/ demo2000/ eng/demo_e /ahs_e7/aki/aki.html.

[26] MIT ITRC, MTL lab. Annual Review for MIT Intelligent Transportation Research Center (ITRC), 2001.

[27] Nicole S.Love, Ichiro Masaki, Berthold K.P.Horn. *Recognition of 3D Compressed Images and Its Traffic Monitoring Applications,* Proceedings of the IEEE Intelligent Vehicles Symposium, 2000, 463-467.

[28] Ichiro Mizunuma, Ichiro Masaki. *Evaluation of Auction and Bidding Algorithms for Resolving Network Resource Competition,* Proceedings of the IEEE Intelligent Vehicles Symposium, 2001, 449-454.

[29] T.Kato, Y.Ninomiya, I.Masaki. *An Obstacle Detection Method by Fusion of Radar and Motion Stereo,* Proceedings of the IEEE Intelligent Vehicles Symposium, 2001, 37-42.

[30] Yajun Fang, Ichiro Masaki, Berthold Horn. *Distance/Motion Based Segmentation under Heavy Background Noise,* submitted to IEEE Intelligent Vehicles Symposium, 2002.

[31] (2001, July). *Nebraska highway safety program and the lincoln-lancaster county health department. [Online].* Available: `http://nncf.unl.edu/eldercare/info/seniordriving/nightdrive.html`

[32] I. T. S. of America, "National intelligent transportation system program plan: A ten-year vision," the United States Department of Transportation, Tech. Rep., January 2002.

[33] C. Curio, J. Edelbrunner, T. Kalinke, C. Tzomakas, and W. von Seelen, "Walking pedestrian recognition," *IEEE Transactions on Intelligent Transportation Systems,* vol. 1, no. 3, pp. 155–163, Sept. 2000.

[34] C. Curio, J. Edelbrunner, T. Kalinke, C. Tzomakas, and W. Seelen, "Walking pedestrian rocognition," *IEEE Transactions on Intelligent Transportation Systems,* vol. 1, no. 3, pp. 155–163, Sept. 2000.

[35] D. Gavrila and J. Giebel, "Shape-based pedestrian detection and tracking," in *Proc. IEEE Intelligent Vehicles Symposium,* 2002.

[36] C. Papageorgiou and T. Poggio, "Trainable pedestrian detection," in *Proc. IEEE ICIP,* 1999, pp. 35–39.

[37] T. Tsuji, H. Hattori, M. Watanabe, and N. Nagaoka, "Development of night-vision system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 3, pp. 203–209, Sept. 2002.

[38] L. Zhao and C. Thorpe, "Stereo and neural network-based pedestrian detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 3, pp. 298–303, Sept. 2000.

[39] M.Bertozzi, A.Broggi, P.Grisleri, T.Graf, and M.Meinecke, "Pedestrian detection in infrared images," in *Proc. IEEE Intelligent Vehicles Symposium*, 2003, pp. 662–667.

[40] H. Nanda and L. Davis, "Probabilistic template based pedestrian detection in infrared videos," in *Proc. IEEE Intelligent Vehicles Symposium*, 2002.

[41] X. Fengliang and F. Kikuo, "Pedestrian detection and tracking with night vision," in *Proc. IEEE Intelligent Vehicles Symposium*, 2002.

[42] Y. Fang, K. Yamada, Y. Ninomiya, B. Horn, and I. Masaki, "Comparison between infrared-image-based and visible-image-based approaches for pedestrian detection," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2003, pp. 505–510.

[43] Y. Guilloux and M. Bruyas, "Aroto project: the benefit of infrared imagery for obstacle avoidance," in *Proc. IEEE Intelligent Vehicles Symposium*, 2002.

[44] C. Stauffer and W. Grimson, "Adaptive background mixture methods for real-time tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vo.2*, 1999, pp. 246–252.

[45] T. Darrell, G. Gordon, M. Harville, and J. Woodfill, "Integrated person tracking using stereo, color, and pattern detection," in *CVPR*, 1998, pp. 601–609.

[46] M. Bertozzi, A. Broggi, and etc., "Vision-based pedestrian detection: will ants help," in *Proc. IEEE Intelligent Vehicles Symposium*, 2002.

[47] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio, "Pedestrian detection using wavelet templates," in *CVPR*, 1997, pp. 193–199.

[48] C. Wohler, J. Aulaf, T. Portner, and U. Franke, "A time delay neural network algorithm for real-time pedestrian detection," in *Proc. IEEE Intelligent Vehicles Symposium*, Oct. 1998, pp. 247–251.

[49] Prem Kuchi, Prasad Gabbur, P. Subbanna Bhat, Sumam David S., Human Face Detection and Tracking using Skin Color Modeling and Connected Component Operators, IETE Jl. of Research, Vol. 38, No. 3&4, p.289-293, May-Aug 2002.

[50] Y. Ishii, H. Hongo, K. Yamamoto, Y. Niwa, Face and head detection for a real-time surveillance system, Proceedings of the 17th. International Conference on Pattern Recognition 2004, p.298-301, 2004.

[51] H. Sidenbladh , F. Torre , M. Black, A Framework for Modeling the Appearance of 3D Articulated Figures, Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000, p.368-371

[52] S. Dockstader, M. J. Berg, and A. M. Tekalp, Stochastic kinematic modeling and feature extraction for gait analysis, IEEE Trans. Image Processing, vol. 12, no. 8, p.962-976, Aug. 2003.

[53] T. Zhao, R. Nevatia, Tracking Multiple Humans in Complex Situations. IEEE Trans on PAMI. Vol.26, No.9, 2004, p.1208-1221.

[54] J. Wang and E. Adelson, Representing moving images with layers, IEEE Trans. on Image Proc., Vol.6, No.3, 1994, p.625-638.

[55] MM. Isard, A. Blake, CONDENSATION-Conditional Density Propagation for Visual Tracking, International Journal of Computer Vision 29, p.5-28, 1998.

[56] A. Utsumi, N. Tetsutani, Texture adaptation for human tracking using statistical shape model, Proceedings of the 16th International Conference on Pattern Recognition, vol.2, 2002, p.973-976.

[57] O.I. Burak, W.H. Wolf, A hierarchical human detection system in (un)compressed domains, IEEE Transactions on Multimedia, Vol.4, No.2, Jun.2002, p.283-300.

[58] M. Seki, T. Wada, H. Fujiwara, K. Sumi, Background Subtraction based on Co-occurrence of Image Variations, CVPR, II. 2003, p.65-72.

[59] Daniel J.Dailey, F.W.Cathey, Suree Pumrin. An Algorithm to Estimate Mean Traffic Speed Using Uncalibrated Cameras, IEEE Transactions on Intelligent Transportation Systems, Vol.1, No.2, June 2000, p.98-107.

[60] J. Wang, E. Adelson. Spatio-Temporal Segmentation of Video Data. SPIE Image and Video Processing II, Vol.2182, p.120-131.

[61] B. Stenger, P. R. S. Mendonca, and R. Cipolla, "Model-Based Hand Tracking Using an Unscented Kalman Filter", In Proc. British Machine Vision Conference, Vol. 1, p.63-72, Manchester, UK, September 2001.

[62] J.J., Jr.LaViola, A comparison of unscented and extended Kalman filtering for estimating quaternion motion, Proceedings of the American Control Conference, 2003, Vol.3, p.2435-2440.

[63] H.Sidenbladh, M.J.Black, and D.J.Fleet. Stochastic tracking of 3D human figures using 2D image motion, ECCV 2000.

[64] T. Zhao, R. Nevatia. Bayesian Human Segmentation in Crowded Situations. CVPR, 2004, vol.II. p.459-466.

[65] D. Comaniciu, V. Ramesh, P. Meer, Real-time Tracking of Non-rigid Objects using Mean Shift. CVPR. 2000. vol.2. p.142-149.

[66] T.Darrell, G.Gorden, M.Harville, J.Woodfill. Integrated Person Tracking Using Stereo, Color, and Pattern Detection. CVPR 1998. p.601-608.

[67] A. Francois, Real-Time Multi-Resolution Blob Tracking, IRIS Technical Report IRIS-04-422, University of Southern California, Los Angeles, April 2004.

[68] Berthold Horn, Yajun Fang, Ichiro Masaki, "Time to Contact Relative to a Planar Surface," Proceedings of IEEE Intelligent Vehicles Symposium 2007.

[69] The Intelligent Vehicle Initiative: Advancing "Human-Centered" Smart Vehicles. *Available:* http://www.tfhrc.gov/pubrds/pr97-10/p18.htm

[70] NHTSA 2020 Report. People Saving People – On the Road to a Healthier Future. *Available:* http://www.nhtsa.gov/nhtsa/whatis/planning/2020Report/2020report.html

[71] 2008 FARS/GES Traffic Safety Facts Annual Report (Early Edition). *Available:* http://www-nrd.nhtsa.dot.gov/Pubs/811170.PDF

http://www-nrd.nhtsa.dot.gov/Pubs/811170.PDF

[72] 2006 Traffic Safety Fact Sheets – older population. *Available:* http://www-nrd.nhtsa.dot.gov/Pubs/811161.PDF

[73] Senior drivers. *Available:* http://www.nysgtsc.state.ny.us/senr-ndx.htm

[74] Asia - New Hotbed for Consumer Automotive Electronics. *Available:* http://www.technewsworld.com/story/52539.html

[75] Frank Cremer, Wim de Jong, Klamer Schutte. Fusion of Polarimetric Infrared Features and GPR Features for Lanmine Detection, *2nd International Workshop on Advanced GPR*, 14-16 May, 2003, Delft, The Netherlands.

[76] Silvia Ferrari, Alberto Vaghi, "Demining Sensor Modeling and Feature-Level Fusion by Bayesian Networks", *IEEE Sensors Journal*, Vol.6, No.2, Apr.2006.

[77] Peter Haapaniemi. "Smart Vehicles Have Minds of Their Own", *Safety + Health*, November 1996, p. 64.

[78] IVI: 8 Major Problems. *Available:* http://www.its.dot.gov/ivi/8MPA.html#SI

[79] Enrico De Micheli, Vincent Torre, Sergio Uras, "The Accuracy of the Computation of Optical Flow and of the Recovery of Motion Parameters", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.15, No.5, 1993, pg.434-447.

[80] F. Meyer, "Time-to-collision from First-Order Models of the Motion Field", *IEEE Transaction on Robotics and Automation*, Vol.10, 1994, 792-798.

[81] F. Meyer, P. Bouthemy, "Estimation of Time-to-collision Maps from First Order Motion Models and Normal Flows", ICPR1992, 78-82.

[82] M. Subbarao, "Interpretation of image flow: a spatio-temporal approach", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.11, No.3, 1989, 266-278.

[83] B.K.P. Horn and B.G. Schunck. *Determining Optical Flow,* Artificial Intelligence, Vol. 16, No. 1-3, August 1981, pp. 185-203.

[84] A.R. Bruss and B.K.P. Horn. *Passive Navigation,* Computer Vision, Graphics, and Image Processing, Vol. 21, No. 1, January 1983, pp.3-20.

[85] B.K.P. Horn and S. Negahdaripour. *Direct Passive Navigation,* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, No.1, January 1987, pp. 168-176.

[86] B.K.P. Horn and E.J. Weldon. Jr. *Direct Methods for Recovering Motion,* International Journal of Computer Vision, Vol. 2, No. 1, Jun. 1988, pp. 51-76.

[87] S. Negahdaripour and B.K.P. Horn. *A Direct Method for Locating the Focus of Expansion,* Computer Vision, Graphics and Image Processing, Vol. 46, No. 3, June 1989, pp. 303-326.

[88] B.K.P. Horn. Parallel Analog Networks for Machine Vision, in Artificial Intelligence at MIT: Expanding Frontiers, edited by Patrick H. Winston and Sarah A. Shellard, MIT Press, Vol. 2, pp. 531573, 1990.

[89] I.S. McQuirk and B.K.P. Horn and H.-S. Lee and J.L. Wyatt. *Estimating the Focus of Expansion in Analog VLSI,* International Journal of Computer Vision, Vol. 28, No. 3, 1998, pp. 261-277.

[90] E. De Micheli and V. Torre and S. Uras. *The Accuracy of the Computation of Optical Flow and of the Recovery of Motion Parameters,* IEEE Transactions on PAMI, Vol. 15, No. 5, May 1993, pp. 434-447.

[91] T.A. Camus. *Calculating Time-to-Contact Using Real-Time Quantized Optical Flow,* Max-Planck-Institut fur Biologische Kybernetik, Technical Report No.14, February, 1995.

[92] P. Guermeur and E. Pissaloux. *A Qualitative Image Reconstruction from an Axial Image Sequence, 30th Applied Imagery Pattern Recognition Workshop,* AIPR 2001, IEEE Computer Society, pp. 175181.

[93] S. Lakshmanan and N. Ramarathnam and T.B.D. Yeo. *A Side Collision Awareness Method,* IEEE Intelligent Vehicle Symposium 2002, Vol. 2, pp. 640645, 1721 June 2002.

[94] H. Hecht and G.J.P. Savelsbergh (eds.). *Time-To-Contact,* Elsevier, Advances in Psychophysics, 2004.

[95] Yajun Fang, Marcelo Mizuki, Berthold Horn, Ichiro Masaki, "TV Camera-based Vehicle Motion Detection and its Chip Implementation", *IEEE Intelligent Vehicles Symposium*, 2000, pp.134-139.

[96] Yajun Fang, Berthold Horn, Ichiro Masaki, "Distance Range Based Segmentation in Intelligent Transportation Systems: Fusion of Radar and Binocular Stereo", *Proceedings of IEEE Intelligent Vehicles Symposium 2001*, 2001, pp.171-176.

[97] Yajun Fang, Berthold Horn, Ichiro Masaki, "Distance/Motion Based Segmentation under Heavy Background Noise", *Proceedings of IEEE Intelligent Vehicles Symposium*, 2002, pp.483-488.

[98] Yajun Fang, Yoshiki Ninomiya, Ichiro Masaki, "Intelligent Transportation Systems - Challenges and Opportunities", *the 2nd International Symposium on Multimedia Mediation Systems*, 2002, pp.72-77.

[99] Yajun Fang, Berthold Horn, Ichiro Masaki, "Depth-Based Target Segmentation for Intelligent Vehicles: Fusion of Radar and Binocular Stereo", *IEEE Transactions on Intelligent Transportation Systems*, Vol.3, No.3, Sept. 2002, pp.196 -202.

[100] Yajun Fang, Keiichi Yamada, Yoshiki Ninomiya, Berthold Horn, Ichiro Masaki,"A Shape-Independent-Method for Pedestrian Detection with Far Infrared-images", *Special Issue on "In-Vehicle Computer Vision Systems" of the IEEE Transactions on Vehicular Technology*, Vol.53, No.6, Nov. 2004 (pp.1679-1697).

[101] Berthold Horn, Yajun Fang, Ichiro Masaki, "Time to Contact Relative to a Planar Surface", *Proceedings of IEEE Intelligent Vehicles Symposium*, 2007.

[102] Yajun Fang, Berthold Horn, Ichiro Masaki, "Systematic information fusion methodology for static and dynamic obstacle detection in ITS," *15th World Congress On ITS*, 2008