Image Enhancement Using Statistical Spatial Segmentation

by

Peter Y. Yao

S.B., Massachusetts Institute of Technology (1995)

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

Master of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1996
[June 1996]

Author.....
.....
and Computer Science
May 28, 1996

Certified by.....
.....
Aaron F. Bobick
Assistant Professor
Thesis Supervisor

Accepted by.....
.....
Morgenthaler
Chairman, Departmental Committee on Graduate Theses

Image Enhancement Using Statistical Spatial Segmentation

by

Peter Y. Yao

Submitted to the Department of Electrical Engineering and Computer Science
on May 28, 1996, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Computer Science and Engineering

Abstract

Many image sequence processing applications are based upon the assumption that the motion of a given region can be well approximated by a parametric motion field. Fitting the entire region to a single motion model is one method of estimating motion, and requires a definition of the region in each frame of the sequence, or other motions in the image will bias the motion estimate. Often this definition is accomplished by a user providing the region in the first frame. Then, the region is subsequently updated by applying the estimated motion in each frame. This is a naive approach because it does not take occlusions, field of view, or region trackability into account.

This thesis accounts for these issues by segmenting an image into pixels consistent with the tracked motion and those that are not. This division is accomplished in three steps: first motion is estimated from the analysis region, whether the region is that specified by the user or the segmented region recovered in the previous frame. Second, all pixels that move according to this estimated motion are found; the test for consistency is based upon measured statistics of the image. Third, with a modified morphological operator, these pixels are grouped into coherent regions which define the new analysis region.

DYNAMO is an image manipulation system into which this thesis is built. It is shown that the motion estimation method described in this thesis improves DYNAMO's object tracking capability. In fact, this method allows DYNAMO to track objects that were previously impossible to track, allowing the system to handle more image sequences.

Thesis Supervisor: Aaron F. Bobick
Title: Assistant Professor

Contents

List of Figures

Chapter 1

Introduction

To humans, objects in movies are clearly defined. Their boundaries are easily discerned, and their motions are usually understandable. To a machine, however, image sequences are simply a set of pixel values with no contextual information. Usually the viewer does not care about the machine's representation of the image sequence, except when the concern is image processing. Depending on the type of processing, the machine must be able to understand what is happening in the scene, what is important to the user and what is not. This thesis is concerned with describing the motion of objects in a scene so that these objects can be latter manipulated. Formulating the description of motion is refered to as motion estimation, since the object's motion can not be exactly recovered. Using the motion estimation to segment the object from the image sequence is refered to as motion segmentation.

Motion estimation and segmentation is a classic machine vision problem that is broadly defined. In some motion estimation applications, the dominant motion of an image sequence is desired [4]. In another, several motions are estimated simultaneously [20]. This thesis is concerned with robustly estimating the motion parameters of a single, user defined object with possible occlusions by dynamically redefining the object features between each pair of frames.

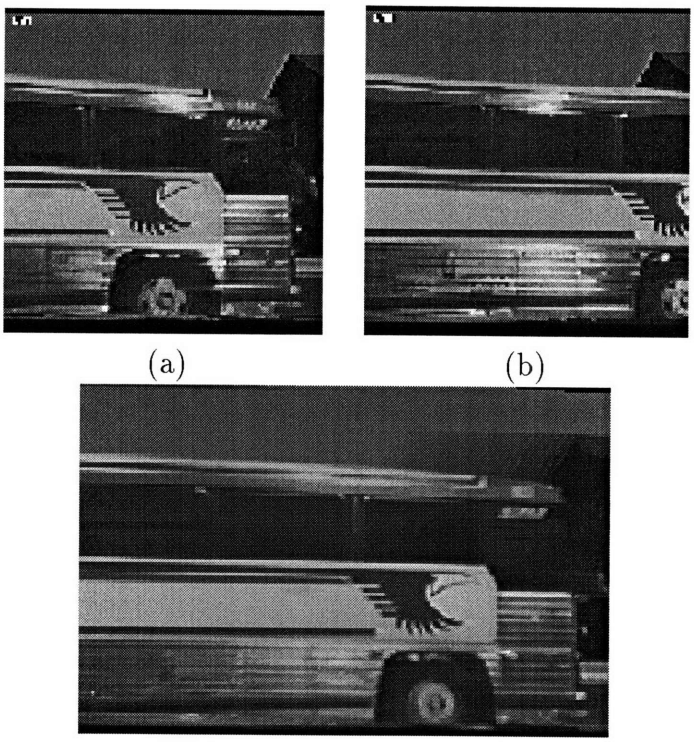


Figure 1-1: Registration of the bus sequence. (a) First frame of a 24 frame bus sequence. (b) Last frame. (c) 24 registered frames.

1.1 Image Sequence Processing

Image sequence processing using motion estimation, is capable of using motion parameters to "undo" the motion of an object in an image sequence. For example, consider two frames of bus sequence in Figure 1-1 in which a bus is moving past the camera. If the motion of the bus is known, that is, if the number of pixels that the bus moves between the pair of frames is known, the two frames can be overlaid, or registered, by applying the inverted motion of the second frame. Thus, the moving bus stays fixed relative to some frame in the sequence and the static background moves relative to the same frame in the resulting sequence. To produce the image in Figure 1-1 (c), the temporal median is taken of the registered sequence.

Notice the image of the registered frames captures the bus's features from all 24 frames. By using all 300 frames of the sequence, this technique can be used to capture an image of the whole bus in one frame, as seen in Figure 5-4.

There are many different applications that use the idea of registration including [14]:

Sequence Stabilization: Sequence stabilization removes motion from an object so that it is stabilized in the image sequence. Although it is implemented in different ways, this operation is popular in new "Steady Cam" camcorders to remove unwanted motion. Take for instance, a crew sequence, one frame of which is pictured in Figure 4-4 (a). In this sequence, the camera shakes from the motorboat motion and the crew shell moves from wave motion. Most "Steady Cams" will remove the motorboat action through mechanical isolation, but will not remove the motion of the crew. Through motion estimation, DYNAMO is able to stabilize the region of interest.

Noise Reduction: Once the object is stabilized in a sequence, a temporal statistic of the sequence, such as the temporal median, will produce a noise and occlusion reduced image of the object. Since the stabilization only stabilizes the moving object and not other "objects" like dirt on the lens, specularities on the object, or occlusions, these other objects will move in the stabilized sequence. Then, the median operator effectively removes them from the resulting image because they are unlikely to occupy the same pixel locations over time. An example of this is shown in Figure 5-2 (d).

Mosaicing: As explained, taking the median image of the stabilized sequence will mosaic the object together if different parts of the object lie in different frames of the sequence. In effect, this increases the camera's field of view [3]. Another way to view this application is tiling a larger image with many frames of a sequence. This allows lens size and focal length to drop without losing resolution. For instance, the image in Figure 1-1, at its resolution, could not have been taking by the camera with one still shot.

Image Sequence Compression: Motion estimation and segmentation can also be used in an image sequence compression scheme, where only the objects and

their motion parameters are transmitted instead of the entire sequence [20]. For instance, an image sequence of a car driving down a street would be transmitted as a still image of a car, a still image of the street, and the motion parameters of the car and street. This is of particular interest for teleconferencing, where bandwidth is restrictive and only a few motions are usually present.

1.2 The DYNAMO Application

DYNAMO is a motion analysis utility and provides a framework to test this thesis. It was originally written by John Wang and Ted Adelson, and a Tcl/Tk X interface was added by this author. It is designed to track moving objects and then perform the above mentioned operations. Both temporal median and temporal average operators are included in the package.

The utility supports raw datafiles and datasets, both grayscale and color. It also deinterlaces and deinterleaves images where appropriate, and supports cropped and time subsampled inputs. A small image cache increases performance, and an incremental median operator makes medians of large image sequences possible. DYNAMO uses the Utah Raster Toolkit (URT) to display images and sequences.

1.3 Goal

The ultimate goal of this work is to increase the robustness of estimating an object's motion. Specifically, this thesis addresses three problems with DYNAMO's current motion estimation method. First, the current method suffers when the tracked object is not always in full view. Second, it is unreliable when another object is occluding the tracked object. Last, the current method depends heavily on the user's definition of the object to be tracked, which is usually not optimal and sometimes insufficient for accurately estimating motion. All three problems are solved by dynamically updating the object's definition between each pair of frames by using previous motion estimates. This definition is the object's spatio-temporal location in the sequence, which is hence

refered to as the analysis region.

1.4 Outline of Thesis

In Chapter 2, DYNAMO's affine motion estimation method is explained, and three problems with DYNAMO's current motion tracking method are described, along with an overview of the dynamic analysis region solution.

Chapter 3 presents work related to the motion estimation and segmentation problem. Specifically, multiple motion analysis [14], hierarchical motion estimation [6], convergence of motion estimation [4], and motion segmentation techniques [1] [16] [21] are described.

Chapter 4 describes how the analysis region is updated between frames, and how the problem of noise and interpolation error is solved.

Chapter 5 compares the performance of the dynamic analysis region to the current static analysis region and concludes with ideas for possible future work.

Chapter 2

Dynamo's Motion Estimation Method

2.1 Parameterized Motion Fields

The image processing applications mentioned in Section 1.1 rely on quantitative knowledge of how objects move in a sequence. Since uncompressed image sequences are not stored with object motion parameters, the object's motion must be estimated before processing. To estimate motion, a description of how two pairs of images relate is formulated,

$$I_1(x, y) = I_2(x - V_x, y - V_y) \tag{2.1}$$

where $I_1(x, y)$ is a pixel in frame 1, and V_x and V_y are motion displacements for these pixels. Instead of assigning a V_x and V_y for every pixel, the motion can be parameterized as a motion model. This parameterization constrains the motion, providing a very simple description of motion. In the real world, pixels that correspond to rigid objects have a definite spatial relation to each other, which should be preserved by the motion model used.

The motion in the image sequence will drive the selection of motion model. For the bus sequence in Figure 1-1, it suffices to use a two parameter model which accounts

for translation:

$$\begin{aligned} V_x &= A_x \\ V_y &= A_y \end{aligned} \tag{2.2}$$

where A_x and A_y are constants describing the horizontal and vertical motion. For sequences with other motions, a six parameter affine motion model as in equation 2.4 corresponds exactly to a plane moving arbitrarily under orthographic projection. The affine model will track rotation, shear, zoom, and translation:

$$V_x = A_x + B_x x + C_x y \tag{2.3}$$

$$V_y = A_y + B_y x + C_y y \tag{2.4}$$

If an object's motion is large compared to its distance from the camera, perspective effects become significant and an 8 parameter projective model may be employed to account for changes in perspective [6]. The projective model describes the general motion of a plane moving in 3D space by:

$$\begin{aligned} V_x &= A_x + B_x x + C_x y + D_x x^2 + E_x xy \\ V_y &= A_y + B_y x + C_y y + D_y xy + E_y y^2 \end{aligned} \tag{2.5}$$

2.2 Affine Motion Estimation

DYNAMO is tailored to track rigid objects, such as vehicles as opposed to people, moving in three dimensional space. Objects such as people or water flow require more than one description of motion because their individual parts do not move with the same motion. In these cases, an optic flow [12] approach would be more appropriate because it attempts to independently describe how each pixel moves between an image pair. Usually optic flow assumes motion is smooth, which makes it suitable for flows. However, this creates an inconsistency at motion boundaries, where there is a sharp change in motion. For instance, the pixels in a boundary between a moving

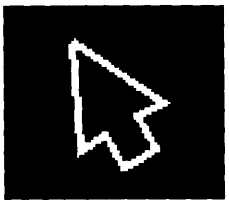


Figure 2-1: An ambiguous scene: a solid arrow, or an arrow outline?

foreground and static background will have contributions of both motions. The global affine motion estimate does not assume motion is always smooth, making its motion estimation more accurate in many cases where a rigid object is being tracked.

The assumption that an entire rigid object undergoes the same motion allows a great deal of expressive power, but it also leads to an ambiguous interpretation in a large region of constant intensity. If a region is comprised of pixels with the same value, the region can fit any motion model because they will fit any small transformation from one frame to the next. In other words, in a constant region, for any small V_x and V_y, $I_1(x,y) = I_2(x-V_x, y-V_y)$. This means that constant regions do not add any useful information that the estimation can use, nor is there any analytical method to determine whether a constant region is moving or not. Figure 2-1 shows an example of this ambiguity. The moving arrow can have two interpretations, either the outline of a moving arrow, or a solid moving arrow because of the constant region in and surrounding the arrow. This ambiguity is not limited to just affine motions, but is a more general problem with image processing.

To estimate the six affine parameters A_x, B_x, C_x, A_y, B_y, and C_y, DYNAMO minimizes equation 2.1 within the bounds of the object to be tracked. The user defines the object they wish to track by indicating a rectangular region, called the analysis region, on some image in the sequence. There is no reason (other than ease of implementation) to require a rectangular analysis region. However, only six linearly independent pixels are needed to determine the six parameters therefore defining the entire object is not necessary. The only important advantage of using more points in the estimation is to reduce the effect of outliers and noise. Also, note that by estimating the motion of a subset of the object, the motion of the whole object is estimated. For example, in Figure 1-1, it makes nominal difference if the analysis

region is the eagle on the side of the bus, or the entire front of the bus.

The minimization is accomplished by using a linear gradient least squares approach similar to Bergen et. al [7] and [6]. A criterion function, is defined by first expanding equation 2.4 to first order:

$$I_2(x,y) \approx I_1(x,y) - I_x V_x - I_y V_y \tag{2.6}$$

Then the error measure E is defined to be:

$$\begin{aligned} E &= (I_2(x,y) - I_1(x,y) - V_x I_x - V_y I_y)^2 \\ &= (I_t - V_x I_x - V_y I_y)^2 \end{aligned} \tag{2.7}$$

where I_x, I_y, and I_t are partial derivatives of the image in x, y, and time respectively. Minimizing equation 2.7 is then linear with respect to the affine parameters.

A linear least squares solution is obtained by substituting the affine motion equations 2.4 into the linearized motion equation 2.7 and solving. Letting $\phi(x,y)$ be the regressor and θ be the vector of affine parameters, then equation 2.7 can be rewritten as:

$$\phi^T(x,y) = [I_x \; xI_x \; yI_x \; I_y \; xI_y \; yI_y] \tag{2.8}$$

$$\theta^T = [A_x \; B_x \; C_x \; A_y \; B_y \; C_y] \tag{2.9}$$

$$I_t = \phi^T \; \theta + E \tag{2.10}$$

Letting $\hat{\theta}$ be the linear least squares solution of θ, the following is obtained:

$$\hat{\theta} = \sum(\phi\phi^T)^{-1} \sum \phi I_t \tag{2.11}$$

where summation is taken over the entire analysis region.

In equation 2.11, the $\phi\phi^T$ term is a 6x6 matrix and the ϕI_t term is a 6x1 vector. The estimation produces meaningful result only if the linearization in equation 2.7 is valid. This is typically true if the motion is small so that the intensity change due to

motion can be modeled by the local intensity gradient.

Two modification are used enhance the linear gradient least square method. To estimate large motions, a multi-level pyramid scheme is used to reduce relative spatial features. An iterative estimation algorithm is used to increase estimation accuracy. In this work, both techniques compliment each other, although they can be used independently.

2.2.1 Coarse-to-Fine Pyramid

The coarse-to-fine algorithm allows for larger displacements by reducing relative spatial distances through a Gaussian pyramid of images. Each image in the pyramid is a copy of the original image convolved with a small filter and reduced in resolution by a power of 2. Let G_l be the l^{th} level of the pyramid [6]:

$$G_l = [G_{l-1} * w]_{\downarrow 2} \tag{2.12}$$

where w is the small kernel filter and $\downarrow 2$ indicates the image is being subsampled, where every other row and column is discarded. Thus, G_0 corresponds to the original image. The conceptual idea is to eliminate small features in the image (perhaps due to noise), and reduce image size so that only a small region needs to be registered. Subsampling by a power of 2 reduces image size, although it is not restricted to powers of 2. Convolving with a small filter removes high frequency content so that these frequencies are not amplified in the reduced resolution image.

The procedure begins by creating a pyramid for both frames being considered. Then, the affine motion estimator determines the motion for the image pair with the lowest resolution. If there are l levels in the pyramid, this pair is subsampled by 2^l. After the affine parameters are computed, this estimate is used as a basis for the $l-1^{th}$ level. This continues until the 0^{th} level in which motion is estimated on the original images. A basis for the motion is obtained from previous levels so that large displacements in the original images can be measured.

Thus, coarse-to-fine analysis can obtain motion estimates of many pixels per frame

by using low resolution images while retaining the accuracy of a small fraction of a pixel by using the high resolution images. By iteration of the warping procedure, the estimation is made more precise.

2.2.2 Iterative Motion Estimation

By iterating the motion estimation process, estimation accuracy is improved. After computing the initial affine parameters, the first frame is warped toward the second and then residual motion is estimated between between the warped frame and second frame. Several estimates is better than one because the distance between the warped frame and the second frame, where the image is smooth, is reduced with each iteration which means with each iteration, pixels are less likely to be confused.

Let V_k be the velocity estimate obtained after the k^{th} iteration of the motion tracking process. During the k^{th} step,

1. The first image frame, $I_1(x, y)$ is warped toward the second image, $I_2(x, y)$ according to the velocity estimate V_{k-1} [14]:

$$I_1^{V_{k-1}}(x, y) = I_2(x - V_x, y - V_y) \tag{2.13}$$

2. The estimation is applied to $I_2^{V_{k-1}}(x, y)$ to obtain the residual motion, ΔV_k.

3. The estimated motion is updated:

$$V_k = V_k + \Delta V_k \tag{2.14}$$

The method of iteratively estimating motion can be used in conjunction with the coarse-fine image pyramid. On each level, motion is estimated until convergence (or for some defined number of iterations is reached). Then, the motion estimate is passed to the next level in the pyramid. The iterative method is especially useful for pyramids where images are greatly subsampled. If subsampling is less drastic, $\sqrt{2}$, for instance, and there are enough levels of the pyramid, the iterative process is not

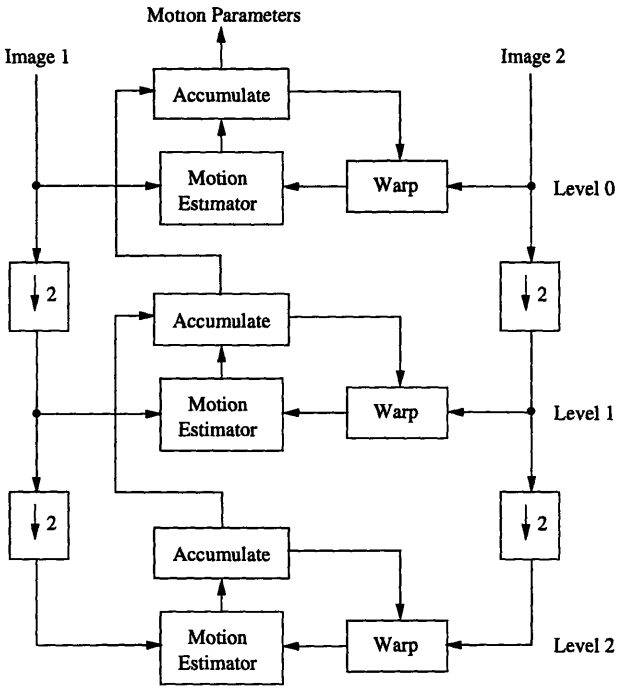


Figure 2-2: Block diagram of the motion estimation procedure

necessary. As long as the lowest level of the image pyramid produces a reasonable estimate, features in subsequent images of the pyramid will be sufficiently close.

To summarize, Figure 2-2 shows a schematic of the motion estimation procedure.

2.3 Motion Tracking

Given the three estimation methods described, linear gradient least squares, coarse to fine pyramid, and iterative estimation, an accurate estimate is achieved if the region under analysis undergoes the same motion. In other words, the motion estimation scheme described does not work if the analysis region contains more than one motion which makes the definition of the analysis region crucial.

The analysis can depend on the initial analysis region because a user defines the region. If the user defines a region with more than one motion, inaccurate motion is due to operator error. It is sufficient to leave this analysis region static throughout the entire sequence if the tracked object shares some common spatial location, recalling

that a constant region is not considered part of the object. This seems reasonable because some feature of the object will be in this common location making it trackable.

For sequences that do not satisfy this constraint, a different analysis region must be chosen for each pair of frames. It is unreasonable to ask the user to define a region in all subsequent image pairs, so a motion tracking scheme is devised in which the (rectangular) analysis region is updated with the results from the current estimation. If $R(x_1, y_1)$ and $R(x_2, y_2)$ define the analysis region in the first frame, $R(x_1+V_x, y_1+V_y)$ and $R(x_2 + V_x, y_2 + V_y)$ defines the analysis region for the second frame. Again, there is no reason to constrain the analysis region to a rectangular region other than implementation simplicity. For an arbitrarily shaped region, each point would be warped with the estimate to produce the analysis region in the subsequent frame.

2.4 Motion Tracking Problems

There are problems with both of the motion tracking methods mentioned in Section 2.3. First, it is often difficult for a user to discern if the tracked object shares some common spatial location throughout the sequence. Therefore, holding the analysis region static is not a reasonable option. Updating the analysis region with the estimated motion also suffers from problems because it does not take changing imagery into account; it requires that the relative spatial features of the tracked object stay fixed throughout the entire sequence and does not take into account the camera's field of view, the initial analysis region, and occlusions found that enter later in the sequence. Some of the typically difficulties are:

Analysis Region Leaves Field of View: Suppose the object is larger than the camera's field of view, as in the bus in Figure 1-1. By laying an analysis region at the front of the bus, its motion can be estimated, but if the analysis region follows the front of the bus, it will eventually disappear from the image sequence as in Figure 1-1(b), leaving nothing for the motion estimator to track. This happens with $x + V_x$ or $y + V_y$ is greater than the dimensions of the image. In this particular example, it suffices to hold the analysis region stationary, since

the bus only undergoes translational motion and always occupies the same pixel locations.

Analysis Region with Few Features: Another problem lies in the fact that the user must choose the analysis region. In the worst case, he might choose a region that lies in a constant region. Even though a human realizes this part of the image belongs to the object, a constant region can fit many motion model, as shown in Figure 2-1. Thus, a constant region will return unpredictable results.

A problem that is more likely and less consequential, is an analysis region with little contrast. If the region has intensities that do not vary very much, or if the region has a periodic pattern, the motion estimator could still return an unpredictable value. The best region to choose is one with high contrast or textures such as edges.

Analysis Region is Occluded: Occlusions biasing the motion estimated is the last problem that the motion updated analysis region encounters. It was previously stated that the analysis region must contain only one motion, or the resulting motion estimate will be biased by all motions. Since the analysis region is updated only with the previous motion estimate, there is no way to ensure that other objects will not enter this region and bias the estimate.

2.5 Possible Motion Tracking Solution

This thesis investigates the possibility of updating the analysis region with the estimated motion, but instead of using the rectangular user defined region, creating a new analysis region comprised of all pixels that seem to move with the estimated motion. To see if pixels move with the same motion, the estimated motion V_x and V_y will be applied to every pixel in I_1. If the pixel moves according to this motion, i.e., if its value after warping is the same as the corresponding pixel value in I_2, then it should be included in the analysis region in the next frame. This method takes advantage of the assumption that pixels moving with the same motion belong to the

same object, and thus will continue to all move with the same motion.

Since the entire image is checked for these pixels, there is no danger of losing an analysis region due to field of view size, unless the object does indeed leave the entire field of view. Because each pixel is chosen individually, a pixel can be chosen only if it lies in a region with sufficient features. Lastly, occlusions are not a problem because their pixel values after warping will not match the pixel in I_2. The rest of this thesis will discuss the implementation and performance of the dynamic analysis region.

Chapter 3

Related Work

The literature on robust motion estimation and segmentation is divided between an non-parametric approach such as optic flow and a parametric motion estimation approach similar to the one described in Chapter 2. The optic flow approach assigns each pixel a translational vector containing local motion information so that the motion in an image pair is described by a flow field [12]. This method is popular ([2] implements and compares nine techniques, namely differential methods, region-based matching, energy-based, and phase-based techniques) because it can simultaneously compute multiple motions by clustering the translational vectors [20]. However, small regions need to be used to ensure that only one motion is present in the region, often making motion computation inaccurate because small regions carry little information.

The parametric motion estimation approach also suffers from the restriction that only one motion is present in the analysis region. The parametric motion estimator, as described in Chapter 2, describes motion over a large spatial region using a motion model so that the estimated motion is constrained. Then, when multiple motions are present in the analysis region, the model parameters should be recovered as well as the motion discontinuities. Ideas such as discontinuity detection [8] or outlier detection [13] have been proposed for this goal. This section will review related work using parametric motion models; we will consider both parameter estimation and motion segmentation.

3.1 Parametric Motion Models

For an excellent background on applications of parametric motion models, Irani and Peleg [14] describe the use of motion estimation to perform temporal integration. This is another way of describing registration and median filtering as discussed in Chapter 1. In [14], the dominant motion is estimated and eliminated from the sequence by registering all frames. This sharpens the region of dominant motion and blurs other regions. This eases a static segmentation process that is necessary to distinguish the dominant regions from other regions. The results of the static segmentation are used to create a mask for use in subsequent processing. After the dominant motion is estimated, the sequence can be registered with respect to the dominant motion and other objects can then be tracked.

Irani and Peleg also discuss the reconstruction of occlusions and the improvement of spatial resolution, as stated in Chapter 1. One deficiency in this work is the dependence on a dominant motion. If a scene is comprised of many small objects in motion, the initial motion estimate will have a component from all objects and thus will not register any one of them. If a dominant motion is assumed to exist, there is also the problem of estimating all object motions even if only one, non-dominant motion is desired.

To increase motion estimation accuracy and efficiency, in [6], Bergen et. al develop their hierarchical motion estimation, as described in Section 2.2.1. Another main motivation of the work was to provide a diverse set of motion models to the motion estimator so that the most appropriate model can be used for the motion estimation. Specifically, they implement fully parametric, quasi-parametric, and non-parametric models, to account for different sequences.

The fully parametric models include an affine flow and planar surface (or projective) model. They are considered fully parametric because they describe the motion of a pixel region with parametric terms. As stated in Chapter 2, the affine model accounts for translation, rotation, skew and zoom. To fully model a plane moving in 3D space, the planar surface model is used.

A rigid body model is implemented as an example of a quasi-parametric model. It is described as quasi-parametric because there is a global parametric component that fits the entire image, as well as a local component which varies from pixel to pixel. The model is the same as the planar surface model, providing the parametric component. However, it assumes a pixel's depth, it's distance from the camera, is the same over a local patch, providing the non-parametric component. This model is used when arbitrary surfaces undergo a rigid motion that a single global model cannot describe. The local range information is also used to determine local image properties, such as the range value.

Finally, the general flow field is presented as a way to describe sequences with more than one motion. Because there are multiple motions present, a global parameterized model will not suffice. In their implementation, the local model is a constant flow model within 5x5 pixel windows. Through the use of these small local windows, the general flow field is able to simultaneously detect multiple motions in a sequence. However, motion errors do occur at motion boundaries, because the local windows assume a smooth motion, and for patches of constant intensity, because the local image structure is ill defined.

Even though the particular implementation of the fully and quasi parametric models assume a dominant motion (which is often false), Bergen raises an interesting point, that the image sequence should drive the motion model selection, and that multiple models can be used within the same motion estimation framework.

Multiple models are also used in [4] where the *convergence* of registration error is examined, rather than the registration error itself. Multiple models are used so that the approach can start with a simple translational model, move to an affine model, and then to a moving planar surface model, as in [6]. In this application, multiple models are used to reduce the effects of multiple motions in the sequence. The translational model finds a single motion of a single image region especially well, providing an initial segmentation so that higher order models will only have components from a single motion.

3.2 Motion Based Segmentation

Segmentation is the decomposition of an image into regions of a uniform attribute. Often, it is used to define or detect objects in an image. An image can be segmented in many ways, by intensity, color, or texture are three possibilities. Motion segmentation decomposes an image into regions based on motion; regions of like motion are grouped together. Motion segmentation often conveys more information about a region's structure than does static segmentation, and in some cases the two methods are used jointly.

One motion based segmentation algorithm is described in [1]. Unlike the work in [4], Ayer et. al attempt to segment moving objects from an image sequence. [4] was simply concerned with segmenting different motions, not necessarily different objects. Again, the method described assumes a dominant motion, and by assigning a "goodness of fit" to regions, motion parameters can be determines for those regions for which the previously estimated motion parameters are not valid.

Motion is estimated by a the least median of squares (LMedS) and the least-trimmed squares (LTS) instead of the least squares method that was described in Section 2.2. These robust methods are much less affected by outliers and deviations from Gaussian noise. The goodness of fit measure is the sum of the residuals which has a chi-square distribution assuming Gaussian noise. A threshold for the measure is obtained such that the probability of a chi-square random variable exceeding it is 5%, although it could be set to any accuracy. Then, if a segmented region from the sequence has a measure lower than the threshold, it's parameters are considered invalid. Then, valid and invalid regions are merged temporally to result in a set of mask sequences that can be used for further motion estimations. It is assumed that the number of different moving objects does not change throughout the sequence.

This algorithm works well for sufficiently texture regions but as expected, not for regions with similar gray values. To handle these regions, a static segmentation method [19] embedded in a multi-resolution framework is applied. Motion parameters are compared to merge regions together during this step of the algorithm. Further-

more, by using the motion parameters from every frame in the sequence, the certainty that a subregion is classified correctly increases, i.e., if a subregion has similar motion parameters in every frame of the sequence, that region can be classified with confidence.

This step of static segmentation is superfluous for this thesis, but raises possibilities for motion segmentation. This thesis is only concerned with segmenting trackable features from an object, not the whole object itself. However, the success of the static segmentation in [19] gives evidence that the thesis could possibly be extended to object segmentation.

Another notable work [16] tracks objects in a sequence by representing them as a silhouette of their projection. The motivation here is to efficiently solve occlusion problems and easily resolve scene interpretations. The algorithm starts with a motion-based segmentation that uses a statistical regularization approach with 2D first-order motion models to ensure stable motion-based partitions [10]. The motion-based segmentation does not depend on optic flow, nor does it require explicit 3D measurements. The partitions are simply the vertices of the object that lie on it's convex hull. This approximation is sufficient since the partition is only used to form a correspondence between it and the predicted regions. Lastly, a recursive filter refines the prediction to obtain the estimates of the region location and shape in the image. Two filters are used toward this goal, a geometric filter and a motion filter, both of which interact to estimate shape, position and motion of the region.

The results of [16] show an approximate spatial description of the tracked object can be useful to solve the problems of some occlusions present in image sequences. It is unclear how well the system handles objects that reappear after being occluded. For instance, a vehicle passing in back of a sign post will split the vehicle into two regions. When the vehicle passes the sign post, it one region will be left, but with a different label than before the occlusion. Despite this problem, if the motion-based segmentation is robust enough, the region-based tracking method seems to perform reasonably well.

This approach is similar to an earlier attempt at automatic expansion of the

analysis region [21] in that some spatial model of the object was to be approximated. In order to simplify the procedure, the initial and resulting analysis regions were constrained to be rectangular. The premise was to repeatedly add rows and columns to the analysis region that had a mean registration error under a certain threshold. The Improvement Measure found in [4] was used to quantify registration error. In some aspects, the threshold used was dynamic; it was defined to be some number, or fraction, of standard deviations away from the mean registration error of the initial analysis region.

The motivation behind the work was to create an analysis region that was as large as possible, to increase estimation accuracy by providing the maximum number of object features to the motion estimator. This method was found to be ineffective for a number of reasons. First, the constraint that the analysis region must be rectangular is unnecessarily prohibitive. Very often, objects being tracked are not rectangular, thus restricting the amount the analysis region could grow.

Second, there was no means for the analysis region to contract. It always assumed the initial analysis region was always correct (i.e., it always contained only a single object motion), therefore, incorrectly adding to the analysis region, would raise the mean registration error of the analysis region and thus make it easier to incorrectly add more to the analysis region. One solution to this problem is to be very conservative when adding rows and columns. However, the threshold between being conservative and aggressive was easily crossed, making the analysis region either static (not expanding) or uncontrollable.

The last problem with this region growing method relates back to the rectangular constraint problem. Each row and column is checked to see if it is within the threshold. If a non-horiztonal or vertical boundary is met, rows may be added even if some pixels lie off the object because these few pixels will not move the mean registration error of the row or column sufficiently far away from the mean of the analysis region. However, these few pixels off the object will increase the mean registration error on the subsequent analysis region. In this way, then mean of the analysis region is either monotonically increasing (if threshold is greater than the mean of the analysis region)

or static (if threshold is less than the mean).

Chapter 4

Segmenting Trackable Features

The underlying goal of this thesis is to remove the dependency on the user's analysis region to estimate motion. This dependency is removed by using a previous motion estimate to augment and correct the analysis region. Once these regions are found, they can be included in an analysis region to stabilize the analysis throughout the entire sequence. By finding these new regions in each frame of the sequence, the motion estimation is no longer dependent on the initial analysis region provided by the user. Even if the user provides an analysis region that does not contain the entire object being tracked, or if it contains parts of other objects, the analysis will converge on trackable features within a few frames.

Motion is estimated as described in Section 2.2. The image is segmented through the use of an *Improvement Measure, M* (equation 4.1). It suffices for this section to mention the Improvement Measure lies between 1 and -1, tends toward $+1$ if registration decreases prediction error, and tends toward -1 if registration increases

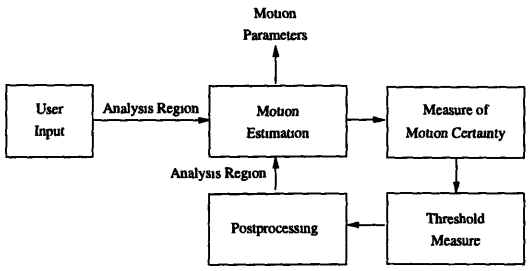


Figure 4-1: Block diagram of the dynamic analysis region algorithm

prediction error. A *Reliability Measure, R* is used to weight the Improvement measure, and a *Segmentation Measure, S* is used to avoid having to threshold the Improvement Measure. The algorithm is as follows:

1. Given two frames $I_1(x, y)$ and $I_2(x, y)$, the global translation of the entire image is calculated. This step converges to the dominant translation even without segmentation.

2. Frame I_2 is warped toward I_1 using the motion parameters that were computed in the previous step.

3. The Segmentation Measure $S(x, y)$ is computed.

4. The above process is repeated for higher order motion models, where the translation motion model is followed by the affine model, which is followed by the moving plane model. For all motion models, except the translation model, motion computation was done when points were weighted by the segmentation mask S, so that only points in the dominant region are influencing the computation [4].

The user's analysis region is only used to seed the segmenting algorithm, as shown in Figure 4-1. After the first motion estimation, motion parameters are used to create a new analysis region implemented as a binary mask for the image. The regions included in the analysis should fall in the corresponding binary 1 values of the mask, and conversely, the regions that are excluded lie in the binary 0 values of the mask. The mask is produced in three steps. First, each pixel is assigned a measure of the certainty that it is moving with the estimated motion model. Then, a mask is created with the set of pixels that fall over a certain threshold. Modified morphological operations are then applied to this mask to remove noise and consolidate patches. The mask is passed to the motion estimator to be used as the analysis region for the subsequent motion estimation.

For the purpose of this thesis, an image sequence with two motions is discussed. In Figure 4-2, R_A is a moving region whose motion model is estimated, and R_B is a

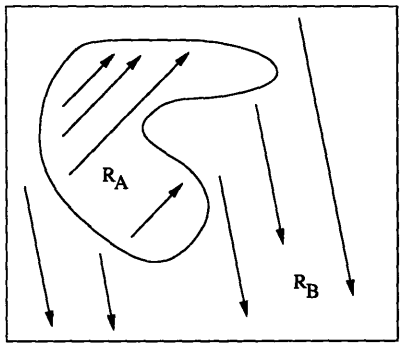


Figure 4-2: Different types of regions in an image sequence. R_A is a region whose motion is estimated, R_B is a region whose motion is disregarded.

background region whose motion model is disregarded. There is no loss of generality by restricting the number of motions; the same principles will hold for arbitrary number of motions.

4.1 Motion Certainty Measure

At the heart of the segmentation is the assignment of a measure to each pixel that indicates the certainty that it moves according to the object's motion parameters. Irani and Peleg describe a measure that does this [4]. A modified measure which will be refered to as the Motion Certainty Measure, or MCM, is defined for a pixel as:

$$M = \frac{I_t^2 - I_{tw}^2}{I_t^2 + I_{tw}^2} \tag{4.1}$$

where $I_t = I_2 - I_1$ and $I_{tw} = I_{2w} - I_1$. I_1 and I_2 are the pixels in images 1 and 2, and I_{2w} is a pixel in image 2 warped toward image 1 using the estimated motion. This measure has three interesting properties:

1. The normalization $I_t^2 + I_{tw}^2$ forces the constraint $-1 \leq M_i \leq 1$.

2. $M_i \approx 1$ if $I_{tw} \ll I_t$, which occurs when the estimated motion decreases the registration error. This indicates the pixel belongs to region R_A.

3. $M_i \approx -1$ if $I_{tw} \gg I_t$, which occurs when the estimated motion increases the registration error. This indicates the pixel belongs to R_B.

M is also defined to be 0 if both $I_t = 0$ and $I_{tw} = 0$, implying that the registration error does not change, even when the motion model is applied. This occurs when the motion parameters are all 0, or when a pixel lies in a constant region where applying the motion simply moves the pixel around the constant region. In this case, it cannot be determined whether the pixel is actually moving according to the motion model, or if it stays static. As Figure 2-1 shows, both physical interpretations are possible.

Ideally, pixels lying in R_A should all have a MCM of +1 because I_{tw} should be 0. However, because of interpolation error, I_{tw} is expected to have some variance around 0. Therefore, the distribution of the MCM to be biased toward +1 for pixels in R_A since I_t is large, and I_{tw} is likely to be close to 0. For pixels in R_B, the distribution of the measure of motion certainty to be centered around 0 since both I_t and I_{tw} have the same distribution and no conclusions about their relative size can be made.

4.1.1 Interpolation Error

Objects in the real world are represented in a digital image as an array of discrete points. This allows images to be easily manipulated, but it creates a problem when warping images some non-integral distance. Suppose an image is translated .2 pixels in some direction. Since this does not map pixels directly onto other pixels, the resulting pixel values are unknown.

The standard solution to this problem is to interpolate the resulting pixel value from neighboring pixels. Figure 4-3 (a) shows the cubic, one dimensional case, in which the value of $F(p')$ is found by fitting a cubic spline to four points around $F(p')$. This is extended to the two dimensional case shown in Figure 4-3 (b), where the interpolated pixel may be expressed in the form

$$F(p', q') = \sum_{m=-1}^{2} \sum_{n=-1}^{2} F(p+m, q+n) R_c[(m-a)] R_c[-(n-b) \qquad (4.2)$$

where a and b are the distance to the closest point as shown in Figure 4-3, and R_c is

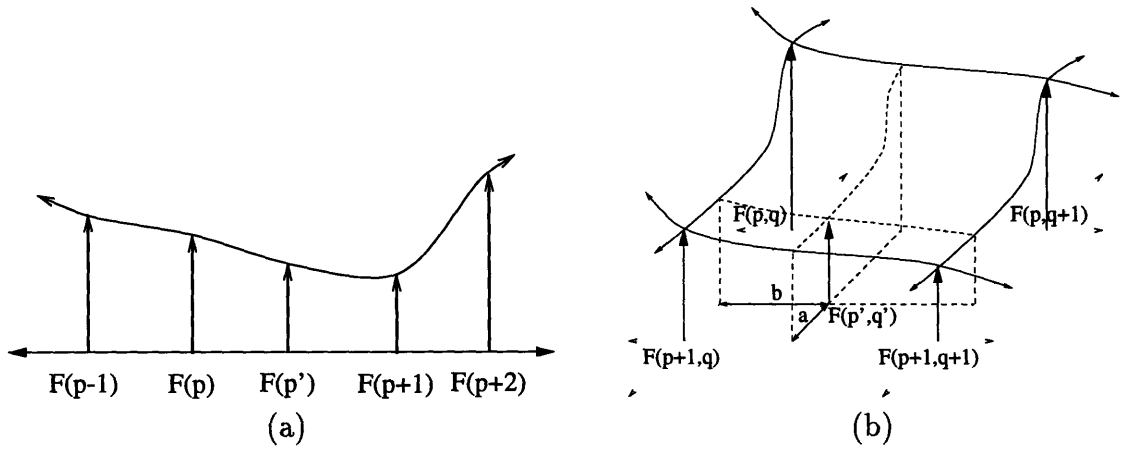


Figure 4-3: Diagrams of cubic interpolation. (a) 1d interpolation to determine the value of $F(p')$. (b) 2d interpolation to determine the value of $F(p', q')$

the bicubic interpolation function [18]:

$$R_c = \begin{cases} \frac{2}{3} + \frac{1}{2}|x|^3 - (x)^2 & 0 \leq |x| \leq 1 \\ \frac{1}{6}(2 - |x|)^3 & 1 \leq |x| \leq 2 \end{cases} \tag{4.3}$$

Bicubic interpolation does well at estimating the value of $F(p', q')$ if the pixels around it describe a cubic function. Since this is not always the case in real imagery, the value of $F(p', q')$ reflects some error.

4.2 Single Measure Threshold

Because of interpolation error, it is not sufficient to define all pixels with a MCM of 1 to belong to the moving region R_A, for some pixels that do belong to the region will have a lower value. Therefore, some non-unity threshold is needed to classify the pixels as belonging to the moving region or not.

Using a lower threshold is a plausible proposal, but as Figure 4-4(b), (c), and (d) show, one global threshold does not produce very good results. The system is attempting to track the crew shell in these figures, with black pixels being included in the analysis region. In Figure 4-4(b), the mask was thresholded at .5, so that only pixels with a MCM value greater than .5 were included in the mask. This image shows the analysis region includes regions of the crew shell, but because the threshold

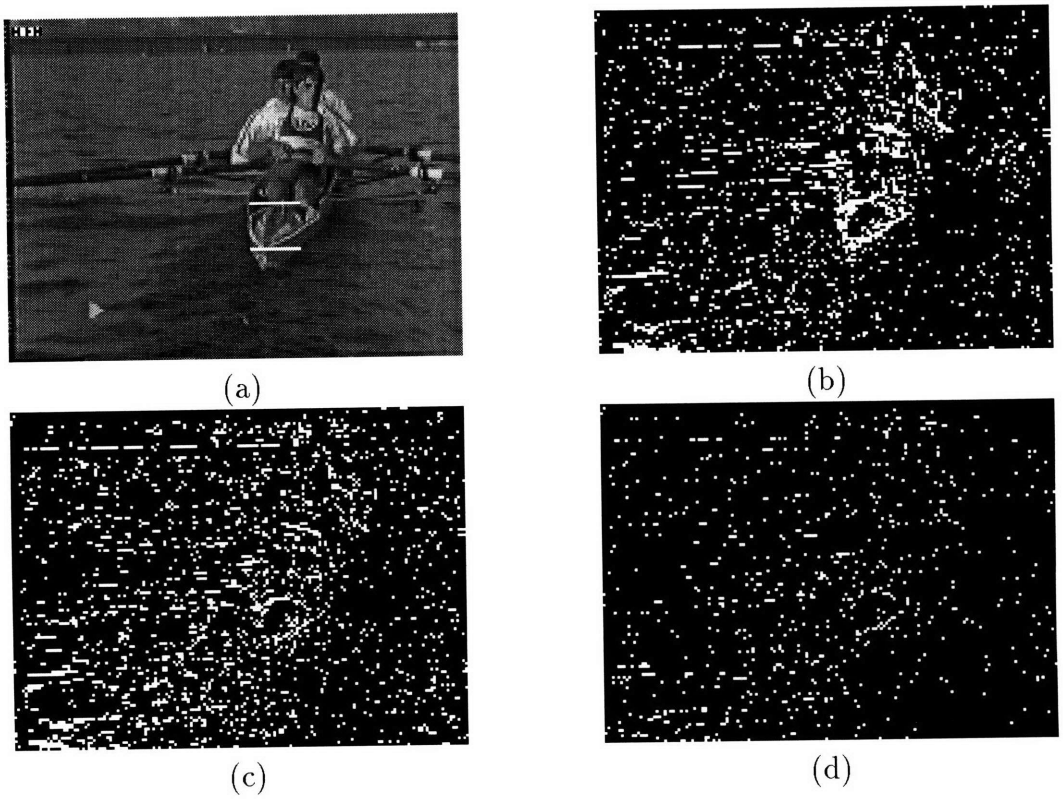


Figure 4-4: Examples of single threshold analysis masks. (a) An original frame from a crew sequence. (b) Mask thresholded at .5, (c) Mask thresholded at .7, (d) Mask thresholded at .9.

is low, it also includes many pixels from the water. Figure 4-4(c) and (d) reflect a threshold of .7 and .9 respectively. These images show a compromise between crew shell and water; the water is not represented in the mask, but the crew shell is not well represented either. A single threshold fails because interpolating a pixel in a constant region will result in less interpolation error than a pixel lying in a highly textured region. Consequently, our measure statistic will be inherently better in constant regions and worse in textured regions, independent of how well the pixels actually move according to the motion.

4.3 Multiple, Texture Dependent Thresholds

To account for interpolation error, a variable threshold is designed to be sensitive to not only the pixel's MCM, but also to the area around the pixel. If a pixel lies in a highly texture region, it's MCM is expected to be lower because I_{tw} of equation 4.1

will be comparatively small due to interpolation error. Therefore, if a pixel is found to be in a textured region, it is considered to belong to region R_A at a lower measure than a pixel in a constant region. Thus a variable threshold should account for the interpolation error.

Whereas previously a single threshold was searched for with an ad hoc approach, now thresholds that are texture dependent are found by through the use of artificial sequences of varying texture. Since motion and texture can be controlled in the artificial sequence, the relation of the MCM and texture, if any, can be discovered through the use of these sequences. Each sequence has two regions, R_A and R_B. Since both of these motions are controlled, any variation in MCM is due to interpolation error, and not estimation error. Therefore, by finding the distribution for these two regions, Bayesian Analysis is used to find the threshold that minimizes the probability of a misclassification.

4.3.1 Artificial Sequences

Artificial sequences are created, one of which is seen in Figure 4-5, by superimposing a texture pattern on top of a background. All texture patterns are found in the VisTex database[1]. With a user defined motion, a two frame sequence is created by warping the top texture pattern while holding the bottom texture fixed. Then the distribution of the Measure of Motion Certainty can be found by inverting the motion and warping the second frame is back toward the first.

4.3.2 Measuring Texture

Along with assigning an MCM to each pixel, another measure is also assigned that predicts the amount of interpolation error. Since a pixel's interpolation error is a function of the frequency of the region around it, a statistic of the region's Fourier Transform is ideal. However, calculating the Fourier Transform for a region around every pixel in the image is too computationally expensive for this application.

[1]http://www-white.media.mit.edu/vismod/imagery/VisionTexture/vistex.html.

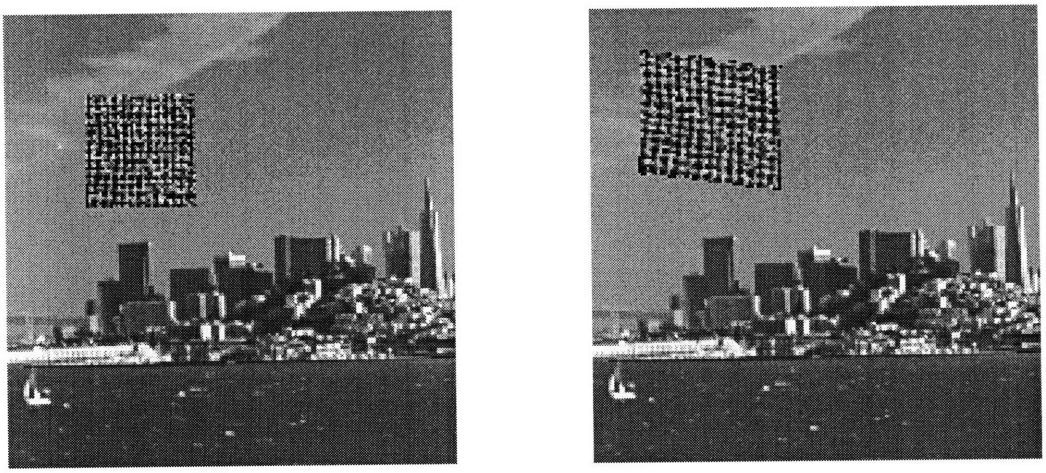


Figure 4-5: The two frames of an artificial sequence used to find the distribution of the Measure of Motion Certainty.

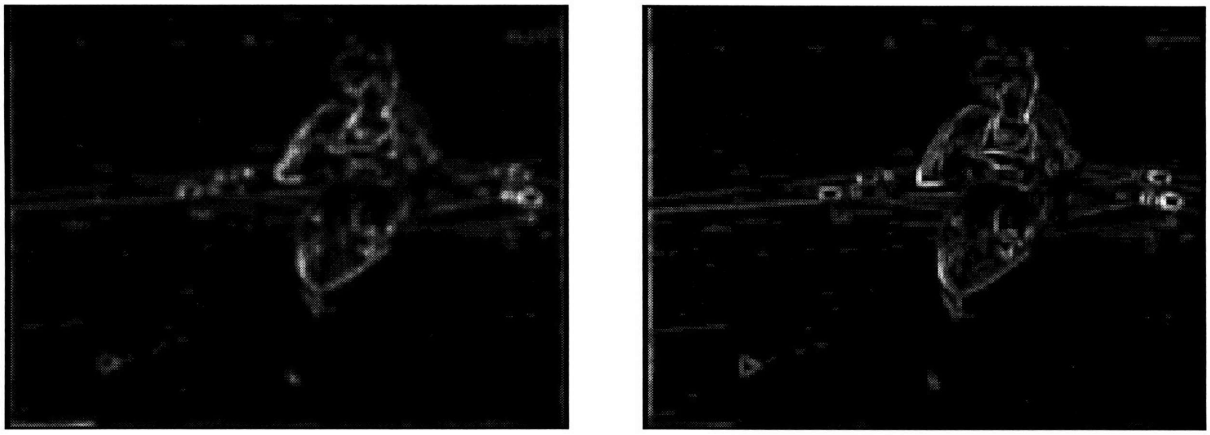
In order to make measuring texture more efficient, spatial statistics are calculated. Since the spatial autocorrelation function relates to the Fourier Transform (the Fourier transform of the spatial autocorrelation function is the square of the magnitude of the function's Fourier transform [18]), a statistic of this function is used. The autocorrelation can be defined by analogy. Imagine two transparencies printed with the same texture, and a light shining from underneath. If the two transparencies are translated about each other, they will allow different amounts of light to shine through. The autocorrelation function is defined as [18]:

$$A_I(m,n) = \sum\sum(I(j,k)I(j-m,k-n)) \tag{4.4}$$

where I is the image, m and n are pixel lags, and j and k are pixel coordinates. The amount of light shining through the transparencies is analogous to the value of A_I, and the translation size is analogous to m and n.

For texture regions, the autocorrelation will have a sharp drop off because pixel values will differ very quickly, restricting the amount of light shining through in the analogy. Likewise, a more constant region will have a shallow drop off because pixel values change much more slowly. The kurtosis, a fourth order statistic, measures the degree of this drop off, thus making it an ideal measure of texture [18].

Since the kurtosis of the autocorrelation function is not a very computationally



(a) (b)

Figure 4-6: Measure of texture (a) autocorrelation method. (b) variance method

efficient measure of texture, the spatial variance around each pixel is also tested. The variance measure is found to work better than the autocorrelation measure, especially for the simple purpose of predicting interpolation error.

Figure 4-6 (a) and (b) show the autocorrelation measure to be smoother than the variance measure. The kurtosis of the autocorrelation function changes more slowly than does the variance because the autocorrelation window is being translated around an area larger than the corresponding area of variance. Not only is variance a more efficient measure, it also better reflects the interpolation error of a pixel. The smooth nature of the autocorrelation function implies that pixels near a textured region will tend to have an artificially higher texture measure. Since bicubic interpolation only takes into account a 5x5 region around the center pixel and variance also considers the same region, the simpler statistic is a better measure of interpolation error.

4.3.3 Distribution of Measure

After calculating both a texture measure and the MCM for every pixel in the image, the distribution of $p(MCM|t, R)$ is found where t is the measure of texture, and R is the region type, either moving with the estimated motion or moving with another motion. Then, by using Bayesian analysis, given an arbitrary t, the threshold of M can be calculated to minimize the probability that the wrong R is chosen.

Modeling these distributions shows promising results. As Figure 4-7 shows, the

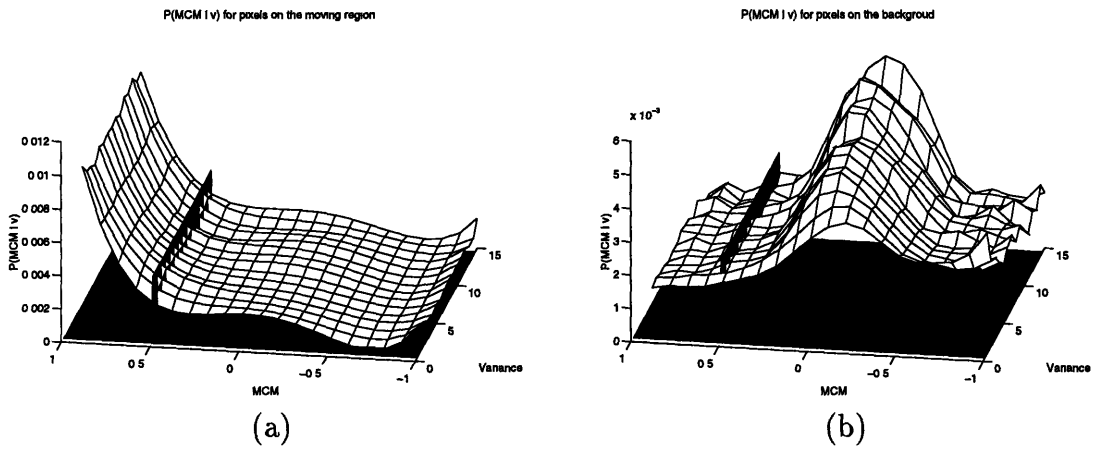


Figure 4-7: The two distribution of MCM. A plausible threshold for the MCM is shown. (a) Distribution for pixels in region R_A (b) Distribution for pixels in region R_B.

distribution of MCM in a region that is moving according to the estimated motion is biased toward +1. For regions moving with a different motion, the distribution is centered just off of 0.

However, there is not a great disparity between different types of texture. Compared to a very constant texture the distribution of the MCM in a steep gradient is biased slightly more toward 0 implying interpolation error does play a role in the calculation of the MCM. This is true for smooth textures. However, at higher textures, there is much less of a difference.

This can be explained because the I_t term of the MCM is sensitive to the amount of texture in a region, given a small motion. The value of I_t is a crude measure of texture; for small motions, it will tend to be larger in textures and smaller in constant regions. I_{tw} is also reflects texture, through interpolation error. Figure 4-8 shows for highly texture regions, those with large I_t, and higher I_{tw} is allowed to produce the same MCM. Thus the MCM compensates for different textures through I_t and I_{tw}. Consequently, the threshold that minimizes the probability of a misclassification only varies slightly between different textures. However, using these values as a guideline for new values is helpful because the system should be conservative when adding pixels that lie in a constant region to the analysis region because these pixels provide less useful information to the motion estimation. Conversely, the system should be more aggressive when adding a pixel that lies in a textured region to the analysis

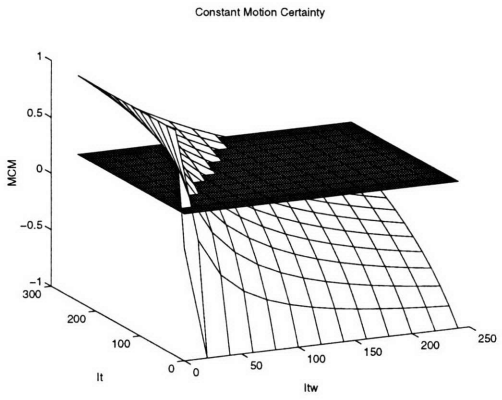


Figure 4-8: M vs I_t vs I_{tw} with constant M shown. Larger I_t and I_{tw} produce the same M as small I_t and small I_{tw}.

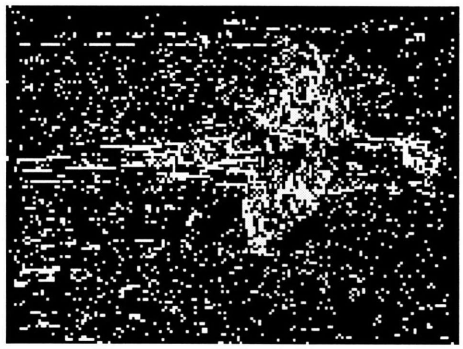


Figure 4-9: The analysis region created by the modified variable threshold.

region. With these heuristics, the Figure 4-9 is the resulting analysis region.

4.4 Postprocessing

Applying the modified variable threshold to the measure mask greatly reduces misclassifications in both constant and texture regions, but Figure 4-9 shows the resulting binary mask is not coherent and suffers from noise. The analysis region should be as coherent as possible for two reasons. First, any noise noise in the mask due to estimation error or signal quality should be eliminated. Second, coherent analysis regions conforms to our notion of the physical world. In image sequences, the set of pixels being tracked corresponds to a physical object rather than a set of incoherent pixels.

Two methods to cohere pixels and regions are tried. First, morphological op-

erations are used to reduce noise. This was shown to be slight too dramatic, so a modified morphological operator is applied to consolidate regions. This, combined with the variable threshold, results in a reliable analysis region that adapts to the dynamic imagery in an image sequence.

4.4.1 Morphology Operator

First, binary morphological operations are tried to remove pixels due to noise and consolidate disconnected patches that are seen in Figure 4-9. Morphological image processing is a type of processing that modifies the structure of objects within an image. It has its roots in spatial set algebra by Minkowski [17] and Haralick [11] provides a comprehensive overview.

Morphological operators can be explained, in a limited way, through the "Hit-and-Miss" transform [11]. Conceptually, a small (usually 3x3), odd sized binary structuring element is passed over the binary image. In terms of set theory, the structuring element is one set and the binary image the other. As the structuring element is scanned over the image, if the image matches the same pattern as the structuring element, the image pixel that corresponds to the center of the structuring element is set to some state. If the image does not match the element's pattern, the center pixel is set to the opposite state.

Two fundamental binary morphological operators are binary dilation, denoted by $\oplus$ and binary erosion, denoted by $\ominus$. A specialized case of binary dilation creates a black pixel when there is at least one pixel in the structuring element. This "grows" pixels in the image by a border of one pixel each iteration of the structuring element. Binary erosion is similar; a pixel is remove if there is at least one pixel in the structuring element. This "shrinks" patches by one pixel for each iteration.

Open and close operations are used to postprocess the binary analysis region. These operations are simply the application of dilations and erosions; the opening of an image I by structuring element K denoted by $I \circ K$ is defined by $I \circ K = (I \ominus K) \oplus K$. The closing of an image, denoted by $I \bullet K$ is $I \bullet K = (I \oplus K) \ominus K$. Since opening starts with the dilation operation, "opening an image with a disk structuring element

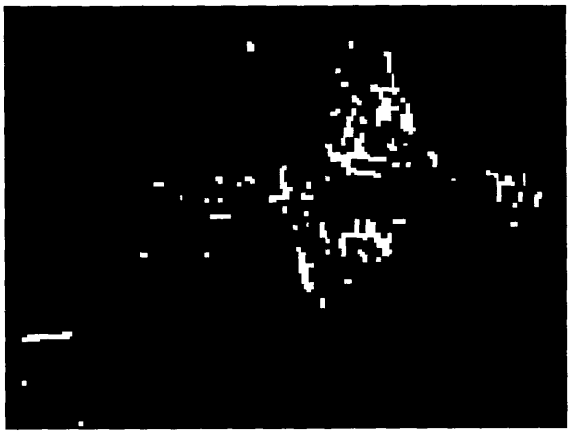


Figure 4-10: Analysis region after applying the open-close morphological operator.

smoothes the contour, breaks narrow isthmuses, and eliminates small islands and sharp peaks or capes". Likewise, "closing an image with a disk structuring element smoothes the contours; fuses narrow breaks and long, thin gulfs; eliminates small holes; and fills gaps on the contours" [11].

In this application, an opening operation followed by a closing operation was tested. The order is important, because the first operation to be performed is erosion, to reduce stray pixels. If a dilation was performed first, much of the noise would be included in the final analysis region. The result is seen in Figure 4-10. From Figure 4-10, it is apparent that almost all of the noise has disappeared from the mask, while many of the regions where the crew shell is located also disappear. This is explained by morphological operations' skill at preserving an object's spatial structure. This is an excellent feature for some applications, but in ours, the initial spatial structure (see Figure 4-9) are not coherent enough to withstand the initial erosion. Likewise, if dilation were performed first, noise would dominate the resulting image.

Consequently, a modified morphological operator, concerned less with preserving spatial structure and more with preserving spatial content, is used.

4.4.2 Modified Morphological Operator

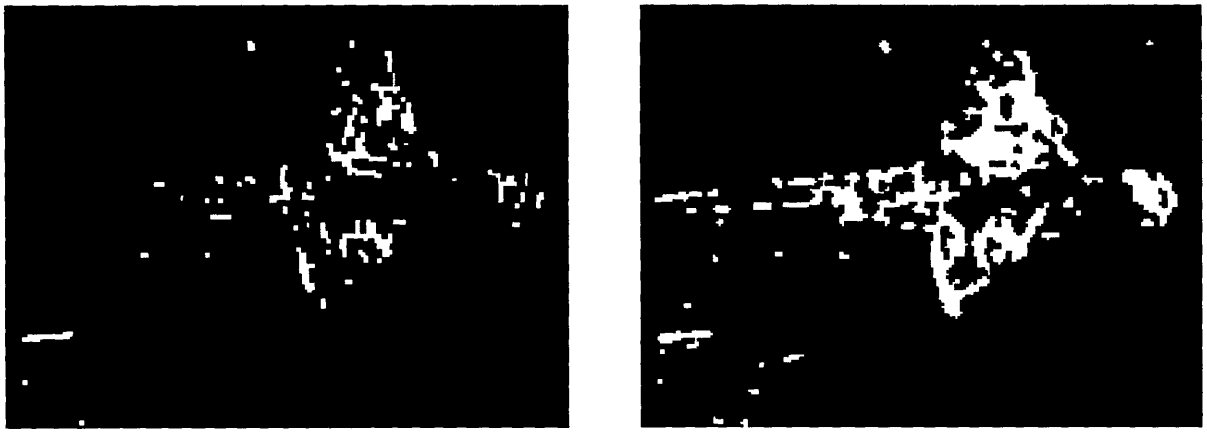
Our morphological operator passes a NxN window over the binary analysis region and counts the number of black and white pixels in the NxN window. Then, the pixel

in the binary analysis region corresponding to the center of the NxN window is set to the value of class with the greatest contribution, i.e., if there are more white pixels in the window, than the center should be turned white. The operator is modified to only turn the middle pixel black if B pixels are already black in the window, and only turn the middle pixel white if W pixels are already white, allowing more pixels to remain their initial state.

This morphological operator is tailored for different functions by altering B and W. For instance, $N = 3, B = 8, W = 8$, will remove and add pixels only if they are not near any other pixels. By lowering B and W, the operator can be used to clean different amounts of noise. N can also be increased to take a larger area into account. Conceptually, the window sees if a pixel lies in a region that is "mostly" R_A or R_B, "mostly" being determined by N, B, and W.

Through empirical tests, three iterations of this morphological operator are found to work well to cohere the analysis region. The first iteration, with parameters $N = 3, B = 6, W = 6$ is performed to reduce noise without disturbing the incoherent analysis region. The second, $N = 5, B = 10, W = 20$ is performed to add pixels to the incoherent region, and the last, $N = 5, B = 20, W = 10$ is performed to remove pixels that have accumulated from residual noise.

The three iterations do not attempt to segment the whole object from the image to use as the analysis region because constant regions do not provide any information to the motion estimation. It is also unnecessary to find the whole object in the image because the estimation is already overconstrained with more than six points. Figure 4-11 shows the difference between results of the open-close operator with the modified morphological operator. The images show the first binary mask generated after seeding the system with an analysis region seen in Figure 4-4(a). The modified morphological operator keeps more of the pixels belonging to the crew shell because no dilation operation is used. The modified operator is not a panacea, however. The lower left side of the analysis mask shows some artifacts still remaining, for the same reason why more of the crew shell remains. Because no dilation is taken, some larger patches of noise still remain. Also note that these particular patches remain because



(a) (b)

Figure 4-11: Comparison between morphological operators. (a) Analysis region after applying the open-close operator. (b) Analysis region after applying the modified morphological operator.

of the lower threshold in that area; the water breaking happened to provide enough texture to warrant a lower threshold (see Figure 4-6).

Chapter 5

Results and Future Work

The dynamic analysis window has different effects on different types of sequences. For some sequences, the dynamic window has no effect; using the standard motion tracking of updating the analysis region achieves the same results. For other sequences, the dynamic window achieves estimation accuracy that can not be produced by the standard window. This section will analyze the performance of the dynamic window in comparison to the static and motion tracked window with examples from four sequences, the bus sequence, the noisy van sequence, the bulletin board sequence, and the crew sequence.

5.1 Single vs. Variable Threshold

The use of a variable threshold in the analysis works well to reduce noisy pixels in the analysis mask. Figure 5-1(c) and (d) show the difference after morphological operations between the two thresholding schemes. Note that the disparity between Figure 5-1(c) and (d) is smaller than the disparity between Figures 5-1(a) and (b). This is due to the morphological operations used; to some extent, they have more of an effect on the resulting mask than does the changing value of the threshold.

To reiterate Section 4.3.3, the thresholds are not chosen with just Bayesian Analysis. Thresholds resulting from the Bayesian Analysis are used as a guide for more aggressive and more conservative thresholds. The reason given in Section 4.3.3 for

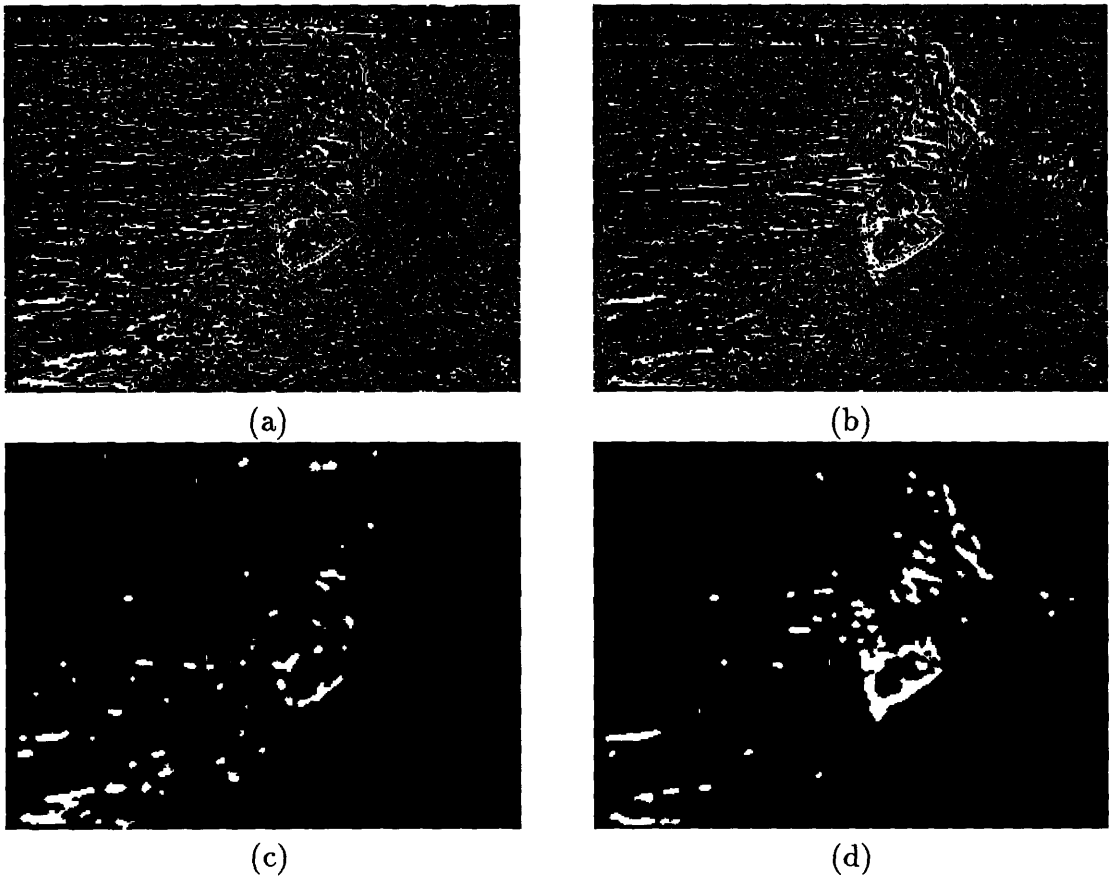


Figure 5-1: Single vs. Variable Threshold, (a) Analysis mask after thresholding with one threshold. (b) Mask after thresholding with a variable threshold. (c) Single threshold mask after morphological operations. (d) Variable threshold mask after morphological operations.

changing the threshold values was that higher textured regions give more useful information to the motion estimation, and constant regions give less.

It can be argued that most of the imagery used is comprised of large constant regions that should not be in the analysis mask. For the bus sequence in Figure 1-1, for instance, the background, house, and most of the bus is a constant texture. This means that for these sequences, not only do constant regions carry less information, they also have a lower *a priori* probability of being included in the analysis mask. If constant regions did have a high probability of being in the mask, if the tracked object was mostly constant, for instance, then it would not be as important to exclude these pixels from the mask thus having a more conservative threshold for constant regions is not necessary. However, these regions still do not provide any more useful information making a conservative threshold not harmful.

5.2 Image Processing Results

As expected, both the conventional method of updating the analysis region and the new dynamic analysis region method work equally well for sequences without the problems discussed in Section 2.4. For instance, consider an analysis region around the license plate of Figure 5-2(a), the first frame of a moving van sequence with heavy Gaussian noise added. The license plate stays within the camera's field of view, it has sufficient texture to be tracked, and no occlusions enter the sequence. To produce the noise reduced image of Figure 5-2(b), the motion of the van is tracked with the conventional tracked analysis region, and a median image of the registered van sequence is found. It is promising for the dynamic analysis region to achieve the same results because the amount of noise might have made it difficult for the modified morphological operations to find the van features. However, as noted previously, it does not take many pixels to produce an accurate estimation. Figure 5-2(c) and Figure 5-2(d) shows the analysis masked the dynamic analysis region produced and the corresponding noise reduced image. Notice in Figure 5-2(c), features other than the license plate are segmented from the image, implying a more accurate motion

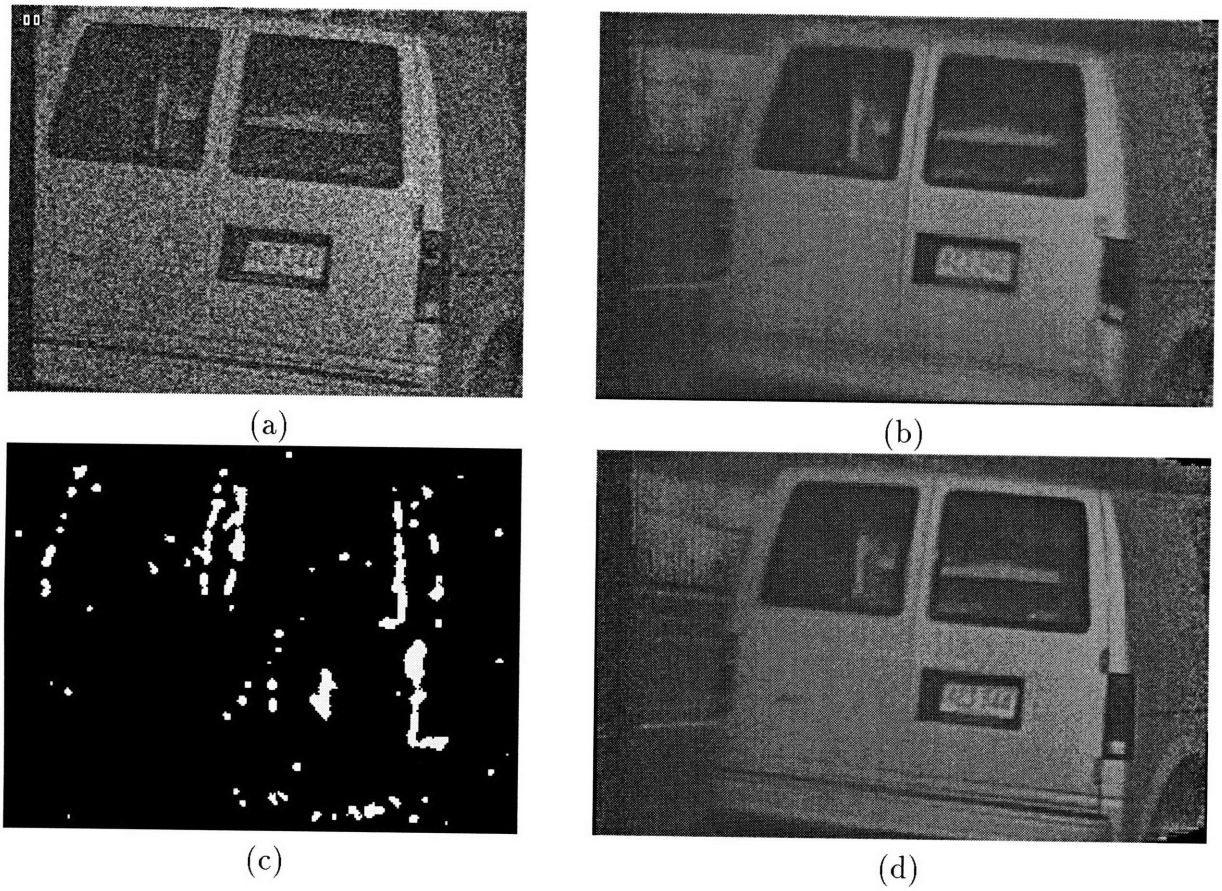
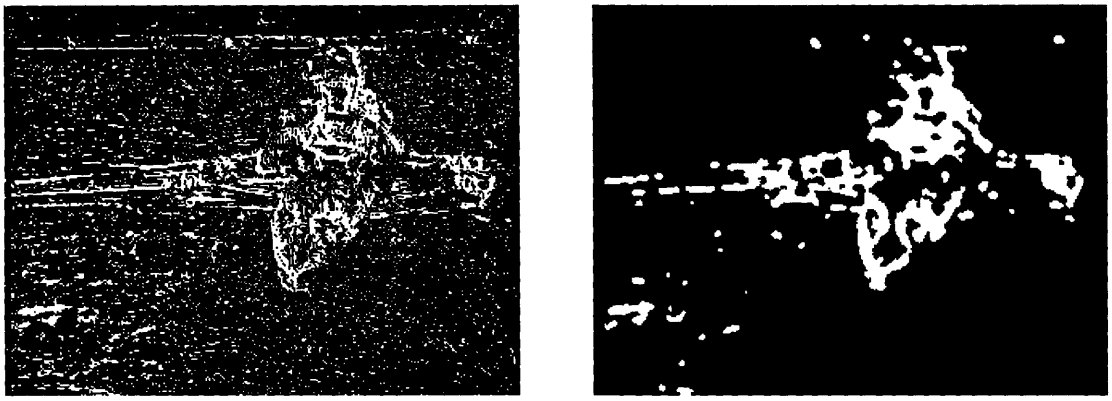


Figure 5-2: Results from estimating motion from the van sequence. (a) First frame of the 15 frame van sequence with Gaussian noise added. (b) Temporal median of the aligned sequence, with motion estimated from the license plate area using the conventional motion tracked method. (c) An example of the dynamic analysis region, seeded by the license plate region. Notice features are still segmented despite the heavy noise. (d) Temporal median of the aligned sequence using the dynamic analysis region method.



(a) (b)

Figure 5-3: Convergence of better analysis regions. (a) second analysis mask, before morphological operations. (b) analysis mask after morphological operations.

estimate if the license plate is occluded or moves out of the camera's field of view.

The dynamic analysis region also has the ability to converge on a better analysis region, given that a user has seeded the process with a somewhat reasonable starting analysis region. For instance, Figure 5-3 shows the analysis region before and after morphological operations for the third frame of the analysis, after using the analysis region to estimate motion as seen in Figure 5-1(d). Notice that many more pixels in the crew shell region receive a higher MCM. This implies that the analysis region shown in Figure 5-1(d) produces an estimate that is more accurate than the estimate from which it was produced. This implies that if the user's analysis region does not have the best trackable features, the dynamic region will converge on them in a few frames (about three frames).

The convergence of an analysis region can also be interpreted as finding a dominant motion. If the entire image is used as an initial analysis region, the initial motion estimate will reflect all motion in the sequence, reflecting the dominant motion the most. Consequently, the pixels in the analysis mask will tend to come from the region of the dominant motion. Since only these pixels are used for subsequent analysis, the motion estimate will eventually converge to estimating only the dominant motion.

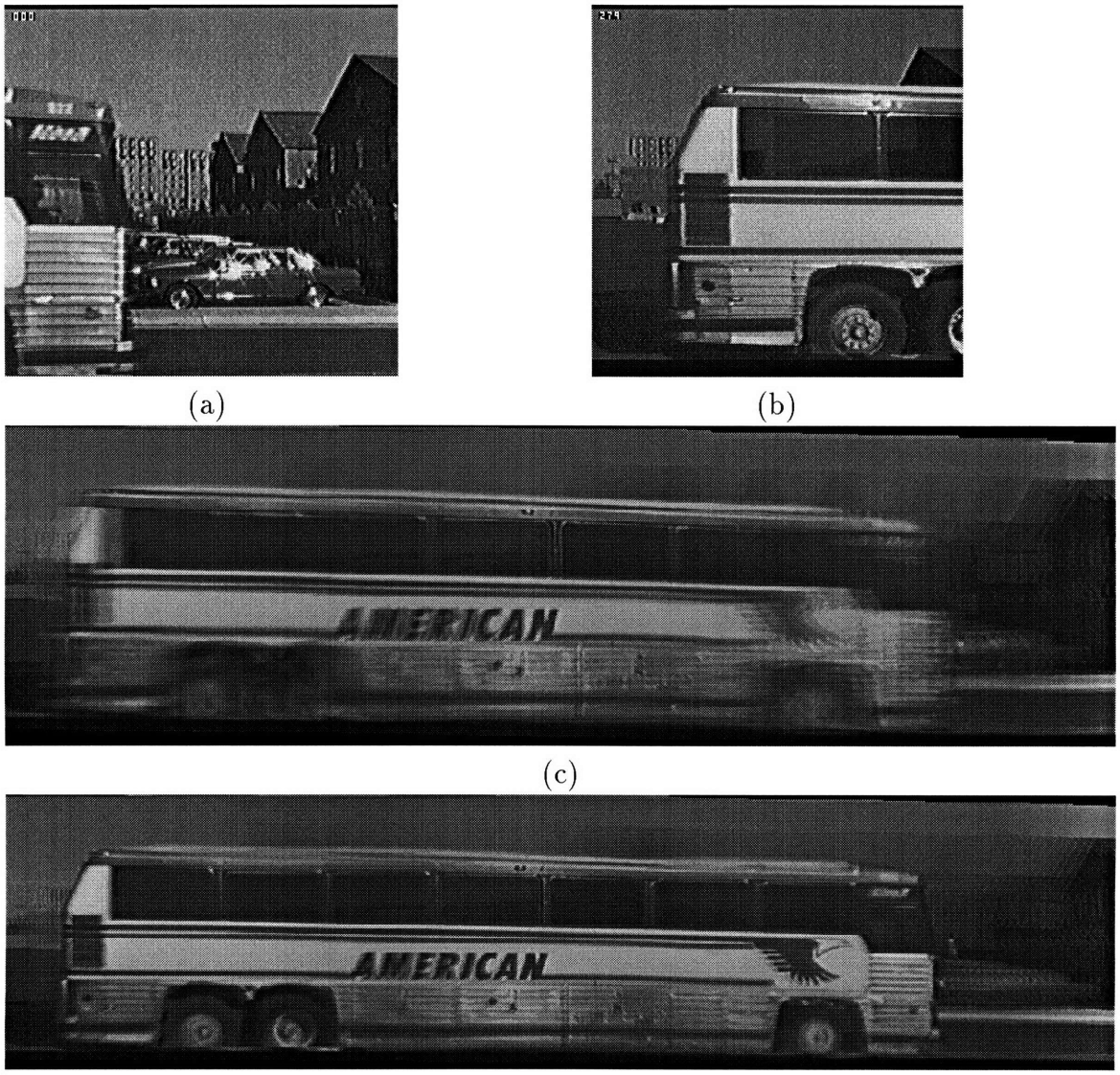
Figure 5-4 compares the results of the conventional region tracking method with the new dynamic region. Figures 5-4(a) and 5-4(b) show the first frame of the bus sequence and the last frame of the 300 frame bus sequence. Notice that the bus

only has a small region common to both frames. Since a motion tracking analysis region will move off the sequence, a static region must be used. Figure 5-4(c) shows the mosaiced bus after registering all 270 frames with the estimated motion from the static region. Only translational motion was considered in this sequence. The front and rear of the bus is blurred because of the analysis region's lack of image information. The features of the bus in its front and rear are mostly horizontal. Thus, these features are essentially useless. If a larger region could be used, some vertical features might be found.

Figure 5-4(d) shows the registered bus when using the estimated motion with the dynamic analysis region. The resulting motion parameters are more accurate because these vertical features are found after the first frame. Notice the background undergoing motion blur because it belongs to a different motion. This is also the reason why Figure 5-4(c) is blurry, it to undergoes a different motion than that which is measured.

The results from image processing in the last examples do not show a big difference between the conventional region tracking method and the new dynamic region method, because the problems listed in section 2.4 are not present. Using the dynamic region with the bus sequence shows some improvement if the entire sequence is used. Figure 5-5 shows an example of a sequence which requires the use of the dynamic analysis region.

This sequence cannot use the conventional method of motion tracking because the multiple motions would occlude any static analysis region. If the region were to track some feature on the initial image, that analysis region would disappear from the camera's field of view. To segment the background from the occlusions, it's motion is estimated, with an arbitrary initial analysis region. As stated before, the dynamic analysis region is able to converge on a better analysis region, making the initial region less important. A translation only motion model is used, as in Equation 2.3. Figure 5-5(c) and (d) show two examples of the dynamic analysis region. Notice that only regions that move with the seeded motion are within the analysis mask, and the analysis mask contains minimal constant regions.



(a) (b)

(c)

(d)

Figure 5-4: Static vs. dynamic analysis region. (a) First frame of the 300 frame bus sequence. (b) Last frame. (c) 300 registered frames from static region. (d) 300 registered frames using dynamic region.

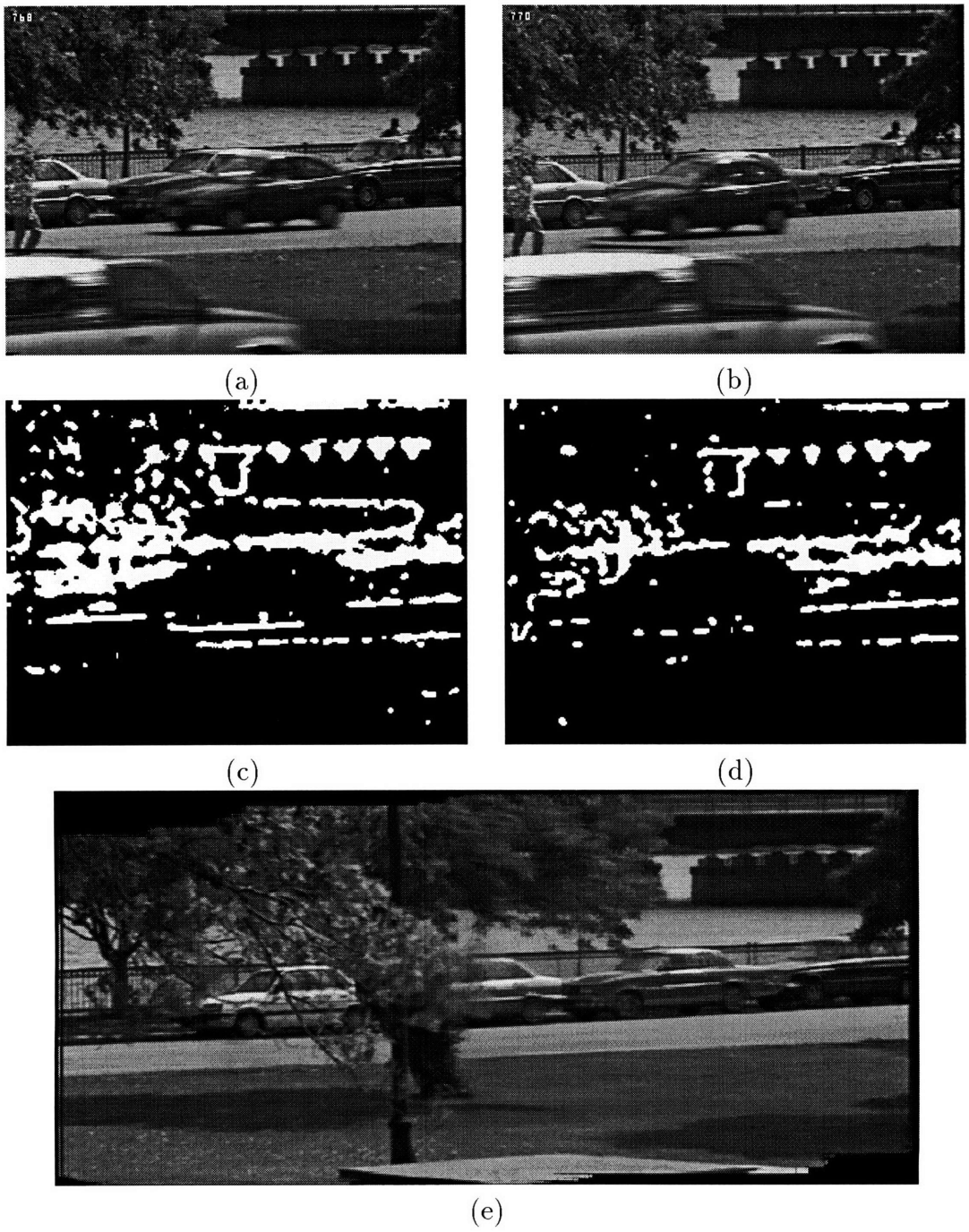


Figure 5-5: Images from the Memorial Drive sequence. (a) and (b) show frames 118 and 120 of the 185 frame sequence. Notice the multiple motions in the sequence, cars, people, leaves, and camera. (c) and (d) show the corresponding analysis regions that tracks the camera motion. (e) shows the temporal median of the registered sequence. Notice occlusions disappear, and the field of view is increased. Black regions in the border are due to lack of information, i.e., these points were never seen by the camera.

After the sequence is registered to a common frame, a temporal median is taken. The result is shown in Figure 5-5(e). Since occlusions appear for a very short time, compared to the background, they are removed by the median operator. There are some imperfections, such as the bridge abutment and two of the cars. This could be due to some rotational component in the sequence. Since the movie was taken by hand (as opposed to a tripod), the strict translation constraint may not be followed. Leaves are also blurred, but this is due to wind.

The original memorial drive movie, the registered sequence and the temporal median sequence can be found at this thesis's url[1]. An example of sequence stabilization, which is difficult to display statically is also shown, using the crew sequence as an example. The original, registered, and integrated bus sequence can also be seen at the web site.

5.3 Conclusion

This thesis presents an iterative, multi-resolution, linear gradient least squares method of robust motion estimation, which is presented in Chapter 2. It is different than other motion estimation techniques because it does not need to compute a dominant motion before other motions are estimated. This is achieved by obtaining an initial object definition, or analysis region, from the user. Then, the motion estimation method is seeded with this definition so that subsequent analysis regions can be derived from the estimated motion parameters.

Chapter 4 describes how the analysis regions are derived. Conceptually, each pixel is classified into two classes through a Motion Certainty Measure, a class that is one the tracked object and a class that is not. Then, the set of pixels that are believed to be on the tracked object are used as the analysis region. By modeling the interpolation error of pixels, the threshold for the Motion Certainty Measure is varied for different textures. Modified morphological operations join the thresholded pixels into a coherent analysis mask.

[1] http://www-white.media.mit.edu/~yao/dynamo.html

The dynamic analysis region method allows an accurate motion estimation even in the presence of an occlusion, since the analysis region will not include the occlusion. This also allows an analysis region to converge to a one with more features, in the case where the initial object definition carries little spatial information. The dynamic analysis region is also another way to measure the dominant motion, if it exists. Experiments on several real image sequences validate the performance of the dynamic analysis region.

5.3.1 Suggestions for Future Work

The work presented in this thesis can be continued in several ways, namely through the use of a priori probabilities to decrease the dependence on morphological operations to reduce noise and cohere region. Also, the magnitude of motion and size of region texture need to be considered for an improved initial analysis mask. Finally, it is proposed that this thesis could be extended to a strict motion segmentation application.

A priori probabilities: One deficiency of the dynamic analysis region is the assumption that the moving and static regions have equal priors. This assumption is reasonable for an arbitrary image; there is no reason to believe a pixel has a greater probability to come from one region and not the other. However, after processing the first image pair, the system has some idea of where the moving object is spatially located. This information should be used to further weight the variable threshold; the pixel threshold should be a function of the type of texture around a pixel and also the distance from a pixel to the proposed moving object. This could be implemented using the modified morphological operator, applied during the thresholding process. The operator would have to be modified again to handle more than two values (it is currently binary), and the result would depend on small frame to frame motion.

Another possible method of incorporating priors into the system would be to use the percentage of the image occupied by the moving region and static region. If

it is known the moving regions occupies a very small portion of the entire image, say 10%, the thresholds for the Motion Certainty Measure should reflect the low probability that a pixel does belong to the moving region. By incorporating priors into the threshold boundary, less importance would be placed on the morphological operations, therefore fewer pixels in the analysis mask would be misclassified.

Other motion and texture considerations: If the dynamic analysis region is to be developed further, it should take into account the amount of motion and what textures to which this motion will map a pixel. The MCM is dependent on two measurements, I_t and I_{tw}. Interpolation error plays a role in the value of I_{tw} and is fairly motion invariant; for any motion, the same amount of interpolation error will be present. The MCM is also dependent on I_t, whose value is dependent on the amount of motion and the texture in which the pixel resides. If the region is textured and the motion is small, I_t is expected to be large, compared to I_{tw}. Likewise, if the region is not very texture, I_t is expected to be small. However, suppose the motion is large compared to the area of the texture in which the pixel resides. This means a pixel in I_1 and the corresponding pixel in I_2 could come from different textures spreading the variance of the MCM. For instance, if a pixel in I_1 lies in a constant texture on the moving region, the MCM is expected to be large because of little interpolation error in I_{tw} and more importantly, I_t is expected to be very small. However, if the motion is large, I_t could very well be large forcing the MCM to tend toward 1, as Figure 4-8 shows. It would be interesting to take both the texture of I_1 and I_2 into consideration before assigning a threshold to the MCM.

Modeling the motion estimation accuracy based on the texture of the analysis region could be another extension to the dynamic region. As stated previously, constant regions provide little information to the motion estimation, whereas textured regions are very useful. It is possible that the estimation accuracy is a function of the amount of texture present in the analysis region. A similar

experiment could be carried out to model the distribution of estimation accuracy given different analysis region textures, and some measure of confidence could be assigned to the estimation. Given a very weak confidence, the initial user supplied analysis region could be given more weight compared to the derived analysis region until the process converged to an analysis region with strong features.

Motion segmentation: At this time, this work is only loosely connected to motion segmentation. It does not attempt to segment the entire object from the rest of the image, it only attempts to segment the trackable features from the object. By using the information from the dynamic analysis region, the method could be extended to extract the entire object from the image sequence. A static segmentation procedure, as in [1] might have to be performed but combined with the motion segmentation, a robust object detection system could be developed. Motion information often gives structural information that a texture, color, or gradient segmentation cannot discern. For instance, consider a moving photograph. The photograph may have many different subregions all moving with the same motion. A static segmentation method will treat these subregions separately whereas the motion segmentation will treat them as one.

Thus, with a simple segmentation procedure, the applications discussed in the Introduction can be performed without background objects undergoing motion blur. This is done by segmenting the tracked object and only processing within it's boundaries. Also, segmentation in this application can provide a cheap method of compression by running the motion estimation for each object in the sequence. Then, these objects are segmented and transmitted as a still image along with its motion parameters. Then, all stills can be composited to recreate the initial sequence [20].

Bibliography

[1] S. Ayer, P. Schroeter, and J. Bigun, "Segmentation of moving objects by robust motion parameter estimation over multiple frames.", *Third European Conference on Computer Vision,* May 1994, pages 316-327.

[2] J.L. Barron, D.J. Fleet, and S.S. Beauchemin, "Performance of Optical Flow Techniques," *International Journal of Computer Vision,* 1994, pages 43-77.

[3] Bender, W., Teodosio, L., "Salient Video Stills: Content and Context Preserved," *ACM Multimedia,* August, 1993.

[4] M. Ben-Erze, S. Peleg, and B. Rousso, "Motion Segmentation Using Convergence Propoerties," *ARPA Image Understanding Workshop,* November 1994.

[5] J. Bergen, P. Burt, R. Hingorani, S. Peleg., "Multiple Component Image Motion: Motion Estimation," Draft, *David Sarnoff Research Center,* 1990.

[6] J. Bergen, P. Anandan, K. Hana, and R. Hingorini al., "Hierarchial model-based motion estimation,"*Proc. Second Eurpoean Conf. on Computer Vision,* December 1990, pages 237-252.

[7] J. Bergen and P. Burt, "Computing two motions from three frames," *Proc. 3rd International Conference on Computer Vision,* Osaka, Japan, 1990, pp. 27-32.

[8] M.J. Black, "Combining intensity and motion for incremental segmentation and trcking over long image sequences," *Second European Conference on Computer Vision,* May 1992, pages 485-493.

[9] Connors, R. W., and Harlow, C. A., "A Theoretical Comparison of Texture Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-2, no.3, December 1980, pages 204-222.

[10] E. Francois and P. Bouthemy, "Multiframe-based identification of mobile components of a scene with a moving camera," *Proc. CVPR,* June 1991, pages 166-172.

[11] R. M. Haralick and L. G. Shapiro, "Computer and Robot Vision, Volume 1," *Addison-Wesley Publishing Company,* 1992, pages 157-185.

[12] B.K.P. Horn, "Robot Vision," *MIT Press*, 1986, pages 278-293.

[13] M. Irani, B. Rousso, and S. Peleg, "Detecting and tracking multiple moving objects using temporal integration," *Second European Conference on Computer Vision,* May 1992, pages 282-287.

[14] M. Irani and S. Peleg, "Image Sequence Enhancement Using Multiple Motions Analysis," *Proc. IEEE Conf. Comput. Vision Patt. Recog.,* Champaign, IL, June 1992, pages 216-221.

[15] M. Levin, "Vision in Man and Machine," *McGraw-Hill, Inc,* 1985, pages 448-451.

[16] F. Meyer and P. Bouthemy, "Region-Based Tracking in an Image Sequence," *Second European Conference on Computer Vision,* May 1992, pages 467-484.

[17] H. Minkowski, "Volumen und Oberflache," *Mat. Ann., Volume 57* 1903, pages 447-459.

[18] W. Pratt "Digital Image Processing." *John Wiley & Sons, Inc.,* 1991, pages 112-116,578.

[19] P. Schroeter and J. Bigun, "Images segmentation by multidimensional clustering and boundary refinement with oriented filters," *Gretsi Fourteenth symposium,*" Sept. 1993, pages 663-666.

[20] J.Y.A. Wang and E.H. Adelson, "Representing Moving Images with Layers," *IEEE Transactions on Image Processing,* September 1994, pages 625-638.

[21] P. Yao, "A Dynamic Window for Motion Analysis," S.B. Thesis, MIT, 1995.