

Terrain Sensing and Estimation  
for Dynamic Outdoor Mobile Robots

by

Christopher Charles Ward

B.S., Mechanical Engineering  
University of California, Berkeley, 2005

Submitted to the Department of Mechanical Engineering  
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Mechanical Engineering

at the

Massachusetts Institute of Technology


June 2007

© 2007 Massachusetts Institute of Technology  
All rights Reserved

Signature of Author . . . . .

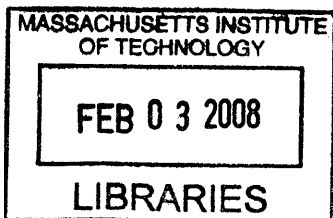
Department of Mechanical Engineering  
May 17, 2007

Certified by . . . . .

  
Dr. Karl Iagnemma  
Principal Research Scientist  
Thesis Supervisor

Accepted by . . . . .

Lallit Anand  
Chairman Committee on Graduate Students



BARKER

# Terrain Sensing and Estimation for Dynamic Outdoor Mobile Robots

by

Christopher Charles Ward

Submitted to the Department of Mechanical Engineering  
On May 17, 2007 in Partial Fulfillment of the  
Requirements for the Degree of Master of Science in  
Mechanical Engineering

## ABSTRACT

In many applications, mobile robots are required to travel on outdoor terrain at high speed. Compared to traditional low-speed, laboratory-based robots, outdoor scenarios pose increased perception and mobility challenges which must be considered to achieve high performance. Additionally, high-speed driving produces dynamic robot-terrain interactions which are normally negligible in low speed driving. This thesis presents algorithms for estimating wheel slip and detecting robot immobilization on outdoor terrain, and for estimating traversed terrain profile and classifying terrain type. Both sets of algorithms utilize common onboard sensors.

Two methods are presented for robot immobilization detection. The first method utilizes a dynamic vehicle model to estimate robot velocity and explicitly estimate longitudinal wheel slip. The vehicle model utilizes a novel simplified tire traction/braking force model in addition to estimating external resistive disturbance forces acting on the robot. The dynamic model is combined with sensor measurements in an extended Kalman filter framework. A preliminary algorithm for adapting the tire model parameters is presented. The second, model-free method takes a signal recognition-based approach to analyze inertial measurements to detect robot immobilization. Both approaches are experimentally validated on a robotic platform traveling on a variety of outdoor terrains. Two detector fusion techniques are proposed and experimentally validated which combine multiple detectors to increase detection speed and accuracy.

An algorithm is presented to classify outdoor terrain for high-speed mobile robots using a suspension mounted accelerometer. The algorithm utilizes a dynamic vehicle model to estimate the terrain profile and classifies the terrain based on spatial frequency components of the estimated profile. The classification algorithm is validated using experimental results collected with a commercial automobile driving in real-world conditions.

Thesis Supervisor: Karl Iagnemma  
Title: Principal Research Scientist

## **ACKNOWLEDGEMENTS**

I would like to thank Karl Iagnemma and Professor Steven Dubowsky for their guidance and support in this research. Karl's advice on both the technical content and writing of this thesis has been invaluable. Professor Dubowsky has helped me ask important questions about my work and has inspired me to continue moving forward.

I would also like to thank my wife, Miriam, for her loving support and for keeping life fun. Thanks Mom and Dad for always having the right priorities and for your love and support. You have set great examples for me. I couldn't have made it here without the support of my family and friends.

I'd also like to thank all the members of the Field and Space Robotics Lab for their help and for always being in the mood for tea hour. I'd especially like to thank Steve Peters for his help with experiments and Chris Brooks for his help with the terrain classification work. Finally, I'd like to thank Andrew Lookingbill at Stanford University for his help in collecting experimental data on the LAGR robot.

This work was supported by the DARPA Learning Applied to Ground Robots program and Ford Motor Company.

# **TABLE OF CONTENTS**

Abstract .....	2
Acknowledgements .....	3
Table of Contents .....	4
List of Figures .....	6
List of Tables .....	8
List of Symbols .....	9
Chapter 1: Introduction .....	14
1.1 Problem Motivation .....	14
1.2 Outline of this Thesis .....	16
Chapter 2: Wheel Slip Detection For Mobile Robots on Outdoor Terrain.....	18
2.1 Introduction.....	18
2.1.1 Problem Description.....	18
2.1.2 Experimental Platform .....	22
2.2 Dynamic Model-Based Wheel Slip Detection .....	23
2.2.1 Introduction .....	23
2.2.2 Dynamic Models .....	24
2.2.3 Slip Detector Algorithm .....	34
2.2.4 Experimental Results.....	45
2.2.5 Conclusions .....	52
2.3 Classification-Based Wheel Slip Detection .....	53
2.3.1 Introduction .....	53
2.3.2 Classification Algorithm Overview .....	54
2.3.3 Feature Vector Selection .....	56
2.3.4 Experimental Results.....	58
2.3.5 Conclusions .....	61
2.4 Detector Fusion.....	61
2.4.1 Introduction .....	61
2.4.2 Fusion Techniques.....	62
2.4.3 Experimental Results.....	63
2.4.4 Conclusions .....	66

2.5	An Adaptive Tire Model.....	67
2.5.1	Introduction.....	67
2.5.2	Dynamic Model.....	67
2.5.3	Adaptation Algorithm.....	68
2.5.4	Results.....	71
2.5.5	Conclusions.....	76
2.6	Summary and Conclusions.....	76
Chapter 3: Speed Independent Vibration-Based Terrain Classification.....		78
3.1	Introduction.....	78
3.2	Algorithm Overview.....	79
3.2.1	Description of Existing Terrain Classification Algorithm.....	79
3.2.2	Enhanced Algorithm for Dynamic Vehicles.....	82
3.3	Waveform Representation.....	84
3.3.1	Profile Estimation.....	84
3.3.2	Impulse Detection and Removal.....	95
3.3.3	Spatial Frequency Components.....	99
3.4	Classification Algorithm.....	101
3.4.1	<i>A Priori</i> Training.....	101
3.4.2	Online Classification.....	104
3.5	Experimental Results.....	105
3.5.1	Experimental Platform.....	105
3.5.2	Results.....	107
3.6	Summary and Conclusions.....	115
Chapter 4: Conclusions and Suggestions for Future Work.....		117
4.1	Contributions of this Thesis.....	117
4.2	Suggestions for Future Work.....	118
References.....		120

## LIST OF FIGURES

Figure 2.1. Inaccuracy of estimating robot velocity by integrating measured acceleration. ....	21
Figure 2.2. The LAGR robot.....	22
Figure 2.3. Rendering of the mobile robot considered in this work. ....	25
Figure 2.4. Robot kinematic parameters. ....	25
Figure 2.5. Diagram showing vehicle and tire forces. ....	25
Figure 2.6. Representative traction coefficient vs. relative velocity curve indicating effect of the 3 traction parameters. The traction coefficient is $F_{traction}/N$ .....	31
Figure 2.7. Comparison of traction force vs. wheel slip curves for the slip-based model and the proposed simplified model at various wheel speeds. ....	32
Figure 2.8. Comparison of robot displacement calculated using proposed relative velocity-based tire model, slip-based model, and kinematic zero slip.....	32
Figure 2.9. Representative rolling resistance coefficient vs. velocity curve indicating the effect of varying the 3 resistance parameters. The rolling resistance coefficient is $F_{roll resist}/N$ . ....	33
Figure 2.10. Example of robot driving normally. Note bottom plot is semi-log scale. ....	48
Figure 2.11. Example of robot becoming immobilized. ....	49
Figure 2.12. Same data as Figure 2.11 processed neglecting GPS. ....	52
Figure 2.13. Experimental results of classifier-based immobilization detection. Each incorrectly classified point is a 0.5 second instance. Wheel velocity is effective linear velocity at wheel radius. ....	60
Figure 2.14. ROC curve for immobilization detection experimental results. ....	61
Figure 2.15. Detector fusion results. Wheel velocity is effective linear velocity at wheel radius. ....	64
Figure 2.16. True velocity profile and velocity errors with and without adaptation. ....	72
Figure 2.17. Evolution of the tire parameters using adaptation. ....	74
Figure 2.18. Adapted velocity estimate error with and without sensor noise.....	76

Figure 3.1. Flowchart for vibration-based terrain classification algorithm as proposed by Brooks [9]. .....	81
Figure 3.2. Metal wheel with grousers used to test the terrain classification algorithm proposed by Brooks. Box shows location of the vibration sensor. ....	82
Figure 3.3. Flowchart for modified speed-independent vibration-based terrain classification algorithm presented in this thesis. ....	84
Figure 3.4. Diagram of quarter-car model. ....	85
Figure 3.5. Bode plot of (3.3), unsprung mass acceleration from terrain profile velocity input. ....	88
Figure 3.6. Bode plot of (3.6), profile estimate from wheel acceleration input. ....	91
Figure 3.7. Bode plot of (3.6) with no tire damping ( $B_t = 0$ ). Note constant gain at high frequency. ....	92
Figure 3.8. Estimated asphalt profile. Tire shown to scale. ....	95
Figure 3.9. Estimated rumble strip profile. Tire shown to scale. ....	95
Figure 3.10. Impulse detection example. ....	97
Figure 3.11. Illustration of impulse removal process. ....	99
Figure 3.12. The experimental vehicle. ....	106
Figure 3.13. Accelerometer mounting location on experimental vehicle suspension. ...	106
Figure 3.14. The experimental electronics package. ....	107
Figure 3.15. Photos of brick, gravel, and grass terrains. ....	109
Figure 3.16. Subset of terrain classification results using enhanced algorithm presented in this thesis. Percentage accuracies are for entire 10 km test set. ....	111
Figure 3.17. Terrain classification results vs. speed using enhanced algorithm. Dotted lines show average accuracy over all speeds. ....	112
Figure 3.18. Subset of terrain classification results using enhanced algorithm without removing impulses. ....	114
Figure 3.19. Subset of terrain classification results using the unmodified time-based algorithm. ....	115

## **LIST OF TABLES**

Table 2.1. Summary of weak constraints used. ....	44
Table 2.2. Summary of tire constants used. ....	46
Table 2.3. Sensitivity of false immobilized flags to changes in tire parameters. ....	51
Table 2.4. Comparison of accuracy of detection and fusion techniques on Section 2.3.4 test set.....	66
Table 2.5. Parameter vectors used for simulations. ....	72
Table 2.6. Velocity errors using adaptation compared with error using $\bar{\theta} = \bar{\theta}_{start}$ .....	75
Table 3.1. Terrain wavelengths corresponding to combinations of vehicle speed and road input frequency. ....	89
Table 3.2. Quarter car model parameters used for experimental vehicle. ....	106
Table 3.3. Number of 4 meter long training and testing instances by terrain type.....	109
Table 3.4. Classification results using enhanced algorithm with 65% threshold. ....	110
Table 3.5. Classification results using enhanced algorithm with SVM parameters chosen to provide maximum combined accuracy. Note overtraining for asphalt classification at expense of accuracy on other terrains.....	113
Table 3.6. Classification results using enhanced algorithm without removing impulses. .....	114
Table 3.7. Classification results using the unmodified time-based algorithm.....	115



## LIST OF SYMBOLS

$a$	Longitudinal distance, CM to front wheel axis
$a_{dist,bx}$	Body $x$ -axis acceleration due to disturbance force
$\hat{a}_{dist,bx,k-1}$	Estimate of body $x$ -axis acceleration due to disturbance force
$a_t$	Threshold for Fusion 2
$A_k$	Process Jacobian matrix, current time step
$A_t$	Traction model constant
$A_{roll}$	Rolling resistance model constant
$b$	Longitudinal distance, CM to rear wheel axis
$b_{ax}$	Accelerometer $x$ -axis walking bias
$\hat{b}_{ax,k-1}$	Estimate of accelerometer $x$ -axis walking bias
$b_{g\psi}$	Gyro $\psi$ -axis walking bias
$\hat{b}_{g\psi,k-1}$	Estimate of gyro $\psi$ -axis walking bias
$b_s$	Sensor walking bias
$B_s$	Suspension damping coefficient
$B_t$	Tire damping coefficient
$c$	Front tire spacing
$c$	Training label vector
$c_{ax}$	Accelerometer constant offset
$c_{gz}$	Gyro constant offset
$c_i$	Label for instance $i$
$c_s$	Sensor constant offset
$C_1$	Traction model constant
$C_2$	Traction model constant
$d$	Number of detectors
$D_i$	Output of detector $i$
$D_s$	Suspension compression distance
$\dot{D}_s$	Time derivative of suspension compression distance
$D_u$	Tire compression distance
$\dot{D}_u$	Time derivative of tire compression distance
$E(*)$	Expected value of *
$f_{controller}(u)$	Wheel acceleration due to speed controller
$f_{max}$	Maximum frequency component

$f_{min}$	Minimum frequency component
$\Delta f_{min}$	Fourier transform frequency resolution
$f_{Nyquist}$	Nyquist frequency
$f_s$	Sampling frequency
$f_{tire}$	Body $x$ -axis acceleration due to tire forces
$\hat{f}$	SVM decision value
$F_{dist}$	Disturbance force
$F_{i,lat}$	Lateral tire force, tire $i$
$F_{i,roll\ resist}$	Longitudinal rolling resistance force, tire $i$
$F_{i,tract}$	Longitudinal traction/braking force, tire $i$
$g$	Acceleration of gravity
$g_{tire}$	Body $z$ -axis angular acceleration due to tire forces
$\mathbf{H}_k$	Measurement Jacobian matrix
$\mathbf{H}_{k,\sigma}$	Measurement Jacobian matrix, for sensor $\sigma$
$i$	Wheel slip
$i$	Index, matrix row index
$i_s$	Wheel skid
$\mathbf{I}$	Identity matrix
$j$	Matrix column index
$J$	Robot $z$ -axis moment of inertia
$k$	Vector index
$k_{ie}$	Impulse end index
$k_{is}$	Impulse start index
$k_s$	Suspension spring constant
$k_t$	Tire spring constant
$K$	Adaptation gain
$\mathbf{K}_k$	Kalman gain
$l$	# of instances
$L$	Profile segment length
$m$	Total vehicle mass
$m_s$	Sprung mass
$m_{us}$	Unsprung mass
$n$	# of features
$n_{f,l}$	Front left tire “normal acceleration”
$n_{f,r}$	Front right tire “normal acceleration”
$n_r$	Rear tire “normal acceleration”
$N$	# of data points per instance

$N_i$	Normal force, tire $i$ . “ $N$ ” if tire not specified
$p$	EMA forgetting factor
$p$	# of frequency elements from Fourier transform
$\mathbf{P}_{a_z,i}$	Vector of power spectrum components of $z$ -axis acceleration for instance $i$
$P_{f_i,y_j}$	Log-scaled power at frequency $i$ of terrain profile segment $j$
$\mathbf{P}_{k-1}$	<i>A posteriori</i> state covariance matrix, previous time step
$\mathbf{P}_k^-$	<i>A priori</i> state covariance matrix
$\mathbf{Q}_{k-1}$	Process noise covariance
$R_1$	Rolling resistance model constant
$R_2$	Rolling resistance model constant
$\mathbf{R}_k$	Measurement noise covariance matrix
$\mathbf{R}_{k,\sigma}$	Measurement noise covariance, for sensor $\sigma$
$s$	Laplace variable
$\mathbf{S}$	Scale factor matrix
$\mathbf{S}_a$	Matrix from SVD
$\mathbf{S}_{signal}$	Signal subset of $\mathbf{S}_a$
$t$	Time
$T$	Time period
<i>threshold</i>	Labeling threshold
$\mathbf{u}_k$	Input vector
$\mathbf{U}_a$	Matrix from SVD
$\mathbf{U}_{signal}$	Signal subset of $\mathbf{U}_a$
$v$	Cross-validation depth
$v_{bx}$	Robot body $x$ -axis velocity
$\hat{v}_{bx,k-1}$	Estimate of robot body $x$ -axis velocity
$\dot{v}_{bx}$	Robot body $x$ -axis acceleration
$\bar{v}_{GPS}$	GPS average velocity over measurement interval
$\hat{\bar{v}}$	Average estimated velocity over time interval
$v_{fwd}$	Tire forward longitudinal velocity
$\mathbf{v}_k$	Measurement noise vector
$v_{rel}$	Velocity of tire relative to ground
$v_s$	Vertical velocity, sprung mass
$\dot{v}_s$	Vertical acceleration, sprung mass
$v_t$	Tire longitudinal velocity
$v_u$	Vertical velocity, unsprung mass
$\dot{v}_u$	Vertical acceleration, unsprung mass
$\mathbf{V}_a$	Matrix from SVD

$V_k$	Measurement noise Jacobian matrix
$V_{k,\sigma}$	Measurement noise Jacobian, for sensor $\sigma$
$w_i$	Zero mean process noise
$\tilde{w}_i$	Weighting factor for detector $i$
$w_{k-1}$	Process noise vector
$W$	Robot weight
$W_k$	Process noise Jacobian matrix, current time step
$W_{training}$	Matrix of principal component instances for training
$W_{test}$	Vector of principal components for one online instance
$x$	Robot longitudinal axis
$\mathbf{x}$	State vector
$\mathbf{x}(k)$	Displacement vector
$x(t)$	Displacement vs. time
$\mathbf{x}_i$	Feature vector, instance $i$
$\mathbf{x}_k$	True state vector, current time step
$x_{i,j}$	Feature $i$ , for instance $j$
$\hat{\mathbf{x}}_k$	<i>A posteriori</i> state vector estimate, current time step
$\hat{\mathbf{x}}_{k-1}$	<i>A posteriori</i> state vector estimate, previous time step
$\hat{\mathbf{x}}_k^-$	<i>A priori</i> state vector estimate, current time step
$\dot{x}_{GPS}$	Measured body $x$ -axis velocity
$\ddot{x}_{IMU}$	Measured body $x$ -axis acceleration
$\Delta x$	Constant spacing distance of profile estimate points
$\mathbf{X}$	Feature-instance matrix
$\tilde{\mathbf{X}}$	Scaled feature-instance matrix
$y$	Robot lateral axis
$\mathbf{y}(k)$	Profile estimate vector
$y(t)$	Profile estimate vs. time
$y(x)$	Profile estimate, constant spatial spacing of $x$
$\tilde{y}(x)$	Profile estimate, non-unique, temporally spaced $x$
$\bar{y}(x)$	Profile estimate, temporally spaced $x$
$\mathbf{y}'$	Truncated profile estimate vector, impulse removed
$\mathbf{y}''$	Truncated profile estimate vector, impulse & discontinuity removed
$y_{road}$	Road height under tire
$\dot{y}_{road}$	Time derivative of road height
$\mathbf{Y}$	Matrix of frequency components and instances for training
$\mathbf{Y}_{test}$	Vector of frequency components for one online instance
$\hat{\mathbf{Y}}$	Mean adjusted version of $\mathbf{Y}$

$z$	Robot vertical axis
$\mathbf{z}$	Measurement vector
$\mathbf{z}_k$	Measurement vector, current time step
$\theta$	Robot body roll angle
$\dot{\boldsymbol{\theta}}_{i,N}$	Roll rate vector, instance $i$ with $N$ elements
$\bar{\boldsymbol{\theta}}$	Parameter vector for adaptation
$\sigma$	Sensor index
$\tau$	Time constant
$\tau_{ax}$	Accelerometer walking bias time constant
$\tau_{gz}$	Gyro walking bias time constant
$\Phi$	Robot body pitch angle
$\dot{\boldsymbol{\Phi}}_{i,N}$	Pitch rate vector, instance $I$ with $N$ elements
$\psi$	Robot body yaw angle
$\dot{\psi}$	Robot body yaw angular velocity
$\hat{\dot{\psi}}_{k-1}$	Estimate of robot body yaw angular velocity
$\dot{\psi}_{IMU}$	Measured body yaw angular velocity
$\ddot{\psi}$	Robot body yaw angular acceleration
$\Omega$	Tire angular velocity
$\omega_l$	Left front tire angular velocity
$\omega_{l,enc}$	Measured left front tire angular velocity
$\hat{\omega}_{l,k-1}$	Estimate of left front tire angular velocity
$\omega_r$	Right front tire angular velocity
$\omega_{r,enc}$	Measured right front tire angular velocity
$\hat{\omega}_{r,k-1}$	Estimate of right front tire angular velocity
$\dot{\boldsymbol{\omega}}_{lft,i,N}$	Left wheel angular acceleration vector, instance $i$ with $N$ elements
$\dot{\boldsymbol{\omega}}_{rt,i,N}$	Right wheel angular acceleration vector, instance $i$ with $N$ elements

# 1

## **Chapter 1: INTRODUCTION**

### ***1.1 Problem Motivation***

Traditionally, mobile robots have been developed for use in indoor laboratory environments or for very low-speed traversal of outdoor environments, such as with planetary exploration rovers. Recently interest has increased in outdoor high-speed mobile robots. A major application for high-speed robots is for the military. In 2007, the Defense Advanced Research Projects Agency (DARPA) continued to identify advanced unmanned systems as a major strategic thrust for the agency [13]. With the first two DARPA Grand Challenges in 2004 and 2005, the agency demonstrated great interest in automating military transport vehicles to minimize possible human casualties. To be effective, such transport vehicles must travel at high speed for efficient supply delivery and be capable of driving on unprepared surfaces in combat areas where roads may not be present.

Vehicle automation is also desired for civilian automobiles. In 2005 there were over 6 million motor vehicle crashes in the United States, resulting in nearly 3 million injuries and costing over \$230 billion [37]. Vehicle automation could potentially eliminate many of these motor vehicle accidents. Of the 1.9 million single vehicle accidents in 2005, at least 1.1 million occurred off the roadway [37]. It is clear that any

vehicle automation, whether fully autonomous or a “driver assist”, must be capable of handling road departure situations at high speeds.

This thesis explores methods for sensing and estimating the effects of natural terrain on a dynamic vehicle. The work in this thesis additionally recognizes that any realizable autonomous system will be constrained by economics and space and can not always have a sensor specifically designed for every task. Thus the algorithms presented here use intelligent algorithm design to estimate vehicle state and terrain information using common and/or affordable sensors normally available on a robotic platform, rather than utilizing costly, targeted sensing. This thesis addresses two topics related to robot mobility on outdoor terrain: wheel slip and terrain identification.

Wheeled vehicles encounter longitudinal and lateral wheel slip when driving over any terrain type. A tire must slip against terrain in order to generate traction and braking forces required for vehicle mobility. Traditional indoor mobile robots typically operate on relatively flat, high-traction terrain where longitudinal wheel slip is relatively small and frequently ignored. However, in general, outdoor terrain is low-traction, resulting in significant levels of wheel slip. Traditional position estimation systems which rely on wheel odometry for robot localization can accumulate large errors unless wheel slip is estimated and directly compensated for or high frequency absolute position measurements are available. Uncontrolled outdoor driving scenarios pose additional challenges to mobile robots, not present in laboratory settings. Traversability of slopes is dependant on both incline angle and terrain traction properties. Attempting to surmount a hill with either too large slope or too low traction or colliding with unobserved terrain features can result in robot immobilization.

This thesis develops two algorithms for detecting robot immobilization on outdoor terrain. The first algorithm utilizes a dynamic model-based approach to solve the general problem of estimating vehicle speed and longitudinal wheel slip. The second algorithm utilizes a model-free, signal recognition-based approach to directly detect robot immobilization. Both algorithms are experimentally validated and two detector fusion approaches are proposed for combining the results of multiple detectors.

Terrain surface properties, including roughness and traction, have a significant effect on vehicle performance, including robot path following ability, and wheel slip. Knowledge of terrain type can provide an indication of surface properties. In addition, terrain type can be useful to autonomous navigation systems, such as for indicating road departure scenarios. This thesis proposes an algorithm for estimating the traversed terrain profile for high-speed vehicles on outdoor terrain and utilizes the estimated profile to classify the underlying terrain type. The wheel slip detection and terrain classification algorithms presented in this thesis provide estimates of some vehicle-terrain interactions for future improved autonomous navigation systems for outdoor mobile robots.

## ***1.2 Outline of this Thesis***

This thesis is organized as follows. This chapter provides motivation for the remainder of the thesis. Chapter 2 introduces two algorithms for estimating wheel slip and detecting robot immobilization on outdoor terrain. Two detector fusion techniques are introduced that combine multiple detector outputs to increase detection robustness and accuracy. The algorithms presented in Chapter 2 are experimentally validated on an autonomous robotic platform in outdoor terrain. An algorithm is presented for adapting tire-terrain traction model parameters and preliminary simulation results are provided.



Chapter 3 proposes an algorithm for classifying the terrain traversed by a dynamic vehicle using tactile information. The algorithm enhances previous terrain classification work by explicitly considering the effects of vehicle speed by creating an estimate of the underlying terrain profile. The terrain classification algorithm is experimentally validated on a commercial automobile driving in real-world conditions. Chapter 4 summarizes the major contributions of this thesis and provides suggestions for future work.

# 2

## **Chapter 2: WHEEL SLIP DETECTION FOR**

### **MOBILE ROBOTS ON OUTDOOR**

#### **TERRAIN**

### ***2.1 Introduction***

#### **2.1.1 Problem Description**

Mobile robot position estimation systems typically rely (in part) on wheel odometry as a direct estimate of displacement and velocity [18],[8]. On high-traction terrain and in combination with periodic GPS absolute position updates, such systems can provide an accurate estimate of the robot's position. However, when driving over low-traction terrain, deformable terrain, steep hills, or during collisions with obstacles, an odometry-based position estimate can quickly accumulate large errors due to wheel slip. With ineffective odometry, periodic absolute position updates can cause large "jumps" in a robot's position estimate. In addition, between updates an odometry-based system is unable to differentiate between a robot that is immobilized with its wheels spinning and one that is driving normally. Autonomous robots should quickly detect that they are immobilized in order to take appropriate action, such as planning an alternate route away

from the low-traction terrain region or implementing traction control. Additionally, robust position estimation is required for accurate map registration.

Wheel slip can be accurately estimated through the use of encoders by comparing the speed of driven wheels to that of undriven wheels [25]; however this does not apply for all-wheel drive vehicles or those without redundant encoders. Ojeda and Borenstein have proposed comparing redundant wheel encoders against each other and against yaw gyros as an indicator of wheel slip, even when all wheels are driven [40], however this technique does not estimate the degree of wheel slip (i.e. whether the robot is fully immobilized). Ojeda and Borenstein have also proposed a motor current-based slip estimator [39]; however this technique requires accurate current measurement and terrain-specific parameter tuning, with proposed tuning techniques requiring either an accurate absolute positioning device or a robot with at least four driven wheels. In [2] Visual Odometry is used to estimate robot velocity and slip for a slip prediction algorithm. Although VO can be accurate on average over time, the authors report VO errors of ~12% on short time scales. In addition, the performance of VO can be degraded in near-featureless environments, such as sand. It should be noted that a body of work exists in the automotive community related to traction control and anti-lock braking systems (ABS); however, this work generally applies at significantly higher speeds than is typical for autonomous robots.

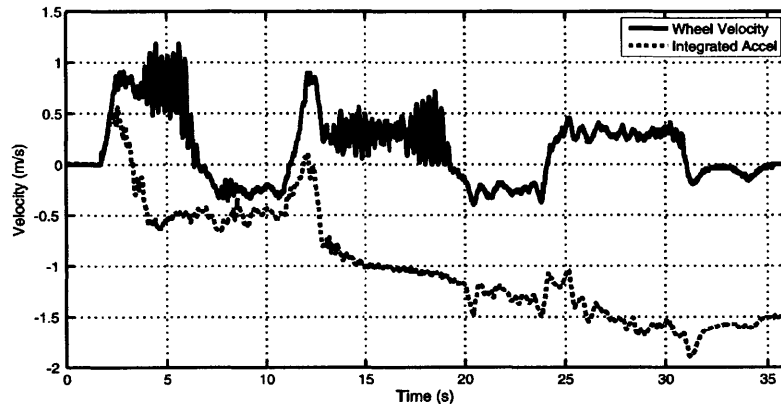
A large amount of work has utilized Kalman filters with inertial and absolute measurements to enhance dead reckoning and estimate lateral slip. In [4] a navigation system is proposed that uses inertial measurements combined with a sensor error model in a Kalman filter to increase measurement accuracy. In [18],[33] traditional dead

reckoning accuracy is improved by including inertial measurements. In [1], absolute position updates from GPS are fused with a model-based Kalman filter to estimate vehicle sideslip and improve position estimation accuracy, and in [42] this work is extended to consider the effects of vehicle roll and pitch. The notion of an effective tire radius, which can indirectly compensate for some longitudinal slip, is presented in [32]. None of this work, however, explicitly considers the effects of longitudinal wheel slip or vehicle immobilization.

A potentially simple approach to detecting robot slip and immobilization is to analyze GPS measurements. In open terrain, GPS can provide accurate position and velocity measurements; however, nearby trees and buildings can cause signal loss and multipath errors and changing satellites can cause position and velocity jumps [44],[27]. Additionally, GPS provides low frequency updates (e.g. typically near 1 Hz [23]) making GPS alone too slow for immobilization detection, where as close to instantaneous detection as possible is desired to avoid excessive position errors.

Another potentially simple slip detection technique is to estimate robot body velocity by integrating acceleration measurements (after subtracting gravitational acceleration due to vehicle pitch) then comparing this estimate against wheel velocity, thereby estimating wheel slip. However, such an approach is not robust at low speeds during travel on rough, outdoor terrain. Figure 2.1 compares wheel velocity with estimated body velocity for a sample experimental data set. At low speeds accelerometer drift errors dominate, causing the velocity estimate to quickly diverge. In this case a detector based on this estimate would detect immobilization for the majority of the data set and be ineffective. Because the velocity estimate error is essentially a random walk, in

some cases such a detector would estimate the velocity to always be larger than the wheel velocity, thus never detecting immobilization. [15] proposes a method of aiding the inertial estimate using vehicle constraints, however the method is not appropriate on uneven, low-traction terrain.

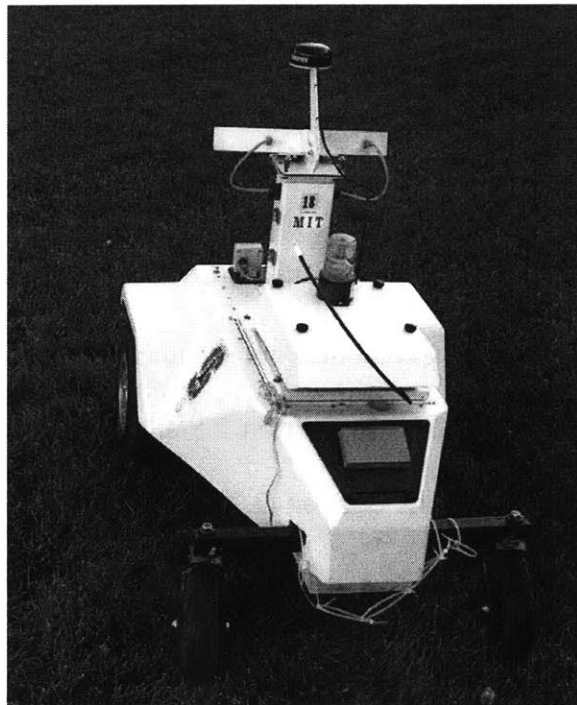


**Figure 2.1. Inaccuracy of estimating robot velocity by integrating measured acceleration.**

This chapter is organized as follows. In the following subsection, the experimental platform used for all the experiments in this chapter is introduced. In Section 2.2 a dynamic-model based approach to estimating wheel slip and immobilization is introduced and experimentally validated. In Section 2.3 a second approach to immobilization detection based on signal recognition is introduced and experimentally validated. Section 2.4 then introduces techniques for combining multiple detectors for one improved result. These techniques are validated by combining the detectors introduced in Sections 2.2 and 2.3. In Section 2.5 an adaptive tire model based on the dynamic framework introduced in Section 2.2 is presented. Finally, in Section 2.6 conclusions are drawn from this work.

## 2.1.2 Experimental Platform

An autonomous mobile robot developed for the DARPA LAGR (Learning Applied to Ground Robots) program has been used to experimentally validate the algorithms discussed in this chapter (Figure 2.2) [35]. The robot is 1.2 m long x 0.7 m wide x 0.5 m and has the kinematic configuration discussed in Section 2.2.2.1. The robot is equipped with 4096 count per revolution front wheel encoders, an Xsens MT9 IMU, a Garmin GPS 16 differential GPS, and two stereo pairs of video cameras (not used in this work). The IMU provides acceleration and angular rate measurements and a filtered estimate of vehicle roll and pitch. The robot has been used to collect data to process offline using a Matlab implementation of the proposed algorithms, as well as to run an online C++ implementation of the algorithm proposed in Section 2.2 on one of the robot's 2.0 GHz Pentium M computers.



**Figure 2.2. The LAGR robot.**

## **2.2 Dynamic Model-Based Wheel Slip Detection**

### **2.2.1 Introduction**

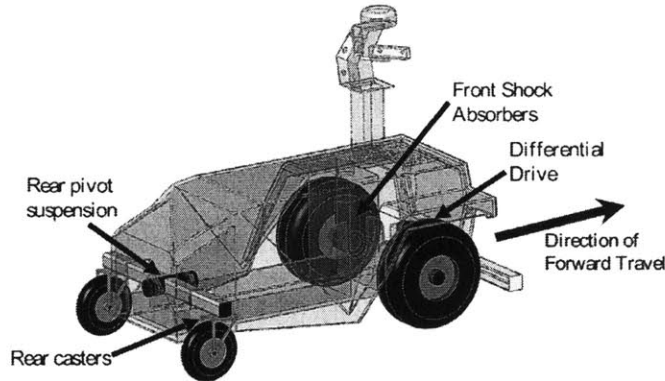
Here a method is presented for detecting robot wheel slip and immobilization that does not require redundant wheel encoders or motor current measurements. The proposed approach uses a dynamic vehicle model fused with wheel encoder, inertial measurement unit (IMU), and (optional) GPS measurements in an extended Kalman filter to create an estimate of the robot's longitudinal velocity. An insight of this approach is the realization that a robot becomes immobilized due to an external force resisting motion, be it a gravitational force resisting movement on an incline or an impact force exerted during a collision. The proposed algorithm utilizes a novel tire traction/braking model in combination with sensor data to estimate external resistive forces acting upon the robot and calculate the robot's acceleration and velocity. Weak constraints are used to constrain the evolution of the resistive force estimate based upon physical reasoning. The algorithm has been shown to accurately detect immobilized conditions on a variety of terrain types and provide an estimate of the robot's velocity during "normal" driving. The algorithm has been run in real time onboard a mobile robot and is shown to be robust to periods of GPS drop out. Preliminary results suggest that algorithm performance degrades gracefully during periods of IMU drop out. The proposed approach captures all relevant dynamics using one continuous model as opposed to approaches which seek to capture the complete dynamics by combining multiple limited models such as in [14],[45].

## 2.2.2 Dynamic Models

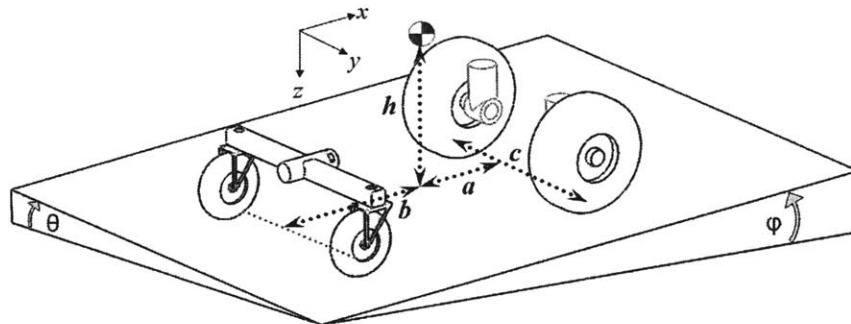
### 2.2.2.1 Robot Configuration

The robot configuration considered in this work is shown in Figure 2.3. The robot has four rubber pneumatic tires and is a front-wheel differential-drive configuration with undriven rear wheels that are freely-rotating castors mounted to a rear pivot joint suspension. The robot body-fixed coordinate system and kinematic parameters are shown in Figure 2.4 and the robot body and tire forces are shown in Figure 2.5. The dynamic models presented below are specific to this robot configuration; however the modeling process is adaptable to other wheeled vehicle configurations.

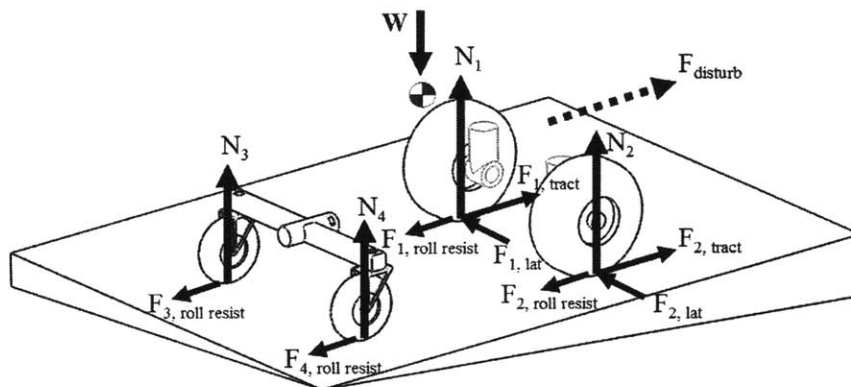




**Figure 2.3. Rendering of the mobile robot considered in this work.**



**Figure 2.4. Robot kinematic parameters.**



**Figure 2.5. Diagram showing vehicle and tire forces.**

#### 2.2.2.2 Vehicle Dynamics

Modeled forces acting on the robot include gravity, a lumped external disturbance force, and tire forces acting at the four tire-terrain contact patches (See Figure 2.5). The disturbance force can represent a variety of external forces such as wind resistance or the force caused by collision with an obstacle. In this work we limit the disturbance force to forces resisting vehicle motion.

Tire forces are composed of a normal component, traction/braking component, rolling resistance component, and lateral force component. The traction/braking forces are negligible for any undriven, freely rolling wheels, as is the case for the rear wheels of the robot considered here. The rear lateral forces can also be neglected because the rear castors spin freely and thus usually align with their velocity vectors.

The vehicle acceleration along the body x-axis is:

$$\begin{aligned}\dot{v}_{bx} &= \frac{1}{m} \left( \sum_{i=1}^2 F_{i,tract} + \sum_{i=1}^4 F_{i,roll\ res} + F_{disturb} - mg \sin(\varphi) \right) \\ &= f_{tire} + a_{dist,bx} - g \sin(\varphi)\end{aligned}\quad (2.1)$$

where  $m$  is the total vehicle mass,  $g$  is the acceleration due to gravity,  $f_{tire}$  and  $a_{dist,bx}$  are the equivalent x-axis body accelerations due to tire forces and the disturbance force. Assuming the vehicle's axis of yaw rotation is approximately the point midway between the front tires and neglecting any yaw moment due to gravity (which is small for moderate vehicle roll), the vehicle's yaw angular acceleration is:

$$\begin{aligned}\ddot{\psi} &= \frac{c}{2J} (F_{1,tract} + F_{1,roll\ res} - F_{2,tract} - F_{2,roll\ res}) \\ &= g_{tire}\end{aligned}\quad (2.2)$$

where  $J$  is the vehicle's moment of inertia about the body z-axis and  $c$  is the distance between front wheel centers. In general, if a robot has non-negligible lateral forces which do not act through the yaw axis, they must be estimated [51] and included in (2.2).

### 2.2.2.3 Normal Forces

Calculation of the robot's normal forces with arbitrary body roll ( $\theta$ ) and pitch ( $\varphi$ ) is in general an underconstrained problem. Methods proposed in the literature [21],[6]

typically consider a simplified 2-wheeled “bicycle” model, which can be applied for 2 or 4 wheeled vehicles when roll effects are ignored. In [28] it is suggested that normal forces be estimated by considering the elasticity of the terrain using tire-soil contact models presented in [5]. A rigid body solution can also be found (utilizing the Moore-Penrose Generalized Inverse), assuming point tire-soil contact [34].

For the robot configuration considered in this work, the assumption of zero moment about the passive rear suspension pivot joint allows the rear left and right normal forces to be assumed equal. With this assumption the normal force calculation is no longer underconstrained and an explicit solution exists. For normal force calculations it is also assumed that the vehicle longitudinal acceleration is negligibly small, which is generally valid for slow-moving robots. The normal forces are:

$$N_1 = \frac{W}{2} \cos(\varphi) \left( \frac{1}{a+b} (b \cos(\theta) - h \tan(\varphi)) - \frac{2h}{c} \sin(\theta) \right) \quad (2.3)$$

$$N_2 = \frac{W}{2} \cos(\varphi) \left( \frac{1}{a+b} (b \cos(\theta) - h \tan(\varphi)) + \frac{2h}{c} \sin(\theta) \right) \quad (2.4)$$

$$N_3 = N_4 = \frac{W}{2(a+b)} (h \sin(\varphi) + a \cos(\varphi) \cos(\theta)) \quad (2.5)$$

As a notational convenience we define the “normal accelerations” as:

$$n_{f,l} = \frac{N_1}{m} \quad (2.6)$$

$$n_{f,r} = \frac{N_2}{m}, \quad (2.7)$$

$$n_r = \frac{N_3}{m} = \frac{N_4}{m}. \quad (2.8)$$

#### 2.2.2.4 Traction/Braking Model

A large body of research has been performed on modeling tire forces on rigid and deformable terrain. Most models are semi-empirical and express tire traction/braking forces as a function of wheel slip  $i$  and wheel skid  $i_s$ , where [51]:

$$i = 1 - \frac{v_t}{r\omega} \quad (2.9)$$

and

$$i_s = 1 - \frac{r\omega}{v_t} \quad (2.10)$$

where  $v_t$  is the tire longitudinal velocity,  $r$  is the tire radius and  $\omega$  is the wheel angular velocity. For example, in [51] the traction force of a pneumatic tire on rigid terrain is formulated as:

$$F_{traction} = \begin{cases} K_t \varepsilon \approx C_i i & i \leq i_{critical} \\ \mu_p N - \frac{\lambda(\mu_p N - K' i)^2}{2l_t K' i} & i > i_{critical} \end{cases} \quad (2.11)$$

where  $K_t = k_t \lambda l_t \left(1 + \frac{l_t}{2\lambda}\right)$ ,  $K' = k_t \lambda l_t$ , and  $i_{critical} = \frac{\mu_p N}{l_t k_t (l_t + \lambda)}$ .

Where  $\lambda$ ,  $\mu_p$ ,  $k_t$ ,  $l_t$ , and  $C_i$  are constants,  $\varepsilon$  is the longitudinal strain (which is proportional to the slip), and  $N$  is the normal force acting on the wheel. A similar formulation is proposed for braking forces.

Implementation of a slip-based tire traction model such as (2.11) has many practical difficulties, including the need to distinguish the cases of traction and braking and driving forward and reverse to correctly calculate slip or skid. Another difficulty is introduced by the fact that the formulations are undefined at zero slip (i.e. when  $\omega = 0$  in (2.9) or  $v_t = 0$  in (2.10)). Additionally, (2.11) requires separate formulations for the low and high slip regimes (distinguished by  $i_{critical}$ ).

Here a unified, explicitly differentiable traction/braking model is proposed that captures the critical elements of the models proposed in the literature. The traction/braking force is expressed as a function of the wheel's relative velocity (also known as the slip velocity), rather than slip. Slip is a normalized version of relative velocity. A relative velocity-based formulation does not introduce the singularities found in slip-based formulations and is consequentially easier to apply within an extended Kalman filter framework. The proposed simplified model is:

$$F_{Traction} = N(\text{sign}(v_{rel})C_1(1 - e^{-A|v_{rel}|}) + C_2 v_{rel}) \quad (2.12)$$

where  $v_{rel}$  is the velocity of the tire relative to the ground:

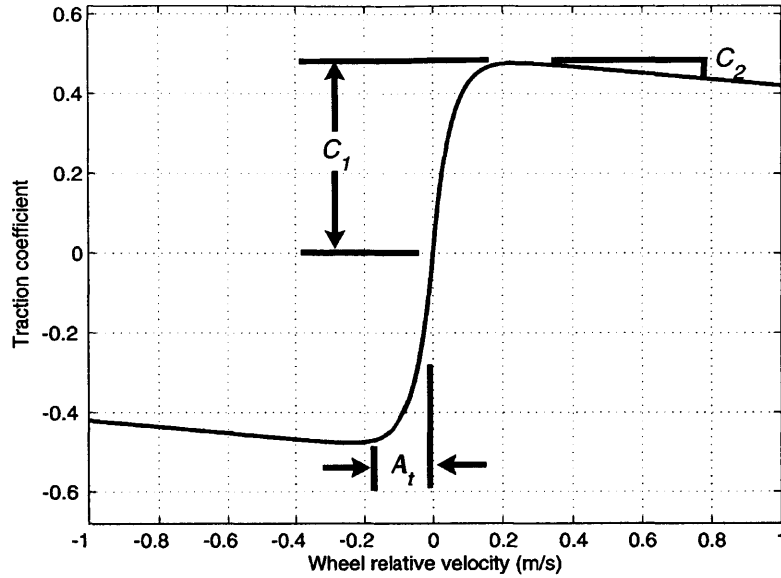
$$v_{rel} = r\omega - v_{fwd}, \quad (2.13)$$

and  $v_{fwd}$  is the tire's forward velocity, computed as:

$$v_{fwd, left} = v_{bx} + \frac{1}{2}c\dot{\psi}, \quad v_{fwd, right} = v_{bx} - \frac{1}{2}c\dot{\psi}, \quad (2.14)$$

where  $\dot{\psi}$  is the robot yaw rate and  $C_1$ ,  $A_t$ , and  $C_2$  are constants.

The simplified model is continuously differentiable and can predict both traction and braking forces, without a need to distinguish the two cases. Additionally, this model requires three terrain/tire dependant parameters ( $C_1$ ,  $A_t$ , and  $C_2$ ). By comparison, the popular ‘‘Magic Formula’’ empirical tire model requires six [51], and (2.11) requires four.  $C_1$  is a positive constant which can be viewed as the maximum tire-terrain traction coefficient.  $A_t$  is also a positive constant and is the slope of the traction curve in the low relative velocity region.  $C_2$  is the slope in the high relative velocity range and can be positive or negative depending on the terrain (Figure 2.6).

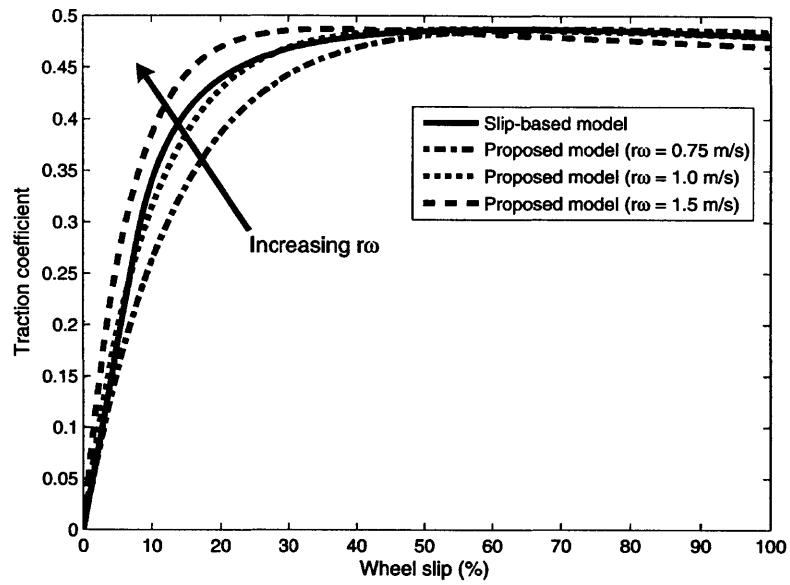


**Figure 2.6. Representative traction coefficient vs. relative velocity curve indicating effect of the 3 traction parameters. The traction coefficient is  $F_{traction}/N$ .**

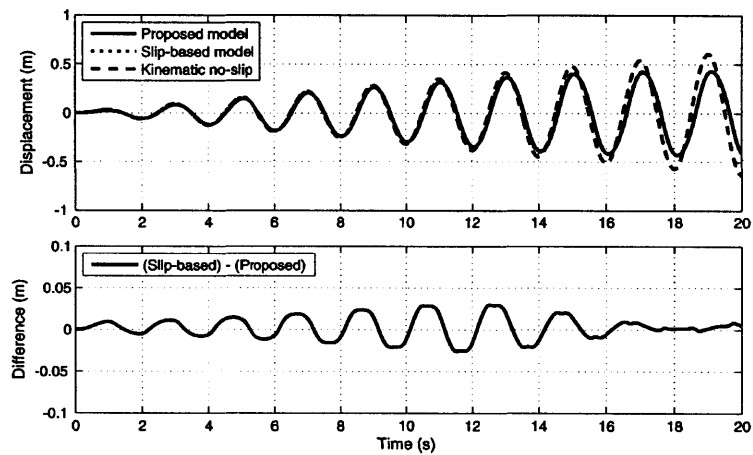
Figure 2.7 shows a plot of traction versus wheel slip for lines of constant wheel velocity using the proposed traction/braking model, as well as a representative traction force curve generated using (2.11). Whereas the traction-slip curve does not vary with wheel velocity using the slip-based model, the proposed model does. Assuming the robot typically operates near a nominal velocity, the proposed model can be interpreted as a pseudo-linearization around the nominal operating velocity.

Figure 2.8 compares displacement estimates calculated using the two traction models of Figure 2.7. The input wheel velocity is a 0.5 Hz sinusoid with amplitude linearly increasing from 0 to 2 m/s at the tire radius. In the top plot the two nearly-indistinguishable dynamic estimates show smaller amplitude due to wheel slip than the kinematic estimate, as expected. The bottom plot shows the difference in displacement between the two dynamic models. With wheel velocities ranging from 0-2 m/s, the difference in displacement between the two estimates is on the order of centimeters. When the range of operating velocities is within an order of magnitude of the nominal

velocity, a single tire model should be sufficient for most applications, however multiple models using different constants at multiple operating points can be employed if needed.



**Figure 2.7. Comparison of traction force vs. wheel slip curves for the slip-based model and the proposed simplified model at various wheel speeds.**



**Figure 2.8. Comparison of robot displacement calculated using proposed relative velocity-based tire model, slip-based model, and kinematic zero slip.**

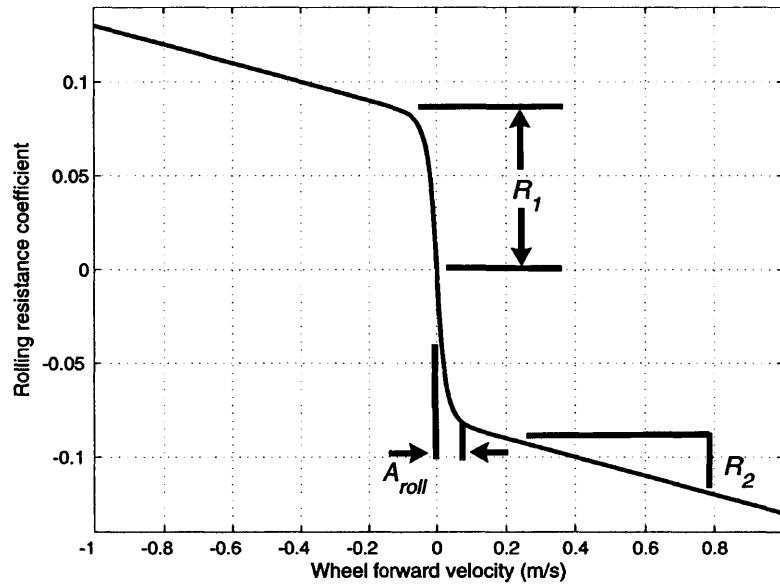


### 2.2.2.5 Rolling Resistance Model

Rolling resistance is generally modeled as a combination of static and velocity dependant forces [51],[6]. Here a function with form similar to (2.12) is proposed as a continuously differentiable formulation of the rolling resistance with the static force smoothed at zero velocity to avoid a singularity. The rolling resistance is:

$$F_{roll\ res} = -\text{sign}(v_{fwd})N\left(R_1\left(1-e^{-A_{roll}|v_{fwd}|}\right) + R_2|v_{fwd}|\right) \quad (2.15)$$

where  $R_1$ ,  $A_{roll}$ , and  $R_2$  are positive constants. Figure 2.9 shows a representative rolling resistance versus wheel translational velocity curve.



**Figure 2.9. Representative rolling resistance coefficient vs. velocity curve indicating the effect of varying the 3 resistance parameters. The rolling resistance coefficient is  $F_{roll\ resist}/N$ .**

### 2.2.2.6 Combined Tire Dynamics

Combining (2.1), (2.6)-. (2.8), (2.12), and (2.15), the vehicle acceleration due to tire traction/braking forces can be calculated as:

$$\begin{aligned}
f_{tire} = & n_{f,l} \left( \text{sign}(v_1) C_1 (1 - e^{-A_t |v_1|}) + C_2 v_1 - \text{sign}(v_3) R_{1,front} (1 - e^{-A_{roll} |v_3|}) - R_{2,front} v_3 \right) \\
& + n_{f,r} \left( \text{sign}(v_2) C_1 (1 - e^{-A_t |v_2|}) + C_2 v_2 - \text{sign}(v_4) R_{1,front} (1 - e^{-A_{roll} |v_4|}) - R_{2,front} v_4 \right) \\
& - 2n_r \left( \text{sign}(v_5) R_{1,rear} (1 - e^{-A_{roll} |v_5|}) - R_{2,rear} v_5 \right) \quad (2.16)
\end{aligned}$$

where:

$$v_1 = v_{rel,frontleft}, v_2 = v_{rel,frontright}, v_3 = v_{fwd,frontleft}, v_4 = v_{fwd,frontright}, v_5 = v_{bx}$$

Combining (2.2), (2.12), and (2.15), the yaw acceleration due to tire forces is:

$$\begin{aligned}
g_{tire} = & \frac{c}{2J} N_{f,l} \left( \text{sign}(v_1) C_1 (1 - e^{-A_t |v_1|}) + C_2 v_1 - \text{sign}(v_3) R_{1,front} (1 - e^{-A_{roll,front} |v_3|}) - R_{2,front} v_3 \right) \\
& - \frac{c}{2J} N_{f,r} \left( \text{sign}(v_2) C_1 (1 - e^{-A_t |v_2|}) + C_2 v_2 + \text{sign}(v_4) R_{1,front} (1 - e^{-A_{roll,front} |v_4|}) - R_{2,front} v_4 \right) \quad (2.17)
\end{aligned}$$

The models in (2.16) and (2.17) will be utilized in the slip estimation algorithm presented in the following section.

## 2.2.3 Slip Detector Algorithm

### 2.2.3.1 Extended Kalman Filter

The slip detector algorithm utilizes an extended Kalman filter (EKF) to integrate sensor measurements with the nonlinear vehicle model. The EKF structure requires that the discrete, nonlinear process model be written in the form:

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k, \mathbf{w}_{k-1}) \quad (2.18)$$

where  $\hat{\mathbf{x}}_k^-$  is the *a priori* estimate of the state vector,  $\mathbf{x}$ , at time step  $k$  and  $f$  is a nonlinear function of the previous state estimate,  $\hat{\mathbf{x}}_{k-1}$ , the current input vector,  $\mathbf{u}_k$ , and process noise,  $\mathbf{w}_{k-1}$ .

The measurement vector,  $\mathbf{z}$ , is a nonlinear function,  $h$ , of the true, current state vector and sensor noise  $\mathbf{v}$  such that:

$$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k). \quad (2.19)$$

The standard EKF time update equations using the notation of [48] are:

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k, 0), \quad (2.20)$$

$$\mathbf{P}_k^- = \mathbf{A}_k \mathbf{P}_{k-1} \mathbf{A}_k^T + \mathbf{W}_k \mathbf{Q}_{k-1} \mathbf{W}_k^T, \quad (2.21)$$

and the EKF measurement update equations using Joseph's form of the covariance update equation [24] are:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{V}_k \mathbf{R}_k \mathbf{V}_k^T)^{-1}, \quad (2.22)$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k (\mathbf{z}_k - h(\hat{\mathbf{x}}_k^-, 0)), \quad (2.23)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^- (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^T. \quad (2.24)$$

The relations  $f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k, 0)$  and  $h(\hat{\mathbf{x}}_k^-, 0)$  express the estimated state and measurement vectors,  $\hat{\mathbf{x}}_k^-$  and  $\hat{\mathbf{z}}$ , by evaluating the nonlinear process and measurement equations,

assuming zero noise.  $\mathbf{Q}$  and  $\mathbf{R}$  are process and measurement noise covariance matrices,  $\mathbf{P}$  is the state error covariance matrix, and  $\mathbf{A}_k$ ,  $\mathbf{W}_k$ ,  $\mathbf{H}_k$ , and  $\mathbf{V}_k$  are process and measurement Jacobian matrices, where:

$$\mathbf{A}_{[i,j]} = \frac{\partial f_i}{\partial x_j}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k, 0), \quad (2.25)$$

$$\mathbf{W}_{[i,j]} = \frac{\partial f_i}{\partial w_j}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k, 0), \quad (2.26)$$

$$\mathbf{H}_{[i,j]} = \frac{\partial h_i}{\partial x_j}(\hat{\mathbf{x}}_k^-, 0), \quad (2.27)$$

and

$$\mathbf{V}_{[i,j]} = \frac{\partial h_i}{\partial v_j}(\hat{\mathbf{x}}_k^-, 0). \quad (2.28)$$

To apply the EKF to real-world sensors with distinct, inconsistent sampling rates, a modified form of the EKF update process is required. The time update equations (2.20), (2.21) are calculated at a constant time step,  $\Delta t$ , such that in the absence of measurements the state estimate is updated based upon the dynamic model. When a new measurement from sensor  $\sigma$  becomes available, the measurement update equations (2.22)-(2.24) are computed for that measurement only, using  $\mathbf{H}_{k,\sigma}$ ,  $\mathbf{V}_{k,\sigma}$ ,  $\mathbf{R}_{k,\sigma}$ , and  $h_\sigma$ , which are the portions of the measurement Jacobians, measurement error covariance, and measurement function corresponding to sensor  $\sigma$  [49],[33]. These equations are repeated for each additional

measurement available at a given time step,  $k$ . If no measurement is available at time step  $k$ , then (2.22)-(2.24) are not used; instead  $\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^-$  and  $\mathbf{P}_k = \mathbf{P}_k^-$ .

### 2.2.3.2 State Space Model Formulation

The vehicle and sensor dynamics are formulated as a state space model using the following state vector:

$$\mathbf{x} = [v_{bx}, b_{ax}, a_{dist,bx}, \omega_l, \omega_r, \dot{\psi}, b_{g\psi}]^T,$$

where  $\omega_l$  and  $\omega_r$  are the angular velocities of the left and right front wheels and  $b_{ax}$  and  $b_{g\psi}$  are the accelerometer x-axis and yaw gyro walking biases, respectively, which are part of the IMU error model suggested in [17].

Typical errors found in accelerometers and rate gyros of low-cost IMU's are due to constant offsets,  $c_s$ , walking biases,  $b_s$ , and sensor noise,  $v$ , such that [17]:

$$z_{meas} = z_{actual} + c_s + b_s + v, \quad (2.29)$$

$$\text{with } \dot{b}_s = -\frac{1}{\tau} b_s + \sqrt{\frac{2f_s \sigma^2}{\tau}} w \quad (2.30)$$

where  $z_{meas}$  is the measured acceleration or angular rate,  $z_{actual}$  is the true value of the measured variable,  $v$  is assumed to be zero mean white noise,  $\tau$  is a time constant,  $f_s$  is the sampling frequency,  $\sigma^2 = E[b_s^2]$ , and  $w$  is zero mean white noise with  $E[w^2] = 1$ .

Using the above state vector, the vehicle dynamics can be written as:

$$\dot{\mathbf{x}} = \begin{bmatrix} f_{tire}(\mathbf{x}, \theta, \varphi) + a_{dist, bx} - g \sin(\varphi) \\ -\frac{1}{\tau_{ax}} b_{ax} \\ 0 \\ f_{controller, l}(u) \\ f_{controller, r}(u) \\ g_{tire}(\mathbf{x}, \theta, \varphi) \\ -\frac{1}{\tau_{gz}} b_{gz} \end{bmatrix} + \begin{bmatrix} \frac{w_1}{\tau_{ax}} \\ \sqrt{\frac{2f_s \sigma_{abx}^2}{\tau_{ax}}} w_2 \\ w_3 \\ w_4 \\ w_5 \\ \frac{w_6}{\tau_{gz}} \\ \sqrt{\frac{2f_s \sigma_{gbz}^2}{\tau_{gz}}} w_7 \end{bmatrix} \quad (2.31)$$

where  $w_i$  are zero mean white noise and  $f_{controller}(u)$  is the wheel acceleration which is a function of the robot's onboard velocity controller and the desired velocity. Note that with constant desired velocity, an ideal wheel speed controller would achieve  $f_{controller, l}(u) = w_5$  and  $f_{controller, r}(u) = w_6$ .

In practice, the high frequency and accuracy of wheel velocity measurements commonly allows an accurate estimate of  $\omega_l$  and  $\omega_r$  without modeling  $f_{controller}$ . When estimating the vehicle dynamics, we therefore neglect  $f_{controller}$ , assuming that the desired velocity is approximately constant between sensor updates. This assumption has the effect of smoothing the wheel speed measurements (helpful for removing pulses which can occur in velocity measurements derived from discrete encoder values). Discretizing the state equations and neglecting the zero-mean process noise,  $w_i$ , the *a priori* estimate of the robot state at time step  $k$  is:

$$\hat{\mathbf{x}}_k^- = \begin{bmatrix} \hat{v}_{bx,k-1} + f_{iire}(\hat{\mathbf{x}}, \boldsymbol{\theta}, \varphi) \Delta t + \hat{a}_{dist,bx,k-1} \Delta t - g \sin(\varphi) \Delta t \\ \hat{b}_{ax,k-1} \left( 1 - \frac{\Delta t}{\tau_{ax}} \right) \\ \hat{a}_{dist,bx,k-1} \\ \hat{\omega}_{l,k-1} \\ \hat{\omega}_{r,k-1} \\ \hat{\psi}_{k-1} + g_{iire}(\hat{\mathbf{x}}, \boldsymbol{\theta}, \varphi) \Delta t \\ \hat{b}_{g\psi,k-1} \left( 1 - \frac{\Delta t}{\tau_{gz}} \right) \end{bmatrix} \quad (2.32)$$

### 2.2.3.3 Measurement Model

The slip detector algorithm utilizes measurements from the IMU, GPS, and front wheel encoders. The measurement vector is:

$$\mathbf{z} = [\ddot{x}_{IMU}, \dot{\psi}_{IMU}, \dot{x}_{GPS}, \omega_{l,enc}, \omega_{r,enc}]^T,$$

where  $\ddot{x}_{IMU}$  and  $\dot{\psi}_{IMU}$  are IMU measurements of x-axis acceleration and yaw rate,  $\dot{x}_{GPS}$  is the component of the GPS velocity measurement along the body x-axis, and  $\omega_{l,enc}$  and  $\omega_{r,enc}$  are the left and right front wheel encoder angular velocity measurements. For the IMU measurements, the sensor model given by (2.29) is used. Note that for  $\ddot{x}_{IMU}$ ,  $z_{actual} = \dot{v}_{bx} + g \sin(\varphi)$ , as the accelerometer measures gravity even if the robot is stopped.

Simplifying, the measurement vector can be modeled as:

$$z = h(x, v) = \begin{bmatrix} f_{iire}(x, \theta, \varphi) + c_{ax} + b_{ax} + a_{dist, bx} \\ \psi + c_{gz} + b_{g\psi} \\ v_{bx} \\ \omega_l \\ \omega_r \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{bmatrix} \quad (2.33)$$

where  $c_{ax}$  and  $c_{gz}$  are constant offsets of the x-axis accelerometer and yaw gyro respectively and  $v_i$  are zero mean white noise. To approximate the constant offsets, they are initialized to the average of the first  $n$  IMU measurements, subtracting out the acceleration due to gravity from the acceleration measurement. When the robot is at rest, the constant offsets are updated with new measurements using the exponential moving average [36]:

$$EMA_{current} = (measurement_{current} - EMA_{prev}) \left( \frac{2}{1+p} \right) + EMA_{prev} \quad (2.34)$$

which is an approximation of the time average of the measurement over the last  $p$  samples, with a higher weight given to the most recent measurements. The EMA is not guaranteed to converge to the true value of the constant offsets, but instead converges to a locally constant offset over a window determined by the forgetting factor  $p$ , which is sufficient in practice. The EMA is easily and recursively calculated making it suitable for online implementation.

The estimated measurement vector is:



$$\hat{\mathbf{z}} = h(\hat{\mathbf{x}}_k^-, 0) = \begin{bmatrix} f_{\text{tire}}(\hat{\mathbf{x}}_k^-, \boldsymbol{\theta}, \boldsymbol{\varphi}) + c_{ax} + \hat{b}_{ax,k}^- + \hat{a}_{\text{dist},bx,k}^- \\ \hat{\psi}_k^- + c_{gz} + \hat{b}_{g\psi,k}^- \\ \hat{v}_{bx,k}^- \\ \hat{\omega}_{l,k}^- \\ \hat{\omega}_{r,k}^- \end{bmatrix}. \quad (2.35)$$

#### 2.2.3.4 Weak Constraints

The disturbance,  $a_{\text{dist},bx}$ , and accelerometer walking bias,  $b_{ax}$ , have both been modeled as random walks. Practically, the only difference between these variables in the model are that  $a_{\text{dist},bx}$  appears in the calculation of  $\dot{v}_{bx}$  while  $b_{ax}$  does not, and that  $a_{\text{dist},bx}$  is assigned a larger covariance in the matrix  $\mathbf{Q}$  so that it can evolve more quickly than  $b_{ax}$ .

Although a direct measure of the disturbance force is generally not available, rules governing its evolution can be developed based upon insight into the physical nature of the disturbance. These rules are implemented using weak constraints described in [20] and implemented in a vehicle model in [32]. Unlike ad hoc solutions, weak constraints are a principled method for integrating rules and constraints into the Kalman filter framework. Weak constraints can be viewed as virtual measurements or observations.

The linear weak constraints considered here are treated the same as physical measurements in the EKF framework. If some user-defined conditions are met (i.e. the physics-based rules), (2.22)-(2.24) are used to update the state vector and system covariance matrix with an associated noise covariance matrix,  $\mathbf{R}_{k,\sigma}$ , for each weak constraint. This is in contrast to ad-hoc techniques which may not propagate state changes though the system covariance matrix. Each weak constraint also has an

associated noise covariance matrix,  $R_{k,\sigma}$ . If the constraint is precisely known, then the covariance is zero and the constraint is considered a *strong* constraint. All of the constraints applied here have nonzero covariance.

The following pseudo code outlines the EKF update process including the weak constraints:

```

while (vehicle operational){
  increment EKFtime by constant dt
  EKF time update (2.20), (2.21)
  if (IMU measurement available){
    Do EKF measurement update (2.22)-(2.24) using:  $H_{IMU}$ ,  $V_{IMU}$ ,  $R_{IMU}$ ,
     $Z_{IMU}$ ,  $h_{IMU}$ 
  }
  if (GPS measurement available){
    Do EKF measurement update (2.22)-(2.24) using:  $H_{GPS}$ ,  $V_{GPS}$ ,  $R_{GPS}$ ,
     $Z_{GPS}$ ,  $h_{GPS}$ 
  }
  if (encoder measurement available){
    Do EKF measurement update (2.22)-(2.24) using:  $H_{encoder}$ ,  $V_{encoder}$ ,
     $R_{encoder}$ ,  $Z_{encoder}$ ,  $h_{encoder}$ 
  }
  for  $i = 1$  : (number of weak constraints) {
    if (Weak Constraint  $i$  condition satisfied){
      Do EKF measurement update (2.22)-(2.24) using:  $H_{wci}$ ,  $V_{wci}$ ,  $R_{wci}$ ,
       $Z_{wci}$ ,  $h_{wci}$ 
    }
  } end for
  (else no measurements or weak constraints)
} end while

```

Table 2.1 summarizes the weak constraints employed in this work. The second column presents the condition that must be met for the constraint to be applied and the third column gives the measurement innovation to be used in (2.23), which becomes  $\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k (z_{k,wci} - h_{wci}(\hat{\mathbf{x}}_k^-, 0))$  for each weak constraint  $i$ . Constraints 1-3 constrain the nature of the disturbance based on physical reasoning, to maintain observability of the state, similar to the implementation in [32]. Constraint 4 allows for calibration of the IMU biases when the robot is stopped. The forth column lists the  $R$  values used for each

weak constraint, normalized by the average of the  $R$  values for the real measurements (a smaller value indicates higher weighting). In practice, as a precaution to prevent the filter from diverging during fault conditions such as malfunctioning sensors, additional constraints could be applied using this framework to limit the magnitude and rate of change of some of the states based on known physical characteristics of the robot (i.e. the robot may have a known top speed). For some of the conditions the variable  $VelDir$  is used, defined as:

$$VelDir = \text{sign}(\omega_l + \omega_r) \tag{2.36}$$

such that  $VelDir$  equals 1 if the wheels are driving forward, 0 if the wheels are stopped or for pure rotation, and -1 if the wheels are driving in reverse.  $i_{EMA}$  is the EMA (2.34) of the average of the left and right front wheel slip.

**Table 2.1. Summary of weak constraints used.**

Description	Condition	“Measurement” Innovation	Normal R value
1) The modeled disturbance force should only oppose motion.	if $(\text{sign}(\hat{\mathbf{a}}_{dist,bx,k}^-) = VelDir)$ AND $(\text{sign}(\hat{\mathbf{a}}_{dist,bx,k}^-) \neq 0)$	$(z_{k,wc1} - h_{wc1}(\hat{x}_k^-, 0)) = (0 - \hat{\mathbf{a}}_{dist,bx,k}^-)$	[0.68]
2) The disturbance should act quickly. The disturbance should not gradually increase such that the wheel EMA of the slip slowly creases. Only applies when the average wheel slip is small. $min_{\Delta i}$ , $thresh_i$ , and $\alpha$ are user-defined constants	if $\left( 0 < \frac{\Delta i}{\Delta t} < min_{\Delta i} \right)$ AND $\left( i_{EMA} < thresh_i \right)$	$(z_{k,wc2} - h_{wc2}(\hat{x}_k^-, 0)) = (\alpha \hat{\mathbf{a}}_{dist,bx,k}^- - \hat{\mathbf{a}}_{dist,bx,k}^-)$ $0 < \alpha < 1$	[0.68]
3) The disturbance can stop the robot, but could not pull the robot backwards. If the robot is moving backwards, then either it is sliding down a hill and the disturbance should be zero, or the estimated disturbance is too high and should be reduced.	if $(\text{sign}(\hat{v}_{bx,k}^-) = -VelDir)$ $\Rightarrow \left( \begin{array}{l} \text{if } \text{sign}(f_{tire} - g \sin(\varphi)) = -VelDir \\ \text{then } \rightarrow \text{a)} \\ \text{else } \rightarrow \text{b)} \end{array} \right)$	$(z_{k,wc3} - h_{wc3}(\hat{x}_k^-, 0)) =$ a) $(0 - \hat{\mathbf{a}}_{dist,bx,k}^-)$ b) $(\max(VelDir[g \sin(\varphi) - f_{tire}, \hat{\mathbf{a}}_{dist,bx,k}^-]) - \hat{\mathbf{a}}_{dist,bx,k}^-)$	[0.14]
4) When robot is fully stopped, the disturbance force and walking biases should end to zero for calibration of the IMU constant biases. $t_{stop}$ is a constant. $T$ is the length of time the condition has been met.	if $(\omega_l = 0)$ AND $(\omega_r = 0)$ for $T \geq t_{stop}$	$(z_{k,wc4} - h_{wc4}(\hat{x}_k^-, 0)) =$ $([0,0,0]^T - [\hat{\mathbf{b}}_{ax,k}^-, \hat{\mathbf{a}}_{dist,bx,k}^-, \hat{\mathbf{b}}_{g\psi,k}^-]^T)$	[0.07] 0 1. 0 0

### 2.2.3.5 Slip and Immobilization Detection

The extended Kalman filter provides an estimate of the robot's forward velocity and the front wheels' angular velocities. Using these estimates, a criterion for detecting when the robot is immobilized is desired. A natural choice for an "immobilized" metric is the wheel slip (2.9). In practice, the calculated wheel slip can be noisy. For example, when the robot is stopped, an incremental wheel motion will yield a calculated slip of 100%, even though the robot is not immobilized. To improve robustness, the EMA (2.34) of the average of the left and right wheel slips is calculated and immobilization is detected if the EMA is larger than a threshold value. The threshold value is chosen empirically. A low value allows the detector to react quickly, however can be prone to falsely detecting immobilized conditions. In practice, since measurement noise can cause large variations in calculated slip at low speeds, the threshold can be chosen to vary with speed. Immobilization is not detected if the robot is braking (i.e.  $v_{bx} > r\omega$ ). The above technique represents one possible criterion for detecting immobilization which has worked well in practice; however other criteria are possible.

## 2.2.4 Experimental Results

### 2.2.4.1 Determination of Model Parameters

The algorithm requires knowledge or estimates of a number of constant parameters. Here, the robot mass and center of gravity location were directly measured. The measurement noise and walking bias process noise covariances were drawn from sensor data sheets and sensor measurements. The process noise and weak constraint

covariances were initially set to values estimated using physical reasoning, before manually tuning the values to achieve improved filter performance.

A series of simple experiments were performed on multiple terrain types to estimate the tire parameter values.  $C_1$  was estimated by measuring the force produced by spinning the robot's wheels while it was restrained with a spring scale.  $C_2$  was zero, as no nominal terrain-independent value was indicated by the test data.  $R_1$  was estimated by measuring the force required to pull the robot forward with the wheels freely spinning.  $A_t$  and  $R_2$  were chosen based upon tire force curves in the literature [10] and upon experimentation with the algorithm.  $A_{roll}$  was chosen to be large to approximate a static rolling resistance force. From these experiments, a set of nominal parameters were extracted which yielded good slip detection performance over many terrain types. Table 2.2 summarizes the tire constants used for this work.

**Table 2.2. Summary of tire constants used.**

Parameter	Nominal Value Used
$C_1$	0.52
$A_t$	$20 \text{ s}\cdot\text{m}^{-1}$
$C_2$	0
$R_{1,front}$	0.08
$R_{2,front}$	0.05
$A_{roll}$	$50 \text{ s}\cdot\text{m}^{-1}$
$R_{1,rear}$	0.0075
$R_{2,rear}$	0.02

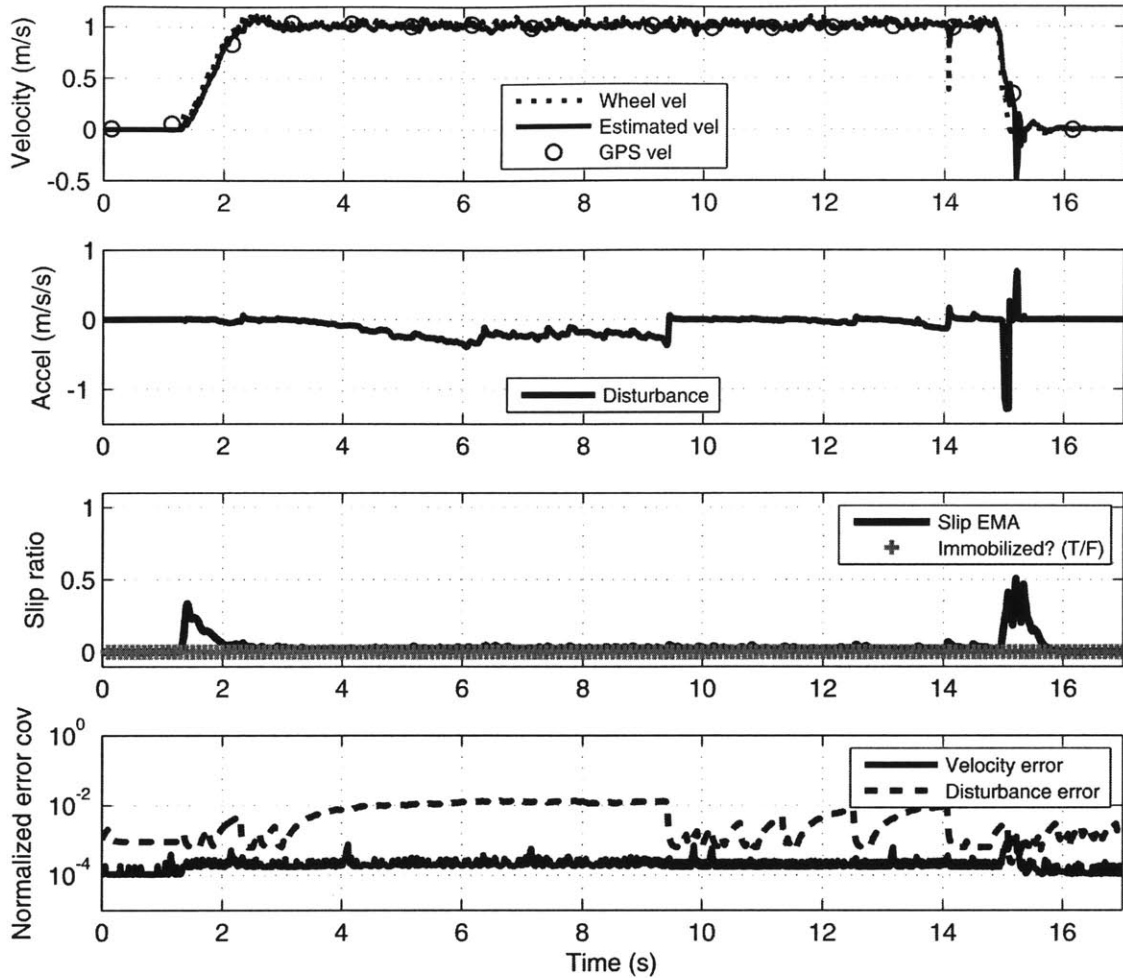
#### 2.2.4.2 Algorithm Performance

The algorithm was applied to 21 outdoor experimental test runs. During these tests, the robot traveled approximately 120 meters over a range of terrain types including loose mulch, loose gravel over hard dry soil, mud, and various grasses. The robot was

driven at speeds ranging from 0.1 m/s to 1 m/s. The test runs include 20 instances of the robot coming to a complete stop with the wheels still spinning, which were initiated by holding the robot back using a spring scale. The tests were performed on nominally level terrain with the robot commanded to drive in a straight line. Preliminary tests show equivalent results on non-level terrain and while the robot autonomously navigates arbitrary paths.

The slip detector correctly identified each of these 20 instances as immobilized with an average detection time of 0.4 seconds. All data with the robot driving freely or sitting at rest was correctly labeled as normal driving, with the exception of two false positives. In total less than 0.2% of the data points were falsely labeled as immobilized.

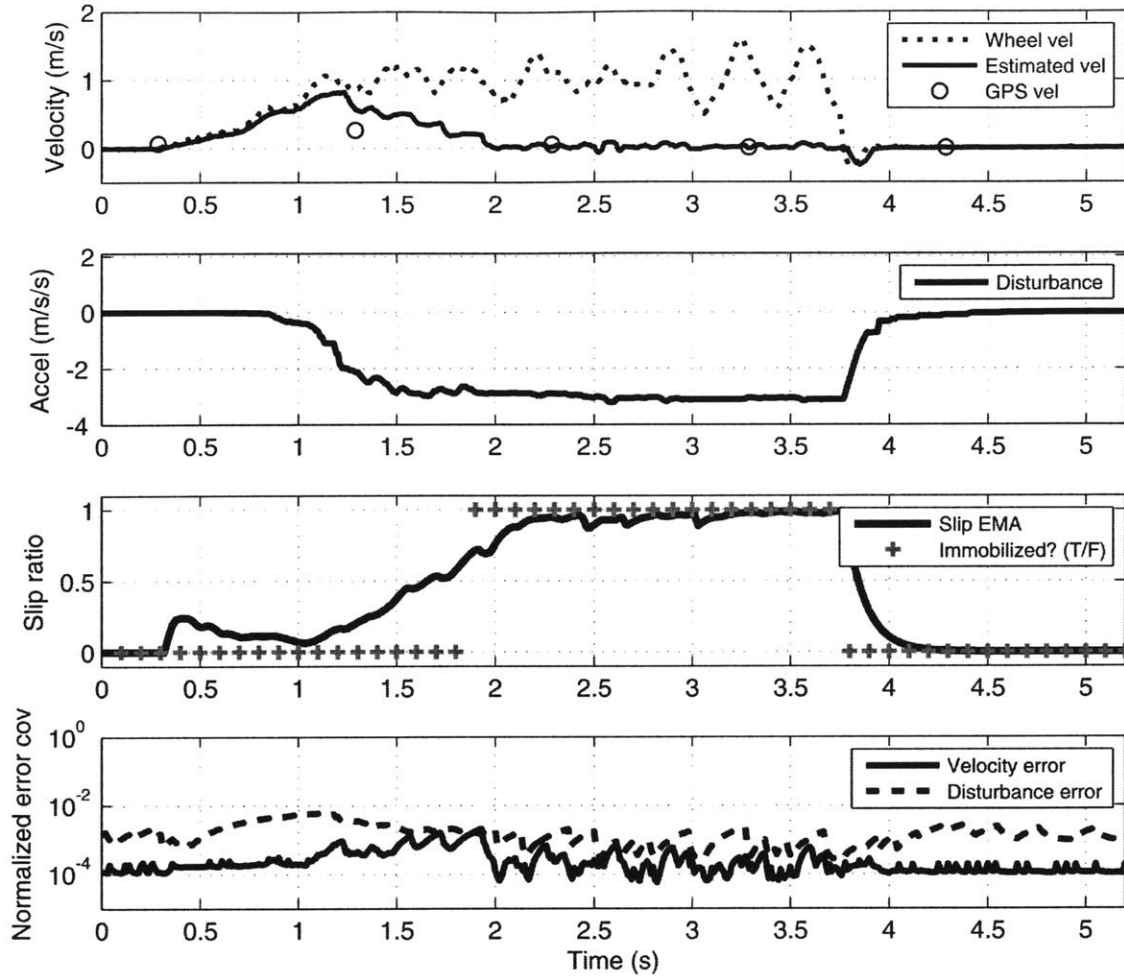
Figure 2.10 shows a plot of the robot driving unconstrained over grass at 1 m/s. In the top plot, it can be seen that the filter's estimated robot velocity follows the measured wheel velocity. At time equals 14 seconds there is a spike in the calculated wheel velocity; however the estimated robot velocity correctly smoothes this quantization error. The second plot shows the estimated disturbance, which remains small while the robot is driving. Just after the robot stops, suspension displacement creates a small spike in the disturbance. The third plot shows the EMA of the wheel slip. While driving, the wheel slip is estimated at approximately 3%, which is physically reasonable. The increased slip while accelerating and braking is also expected. The detector correctly labeled the entire data set as driving normally (i.e. "immobilized?" = 0). The fourth plot shows the error covariance from the Kalman filter  $P$  matrix for  $v_{bx}$  and  $a_{dist,bx}$ , normalized by the process noise covariance. In all results the normalized error remains well bounded, below unity, indicating that the filter is consistent.



**Figure 2.10. Example of robot driving normally. Note bottom plot is semi-log scale.**

Figure 2.11 shows a plot of the robot attempting to drive forward at 1 m/s on grass while restrained with a spring scale to produce 100% wheel slip. The velocity estimate shows that the robot accelerates against the spring, but quickly becomes immobilized. The disturbance estimate approaches a near-constant resistive value ranging from  $-2.8$  to  $-3.1 \text{ m}\cdot\text{s}^{-2}$  while the robot is immobilized, before returning to zero when the wheels stop spinning. During this test, the spring scale measured a 325 N force holding the robot back. The equivalent body acceleration for the 117 kg robot is  $2.8 \text{ m}\cdot\text{s}^{-2}$ , which closely agrees with the estimated disturbance. The slip EMA quickly approaches 100% and the detector identifies the robot as immobilized at time equals 1.85 s.





**Figure 2.11. Example of robot becoming immobilized.**

Note that the GPS velocity estimate for these two tests (conducted in open terrain with few trees or tall buildings) was very accurate and thus GPS-based slip detection is possible. However GPS measurements were available at 1 Hz, slower than desired for detection. Additionally, GPS returns the average velocity over the previous time step, and thus the measurement is truly accurate for 0.5 seconds prior to the reported measurement time (in Figure 2.10 & Figure 2.11 the GPS velocity plots are time shifted by 0.5 s to account for this). In the example shown in Figure 2.2, immobilization could not be detected by GPS until  $t \sim 2.8$  s, nearly one second slower than the proposed algorithm.

#### 2.2.4.3 Sensitivity to Tire Model Parameters

To study the algorithm's sensitivity to tire model parameter values, the 21 experimental data sets were reprocessed, individually varying one of the five tire constants by  $\pm 20\%$ . In all 210 tests, the algorithm correctly identified all 20 immobilizations. The number of false positives for each case is summarized in Table 2.3. It was observed that the algorithm performance was most sensitive to changes in  $C_f$ . Increasing  $C_f$  increases the maximum modeled traction, making the model less likely to estimate that traction has been lost and the wheels are slipping. Conversely, decreasing  $C_f$  reduces the modeled available traction, increasing the likelihood of wheel slip in the model and causing an increase in the number of false immobilization detections. Even in the worst case, only 0.3% of the data points were falsely labeled immobilized.

Two additional cases were evaluated as a limited study of second-order sensitivities. The first should be the worst case for false positives, with both traction parameters -20% and all resistance parameters +20%. In this case, all immobilizations were detected and there were 6 false detections. The second should be the worst case for correct detections, with both traction parameters +20% and all resistance parameters -20%. In this case there was only 1 false detection; however one immobilization was not detected. In summary, the algorithm appears to be quite robust to errors in the estimated tire model parameters. It should be noted that the algorithm's velocity estimate accuracy will depend on the accuracy of the tire model for the current terrain.

**Table 2.3. Sensitivity of false immobilized flags to changes in tire parameters.**

Parameter	$C_l$		$A_r$		$R_l$		$R_2$		$A_{roll}$		
Parameter Change	Nominal	+20%	-20%	+20%	-20%	+20%	-20%	+20%	-20%	+20%	-20%
# False Positives	2	1	5	2	3	2	2	2	2	2	2

#### 2.2.4.4 Algorithm Performance without GPS

The 21 experimental datasets were reprocessed without including GPS velocity measurements (i.e. using wheel encoder and IMU measurements only). The algorithm again correctly identified all 20 immobilizations. Without GPS, the false immobilization detections increased from two to four (0.35% of all data points). Figure 2.12 shows the results of processing the data set of Figure 2.11 without using GPS measurements. In this case the two plots are nearly indistinguishable. These results suggest the algorithm can be applied on systems lacking reliable GPS, such as mobile robots in urban surroundings, underwater, or where GPS is not available such as for Mars rovers. However, GPS can increase accuracy and improve performance when available.

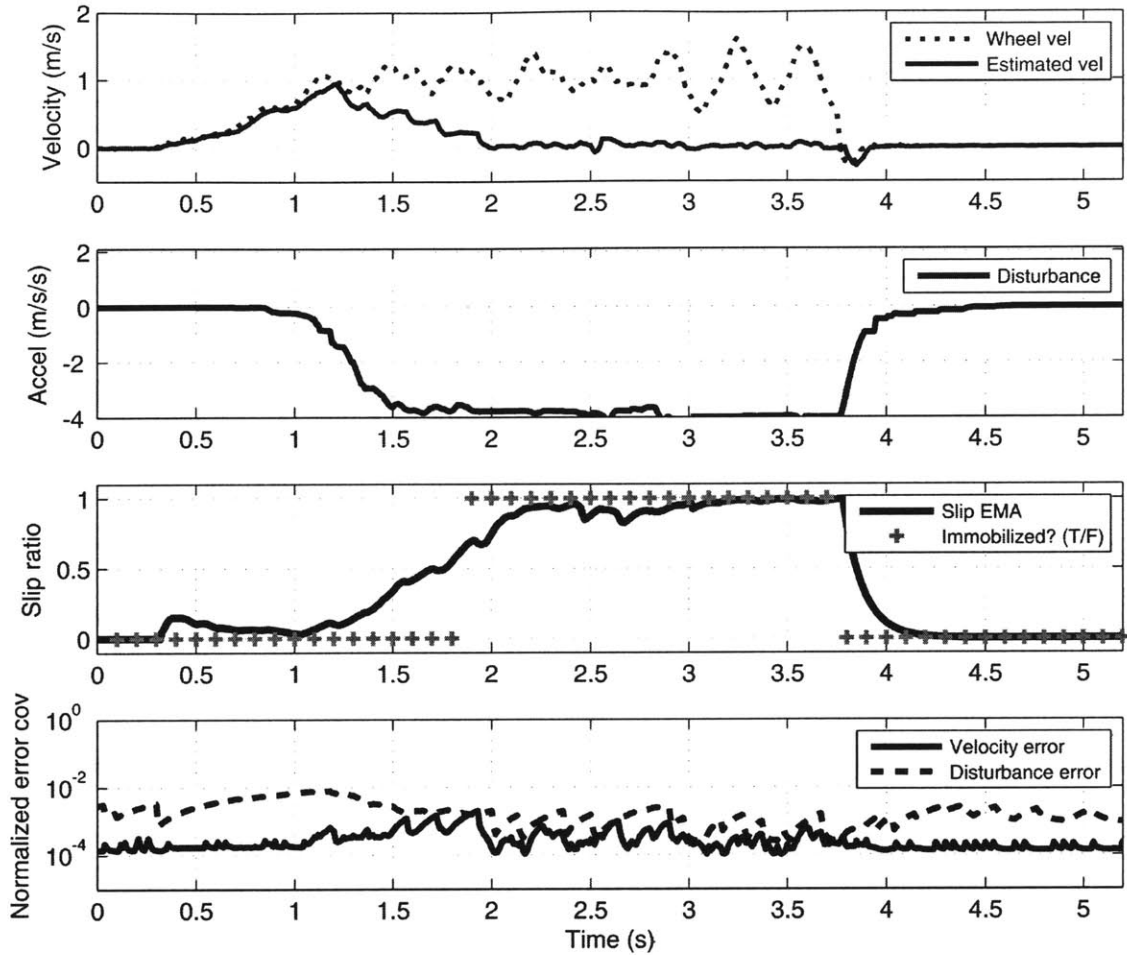


Figure 2.12. Same data as Figure 2.11 processed neglecting GPS.

## 2.2.5 Conclusions

A dynamic model-based slip detector has been proposed that has proven effective at detecting robot immobilization over a variety of outdoor terrains. The detector utilizes a novel tire traction/braking model and weak constraints to estimate external forces acting on the robot. The algorithm can be applied to any vehicle with an IMU, wheel encoders, and (optionally) GPS. Sensitivity analysis has indicated that accurate immobilization detection is possible with relatively coarse engineering estimates of tire/terrain model parameters. The algorithm also yields reasonably accurate estimates of the robot's

velocity and could potentially be implemented in a position estimation system that is robust to wheel slip. Inclusion of a lateral tire-force model could potentially allow estimation of side slip within the presented framework.

In Section 2.5 a technique for autonomously adapting the tire model parameters is presented which allows the algorithm to provide accurate velocity estimates as well as improve the slip detection time and reliability over variable terrain. Section 2.4 explores fusing the output of multiple slip detection algorithms to increase detection speed and accuracy.

## **2.3 Classification-Based Wheel Slip Detection**

### **2.3.1 Introduction**

Here a method is presented for detecting robot immobilization using a signal-recognition approach. Offline, a support vector machine (SVM) classifier is trained to recognize immobilized conditions within a feature space formed using inertial measurement unit (IMU) and optional wheel speed measurements. The trained SVM can then be used to quickly detect immobilization with little computation. Experimental results show the algorithm to quickly and accurately detect mobile robot immobilization in various scenarios.

One drawback of the model-based slip detection algorithm presented in Section 2.2 is that it requires identification of a small number of physical tire model parameters. The classification-based approach presented here was developed as an alternative, model-free approach to detecting robot immobilization. The classification-based approach,

however, only produces a binary immobilization detection output and does not produce an estimate of the robot's velocity.

Machine learning/classification techniques have been employed in various mobile robotics applications including vibration-based terrain classification [10] and self-supervised vision-based road detection [12], as well as other applications such as speech recognition [19]. The author is aware of no previous work utilizing these techniques for robot immobilization detection.

### 2.3.2 Classification Algorithm Overview

The algorithm proposed in this work was inspired by the observation that a human in a vehicle with eyes closed can quickly and robustly distinguish whether the vehicle is:

- 1) completely stopped with wheels stopped,
- 2) driving normally over outdoor terrain, or
- 3) immobilized, with the wheels rotating but slipping.

Even in the absence of training for this task and without visual feedback, a human can interpret clues such as vehicle heave/jounce and motor/engine sound signature to discriminate between cases 1-3 with reasonable accuracy.

The proposed algorithm uses a signal-recognition approach to detect mobile robot immobilization (case 3 above) based on inertial and wheel speed measurements. The measurements are used to form  $n$  features that can be used to distinguish between the two classes “immobilized” and “normal driving.” A support vector machine (SVM) is used to determine class boundaries within the  $n$ -dimensional feature space [11].

The SVM is trained using a hand-labeled data set of  $l$  instance-label pairs  $(\mathbf{x}_1, c_1), \dots, (\mathbf{x}_i, c_i), \dots, (\mathbf{x}_l, c_l)$  with  $\mathbf{x}_i \in \mathfrak{R}^n$  and  $c_i \in \{-1, 1\}$  [26],[29]. In this work, “normal” is

labeled as  $c_i = -1$  and “immobilized” as  $c_i = 1$ . The  $l$  training instance feature vectors,  $\mathbf{x}_i$ , are combined to form the  $l \times n$  feature matrix,  $\mathbf{X} = [\mathbf{x}_1 \ \cdots \ \mathbf{x}_l]^T$ , and the labels form the  $l \times 1$  training label vector,  $\mathbf{c} = [c_1 \ \cdots \ c_l]^T$ .

Classification accuracy is improved by scaling each feature type to have similar magnitudes [29]. To scale each feature to the range  $[-1, 1]$ , the  $n \times n$  scale factor matrix,  $\mathbf{S}$ , is formed such that:

$$\mathbf{S}_{i,j} = \begin{cases} \frac{1}{\max(|\text{column } j \text{ of } \mathbf{X}|)} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \quad (2.37)$$

and the scaled training feature matrix,  $\tilde{\mathbf{X}}$ , is then:

$$\tilde{\mathbf{X}} = \mathbf{X} \cdot \mathbf{S}. \quad (2.38)$$

$\tilde{\mathbf{X}}$  and  $\mathbf{c}$  are used to train a SVM using a radial basis function (RBF) kernel. An RBF kernel was chosen because it performs well with both non-linear and linear class relations and requires few kernel parameters [29]. SVM parameters are found using a grid search to systematically find a parameter set that minimizes the average classification error and error standard deviation of a  $\nu$ -fold cross-validation [29]. The final SVM model is trained using the best SVM parameter set and the entire training data set.

The parameter search and SVM training can be computationally expensive. However training is performed only once, offline, producing an SVM model suitable for computationally inexpensive online classification. Note that during online classification,

each measured feature vector,  $\mathbf{x}$ , is first multiplied by the scale factor matrix,  $\mathbf{S}$ , before classification by the trained SVM.

During online classification, the output of the SVM's decision function is a scalar decision value,  $\hat{f} \in (-\infty, \infty)$ , where the value of  $\hat{f}$  is a measure of the distance of the instance from the class boundary in the  $n$ -dimensional feature space. Typically an instance is labeled as:

$$label = \begin{cases} \text{immobilized (1)} & \text{if } \hat{f} > 0 \\ \text{normal (-1)} & \text{if } \hat{f} < 0 \\ \text{unknown (0)} & \text{if } \hat{f} = 0 \end{cases} \quad (2.39)$$

However, increased accuracy can usually be achieved at the expense of lowered labeling completeness (i.e. labeling more instances "unknown") using the following:

$$label = \begin{cases} \text{immobilized (1)} & \text{if } \hat{f} > threshold \\ \text{normal (-1)} & \text{if } \hat{f} < -threshold \\ \text{unknown (0)} & \text{if } -threshold \leq \hat{f} \leq threshold \end{cases} \quad (2.40)$$

where  $threshold \geq 0$ . In this work (2.39) has been used unless otherwise specified, meaning that all data has been classified.

### 2.3.3 Feature Vector Selection

In this work four features have been chosen to form the feature vector  $\mathbf{x}_i = [x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}]$ . Each feature is a numerical representation of sensor data that attempts to mimic the sensory cues a human operator would exploit when attempting to detect immobilized conditions. Data is sampled at a rate  $f_s$  and a numerical transform is



calculated on a set of  $N$  data points for each feature instance. Figure 2.3 and Figure 2.4 illustrate the coordinate system used in feature definitions.

The first two features were chosen as the variance of the  $N$  element groupings  $i$  of roll rate,  $\dot{\theta}_{i,N}$ , and pitch rate,  $\dot{\phi}_{i,N}$ , such that:

$$x_{i,1} = \text{var}(\dot{\theta}_{i,N}) = \text{E}\left(\left(\dot{\theta}_{i,N} - \text{E}(\dot{\theta}_{i,N})\right)^2\right), \quad (2.41)$$

$$x_{i,2} = \text{var}(\dot{\phi}_{i,N}) = \text{E}\left(\left(\dot{\phi}_{i,N} - \text{E}(\dot{\phi}_{i,N})\right)^2\right). \quad (2.42)$$

These two features are a measure of the degree of roll and pitch experienced by a vehicle during travel over uneven outdoor terrain.

The third feature was chosen as a measure of the variation in the z-axis (vertical) acceleration. The variance is a measure of the total variation from the mean over all frequencies; however empirical results have shown that only high frequency z-axis acceleration signal variation effectively distinguishes immobilized conditions. For feature three,  $\mathbf{P}_{a_z,i}$  the  $p$  element vector of the power spectrum coefficients of grouping  $i$  of z-axis acceleration is calculated using a discrete Fourier transform, where:

$$p = \left\lceil \frac{N+1}{2} \right\rceil. \quad (2.43)$$

where  $\lceil \cdot \rceil$  is the ceiling function. Then feature three is calculated as:

$$x_{i,3} = \sum_{k=\lceil p/2 \rceil}^p P_{a_2,i,k} \cdot \quad (2.44)$$

For this work,  $N = 50$  was chosen and  $f_s = 100$  Hz, resulting in a sum of the frequency content from 25 to 50 Hz. This frequency range was empirically determined to perform well for the robot system used in this work.

Feature four was chosen as the mean of the magnitude of the wheel angular accelerations:

$$x_{i,4} = \text{mean}(|\dot{\omega}_{lf,i,N} + \dot{\omega}_{rt,i,N}|) = E(|\dot{\omega}_{lf,i,N} + \dot{\omega}_{rt,i,N}|) \quad (2.45)$$

where  $\dot{\omega}_{lf,i,N}$  and  $\dot{\omega}_{rt,i,N}$  are the  $N$  element groupings  $i$  of the left and right wheel angular accelerations, respectively. During outdoor driving, terrain unevenness leads to variations in wheel torque, leading to variations in wheel angular acceleration. This variation is minimized when the robot is immobilized.

### 2.3.4 Experimental Results

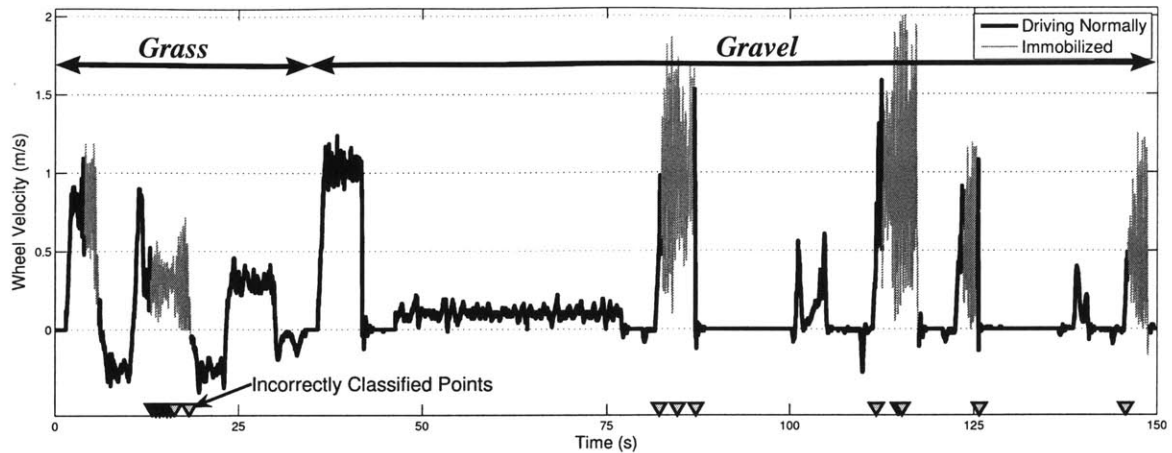
The SVM classifier was trained on data gathered during traversal of mud, loose mulch, and various grasses at speeds ranging from 0 to 1.0 m/s. The training data included 14 instances of the robot coming to a complete stop with the wheels still spinning, which were initiated by retarding robot motion with a spring scale. Using  $N = 50$ , the classifier was trained with 408 instance-label pairs, 18% of which were labeled as immobilized.

The classifier was tested using two distinct data sets. In the first set, the robot was driven once again over grass; however immobilization was initiated when the robot

experienced significant wheel slip while attempting to surmount a hill. In the second set, the robot was driven over loose gravel mixed with dry, brittle soil, and immobilization was initiated by retarding robot motion with a spring scale. Note that this terrain type was not present in the training data set.

Test results using all four features described in Section IIb are shown in Figure 2.13. Total classification accuracy was 94.7%. The figure shows that all incorrectly labeled points were near an actual immobilized period, with 98.1% of normal points correctly classified. The 1.9% of normal points classified as immobilized were all near the start or end of an immobilized period, which could indicate small errors in hand labeling of these extremal points. 75% of immobilized instances were classified correctly; however all immobilized periods were recognized as immobilized in at least some of the data instances comprising that occurrence.

Using only the first three features so that only IMU measurements were required, total classification accuracy was 92.0%, with 97.7% of normal instances and 59.1% of immobilized instances correctly classified. With only three features, classification accuracy was reduced, however false immobilized detections remained low and all immobilized occurrences were again detected.



**Figure 2.13. Experimental results of classifier-based immobilization detection. Each incorrectly classified point is a 0.5 second instance. Wheel velocity is effective linear velocity at wheel radius.**

Figure 2.14 shows a receiver operating characteristic (ROC) curve for classification of the test data set using all four features. The vertical axis shows the percentage of total instances that are classified correctly while the horizontal axis shows the percentage classified incorrectly. The curves are generated by progressively increasing *threshold* in (2.40), causing fewer points to be classified and more points to be “unknown.” Thus, increasing *threshold* results in a more conservative classifier. The upper-right endpoint of each line is the classifier accuracy with all instances classified (*threshold* = 0).

It can be seen that as *threshold* is increased, the percent of incorrect classification initially decreases rapidly, while the percent correct remains near constant, meaning in this region the majority of correctly labeled points were further than *threshold* from the class boundary. This curve shows the possible tradeoffs between number of instances labeled and labeling accuracy and can be a useful design tool.

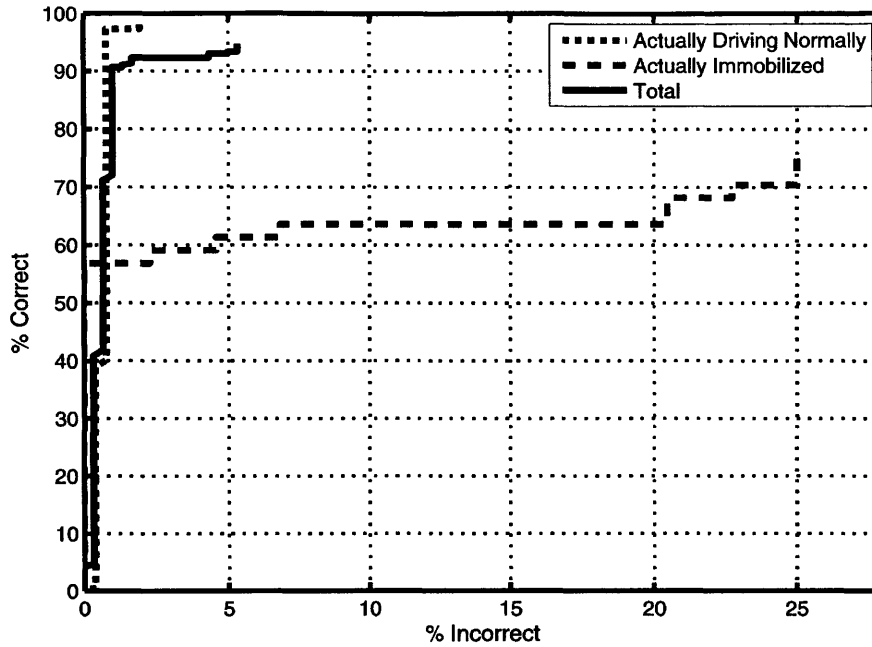


Figure 2.14. ROC curve for immobilization detection experimental results.

### 2.3.5 Conclusions

A signal recognition based approach to detecting robot immobilization has been proposed and experimentally validated. Four distinguishing features have been proposed for the algorithm requiring an IMU and (optionally) wheel encoders or tachometers, both common sensors on outdoor mobile robots. Future work will explore the effects of SVM kernel selection and robot speed and configuration on algorithm performance and test the algorithm on alternate terrain types and situations.

## 2.4 Detector Fusion

### 2.4.1 Introduction

To improve detection accuracy and robustness, immobilization detector fusion techniques have been explored. One technique is proposed to minimize false immobilization detections. A second is proposed to increase overall detection accuracy

while maintaining rapid detector response. The two fusion techniques are demonstrated with experimental data using the immobilization detection algorithms proposed in Sections 2.2 and 2.3.

## 2.4.2 Fusion Techniques

To increase immobilization detection accuracy two techniques have been explored to fuse multiple detector outputs. The first technique (termed Fusion 1) is designed to minimize false immobilization detections at the expense of increasing the number of immobilized instances incorrectly classified as normal. For  $d$  detectors,  $D_i$ , each with output 1 for “immobilized” and -1 for “normal”:

$$\text{Fusion 1} = \begin{cases} 1 & \text{if } (D_1=1) \text{ AND } (D_2=1) \dots \\ & \dots \text{ AND } (D_i=1) \dots \text{ AND } (D_d=1) \\ -1 & \text{otherwise} \end{cases} \quad (2.46)$$

Thus Fusion 1 detects immobilized only if all detectors agree that the robot is immobilized.

The second technique (termed Fusion 2) is designed to increase total detection accuracy and yield faster immobilization detection than Fusion 1. For Fusion 2, each detector output,  $D_i$ , is expressed as a continuous variable on the interval  $[-1, 1]$ , with an output of 1 meaning the detector is completely confident that the robot is immobilized, -1 meaning the detector is completely confident the robot is driving normally, and 0 meaning there is an equal probability of the robot being immobilized or driving normally. Fusion 2 is a weighted average of the detector outputs:

$$\text{Fusion 2} = \begin{cases} 1 & \text{if } \sum_{i=1}^d w_i D_i > a_t, \\ -1 & \sum_{i=1}^d w_i D_i < -a_t, \\ 0 & \text{otherwise} \end{cases} \quad (2.47)$$

where  $a_t$  is a threshold value and  $\tilde{w}_i$  are weights with:

$$\sum_{i=1}^d \tilde{w}_i = 1. \quad (2.48)$$

### 2.4.3 Experimental Results

The performance of the fusion techniques described in Section IVa was studied using the detector proposed in Section 2.3 (i.e. the SVM method) and the detector proposed in Section 2.2 (i.e. the EKF method). The output of the SVM method was scaled for Fusion 2 by first determining the smallest *threshold* for which all classified training points are classified correctly,  $threshold_{100\%}$ . Then:

$$D_{SVM} = \text{sat}\left(\frac{\hat{f}}{threshold_{100\%}}, 1\right) \quad (2.49)$$

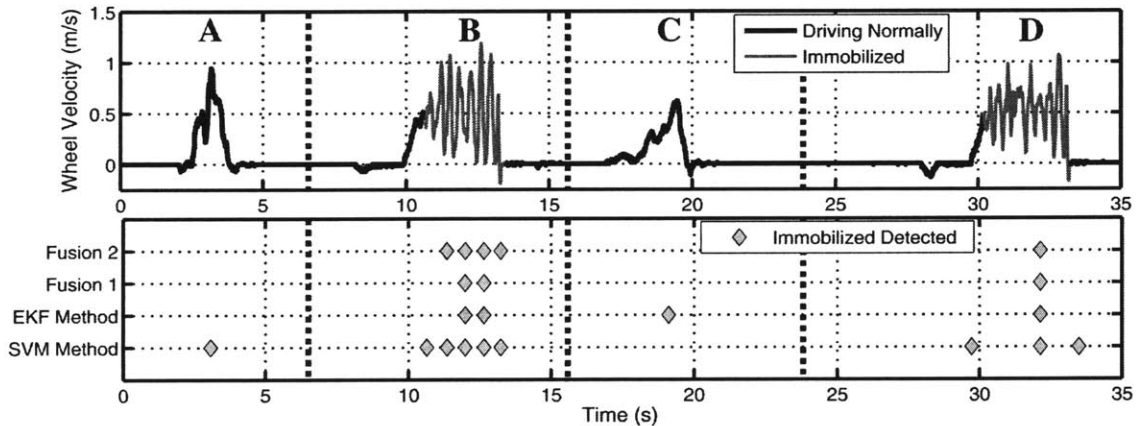
where the saturation function,  $\text{sat}(x, y)$ , is defined here as:

$$\text{sat}(x, y) = \begin{cases} x & \text{if } |x| < |y| \\ \text{sign}(x) \cdot |y| & \text{otherwise} \end{cases}. \quad (2.50)$$

The EKF method outputs a detected class for each of the  $N$  data points that make up instance  $i$  of the SVM method, but does not output a confidence value.  $D_{EKF}$  is

therefore taken as the mean of the  $N$  data points, providing an estimate of the detector's confidence. If half of the  $N$  points are classified as immobilized, then there is approximately a 50% chance the robot was immobilized during those data points and  $D_{EKF} = 0$ . This estimate has the drawback of assigning low confidence when immobilization begins near the end of the  $N$  points, possibly leading to sub-optimal detection time; however it provides a computationally simple method to test the fusion technique performance. For Fusion 2,  $\tilde{w}_1 = \tilde{w}_2 = 0.5$  and  $a_t = 0$  were used.

Figure 2.15 shows a dataset of the robot driving over loose mulch and demonstrates the relative performance of the fusion techniques. In sections A and C the robot was driven normally under remote control, and in sections B and D the robot was commanded to drive forward at 0.5 m/s but was restrained with a spring scale, causing immobilization. The bottom plot indicates the moments when immobilization was detected by the two detectors and two fusion techniques.



**Figure 2.15. Detector fusion results. Wheel velocity is effective linear velocity at wheel radius.**

It can be seen that in section A the SVM method falsely detects immobilization, likely due to rapid wheel speed oscillation. The EKF method, however, correctly labels this instance as normal driving, allowing both fusion techniques to correctly label this



section as normal. Similarly, in section C the EKF method misclassified an instance as immobilized, but the SVM method and both fusion methods correctly classified this section.

In section B, the SVM method detected immobilization very rapidly, while the EKF method's detection time was approximately 1.0 second slower. In this case, the SVM method detected immobilized immediately after the robot begins to decelerate, while the EKF method detected immobilized when the robot came to a stop. As expected, Fusion 1 only detected immobilization when both detectors agreed. Fusion 2 was able to detect immobilization approximately 0.5 seconds sooner than Fusion 1 because the SVM method expressed high confidence in its output while the EKF method expressed an uncertain output (i.e. an output near 0). In section D, the SVM method expressed a low confidence in its early immobilization detection and neither fusion technique detected immobilization until the EKF method was in agreement.

A comparison of detection accuracy of the four methods when performed on the Section 2.3.4 test set, which included 301 half-second instances, is shown in Table 2.4. All four techniques detected the 6 immobilized periods. The SVM method detected immobilization the fastest followed by Fusion 2; however in some cases the SVM method detected immobilization before the vehicle was stopped, accounting for the 3 false positives. Both fusion techniques eliminated these false positives, with Fusion 2 demonstrating the highest total accuracy.

**Table 2.4. Comparison of accuracy of detection and fusion techniques on Section 2.3.4 test set.**

	SVM Method	EKF Method	Fusion 1	Fusion 2
Total Accuracy:	94.7%	95.7%	91.7%	98.0%
# False Positives:	3	0	0	0
# False Negatives:	13	13	25	6

Although not shown in Figure 2.15 or Table 2.4, it is possible that a detector could falsely label an instance with high enough confidence for the point to be mislabeled by Fusion 2 but not Fusion 1. Fusion 1 should therefore be more robust to false positives. If both detectors mislabel an instance, it will be mislabeled under both fusion techniques.

#### **2.4.4 Conclusions**

Two simple immobilization detector fusion techniques have been proposed to combine the output of the classifier-based immobilization detector and a dynamic model-based detector. Fusion 1 resulted in a conservative approach to minimize false detections, while Fusion 2 provided faster performance while potentially allowing more false detections. Both fusion techniques were shown to eliminate false immobilization detections on the experimental data set and increase overall accuracy compared to each individual detector. Future work should explore using various fusion techniques to combine more than two detectors and for alternative applications including terrain classification.

## **2.5 An Adaptive Tire Model**

### **2.5.1 Introduction**

In Section 2.2, a dynamic model-based wheel slip estimator was presented. The algorithm estimates a robot's longitudinal velocity in the presence of wheel slip. A valuable application of this algorithm would be to extend the method to an accurate and robust robot position estimate. However, the presented algorithm is dependent on several tire/terrain parameters that vary with terrain type. For accurate position or velocity estimation, accurate identification of these traction parameters is required. This section presents a method for automatic adaptation of the tire model parameters. Preliminary simulation results show that the method adapts the tire parameters toward their true values and increases velocity estimation accuracy.

### **2.5.2 Dynamic Model**

The dynamic model used for tire parameter adaptation is identical to the one presented in Section 2.2.2. Recall that the model-based slip detector algorithm estimates the disturbance force causing immobilization; however the adaptation algorithm proposed here will only be run when the vehicle is driving freely. Therefore the disturbance force is neglected here. Using the notation of Figure 2.5, the vehicle acceleration along the body x-axis when the robot is unconstrained is:

$$\dot{v}_{bx} = \frac{1}{m} \left( \sum_{i=1}^2 F_{i,tract} + \sum_{i=1}^4 F_{i,roll\ res} - mg \sin(\varphi) \right). \quad (2.51)$$

Using the same tire models proposed in Section 2.2, we define the following parameter vector of tire constants to be identified:

$$\begin{aligned} \bar{\theta} &= (\theta_1 \ \theta_2 \ \theta_3 \ \theta_4 \ \theta_5 \ \theta_6 \ \theta_7 \ \theta_8 \ \theta_9)^T \\ &= (C_1 \ -A_t \ C_2 \ -R_{1,front} \ -A_{roll,front} \ -R_{2,front} \ -R_{1,rear} \ -A_{roll,rear} \ -R_{2,rear})^T \end{aligned} \quad (2.52)$$

Using the parameter vector  $\bar{\theta}$ , and following the wheel velocity notation from Section 2.2:

$$v_1 = v_{rel,frontleft}, v_2 = v_{rel,frontright}, v_3 = v_{fwd,frontleft}, v_4 = v_{fwd,frontright}, v_5 = v_{bx},$$

the vehicle forward acceleration can be written as:

$$\begin{aligned} \dot{v}_{bx} &= n_{f,l} (\text{sign}(v_1) \theta_1 (1 - e^{\theta_2 |v_1|}) + \theta_3 v_1 + \text{sign}(v_3) \theta_4 (1 - e^{\theta_5 |v_3|}) + \theta_6 v_3) + \\ &\quad n_{f,r} (\text{sign}(v_2) \theta_1 (1 - e^{\theta_2 |v_2|}) + \theta_3 v_2 + \text{sign}(v_4) \theta_4 (1 - e^{\theta_5 |v_4|}) + \theta_6 v_4) + \\ &\quad 2n_r (\text{sign}(v_5) \theta_7 (1 - e^{\theta_8 |v_5|}) + \theta_9 v_5) \end{aligned} \quad (2.53)$$

And with initial conditions the above can be used to solve for the vehicle's velocity.

### 2.5.3 Adaptation Algorithm

The tire model parameters,  $\bar{\theta}$ , are generally unknown or poorly known functions of many factors including tire type, tire pressure, tire wear, terrain type, terrain moisture, etc. Rather than attempting to define *a priori* information for tire parameters for all possible driving conditions, it is desired to have online, automated parameter adaptation. This is a significant challenge as three of the parameters are nonlinearly involved in the

model. Sensor data useful for adaptation is available from the onboard GPS and IMU. The GPS provides an estimate of the vehicle's velocity at 1 Hz. The reported GPS velocity estimate appears to be the average velocity over the one second interval,  $T$ , between updates:

$$\bar{v}_{GPS} = \frac{1}{T} \int_{t-T}^t v_{bx}(\tau) d\tau \quad (2.54)$$

After estimating the vehicle's velocity by solving (2.53), we can calculate the average estimated vehicle velocity,  $\hat{v}(\bar{\theta})$ . The IMU returns acceleration measurements at 100 Hz which can be directly compared to the estimated acceleration calculated with (2.53). Although the IMU data is a more direct measurement of the modeled variable, it is desired to also utilize the GPS measurement as the IMU provides a fairly noisy signal.

The adaptation problem has been formulated as a constrained minimization problem [38]. The adaptation is performed after each GPS update by minimizing:

$$f = \frac{1}{2} \bar{v}^2 + \frac{1}{2} (\Delta \bar{\theta})^T K^{-1} (\Delta \bar{\theta}) + p \sum_k \left( \dot{v}_{bx,k} - \hat{\dot{v}}_{bx,k} \right)^2 \Big|_{\text{last 100 accel measurements}} \quad (2.55)$$

subject to:

$$A\bar{\theta} \leq \bar{0} \quad (2.56)$$

where:

$$A = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad (2.57)$$

$$\tilde{v} = (\hat{v}(\bar{\theta}) - \bar{v}_{GPS}), \quad (2.58)$$

and

$$\Delta\bar{\theta} = \bar{\theta}_{adapted} - \bar{\theta}_0 \quad (2.59)$$

The term  $\frac{1}{2}(\Delta\bar{\theta})^T K^{-1}(\Delta\bar{\theta})$  is included to control the adaptation rate, where  $K$  is a positive definite gain matrix. Due to sensor noise, it is not generally desired for the adaptation to exactly match the model to the sensor data at every time step, but rather to tend to match the data over time.  $p$  is a positive constant used to tune the tradeoff between GPS and IMU data preference.

The constrained minimization formulation could theoretically be solved using Lagrange multipliers, however a closed form solution does not appear to exist for the

proposed non-linear optimization. Instead the Matlab minimization function “fmincon” is used which utilizes a sequential quadratic programming method. Running the adaptation algorithm on 37 seconds of test data using non-optimized Matlab code, requires approximately 23 seconds, suggesting that the algorithm could potentially be implemented in real time.

## 2.5.4 Results

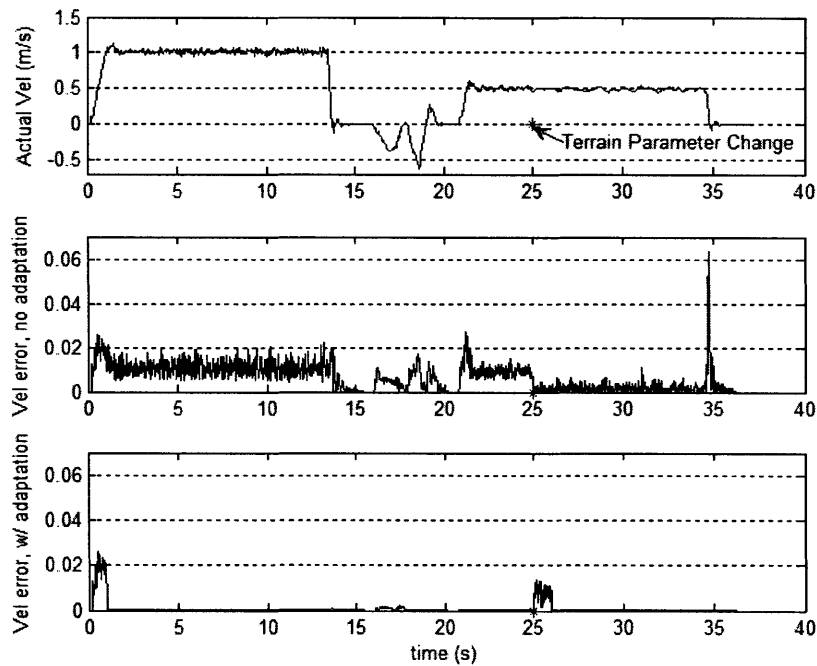
Since a ground truth is necessary for evaluating the accuracy of the tire parameter adaptation algorithm, it is not easily experimentally validated. The algorithm was instead tested against simulated data. 37 seconds of actual wheel encoder data from the vehicle driving over green grass was fed into the model given by (2.53), as well as the yaw rate model discussed in Section 2.2.2. This calculation was done using a parameter vector  $\bar{\theta}_{true}$ , which was held constant at  $\bar{\theta}_{true,1}$  for the first 25 seconds of simulation and then stepped to  $\bar{\theta}_{true,2}$  for the remainder of the simulation, to simulate driving onto a new terrain type. This simulation produced the “true” velocity profile, shown in Figure 2.16. This simulation was also used to generate simulated IMU and GPS “measurements”. A third parameter vector,  $\bar{\theta}_{start}$ , was chosen as a nominal vector to initialize the adaptation algorithm. Table 2.5 lists the parameters used for the three parameter vectors.

Using  $\bar{\theta}_{start}$ , two additional velocity profiles were generated. The first was generated using  $\bar{\theta} = \bar{\theta}_{start}$  for the duration of the simulation, while the second was generated using the adaptation algorithm. Figure 2.16 shows the absolute velocity error with and without adaptation. The robot drives forward at nearly constant 1 m/s for the first 14 seconds, then backs up and turns around, followed by driving at nearly constant

0.5 m/s from time equals 21 to 35 seconds. We can see that after a transient period, the error is significantly reduced using adaptation. The algorithm is also able to quickly adapt to the parameter change at time equals 25 seconds. In this example, the average error is reduced by 89%. The amount of error reduction will depend on how distant the initial parameter guess is from the true value, however the improvement appears significant since the traction parameters can change greatly over different terrains.

**Table 2.5. Parameter vectors used for simulations.**

	$\theta_1$	$\theta_2$	$\theta_3$	$\theta_4$	$\theta_5$	$\theta_6$	$\theta_7$	$\theta_8$	$\theta_9$
$\bar{\theta}_{true,1}$	0.52	-20	0	-0.08	-50	-0.05	-0.0075	-50	-0.02
$\bar{\theta}_{true,2}$	0.40	-16	0	-0.08	-50	-0.05	-0.0075	-50	-0.02
$\bar{\theta}_{start}$	0.50	-18	0	-0.10	-45	-0.05	-0.10	-45	-0.01

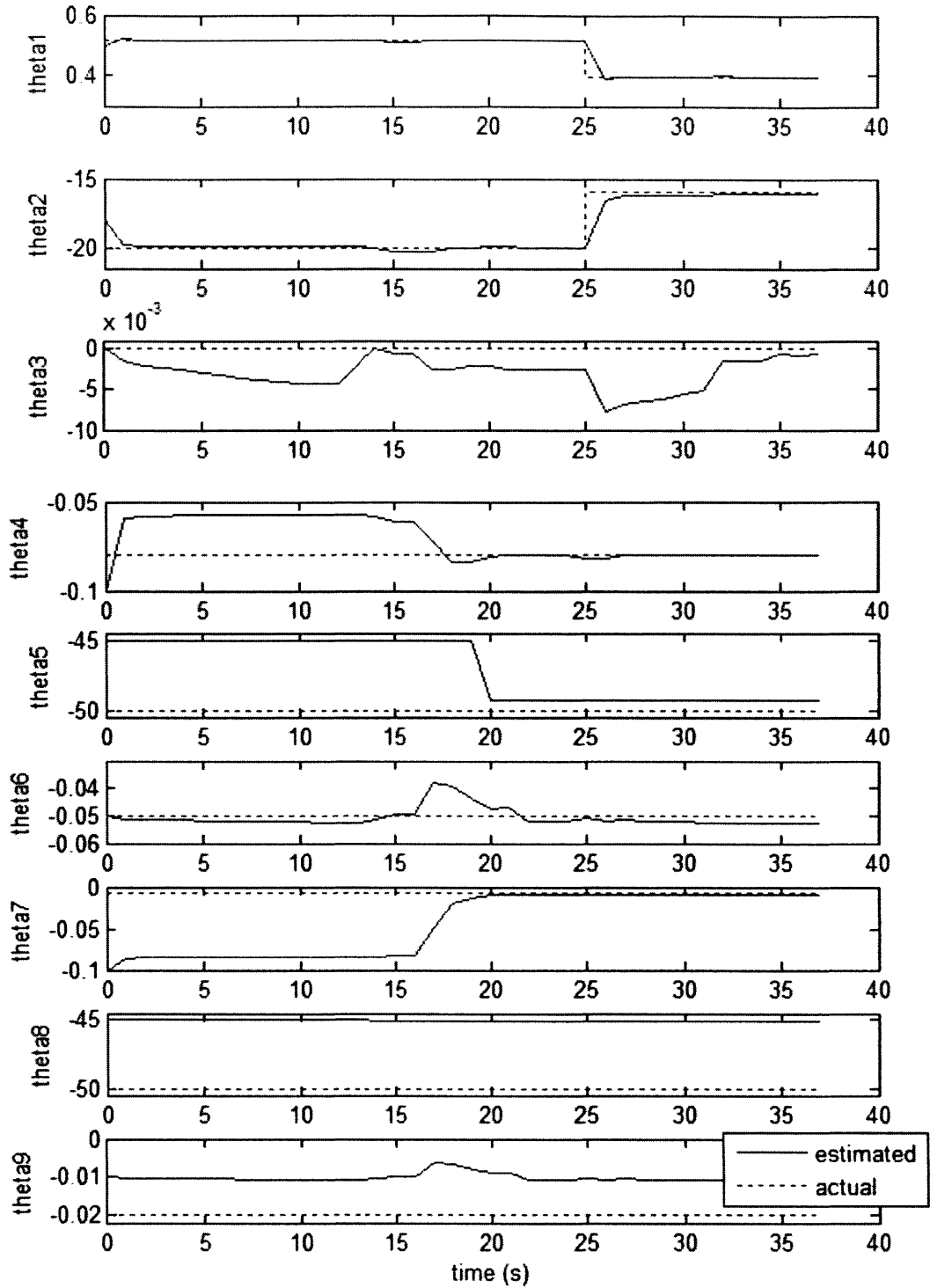


**Figure 2.16. True velocity profile and velocity errors with and without adaptation.**

Figure 2.17 shows the evolution of the parameter vector using the adaptation algorithm. The velocity estimate is most sensitive to  $\theta_1$  and  $\theta_2$ , and we see that these terms are estimated very effectively. We see that none of the other terms are divergent,



and most tend toward their true values, although some adapt very slowly, such as  $\theta_8$ . We should note that the algorithm does not guarantee that the parameters will reach their true values, but simply that the velocity and acceleration errors will be reduced.



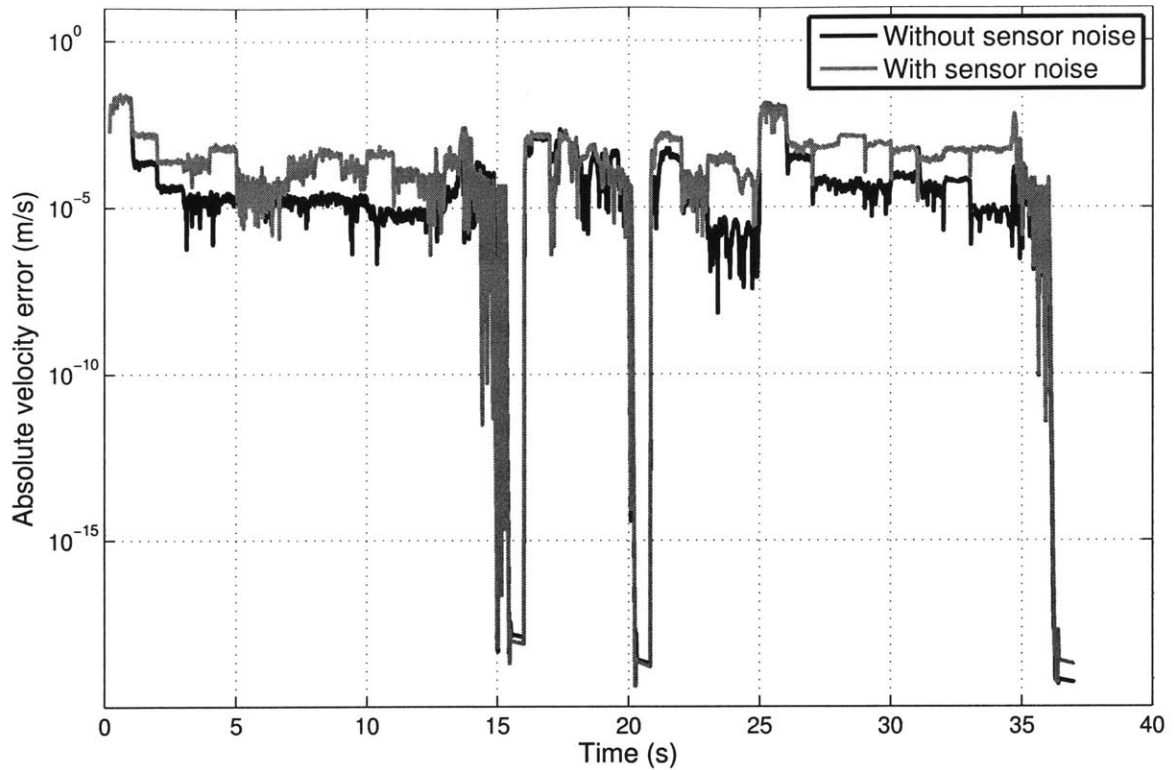
**Figure 2.17. Evolution of the tire parameters using adaptation.**

In practice, robot sensors will introduce noise to the measured velocity and acceleration used for the algorithm. To test the algorithm under more realistic conditions,

an additional test was performed by adding generated sensor noise to the “measurements”. Zero mean white noise with variance of  $0.001 \text{ m/s}^2$  was added to the acceleration “measurements” and zero mean  $0.05 \text{ m/s}$  RMS noise was added to the GPS velocity “measurements”. These values were taken from the product literature and are roughly equivalent to noise seen in practice on mid-grade commercial sensors. Figure 2.18 compares the velocity estimation error with adaptation, with and without sensor noise. Table 2.6 compares three measures of the estimated velocity error for the three cases of no adaptation ( $\bar{\theta} = \bar{\theta}_{start}$ ), adaptation without sensor noise, and adaptation with simulated sensor noise. The velocity error is the difference between the particular velocity estimate and the true vehicle speed. The percentages given for the two adaptive cases compare the error with and without adaptation. As expected, the presence of sensor noise causes an increase in the velocity error, however the adaptation algorithm still significantly improves the velocity estimate over the baseline approach. Note that error values are dependent on how close the initial parameter estimate is to the true value and on the degree of terrain variation.

**Table 2.6. Velocity errors using adaptation compared with error using  $\bar{\theta} = \bar{\theta}_{start}$ .**

	RMS $(v_{bx} - \hat{v}_{bx})$	Median $ v_{bx} - \hat{v}_{bx} $	Mean $ v_{bx} - \hat{v}_{bx} $
No adaptation	0.0087	0.0062	0.0065
Adaptation, no noise	0.0030 (34%)	0.000023 (0.4%)	0.00071 (11%)
Adaptation, w/ noise	0.0031 (36%)	0.00029 (4.7%)	0.0010 (15%)



**Figure 2.18. Adapted velocity estimate error with and without sensor noise.**

### 2.5.5 Conclusions

In this section, a tire model parameter adaptation algorithm has been proposed. Preliminary simulations show that the algorithm increases velocity estimation accuracy, adapts toward the “true” tire/terrain parameters, and responds rapidly to changes in terrain. Experimental validation of the algorithm is still required; however preliminary results are promising.

## 2.6 Summary and Conclusions

In this chapter, two novel techniques for detecting robot immobilization were presented. The dynamic model-based approach presented in Section 2.2 estimates the vehicle speed at all time and thus has additional utility for position estimation. This technique requires identification of a small number of vehicle traction parameters,

although it is shown that for immobilization detection, engineering estimates of the parameters are sufficient. Section 2.5 proposes a preliminary parameter adaptation algorithm which may allow for increased slip estimation accuracy over highly variable terrain. An alternate immobilization detection algorithm was presented in Section 2.3, which uses a signal recognition based approach. This model-free technique does not require identification of vehicle traction parameters, however it does require *a priori* classifier training. Additionally, this approach only provides discrete outputs of “immobilized” or “normal”, rather than a continuous slip estimate as provided by the more complex model-based approach. In Section 2.4 two detector fusion techniques are proposed which were shown to increase detection accuracy and speed.

This chapter showed that common robot sensors can be used to effectively estimate robot state information beyond their intended use. The common approach to the methods presented is that they considered the underlying vehicle dynamics of a robot traveling on outdoor terrain in order to find a solution.

# 3

## **Chapter 3: SPEED INDEPENDENT VIBRATION- BASED TERRAIN CLASSIFICATION**

### **3.1 Introduction**

Terrain characteristics can have a significant effect on vehicle handling, ride quality, and stability. Terrain surface properties determine tire traction properties, with large implications for longitudinal and lateral wheel slip. Additionally, terrain roughness has the effect of varying the normal force acting on a tire, which in turn affects vehicle characteristics [7]. For low-speed robots, terrain traction properties are of primary concern [30], however for high speed vehicles, dynamic terrain effects must also be considered. Classification of the terrain traversed by a vehicle can provide useful information in many scenarios. Knowledge of terrain type can dictate the range of possible maneuvers in hazard avoidance or road departure situations, and provide feedback for tuning of traction control and suspension properties.

In this work we employ tactile sensing to classify terrain. Tactile sensors are those that directly interface with the environment, such as a sensor which measures the vehicle vibrations due to the tire ground interaction. Such sensors can provide information unavailable to visual sensors, such as detecting a hard load-bearing surface under a thin surface material. Brooks and Iagnemma have previously proposed a wheel

vibration-based approach to terrain classification suitable for low speed planetary exploration rovers with rigid wheels [9],[10]. Additionally, similar algorithms have explored alternate sensor modalities, such as a body mounted IMU, for low-speed terrestrial robots with pneumatic tires [16],[41]. None of these methods, however consider the effect of vehicle speed on terrain vibration signature.

This chapter introduces an improved version of the vibration-based terrain classification algorithm proposed by Brooks and Iagnemma [9],[10] for high speed motion. The Brooks algorithm was developed for low-speed planetary exploration rovers with rigid metallic wheels and demonstrated very good performance in its intended application. Terrestrial motor vehicles, however, experience a much larger range of velocities. Additionally, rubber pneumatic tires dampen out many of the high frequency vibrations exploited in the Brooks algorithm. The algorithm presented here enhances the Brooks algorithm through the use of a dynamic model that explicitly accounts for the effects of speed variations on vibration signature.

## **3.2 Algorithm Overview**

### **3.2.1 Description of Existing Terrain Classification Algorithm**

The vibration-based terrain classification algorithm proposed by Brooks takes a signal recognition approach to classify wheel vibration time-series data. A flowchart of the Brooks algorithm is shown in Figure 3.1. A classifier is first trained offline using labeled training data to recognize the vibration signatures created by driving over different terrains. The trained classifier is then utilized for online classification of unknown terrain. In the Brooks algorithm, vibration data is recorded from a wheel

mounted accelerometer at a fixed sampling rate (44.1 kHz). Prior to classification, the vibration data is then broken into multiple training or testing instances of a specified number of samples, such that each instance comprises a fixed length of time of data. The power spectral density (PSD) is calculated for each instance and then the frequency components are log scaled. To reduce the dimensionality of the problem, only the first  $k$  principal components of the PSD are then used to train the classification algorithm. The same principal components are calculated and used during online classification.

The Brooks algorithm was developed for low-speed planetary rovers with rigid metallic wheels with grousers (Figure 3.2). In this application, the dominant wheel vibrations are primarily due to the direct interaction between the grousers and the terrain material, and are of relatively high frequency. At low speeds, dynamic interactions between the wheel and the terrain geometry are minimal. For example, if the rover drives over a rock, the wheel will most likely remain in contact with the rock during traversal, as opposed to a higher speed vehicle which might “jump” off the rock, inducing vibrations due to suspension dynamics.



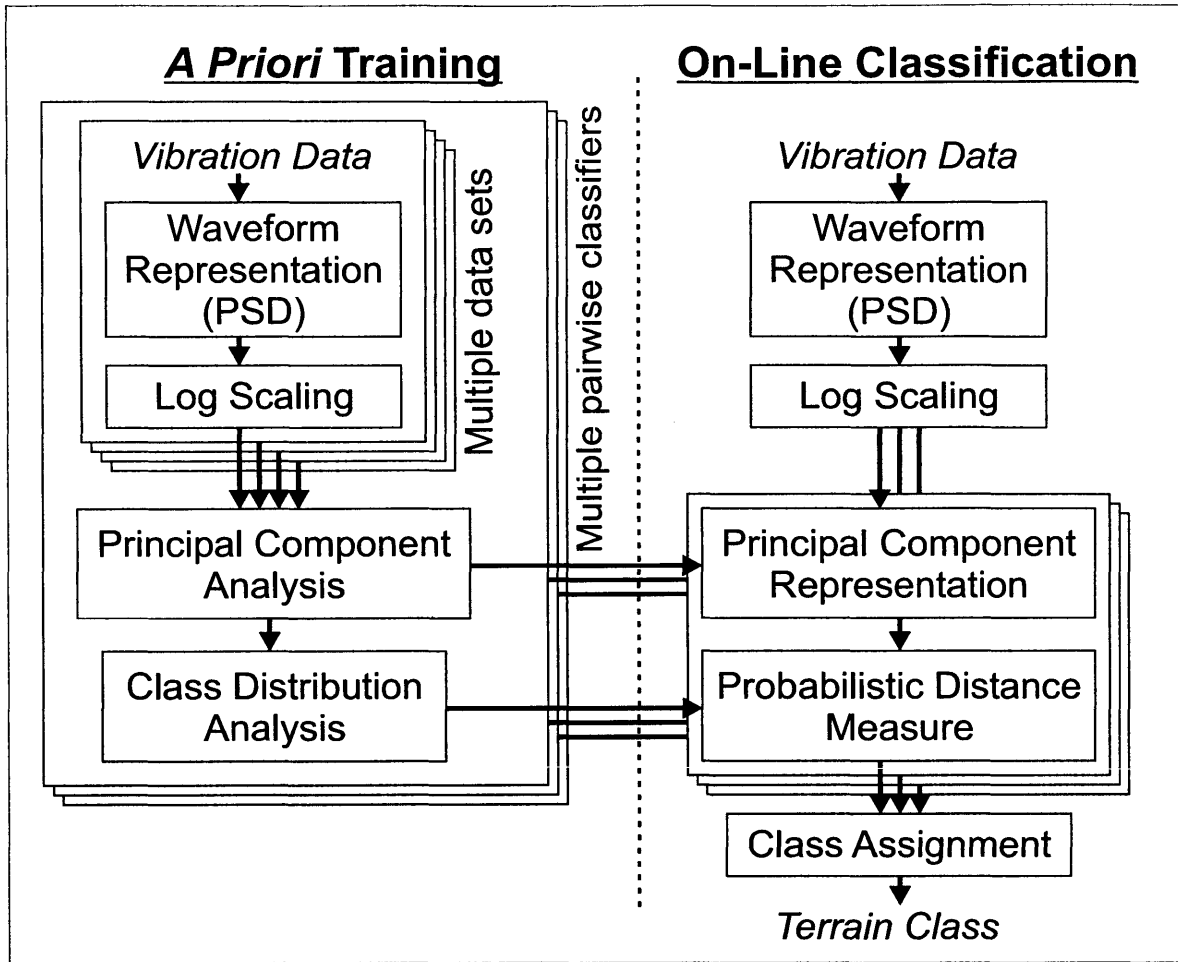
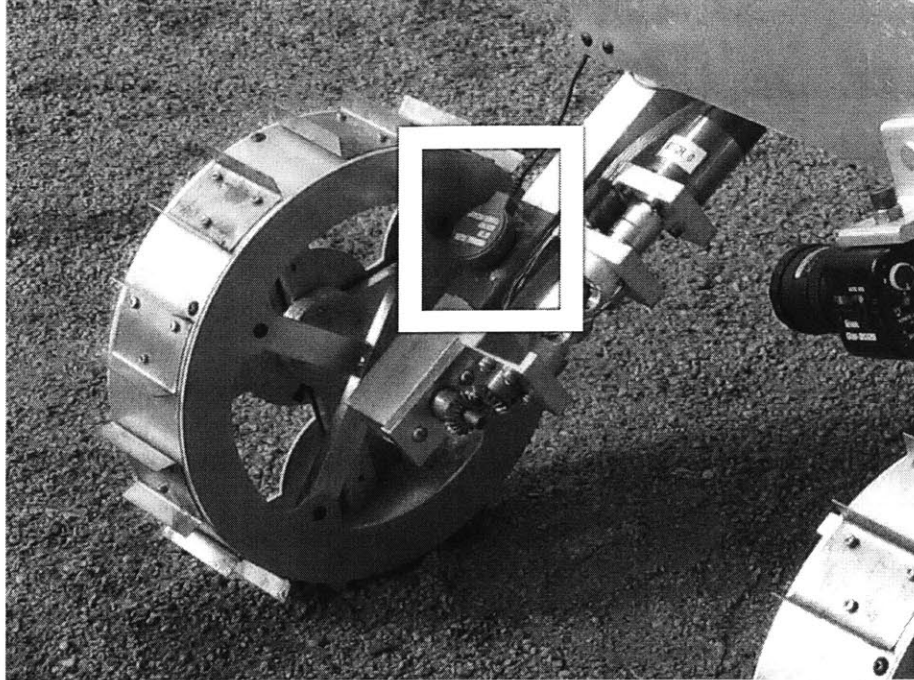


Figure 3.1. Flowchart for vibration-based terrain classification algorithm as proposed by Brooks [9].



**Figure 3.2. Metal wheel with grousers used to test the terrain classification algorithm proposed by Brooks. Box shows location of the vibration sensor.**

### **3.2.2 Enhanced Algorithm for Dynamic Vehicles**

For vehicles with rubber pneumatic tires, high frequency vibrations due to low-speed interaction between the tire and the terrain material, such as those utilized by the Brooks algorithm, are likely to be damped by the tires and vehicle suspension. At high speeds, terrain geometry will exert measurable accelerations on the wheel, and these relatively low-frequency effects are typically the dominant vibrations. Terrain material does still affect the measurable vibrations, since the deformability of the terrain can modulate the impact of terrain geometry on wheel acceleration.

Varying vehicle speed over a given terrain profile has two primary effects on the measurable wheel accelerations: modulating frequency and amplitude. The temporal frequency of the wheel accelerations is directly proportional to the vehicle speed. For example, a vehicle driving over a bumpy road will impact each bump quicker as the vehicle speed increases, increasing the frequency of the measured accelerations. The

amplitude of the wheel accelerations is also related to the vehicle speed. In general, driving over a bump at higher speed produces larger vertical wheel accelerations than driving at lower speed, as the wheel must traverse the bump in a shorter time. The amplitude effect is a function of the vehicle suspension and input frequency and will be explored further in Section 3.3.1.

The classification algorithm presented in this thesis explicitly considers the frequency and magnitude effects, resulting in an algorithm that can accurately classify terrain at varying vehicle speeds. The key idea of the modification is that the measurable wheel accelerations are the product of an underlying terrain profile/material combination. If the profile can be estimated, then terrain classification can be performed on this estimate, which is decoupled from the vehicle dynamics.

Figure 3.3 shows a flowchart for the modified algorithm. The measured time-domain wheel acceleration is passed through the inverse of the combined tire and suspension dynamics to produce an estimate of the road profile as a function of time ( $y(t)$ ). Next, the absolute value of the vehicle speed is integrated to estimate the vehicle displacement ( $x(t)$ ), which is combined with the road profile to form an estimate of the terrain elevation versus displacement ( $y(x)$ ). Similar to the Brooks algorithm (though in the spatial, rather than temporal, domain), the profile signal is segmented into instances of constant displacement, broken into spatial frequency components, and then the first  $k$  principal components are used for classification. The algorithm uses a similar approach of training a classifier offline with labeled training data, before performing online classification of unknown terrain.

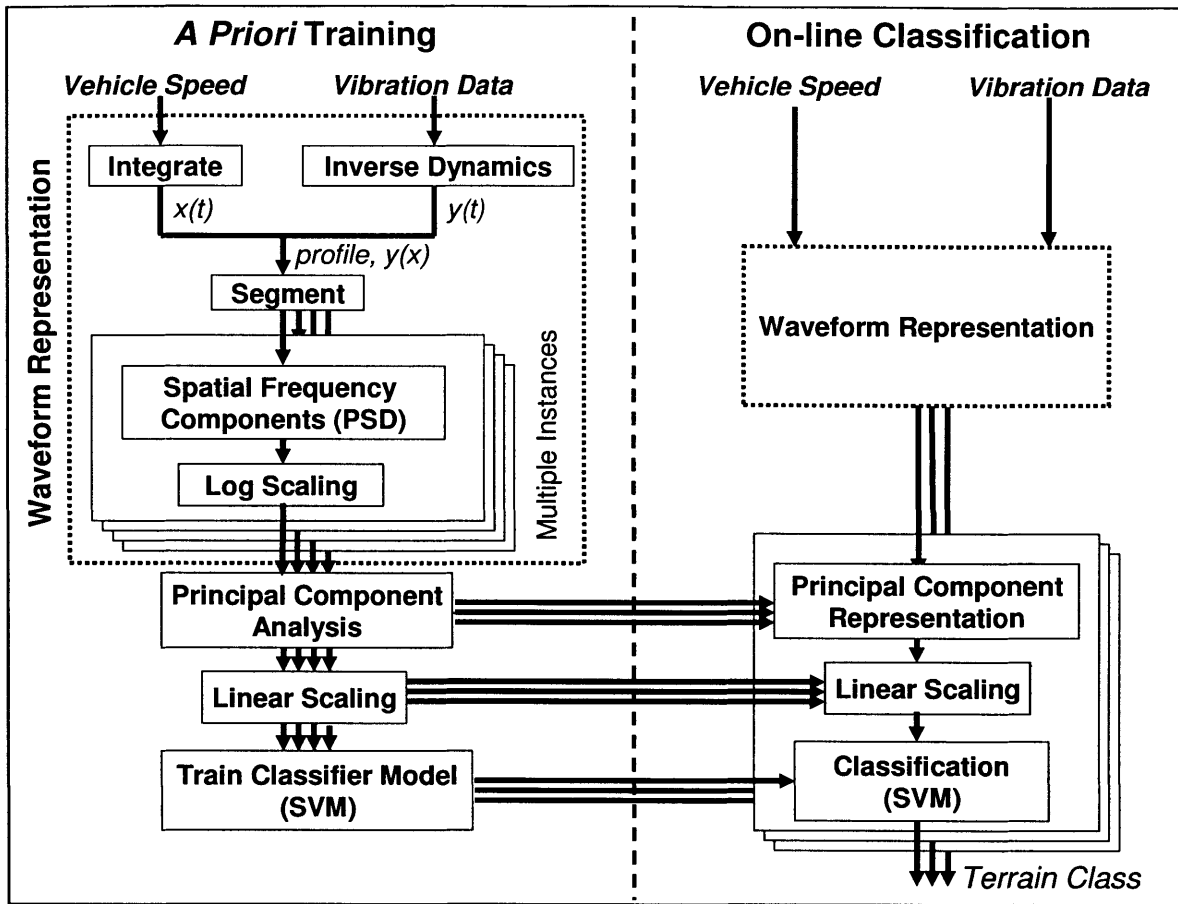


Figure 3.3. Flowchart for modified speed-independent vibration-based terrain classification algorithm presented in this thesis.

### 3.3 Waveform Representation

#### 3.3.1 Profile Estimation

To estimate the terrain surface profile, we calculate the tire-suspension dynamics using a standard quarter-car model [51], shown in Figure 3.4. The quarter-car model represents the dynamics of one wheel of the vehicle attached to the vehicle body through a suspension. The model consists of the “sprung” mass,  $m_s$ , which is one quarter the mass of the vehicle body, and the “unsprung” mass,  $m_{us}$ , which is the mass of a single wheel and attached suspension components. The road height under the tire at time  $t$  is  $y_{road}(t)$ , and the rate of change of the road height,  $\dot{y}_{road}(t)$ , acts as a flow source input to

the rim of the tire. The rubber pneumatic tire is modeled as a spring and damper,  $k_t$ ,  $B_t$ , between the road and the unsprung mass. The vehicle's suspension is modeled as an additional spring and damper,  $k_s$ ,  $B_s$ , between the unsprung and sprung masses. Linear spring-damper models are used in this work, however in general a nonlinear spring-damper could be used if such a model is known for the test vehicle. The vertical velocities of the unsprung and sprung masses are  $v_u$  and  $v_s$ , respectively.  $D_u$  and  $D_s$  are the spring compression distances.

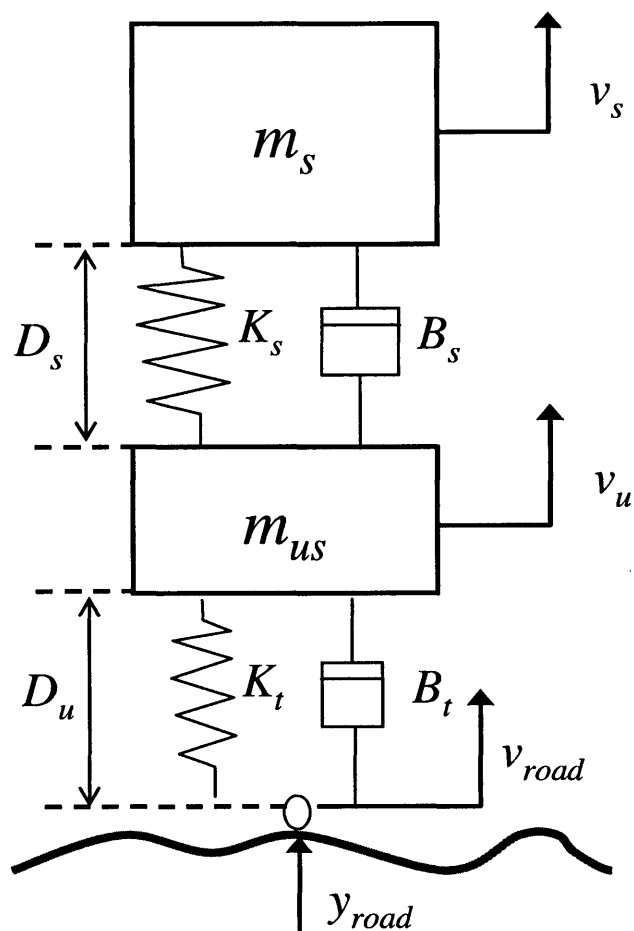


Figure 3.4. Diagram of quarter-car model.

After summing the forces acting on each mass, the following state-space model of the quarter car dynamics can be obtained:

$$\begin{bmatrix} \dot{v}_s \\ \dot{v}_u \\ \dot{D}_s \\ \dot{D}_u \end{bmatrix} = \begin{bmatrix} -\frac{B_s}{m_s} & \frac{B_s}{m_s} & \frac{K_s}{m_s} & 0 \\ \frac{B_s}{m_{us}} & -\frac{B_s+B_t}{m_{us}} & -\frac{K_s}{m_{us}} & \frac{K_t}{m_{us}} \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_s \\ v_u \\ D_s \\ D_u \end{bmatrix} + \begin{bmatrix} 0 \\ B_t \\ 0 \\ 1 \end{bmatrix} \dot{y}_{road} \quad (3.1)$$

From the above state-space model, the following transfer function of the unsprung mass vertical velocity,  $v_u$ , from the road input,  $\dot{y}_{road}$ , is obtained:

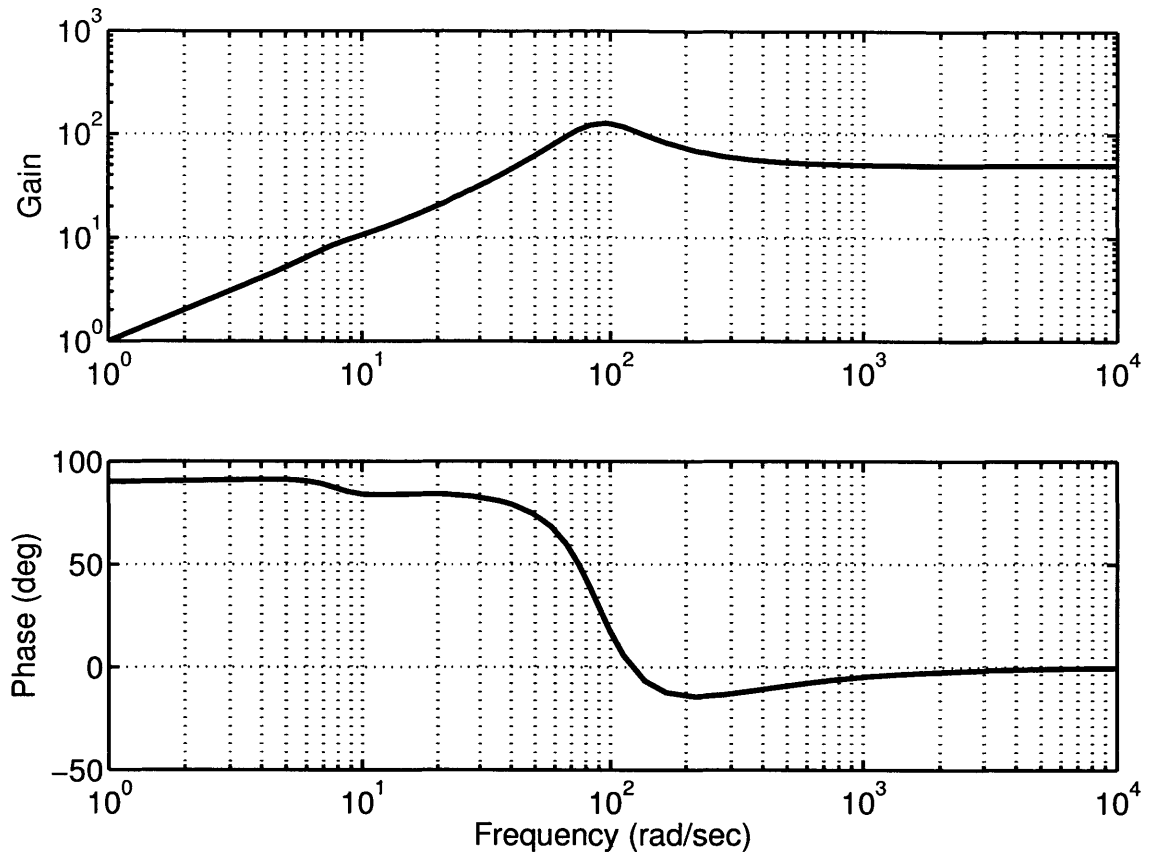
$$\frac{v_u(s)}{\dot{y}_{road}(s)} = \frac{m_s m_{us} B_t s^3 + (m_s K_t + m_{us} B_s B_t) s^2 + (m_{us} B_t K_s + B_s K_t) s + K_s K_t}{m_s m_{us} s^4 + (m_s B_s + m_s B_t + m_{us} B_s) s^3 + (m_s K_s + m_s K_t + B_s B_t + m_{us} K_s) s^2 + (B_s K_t + B_t K_s) s + K_s K_t} \quad (3.2)$$

The above transfer function relates the wheel vertical velocity to changes in the road elevation. However, the wheel vertical acceleration can be more easily measured with low cost sensors than the vertical velocity. In transfer function form, it is simple to find the desired relation for wheel acceleration from road input as:

$$\frac{\dot{v}_u(s)}{\dot{y}_{road}(s)} = \frac{s v_u(s)}{\dot{y}_{road}(s)} = s \left( \frac{v_u(s)}{\dot{y}_{road}(s)} \right) \quad (3.3)$$

Figure 3.5 shows a Bode plot of this transfer function using the parameters for the experimental test vehicle described in Section 3.5.1. The tire and suspension act as a high-pass filter (from road velocity to tire acceleration), with road input frequencies above about 50 radians/sec (8 Hz) significantly affecting wheel accelerations. This is intuitive, as a constant road input velocity would drive the unsprung mass at a constant velocity, thus the acceleration would be zero.

The road input frequency is a function of the wavelength of terrain features and the vehicle speed. Table 3.1 shows terrain wavelengths corresponding to combinations of input frequency and two vehicle speeds representative of city/off-road and highway driving. The table also gives representative feature types for each wavelength, with some type definitions from [42]. At typical automotive driving speeds, road inputs below 10 radians/sec correspond to changes in slope and elevation and do not contain significant information on terrain composition. Additionally, the quarter car model assumes a point contact between the tire and terrain, however a typical automotive tire will have a contact patch length on the order of 10 cm due to deformation of the rubber tire. The contact patch will provide additional damping to high frequency, small wavelength terrain inputs. Thus the bulk of useful terrain information will be contained between 10 to 1,000 radians/sec (1.6 to 160 Hz) at typical automotive speeds.



**Figure 3.5. Bode plot of (3.3), unsprung mass acceleration from terrain profile velocity input.**



**Table 3.1. Terrain wavelengths corresponding to combinations of vehicle speed and road input frequency.**

Profile Input Frequency	Vehicle Speed	Wavelength	Feature types
1 rad/sec (0.16 Hz)	11 m/s (~ 25 mph) 27 m/s (~ 60 mph)	69 m 170 m	Hills / slopes Hills / slopes
10 rad/sec (1.6 Hz)	11 m/s (~ 25 mph) 27 m/s (~ 60 mph)	6.9 m 17 m	Small hills / slopes Small hills / slopes
100 rad/sec (16 Hz)	11 m/s (~ 25 mph) 27 m/s (~ 60 mph)	0.69 m 1.7 m	Very coarse gravel Boulders, speed bumps
1,000 rad/sec (160 Hz)	11 m/s (~ 25 mph) 27 m/s (~ 60 mph)	0.069 m 0.17 m	Coarse gravel Very coarse gravel, cobblestone
10,000 rad/sec (1,600 Hz)	11 m/s (~ 25 mph) 27 m/s (~ 60 mph)	0.007 m 0.017 m	Very fine gravel Fine/medium gravel

The model in (3.3) provides an indication of what terrain signal will be measurable via an accelerometer mounted on the unsprung mass. However, for classification, we wish to estimate the terrain profile, given the measured acceleration. First the transfer function for the unsprung mass acceleration as function of profile height is found as:

$$\frac{\dot{v}_u(s)}{y_{road}(s)} = \dot{v}_u(s) \left( \frac{s}{\dot{y}_{road}(s)} \right) = s^2 \left( \frac{v_u(s)}{\dot{y}_{road}(s)} \right) \quad (3.4)$$

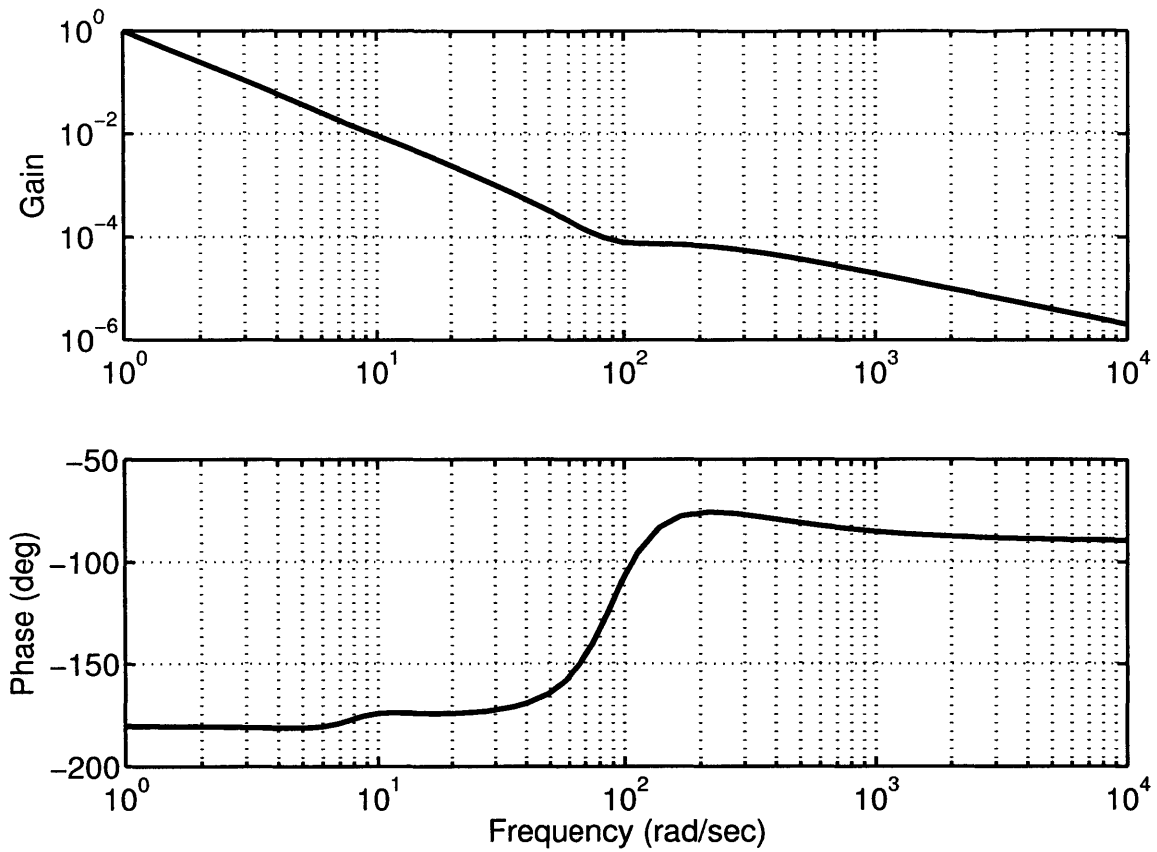
Then the desired transfer function for road profile from unsprung mass acceleration is taken as the inverse of (3.4) such that:

$$\frac{y_{road}(s)}{\dot{v}_u(s)} = \frac{1}{\left(\frac{\dot{v}_u(s)}{y_{road}(s)}\right)} = \frac{1}{s^2 \left(\frac{v_u(s)}{\dot{y}_{road}(s)}\right)} \quad (3.5)$$

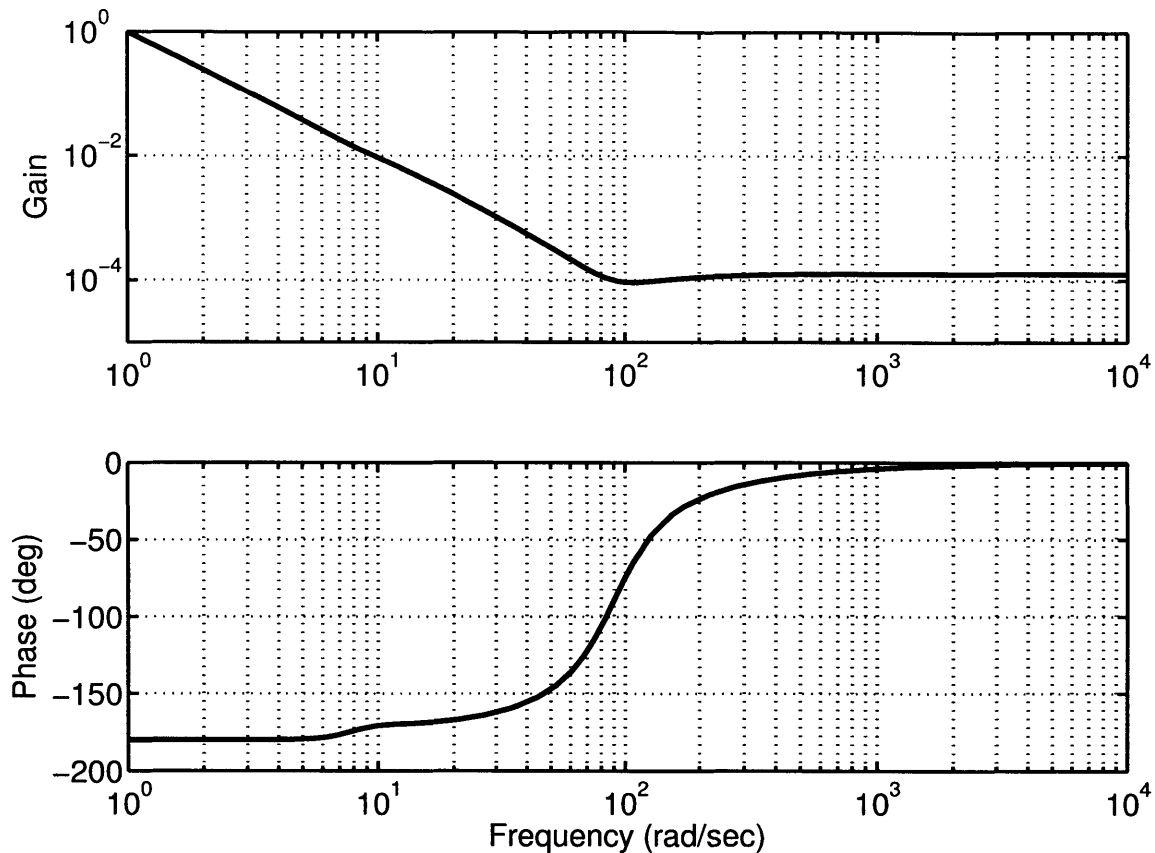
Substituting in (3.2) yields the desired model for estimating road profile from measured vertical wheel acceleration:

$$\frac{y_{road}(s)}{\dot{v}_u(s)} = \frac{m_s m_{us} s^4 + (m_s B_s + m_s B_t + m_{us} B_s) s^3 + (m_s K_s + m_s K_t + B_s B_t + m_{us} K_s) s^2 + (B_s K_t + B_t K_s) s + K_s K_t}{m_s m_{us} B_t s^5 + (m_s K_t + m_{us} B_s B_t) s^4 + (m_{us} B_t K_s + B_s K_t) s^3 + K_s K_t s^2} \quad (3.6)$$

Figure 3.6 shows a Bode plot of (3.6). The response is similar to a double integrator below 6 radians/sec (1 Hz) and a single integrator above 1,000 radians/sec (160 Hz). A constant acceleration measurement at the unsprung mass requires the wheel to be constantly accelerating, such as while free falling; thus the high gain at low frequencies corresponds to a step in the road profile. Typical accelerometers, however, have near constant measurement biases. These biases can be partially estimated and removed, however any uncompensated accelerometer bias can cause large profile deviations, which appear as terrain slopes. For classification, only profile features on the relatively small scale of terrain composition are desired. Large scale features, whether part of the true profile or caused by sensor errors, will be removed before classification. In some cases, damping in the pneumatic tire is not included in quarter-car models. Figure 3.7 shows a bode plot of (3.6) using  $B_t = 0$ . Comparing this with Figure 3.6, we see that the tire damping provides high frequency damping in the transfer function from tire acceleration to road profile. Thus, inclusion of the tire damping has the added effect of reducing the impact of high frequency sensor noise.



**Figure 3.6. Bode plot of (3.6), profile estimate from wheel acceleration input.**



**Figure 3.7. Bode plot of (3.6) with no tire damping ( $B_t = 0$ ). Note constant gain at high frequency.**

To estimate the road profile, a discrete-time transformation of (3.6) is applied to the measured acceleration using standard techniques, after first subtracting the acceleration due to gravity from the measurement. To remove large-scale terrain elevation content a quadratic best fit is performed on a moving data window (A one second window has been used here) and subtracted from the estimated profile, leaving only the relatively high frequency terrain profile information,  $y(t)$ . Some remaining low frequency content will be removed in the spatial domain, after impulses are filtered as discussed in Section 3.3.2. Large-scale terrain elevation removal could also be accomplished using a properly designed high pass filter.

Next the vehicle displacement is estimated from the measured vehicle speed as:

$$x(t) = \int_0^t |\text{vehicle speed}(t)| dt \quad (3.7)$$

Then it is straightforward to restate the profile in spatial, rather than temporal, coordinates as  $\tilde{y}(x)$ .

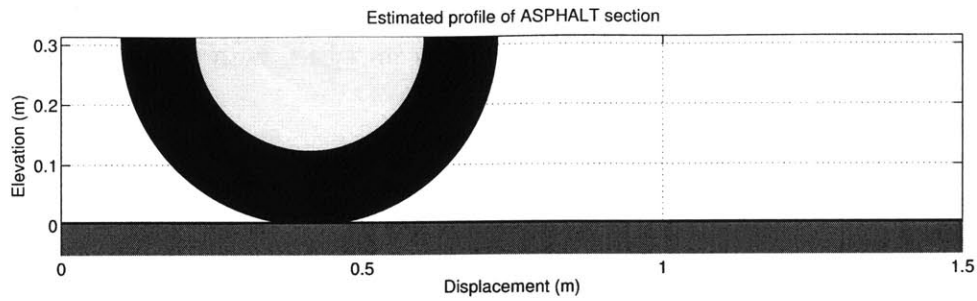
Two additional manipulations are required to produce a profile estimate suitable for classification. When the vehicle is stopped, unique accelerometer data is collected in the temporal domain; however, because vehicle displacement is zero during this time, redundant data points,  $\tilde{y}(x(t_{\text{stopped}}))$ , are created at  $x(t_{\text{stopped}})$ . The  $N$  redundant points are removed and assigned the mean value of the profile each time the vehicle stops such that:

$$\tilde{y}(x(t_{\text{stopped}})) = \frac{1}{N} \sum_{t_{\text{stop}}}^{t_{\text{start}}} \tilde{y}(x) \quad (3.8)$$

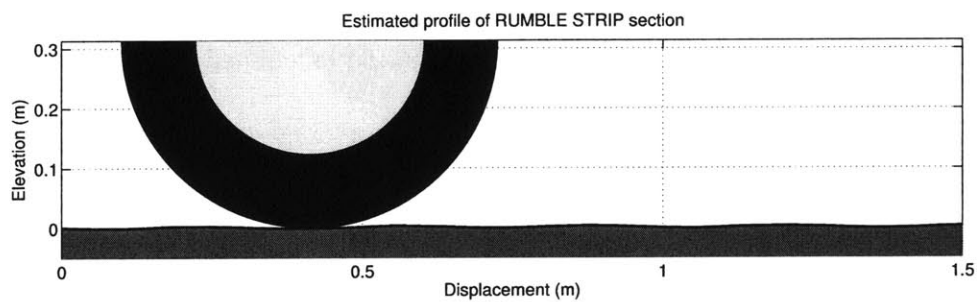
$\tilde{y}(x)$  is the profile estimated at constant temporal sampling frequency. However, for the spatial frequency decomposition performed in Section 3.3.3, the profile must be sampled at constant spatial frequency. The final profile estimate,  $y(x)$ , is obtained by interpolating  $\tilde{y}(x)$  at constant spatial intervals,  $\Delta x$ . A piecewise cubic spline interpolation has been used. The choice of  $\Delta x$  determines what the maximum spatial frequency will be and will be discussed further in Section 3.3.3.

It should be noted that the profile estimate is not expected to exactly match the true road profile, but instead is a representation of the profile as seen by the vehicle. The quarter car model assumes that the tire has point contact with rigid terrain. As mentioned previously, the tire has a finite contact patch due to deformation of the tire. The tire therefore filters some small wavelength terrain features from affecting the vehicle

dynamics. These features are not measurable via the wheel mounted accelerometer and will thus not be included in the profile estimate. Additionally, terrain is not, in general, perfectly rigid. Driving over a speed bump-sized lump of a deformable terrain, such as soil, will excite the suspension dynamics less than driving over a rigid asphalt speed bump. In the deformable case, the profile estimate will be a representation of a rigid profile with the same effective suspension excitation as the true deformable profile. Although the profile estimate is not expected to be exactly accurate, it is believed to capture the effective profile information relevant for suspension excitation. Figure 3.8 show a relatively flat asphalt profile estimate and Figure 3.9 shows a rumble strip profile estimate with a low-frequency, periodic profile. Despite the inherent inaccuracies, the profile estimates appear qualitatively correct.



**Figure 3.8. Estimated asphalt profile. Tire shown to scale.**



**Figure 3.9. Estimated rumble strip profile. Tire shown to scale.**

### 3.3.2 Impulse Detection and Removal

In Section 3.3.1 the road profile,  $y(x)$ , was estimated. Terrain-type classification will be performed on the estimated profile after one additional pre-filtering step. The profile of terrain impulses, such as potholes in streets, is not representative of the profile of the underlying terrain type. Therefore, impulses are detected and removed from the profile estimate before performing classification. Knowledge of impulse locations may be useful and this information can be retained to be processed separately from the terrain classification algorithm.

The impulse detection developed for this algorithm is based upon the following idea: variation (standard deviation) in a road profile should be relatively constant over a given terrain type. An impulse in the terrain will create a sharp, brief increase in the

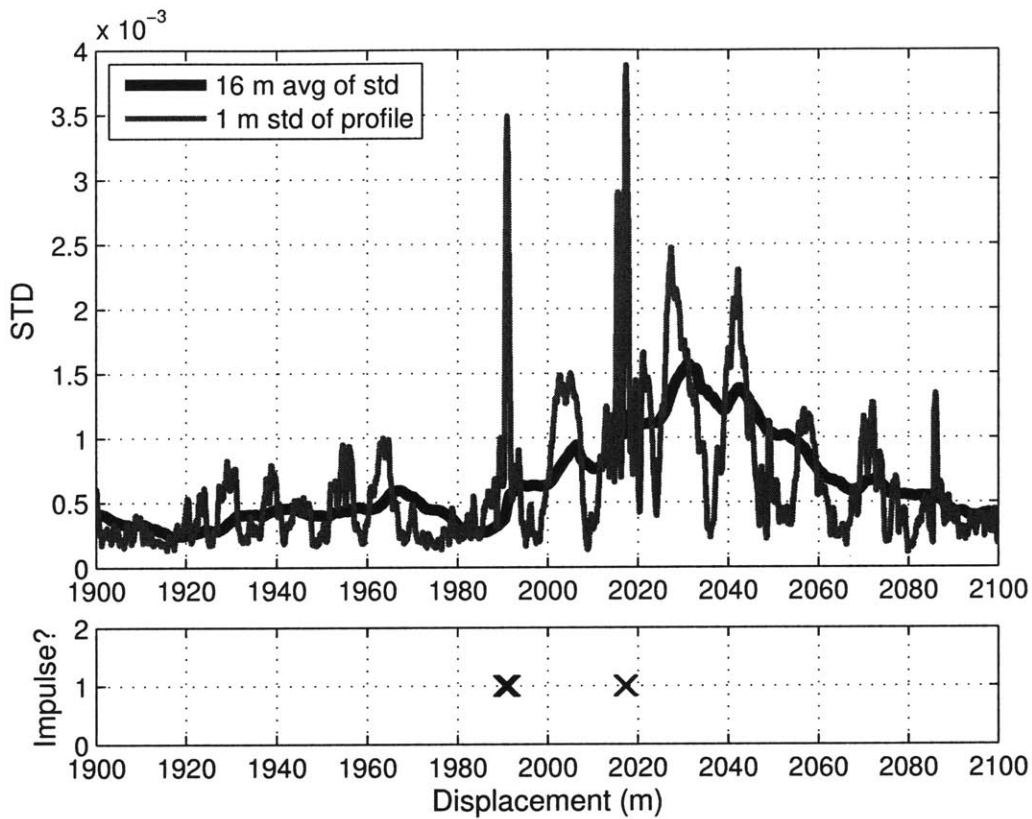
profile variation, whereas changes in terrain provide relatively long term changes. The following process has been developed to identify impulses, with values adopted for this work given in parentheses:

1. Calculate the standard deviation of the road height over a short moving window (1 meter).
2. Calculate a moving average of the standard deviation calculated above, over a longer window (16 meters).
3. Check if the short-scale standard deviation calculated in 1) is greater than a specified multiple (3x) of the large-scale average calculated in 2). If larger, label the point as an impulse.
4. For each detected impulse, “grow” the impulse back by a set distance (0.2 meters). There is a delay in the detection of the impulses due to 1) being calculated over a window and because the spike must exceed a threshold before the impulse is detected. Growing the labeled section back accounts for this delay. Since the goal is to remove the data including the impulse before classification, the growth should err on the large side.

Figure 3.10 shows an example of the impulse detection from the experimental data. The bottom plot shows the points where impulses have been detected. Analysis of driving video has shown that impulses are correctly identified using this detection scheme when the vehicle hits sharp bumps or dips in the terrain. It should be noted that this impulse detection scheme may initially label transitions from relatively smooth to rough terrains as an impulse, however the impulse will only be detected briefly at the initial



terrain transition point because the standard deviation is compared to a moving average of itself.



**Figure 3.10. Impulse detection example.**

After impulses are detected, they are removed from the data used for classification. Figure 3.11 illustrates the impulse removal process. The procedure begins with a profile estimate vector,  $y(k)$ , where  $k$  is the index of the terrain height vector developed in Section 3.3.1, at uniformly spaced displacements  $x(k)$ . Impulses are identified using the above procedure, with a given impulse having “impulse start” index  $k_{is}$  and “impulse end” index  $k_{ie}$ . The impulse is removed (Figure 3.11.c) by eliminating  $y(k_{is}+1)$  through  $y(k_{ie}+1)$  to form the truncated profile vector  $y'$  where:

$$\mathbf{y}' = [\dots, \mathbf{y}(k_{-2}), \mathbf{y}(k_{-1}), \mathbf{y}(k_{is}), \mathbf{y}(k_1), \mathbf{y}(k_2), \dots]. \quad (3.9)$$

Finally the profile estimate following the removed impulse is adjusted to be zeroth-order continuous (Figure 3.11.c):

$$\mathbf{y}''(k) = \begin{cases} \mathbf{y}'(k) & \text{for } \forall k \leq k_{is} \\ \mathbf{y}'(k) + (\mathbf{y}(k_{is}) - \mathbf{y}(k_{ie})) & \text{for } \forall k > k_{is} \end{cases} \quad (3.10)$$

The above procedure is repeated to remove all detected impulses from the profile vector before classification.

After the impulses are identified and removed, a moving average is subtracted from the estimated profile to eliminate any remaining DC offset and low frequency terrain effects. This step should be performed even if impulses are not removed. In this work a moving average window of 10.24 meters (512 instances) has been used. For notational convenience, the profile vector in the following sections will be referred to as  $\mathbf{y}$ , without the double prime notation. Unless otherwise noted, it is assumed in future sections that impulses have been removed from the profile estimate vector such that  $\mathbf{y} = \mathbf{y}''$ .

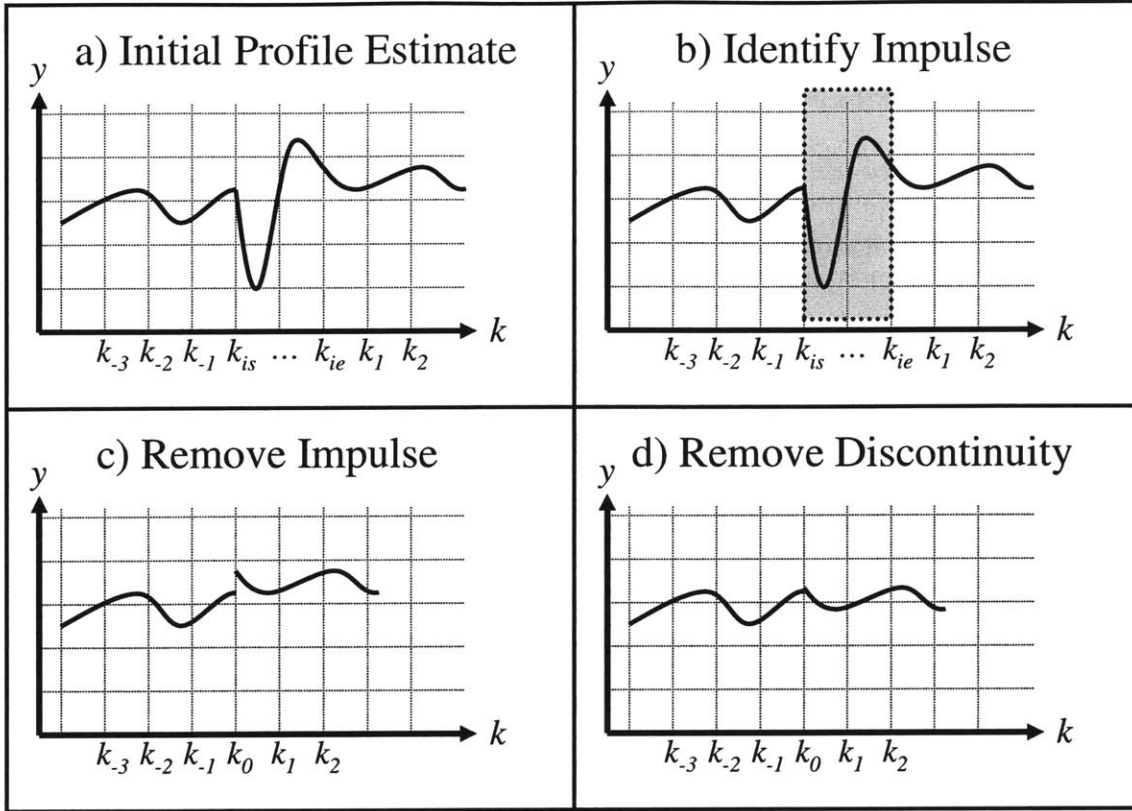


Figure 3.11. Illustration of impulse removal process.

### 3.3.3 Spatial Frequency Components

Next the profile estimate is broken into  $l$  short segments of length  $L$ .  $L$  is chosen as a compromise between two competing requirements.  $L$  sets the classification resolution; a terrain patch should be at least  $L$  long to be correctly classified, making a smaller value of  $L$  desired.  $L$  also determines the resolution of spatial frequencies the signal can be decomposed to using a Fourier transform, where:

$$\Delta f_{\min} = \frac{f_s}{\text{numPoints}} = \left( \frac{1}{\Delta x} \right) \left( \frac{\Delta x}{L} \right) = \frac{1}{L} \quad (3.11)$$

where  $\Delta x$ , which cancels in this equation, is the spatial spacing of the profile data points from Section 3.3.1. Small  $\Delta f_{\min}$  is desired to extract the maximum information for classification from the data. For this work,  $L$  was chosen as 4 meters, which allows

classification of terrain patches on the order of one car length and a spatial frequency resolution of 0.25 cycles/m.

The choice of  $\Delta x$  in Section 3.3.1 is also determined by frequency domain considerations. The spatial sampling frequency is:

$$f_s = \left( \frac{1}{\Delta x} \right). \quad (3.12)$$

And the Nyquist frequency is then:

$$f_{Nyquist} = \frac{f_s}{2} = \left( \frac{1}{2\Delta x} \right). \quad (3.13)$$

The Nyquist frequency is the highest spatial frequency that can be resolved from the profile data and therefore corresponds to the minimum feature size which can be distinguished from the terrain. However, it does not make sense to choose  $\Delta x$  significantly smaller than the distance traversed by the vehicle in one temporal sampling interval traveling at normal operating speeds. In this work,  $\Delta x$  was chosen as 0.02 meters.

The spatial power spectral density (PSD) of each of the terrain profile segments is calculated using Welch's method [50]. The PSD is then log-scaled. We will use the notation  $P_{f_i, y_j}$  to represent the log-scaled power at frequency  $i$  of terrain profile segment  $j$ . The frequencies range from  $f_{min} = 0$  to  $f_{max} = f_{Nyquist}$ . To reiterate, the frequency components are spatial frequency components of the estimated road profile in units of cycles/meter, which are independent of the vehicle speed. These frequency components will be utilized in the following classification algorithm.

### **3.4 Classification Algorithm**

The remainder of the terrain classification algorithm, described below, closely follows the approach suggested by Brooks [9],[10], except using the profile spatial frequency components, rather than vibration temporal frequency components. Additionally a support vector machine (SVM) classifier is used for speed and convenience, rather than the Fisher linear discriminant analysis based classifier utilized by Brooks. The classification algorithm begins with offline, *a priori*, training of the classifier using labeled training data of desired terrain types, followed by online classification of unknown terrain.

#### **3.4.1 A Priori Training**

The purpose of *a priori* training is to teach the classifier to recognize the profile signatures of different desired terrain types. Data is collected by driving over known terrain types. In theory the driving speed for the training data can be arbitrary, however in practice improved results are likely achieved by collecting data at the range of expected driving speeds, due to unmodeled nonlinearities in the vehicle suspension. The training data collected for this work was collected during normal driving at non-constant speeds, which did not necessarily correspond to testing data driving speed.

Data is collected using an accelerometer mounted to the vehicle suspension with its axis inline with the direction of suspension travel. Measured unsprung mass vertical accelerations from the entire training data set are converted to a profile estimate, impulses are removed, and the profile is broken into segments which are decomposed into spatial frequency components as discussed in Section 3.3.3. For training, the profile is broken

into  $l$  non-overlapping segments such that all data is only used once. The frequency components of each terrain segment are then placed in a matrix,  $Y$ , as:

$$Y = \begin{bmatrix} P_{f_{min}, \mathcal{Y}_1} & \cdots & P_{f_{min}, \mathcal{Y}_l} \\ \vdots & \ddots & \vdots \\ P_{f_{max}, \mathcal{Y}_1} & \cdots & P_{f_{max}, \mathcal{Y}_l} \end{bmatrix}. \quad (3.14)$$

In this form, each column of  $Y$  contains the frequency components of a single terrain segment.  $Y$  has a large number of frequency components which represents an unwieldy feature space for efficient classification. To reduce the dimensionality, principal component analysis is used to form a smaller set of components which contain the most signal to perform classification on.

First the rows of  $Y$  are mean-adjusted to form the matrix  $\hat{Y}$ :

$$\hat{Y} = Y - \begin{bmatrix} \text{mean}(P_{f_{min}}) \\ \vdots \\ \text{mean}(P_{f_{max}}) \end{bmatrix} [1 \cdots 1]. \quad (3.15)$$

Singular value decomposition [9],[22] is next used to separate  $\hat{Y}$  into three matrices,  $U_a$ ,  $S_a$ , and  $V_a$ , such that:

$$\hat{Y} = U_a S_a V_a^T. \quad (3.16)$$

$U_a$  is a unitary matrix with the principal components of  $\hat{Y}$  as columns,  $S_a$  is a diagonal matrix of singular values, and  $V_a$  is a unitary matrix with the principal components of  $\hat{Y}^T$  as columns. To reduce the dimensionality of the classification problem, only the

principal components containing the majority of the signal variation are desired. To derive the first  $n$  principal components, first the matrix  $U_{signal}$  is formed from the first  $n$  columns of  $U_a$ , and  $S_{signal}$  is formed from the upper-left  $n \times n$  block of  $S_a$ . Choosing too high a value for  $n$ , can result in overtraining the classifier to recognize noise in the training data and decrease the performance when classifying new data [9]. For this work, the first ten principal components have been used ( $n = 10$ ). Finally the principal components of the terrain profile spatial frequency content are calculated as:

$$\mathbf{W}_{training} = \mathbf{S}_{signal}^{-1} \mathbf{U}_{signal}^T \mathbf{Y}. \quad (3.17)$$

$\mathbf{W}_{training}$  will be an  $n \times l$  matrix of the form:

$$\mathbf{W}_{training} = \begin{bmatrix} PC_{1,y_1} & \cdots & PC_{1,y_l} \\ \vdots & \ddots & \vdots \\ PC_{n,y_1} & \cdots & PC_{n,y_l} \end{bmatrix}. \quad (3.18)$$

Terrain classification will be performed using an SVM classifier, as used in Section 2.3.2 for immobilization classification. Each distinct terrain type is assigned a unique positive integer label. And an  $l \times 1$  training label vector,  $\mathbf{c} = [c_1 \cdots c_l]^T$ , is formed using the known terrain labels for the training data. The SVM is then trained using the same procedure discussed in Section 2.3.2 for choosing SVM parameters and linearly scaling the features.  $\mathbf{W}_{training}$  is used for the feature-instance matrix,  $\mathbf{X}$ . The principal component vectors (defined by  $\mathbf{S}_{signal}^{-1} \mathbf{U}_{signal}^T$ ), feature scale factors, and trained SVM model are retained for online classification.

### 3.4.2 Online Classification

During online classification, wheel accelerometer data is collected at the same rate as for the training data. The accelerometer data is converted to a profile estimate as discussed above. For this work, the terrain is classified each time the vehicle travels over a unique terrain patch of length  $L$ . In practice, the classified terrain patches can overlap for increased terrain transition detection resolution (In general, a moving patch of length  $L$  will include greater than 50% of a new terrain type before a sequential, non-overlapping patch).

For each profile segment, the PSD is calculated, and the log-scaled elements are placed in the vector:

$$\mathbf{Y}_{test} = \begin{bmatrix} P_{f_{min}, y_{test}} \\ \vdots \\ P_{f_{max}, y_{test}} \end{bmatrix}. \quad (3.19)$$

And the principal components calculated for the training data are calculated for the online segment using the same transformation matrices,  $\mathbf{S}_{signal}$  and  $\mathbf{U}_{signal}$ :



$$W_{test} = S_{signal}^{-1} U_{signal}^T Y_{test} \quad (3.20)$$

$$W_{test} = \begin{bmatrix} PC_{1,y_{test}} \\ \vdots \\ PC_{n,y_{test}} \end{bmatrix} \quad (3.21)$$

## 3.5 Experimental Results

### 3.5.1 Experimental Platform

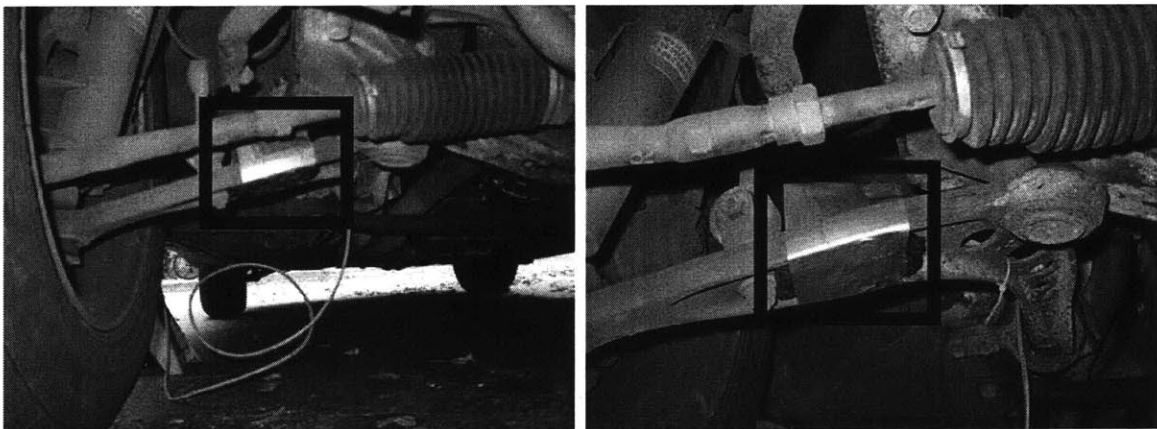
A standard coupe-style passenger vehicle, a 1994 BMW 325is, has been instrumented to experimentally validate the terrain classification algorithm presented in this work (Figure 3.12). Approximate quarter-car model parameters have been identified for this vehicle and are listed in Table 3.2. An Analog Devices ADXL320 dual-axis accelerometer has been mounted to the suspension as shown in Figure 3.13, with one axis aligned with the vertical suspension travel. The accelerometer has a dynamic range of  $\pm 5.0$  g's and is low pass filtered at 250 Hz, yielding a 5 mg resolution. The accelerometer is sampled at 512 Hz using a PC104-based data logging system equipped with a Diamond Systems Diamond-MM-32-AT data acquisition board with a 16-bit A/D resolution. The electronics package (Figure 3.14) is also equipped with a Novatel ProPak-G2*plus* GPS receiver used to measure vehicle speed and a Crossbow AHRS400 inertial measurement unit (IMU) to measure vehicle tilt and acceleration. The IMU-based vehicle acceleration measurements were used in a simple Kalman filter to augment the GPS velocity measurements between samples and during periods of poor GPS reception.



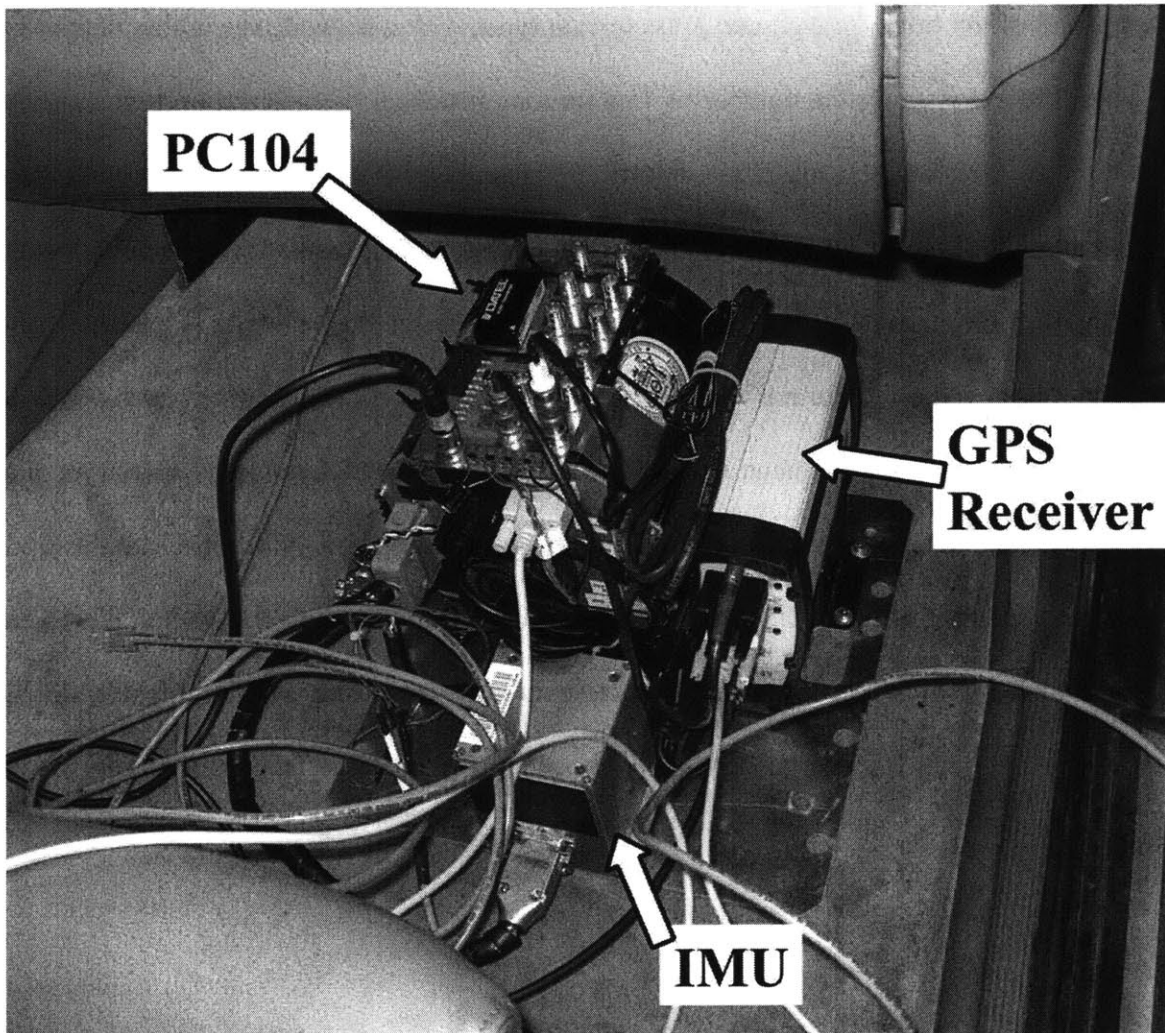
**Figure 3.12. The experimental vehicle.**

**Table 3.2. Quarter car model parameters used for experimental vehicle.**

Parameter	Value
$m_s$	325 kg
$m_{us}$	32.5 kg
$k_s$	22.22 kN/m
$k_t$	254.8 kN/m
$B_s$	2,250 kg/s
$B_t$	50 kg/s



**Figure 3.13. Accelerometer mounting location on experimental vehicle suspension.**

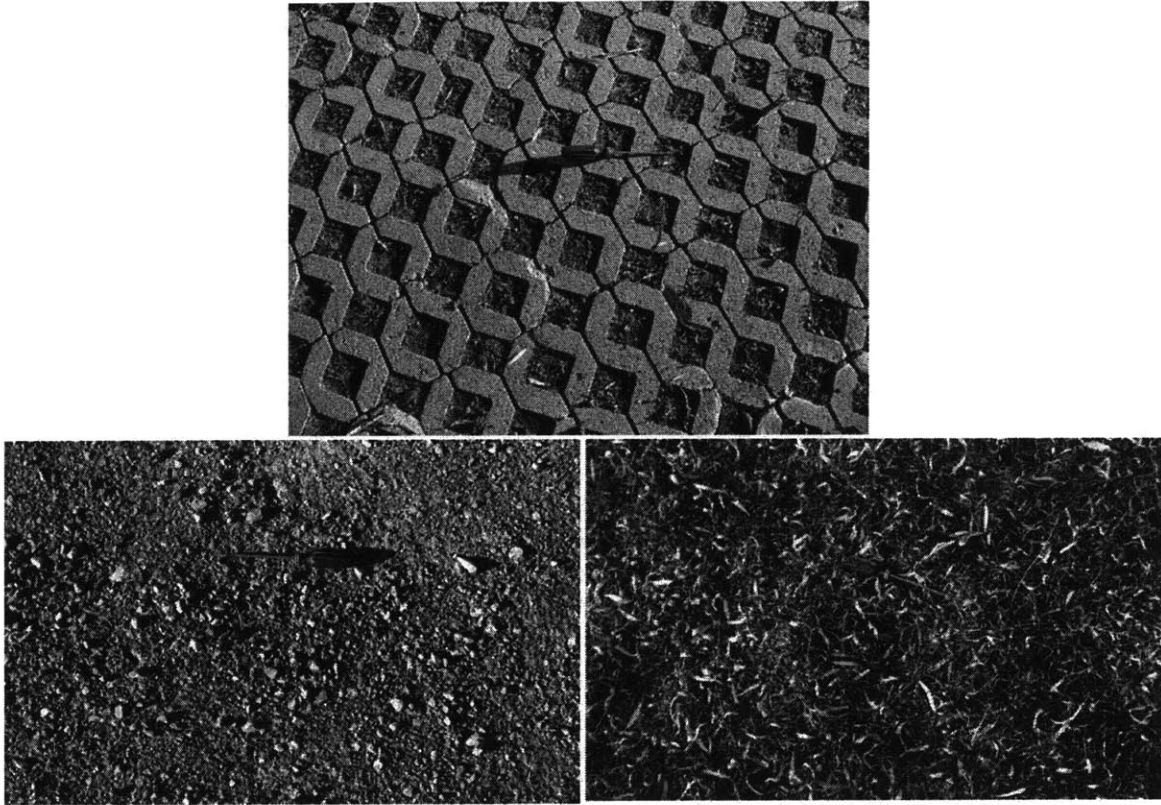


**Figure 3.14.** The experimental electronics package.

### **3.5.2 Results**

The experimental vehicle was driven over ~12 km of real-world driving conditions near Middlesex Fells, Massachusetts, in fair weather. The driving was on public roads and included many turns, starts, and stops. The vehicle was driven over a large variety of asphalt road surfaces including smooth highways, town roads, rough parking lots, and numerous potholes. Additionally the vehicle was driven on a brick road, a gravel parking lot, a highway rumble strip, and a grass shoulder. Figure 3.15 shows

examples of the brick, gravel, and grass terrain types with a screwdriver in the picture for scale. Table 3.3 shows the number of 4 meter long instances of each terrain type used for training and testing the classifier algorithm. Due to the small amount of grass data available, the classifier was not trained to detect grass. The grass data was included when testing the algorithm to give an example of how an untrained terrain type is handled. The vehicle was driven at “normal” speeds ranging from 0 to 104 km/hr (0 to 29 m/s). The training data had a spatial mean vehicle speed of 41 km/hr (11.4 m/s) and spatial median of 21 km/hr (5.8 m/s), while the testing data had a spatial mean speed of 57 km/hr (15.8 m/s) and spatial median of 55 km/hr (15.3 m/s). Both the training and testing data were hand-labeled with the “true” terrain type, to compare against the algorithm result. Small errors in the hand labeling are possible, and thus perfect classification accuracy should not be expected.



**Figure 3.15. Photos of brick, gravel, and grass terrains.**

**Table 3.3. Number of 4 meter long training and testing instances by terrain type.**

<b>Terrain Type</b>	<b>Training Instances</b>	<b>Testing Instances</b>
Asphalt	258	2200
Brick	68	52
Gravel	106	177
Rumble Strip	34	35
Grass	0	7
<b>Total Instances</b>	466	2471
<b>Total Distance</b>	1.9 km	9.9 km

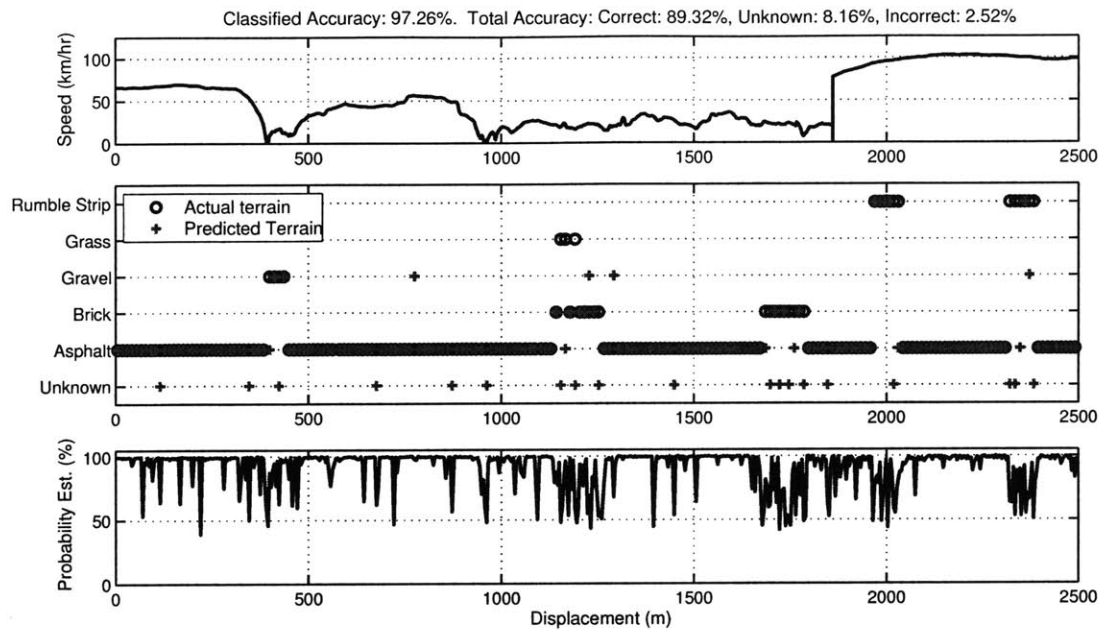
The proposed classification algorithm was run on the ~10 km of testing data. Table 3.4 shows the classification results by terrain class with data classified with less than 65% confidence labeled as “unknown.” For all terrain types the majority of data was either labeled correctly, or as “unknown.” 89.3% of data instances were correctly labeled, 8.2% were labeled “unknown,” and only the remaining 2.5% were incorrectly

labeled. Of the labeled data, 97.3% was correctly labeled. Additionally, the majority of the grass data was labeled as unknown, demonstrating the algorithm is reasonably robust to untrained terrain types. Overall the classification results are quite good, particularly considering that besides the grass, each terrain type is a “hard” surface and might be expected to give a similar vibration signature.

Figure 3.16 shows a 2.5 km subset of the 10 km of test results. The top plot shows the vehicle speed, the middle plot shows the actual and predicted terrain types, and the bottom plot shows the probability estimate. For readability, the terrain types in the middle plot have been decimated by a factor of 3.

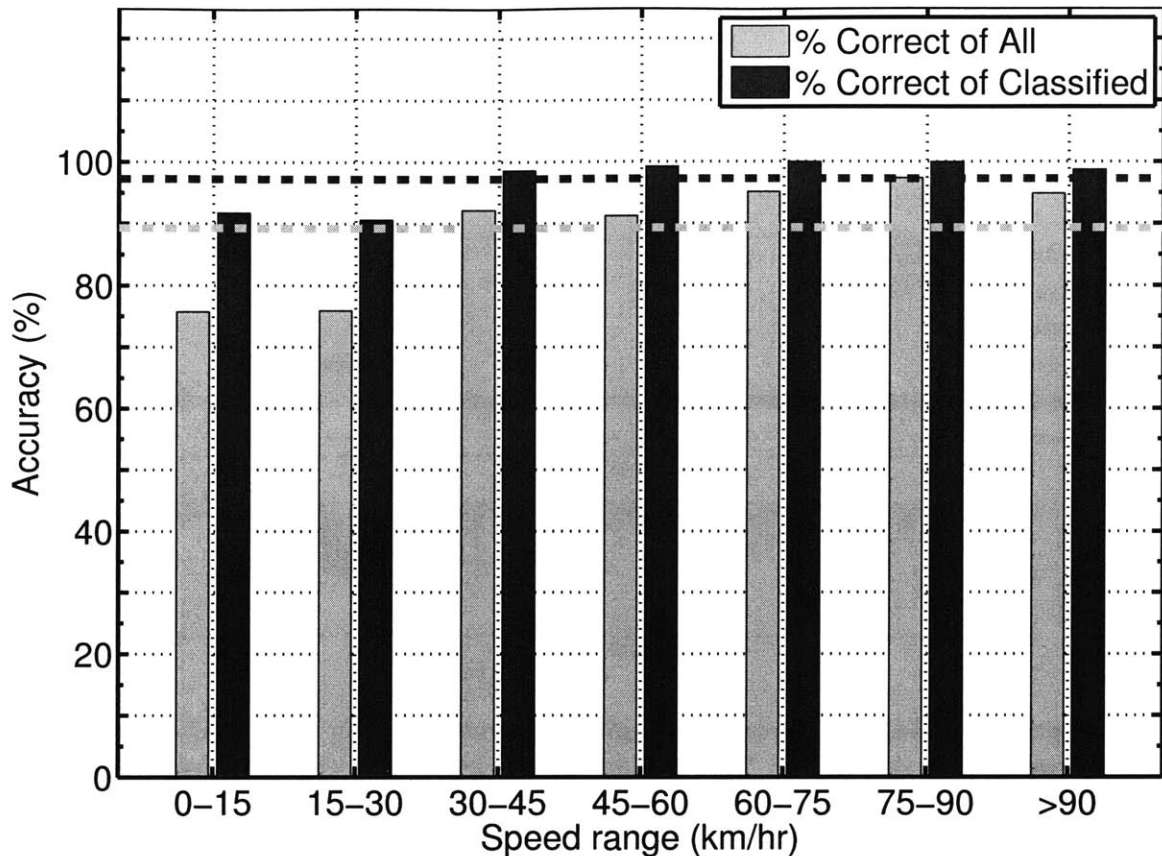
**Table 3.4. Classification results using enhanced algorithm with 65% threshold.**

		<b>Actual Label:</b>				
		Asphalt	Brick	Gravel	Rumble Strip	Grass
<b>Classified as:</b>	Asphalt	<b>92.9%</b>	13.5%	4.0%	16.7%	42.9%
	Brick	0.1%	<b>40.4%</b>	4.0%	0.0%	0.0%
	Gravel	1.1%	9.6%	<b>70.6%</b>	2.8%	0.0%
	Rumble Strip	0.0%	0.0%	0.0%	<b>50.0%</b>	0.0%
	Unknown	5.9%	36.5%	21.5%	30.6%	<b>57.1%</b>



**Figure 3.16. Subset of terrain classification results using enhanced algorithm presented in this thesis. Percentage accuracies are for entire 10 km test set.**

The intent of the proposed algorithm is to develop an algorithm suitable for dynamic vehicles traveling at arbitrary speeds. Figure 3.17 shows the classification accuracy versus vehicle speed. The figure shows both the percentage of correctly classified instances out of all instances in the speed range, and the percentage correctly classified instances out of the classified instances. Over all speed ranges, over 90% of the labeled data is correctly labeled. The slightly lower accuracy at low speed is likely due to the increased amount of non-asphalt data at lower speed.



**Figure 3.17. Terrain classification results vs. speed using enhanced algorithm. Dotted lines show average accuracy over all speeds.**

Some algorithm tuning is possible with the choice of SVM parameters. For the results presented above, SVM cross-validation was performed to maximize the classification accuracy for all terrain classes, giving equal weight to each class. An alternative metric is to choose parameters resulting in the highest total accuracy. Using such a metric, 90.2% of data was classified correctly; a small improvement over the 89.3% accuracy given above. However, as shown in Table 3.5, because more asphalt data was available than other terrain types, this metric has the effect of overtraining the classifier to detect asphalt and results in decreased classification accuracy for the other terrain types.



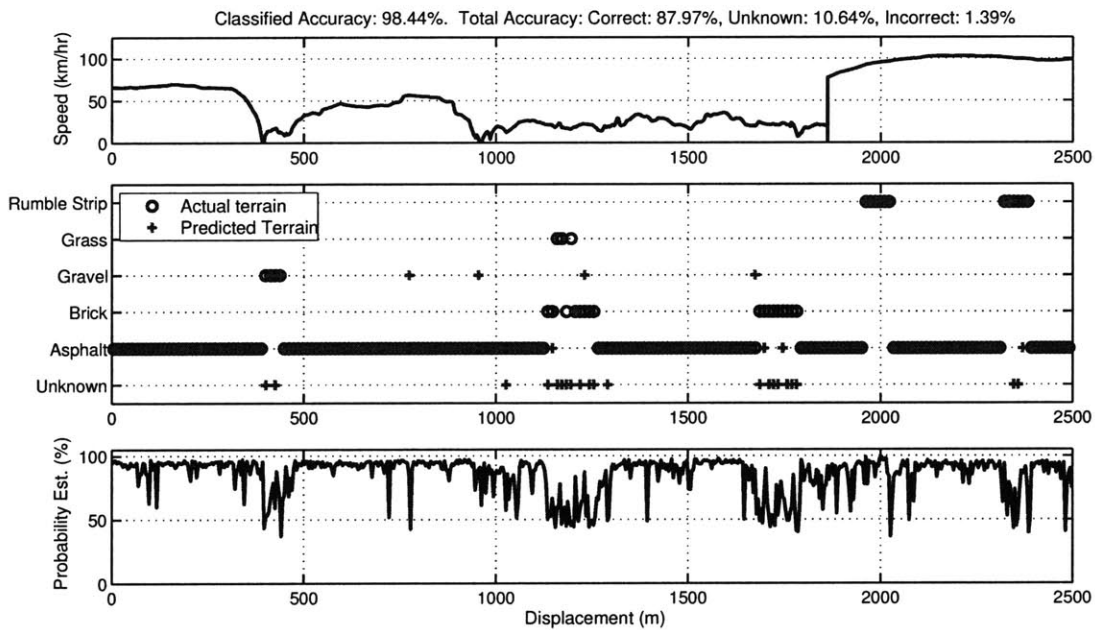
**Table 3.5. Classification results using enhanced algorithm with SVM parameters chosen to provide maximum combined accuracy. Note overtraining for asphalt classification at expense of accuracy on other terrains.**

		Actual Label:				
		Asphalt	Brick	Gravel	Rumble Strip	Grass
Classified as:	Asphalt	<b>95.5%</b>	23.1%	7.3%	22.2%	14.3%
	Brick	0.0%	<b>23.1%</b>	1.1%	0.0%	0.0%
	Gravel	0.5%	17.3%	<b>56.5%</b>	0.0%	0.0%
	Rumble Strip	0.0%	0.0%	0.0%	<b>47.2%</b>	0.0%
	Unknown	4.0%	36.5%	35.0%	30.6%	<b>85.7%</b>

Table 3.6 and Figure 3.18 show the classification results using the enhanced terrain classification algorithm, but without filtering impulses from the data before classification. The classified accuracy is actually increased to 98.4%, from 97.3%, however this is at the expense of more data being labeled “unknown”. The algorithm appears to assign an “unknown” class to the majority of instances containing impulses. However there are two benefits to impulse removal. First, detecting the presence of an impulse provides more information than labeling the instance containing the impulse as “unknown.” Second, an impulse is generally shorter than the 4 meter terrain segments. If labeled as an impulse, only the data containing the impulse is removed, however, if labeled as unknown, the entire 4 meter segment is unclassified.

**Table 3.6. Classification results using enhanced algorithm without removing impulses.**

		Actual Label:				
		Asphalt	Brick	Gravel	Rumble Strip	Grass
Classified as:	Asphalt	<b>93.7%</b>	13.2%	1.1%	2.7%	14.3%
	Brick	0.2%	<b>3.8%</b>	1.7%	0.0%	0.0%
	Gravel	0.6%	5.7%	<b>48.3%</b>	0.0%	0.0%
	Rumble Strip	0.0%	0.0%	0.0%	<b>73.0%</b>	0.0%
	Unknown	5.5%	77.4%	48.9%	24.3%	<b>85.7%</b>



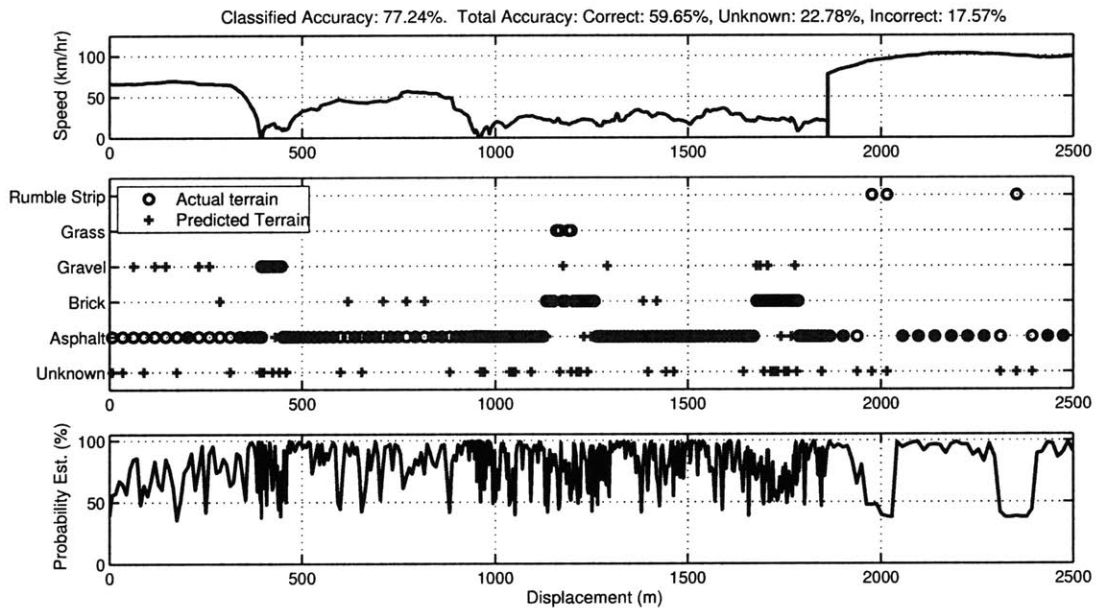
**Figure 3.18. Subset of terrain classification results using enhanced algorithm without removing impulses.**

The data was also classified using the original time-based algorithm proposed by Brooks [9], where the frequency components are calculated directly from the measured time-domain vibrations. The Brooks algorithm originally used a Fisher linear discriminant analysis based classifier; however the SVM-based classifier was used here. Otherwise the Brooks algorithm has not been modified. As shown in Table 3.7 and

Figure 3.19, the unmodified algorithm yields significantly poorer results for a dynamic vehicle with rubber-pneumatic tires than the enhanced algorithm presented here.

**Table 3.7. Classification results using the unmodified time-based algorithm.**

		Actual Label:				
		Asphalt	Brick	Gravel	Rumble Strip	Grass
Classified as:	Asphalt	<b>68.7%</b>	15.2%	26.2%	0.0%	0.0%
	Brick	6.0%	<b>27.8%</b>	10.5%	0.0%	63.6%
	Gravel	5.3%	21.5%	<b>34.2%</b>	0.0%	0.0%
	Rumble Strip	0.0%	0.0%	0.0%	<b>0.0%</b>	0.0%
	Unknown	20.0%	35.4%	29.2%	100.0%	<b>36.4%</b>



**Figure 3.19. Subset of terrain classification algorithm results using the unmodified time-based algorithm.**

### 3.6 Summary and Conclusions

An algorithm has been presented for supervised classification of terrain for dynamic vehicles. The algorithm creates an estimate of the terrain profile from measured wheel accelerations. Spatial frequency components of the estimated profile are used as speed-independent features for classification. Excellent algorithm performance has been

experimentally demonstrated driving on multiple terrain types at a wide range of vehicle speeds. Algorithm performance has been shown to be superior to a baseline algorithm which does not consider the effects of vehicle speed. In addition to the profile estimation algorithm, a novel technique for detecting terrain impulses has been developed which can accurately detect impulses such as potholes.

Future work should explore the accuracy of the road profile estimate versus measured profiles and explore alternate uses of the profile estimate, such as road roughness estimation. Additional future work should validate the algorithm on additional vehicle and terrain types.

# 4

## **Chapter 4: CONCLUSIONS AND SUGGESTIONS FOR** **FUTURE WORK**

### **4.1 Contributions of this Thesis**

This thesis has presented novel sensing and estimation algorithms for two significant terrain related problems for dynamic outdoor mobile robots. In Chapter 2, algorithms were presented for detecting wheel slip and robot immobilization using non-task specific sensors. In Chapter 3, an algorithm was presented for estimating the traversed terrain profile and classifying terrain type.

The major contribution of Chapter 2 is the development of longitudinal wheel slip estimation algorithms suitable for specific sensor deprived situations. Wheel slip estimation is trivial if a vehicle has at least one undriven wheel equipped with a speed sensor. However, in some situations, mounting a sensor on an undriven wheel is not practical, and for all-wheel-drive vehicles, wheel slip is expected on all wheels, making such an approach impossible. Prior approaches have utilized electric drive motor current or vehicle mounted video to solve the wheel slip problem, however these approaches are not always feasible, as discussed in the text. The author is aware of no prior work which has solved the problem using only driven wheel speed and noisy inertial measurements on outdoor terrain. This thesis has presented two novel techniques for solving this

problem, a dynamic model-based estimator, and a signal recognition-based approach. For the dynamic-model based approach, a simplified tire traction/braking model was developed, which is suitable for fast online estimation of tire forces. A preliminary adaptation algorithm was also presented for automatic identification of traction parameters.

The major contribution of Chapter 3 is the development of a tactile, vehicle speed independent, terrain classification algorithm. Prior work has utilized both wheel and body mounted sensors to measure the tactile response of a vehicle to different terrain types. However, the author is aware of no previous work which has specifically considered the effects of changing vehicle speed on the measured vehicle terrain response. All prior work the author is aware of has been demonstrated at only very low, constant speed. Excellent performance with the proposed algorithm has been experimentally demonstrated on an automobile driving over multiple terrain types at the full range of typical driving speeds. Additionally, as part of the classification algorithm, an algorithm was developed for estimating the traversed terrain profile using an inexpensive wheel mounted accelerometer.

## ***4.2 Suggestions for Future Work***

Many extensions to the work presented in this thesis are possible ranging from additional experimental validation to further theoretical development. The dynamic model-based wheel slip estimation algorithm presented in Section 2.2 estimates true vehicle speed, however the experimental data available only included enough information to validate immobilization detection. Future work should compare estimated vehicle speed with a ground truth measurement and investigate the effect of tire model

parameters on the velocity estimate. Future work should then expand the presented framework to create a vehicle position estimate that is robust to wheel slip. Future work should also experimentally validate and improve the proposed tire model adaptation algorithm.

The proposed terrain classification algorithm has been experimentally validated using one vehicle type on five terrain types. Future work should validate the algorithm on additional terrain and vehicle types. Additional work should investigate the preferred applications of the proposed algorithm. For example, terrain classes assigned in this work were specific terrain names, such as “gravel” or “asphalt.” However these terrain names may not effectively capture the specific effect of the terrain on the vehicle. Rough asphalt may affect the vehicle response the same as brick, but very differently than smooth asphalt. Thus, it may be more useful to define terrain classes as more descriptive groups, such as “hard, flat surface” and “hard, rough surface.” These definitions should be formulated with a specific application, such as traction control or automatic suspension tuning, in mind. Future work should also explore the accuracy of the profile estimate and explore direct applications of this estimate. Some possible uses include creating a “roughness” measure or using the estimated terrains as simulated terrain in vehicle simulations.

## REFERENCES

- [1] R. Anderson, D Bevly, "Estimation of slip angles using a model based estimator and GPS," *Proceedings of the 2004 American Control Conference*, vol. 3, pp 2122-2127, July 2004.
- [2] A. Angelova, L. Matthies, D. Helmick, G. Sibley, P. Perona, "Learning to predict slip for ground robots," *ICRA, 2006*.
- [3] *Automotive Handbook*, 5<sup>th</sup> ed., Robert Bosch GmbH, Germany, 2000.
- [4] B. Barshan, H. Durrant-Whyte, "Inertial navigation systems for mobile robots," *IEEE Trans. on Robotics and Automation*, vol. 11, no. 3, 1995.
- [5] M.G. Bekker, *Introduction to Terrain-Vehicle Systems*, Univ. of Michigan Press, Ann Arbor, 1969, part II, ch. 2.
- [6] M. G. Bekker, *Theory of Land Locomotion*, Univ. of Michigan Press, Ann Arbor, 1956, ch. 9 & 6.
- [7] M. Blank, D. Margolis, "The effect of normal force variation on the lateral dynamics of automobiles," *Society of Automotive Engineers, paper # 960484*, 1996.
- [8] J. Borenstein, H. R. Everett, L. Feng, "Where am I?" *Sensors and Methods for Mobile Robot Positioning*, Univ. of Michigan, April 1996, Available: <http://www-personal.umich.edu/~johannb/shared/pos96rep.pdf>.
- [9] C. Brooks, "Terrain Identification Methods for Planetary Exploration Rovers," Master's Thesis, Massachusetts Institute of Technology, September 2004.



- [10] C. Brooks, K. Iagnemma, "Vibration-Based Terrain Classification for Planetary Exploration Rovers," *IEEE Transactions on Robotics*, vol. 21, no. 6, December 2005.
- [11] C.-C. Chang, C.-J. Lin, "LIBSVM: a library for support vector machines," 2001. <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
- [12] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, G. Bradski, "Self-supervised monocular road detection in desert terrain," *Proc. of the Robotics Science and Systems Conference*, Philadelphia, PA, 2006.
- [13] *Defense Advanced Research Projects Agency: Strategic Plan, February 2007*, Accessed March 30, 2007, Available: <http://www.darpa.mil/body/pdf/DARPA2007StrategicPlanfinalMarch14.pdf>.
- [14] N. De Freitas, R. Dearden, F. Hutter, R. Morales-Menendez, J. Mutch, D. Poole, "Diagnosis by a Waiter and a Mars Explorer," *Proceedings of the IEEE*, vol. 92, no. 3, 2004.
- [15] G. Dissanayake, S. Sukkarieh, E. Nebot, H. Durrant-Whyte, "The aiding of a low-cost strapdown measurement unit using vehicle model constraints for land vehicle applications," *IEEE Trans on Robotics and Automation*, vol. 17, no. 5, 2001.
- [16] E. DuPont, R. Roberts, C. Moore, M. Selekwa, E. Collins, "Online terrain classification for mobile robots," *Proceedings of IMECE 2005*, Orlando, USA, November 2005.
- [17] W. Flenniken, J. Wall, D. Bevly, "Characterization of various IMU error sources and the effect on navigation performance," *ION GNSS 2005*.

- [18] Y. Fuke, E. Krotkov, "Dead reckoning for a lunar rover on uneven terrain," *Proc. of 1996 IEEE Intl. Conf. on Robotics and Automation*, Minneapolis, MN, April 1996.
- [19] A. Ganapathiraju, J. Hamaker, J. Picone, "Applications of support vector machines to speech recognition," *IEEE Trans. on Signal Processing*, vol. 52, no. 8, August 2004.
- [20] J. Geeter, H. Brussel, J. Schutter, "A Smoothly Constrained Kalman Filter," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 10, October 1997.
- [21] T. Gillespie, *Fundamentals of Vehicle Dynamics*. Society of Automotive Engineers, Warrendale, PA. 1992, ch. 1.
- [22] G. H. Golub, C. F. Van Loan, "The singular value decomposition and unitary matrices," In *Matrix Computations*, 3<sup>rd</sup> ed, Johns Hopkins University Press, Baltimore, MD, 1996.
- [23] *GPS 16/17 Series Technical Specifications*, Revision A, Garmin International, Inc., Olathe, KS, July 2005.
- [24] M. Grewal, A. Andrews. *Kalman Filtering: Theory and Practice*, Englewood Cliffs, NJ: Prentice Hall, 1993, ch. 7.
- [25] F. Gustafsson, "Slip-based tire-road friction estimation," *Automatica*, vol. 33, no. 6, pp. 1087-1099, 1997.
- [26] T. Hastie, R. Tibshirani, J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Springer, NY, 2001.
- [27] B. Hofmann-Wellenhof, H. Lichtenegger, J. Collins, *Global Positioning System: Theory and Practice*, 5<sup>th</sup> ed., New York: Springer, 2001, ch. 6.

- [28] M. Hung, D. Orin, "Dynamic simulation of actively-coordinated wheeled vehicle systems on uneven terrain," *Proc. of the 2001 IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, May 2001.
- [29] C.-W. Hsu, C.-C. Chang, C.-J. Lin, "A practical guide to support vector classification," Accessed: August 2006, Available: <http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>.
- [30] K. Iagnemma, S. Kang, H. Shibly, S. Dubowsky, "Online terrain parameter estimation for wheeled mobile robots with application to planetary rovers," *IEEE Transactions on Robotics*, vol. 20, no. 5, October 2004.
- [31] K. Iagnemma, S. Dubowsky, "Traction Control of Wheeled Robotic Vehicles with Application to Planetary Rovers." *Intl Journal of Robotics Research*, vol. 23, no. 10, pp. 1029-1040, October 2004.
- [32] S. Julier, H. Durrant-Whyte. "On the role of process models in autonomous land vehicle navigation systems," *IEEE Trans. on Robotics and Automation*, vol. 19, no. 1, February 2003.
- [33] A. Kelly, "A 3D state space formulation of a navigation Kalman filter for autonomous vehicles," CMU Technical Report CMU-RI-TR-94-19.
- [34] V. Kumar, K. Waldron, "Force distribution in closed kinematic chains," *IEEE Journal of Robotics and Automation*, vol. 4, no. 6, December 1988.
- [35] "Learning Applied to Ground Robots," accessed July 24, 2006, <http://www.darpa.mil/ipto/Programs/lagr/vision.htm>.
- [36] "Moving Averages", accessed July 2006, [http://www.stockcharts.com/education/IndicatorAnalysis/indic\\_movingAvg.html](http://www.stockcharts.com/education/IndicatorAnalysis/indic_movingAvg.html)

- [37] *National Highway Traffic Safety Administration*, "Traffic Safety Facts 2005," Accessed March 30, 2007, Available: <http://www-nrd.nhtsa.dot.gov/pdf/nrd-30/NCSA/TSFAnn/TSF2005.pdf>.
- [38] O. Nelles, *Nonlinear System Identification*. Springer-Verlag, Germany, 2001.
- [39] L. Ojeda, D. Cruz, G. Reina, J. Borenstein, "Current-based slippage detection and odometry correction for mobile robots and planetary rovers," *IEEE Transactions on Robotics*, vol. 22, no. 2, April 2006.
- [40] L. Ojeda, G. Reina, J. Borenstein, "Experimental results from FLEXnav: An expert rule-based dead-reckoning system for Mars rovers," *IEEE Aerospace Conference 2004*, Big Sky, MT, March 2004.
- [41] L. Ojeda, J. Borenstein, G. Witus, R. Karlsen, "Terrain characterization and classification with a mobile robot," *Journal of Field Robotics*, vol. 23, no. 2, 2006.
- [42] "Particle size," *Wikipedia, the Free Encyclopedia*, accessed March 20, 2007, [http://en.wikipedia.org/wiki/Particle\\_size](http://en.wikipedia.org/wiki/Particle_size).
- [43] J. Ryu, E. Rosseter, J.C. Gerdes, "Vehicle sideslip and roll parameter estimation using GPS," *AVEC*, Hiroshima, Japan, 2002.
- [44] S. Sukkarieh, E. Nebot, H. Durrant-Whyte, "A high integrity IMU/GPS navigation loop for autonomous land vehicle applications," *IEEE trans. on Robotics and Automation*, vol. 15, no. 3, June 1999.
- [45] P. Sundvall, P. Jensfelt, "Fault detection for mobile robots using redundant positioning systems," *ICRA 2006*.
- [46] H. Tan, Y. Chin, "Vehicle antilock braking and traction control: a theoretical study," *Intl. Jrnl of Systems Science*, vol. 23, no. 3, 1992.

- [47] M. Wadda, K. Yoon, H. Hashimoto, "High accuracy road vehicle state estimation using extended Kalman filter," *Proc. of 2000 IEEE Intelligent Transportation Systems*, Dearborn, MI, October 2000.
- [48] G. Welch, G. Bishop, "An introduction to the Kalman filter," *SIGGRAPH 2001*.
- [49] G. Welch, G. Bishop. "SCAAT: Incremental tracking with incomplete information", *SIGGRAPH 1997*, pp..333-344.
- [50] P.D. Welch, "The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Transactions on Audio and Electroacoustics*, vol. AU-15, pp. 70-73, June 1967.
- [51] J. Y. Wong, *Theory of Ground Vehicles*, 3<sup>rd</sup> ed., New York, NY: John Wiley & Sons, 2001, ch. 1.

3369-97