A Probabilistic Framework for Real-Time Mapping on an Unmanned Ground Vehicle

Senior Thesis in Control and Dynamical Systems by:
Jeremy H. Gillula

Thesis Advisor:
Professor Joel Burdick

June 8, 2006

Abstract

In the course of preparing for the 2005 DARPA Grand Challenge, an off-road race for autonomous vehicles, a group of undergraduates from Caltech developed a set of deterministic algorithms for performing sensor fusion on maps generated by different range sensors on a mobile robot. That framework had serious limitations, however, including "disappearing" obstacles and lack of confidence data associated with features in the maps. In this thesis, we present a probabilistic framework that attempts to solve some of these problems by using error models of two typical types of range sensors, as well as by making use of Kalman filtering techniques from control theory to fuse the resulting measurements into an accurate digital elevation map. Our results indicate that this probabilistic framework has several advantages over the determinisic framework used by Team Caltech in the 2005 Grand Challenge.

Contents

List of Figures

Chapter 1

Introduction

Autonomous mobile robots (also known as unmanned ground vehicles, or UGVs) have a variety of applications, including use in the military to reduce the risk to soldiers during combat; in industrial settings, especially in autonomous inspections of hazardous waste facilities; and in planetary exploration, of which NASA's Mars Pathfinder and Mars Exploration Rovers are the most famous examples. However, one of the major limitations of UGVs is the accuracy with which they can sense and map their surroundings as they explore, since they must frequently work in unstructured, previously unmapped environments. Complicating the problem further, UGVs must generate maps of their environments using measurements from potentially noisy sensors. Designers frequently attempt to reduce the impact of this problem by using multiple sensor systems with different strengths and weaknesses (Luo and Kay, 1989). Stereovision systems, for example, can provide dense but potetially noisy range data, whereas laser range finders can provide relatively more accurate, but also sparser range data. Using multiple sensors to map an environment brings up two potential problems, however.

First, how do you transform the robot's perception of the world into a data structure that can be used to efficiently plan the robot's path? Of course, the answer to this question depends on the type of path-planning algorithm that is used. One common data structure used for path planning is a goodness map, essentially a grid of discrete cells, where every cell is assigned a goodness value corresponding to its traversability as determined by a variety of measures (Goldberg et al., 2002). In unstructured environments, however, the problem is non-trivial; there is generally no obvious method of computing goodness (traversability) based on raw sensor data, especially when that data is sparse. (One exception would be stereovision data, which frequently provides enough data in a single measurement to compute traversability (Bellutta et al., 2000).) Instead, one can introduce an intermediate data structure between the raw sensor data and the final goodness map, which can be used to collect the data from the measurements over time. This data structure is known as a digital elevation map (DEM), and is similar to a goodness map in that it is made up of a grid of discrete cells. Its contents are fundamentally different, however, in that every cell contains a height that corresponds to the elevation of the terrain at the coordinates of that cell, instead of an abstract goodness value.

Figure 1.1: Alice, Caltech's entry in the 2005 DARPA Grand Challenge.

The second question one must answer when using multiple sensors on a UGV is whether or not one can combine the information from the different sensors into a representation of the world that not only makes sense, but is also optimally accurate. In other words, how should one fuse together the data from different sensors? In this thesis we provide an answer to that question, drawing on techniques from control theory to create a probabilistic framework in which to perform sensor fusion.

1.1 A Previous Framework and Its Limitations

The sensor fusion problem is not a new one, of course, and has been explored in the past. One potential solution was explored by a group of undergraduate and graduate students from Caltech on their entry into the DARPA Grand Challenge: a 10-hour, 175-mile off-road race for unmanned autonomous vehicles that was held for the second time in October 2005 (see figure 1.1). In preparation for the Grand Challenge, a particularly flexible framework was developed for performing sensor fusion, using both DEMs and goodness maps. In particular, every sensor created its own DEM and corresponding goodness map using methods described in Cremean et al. (2006). The resulting goodness maps were then fused using a heuristic algorithm that was biased toward sensors whose coverage areas were closer to the vehicle. A diagram of this framework is depicted in figure 1.2. Unfortunately, this solution had several limitations, which we briefly review here.

The first such limitation is something we have termed the "disappearing obstacle" problem, which is also described in more detail in Cremean et al. (2006). Essentially, a chain of events can occur in which a long-range sensor accurately detects an obstacle of a given height, but the data from that sensor is overruled by a short-range sensor, which detects only the lower portions of the obstacle. While one could adjust the weight attached to each sensor so that sensors pointed closer to the vehicle had lower confidence than those pointed further away, this would remove the potential advantages of using short-range sensors, namely that short-range sensors tend to produce data with less noise due to a shorter moment-arm about

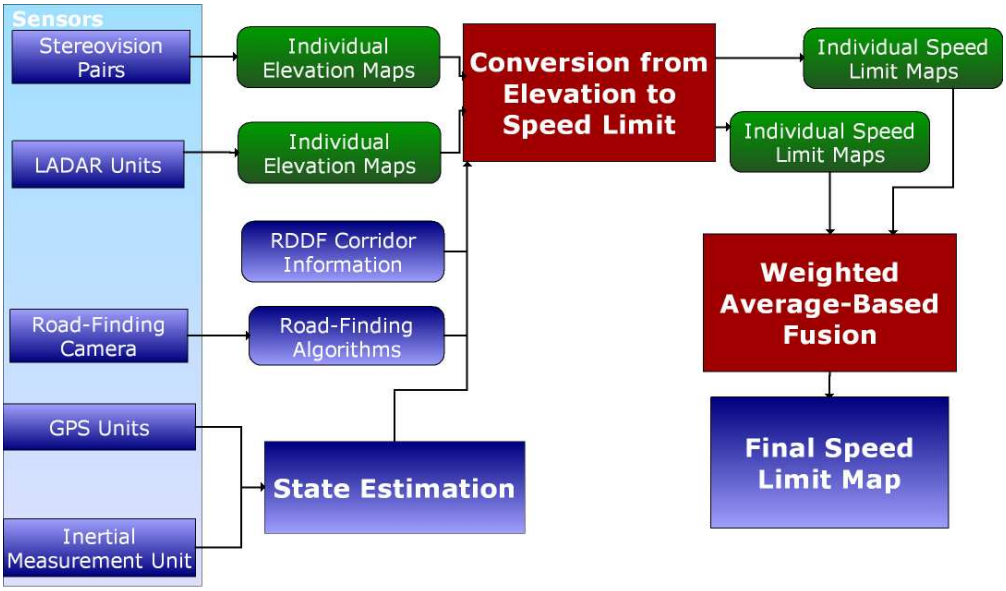

Figure 1.2: A diagram of the information flow in the software system used on Team Caltech's 2005 Grand Challenge entry.

which the range measurements are taken. We describe this problem in more detail in section 2.3.

A second such limitation has to do with what values the goodness map should have in areas which are void of sensor data (no-data areas). There are arguments against both setting the goodness of no-data cells to be high and setting them to be low (again, see Cremean et al. 2006). One potential solution would be to use a probabilistic framework instead, so that meaningful statements could be made about the probability of a given cell having a certain goodness value. Again, this would require a new framework in which to fuse the sensor data.

The final limitation is similar to the no-data limitation described above, in that the system has no confidence data associated with the goodness values (or the elevation values) in a given cell. This limitation was one of the major causes of the failure of Caltech's vehicle to complete the 2005 Grand Challenge. Essentially, two midrange sensors failed during the course of the race, leaving only a significantly less accurate long-range sensor to guide the vehicle. The vehicle did not slow down while driving through areas of its map about which it should have had less confidence, however, leading it to eventually crash.

1.2 A Probabilistic Framework

In light of these limitations, it is clear that another sensor fusion strategy is necessary. In this thesis, we will demonstrate how performing sensor fusion using a probabilistic framework at the DEM stage of mapping to fuse all of the incoming sensor data can eliminate some of the

limitations described. Although similar sensor fusion algorithms have been proposed before (as in Cremean (2006)), we believe that this is the first system to probabilistically fuse data from disparate sensor types. Thus, the contributions of this thesis are threefold:

1. We present error models for two common types of sensors, stereovision and LADAR (building on results in Ye and Borenstein (2002) and Matthies (1992)), which can then be used in a probabilistic sensor fusion framework.

2. We present a probabilistic framework (inspired by Cremean (2006)) using techniques adapted from control theory (*i.e.* the Kalman filter), which, given certain conditions on the form of the sensor noise, provide the optimal estimate of the elevation of the surrounding terrain.

3. We take advantage of the framework's probabilistic nature by extending it to present a solution to the "disappearing obstacle problem."

Chapter 2

Technical Approach

According to control theory, the optimal estimator for a linear system undergoing Gaussian white noise disturbances is a Kalman filter. Therefore, we would expect that a Kalman filter would perform reasonably well as the means of data fusion for the problem described above. To test this expectation, we implemented a discrete Kalman filter on a cell-by-cell basis for the digital elevation map, with the inputs consisting of the measurements and variances from the different sensors for a given cell, and the output consisting of the estimated elevation and covariance of that cell. The algorithm we propose proceeds as follows, and is depicted graphically in figure 2.1:

1. A range measurement of the terrain surrounding the vehicle is taken, and an error model is used to describe the uncertainty surrounding the measurement.

2. The measurement is transformed into a global coordinate frame, and the uncertainty in the vehicle's state estimate adds to the uncertainty of the measurement's position.

3. The measurement (and more specifically, its uncertainty) is discretized so that only a small number of cells in the elevation map need to be updated.

4. In every cell that needs to be updated, the new data is fused by an individual Kalman filter in the particular cell using the discretized information taken from the sensor's uncertainty model.

One important aspect of this algorithm is that it is performed aysnchronously – data fusion is performed only when new measurements are taken, and no additional processing is required otherwise. Additionally, the computational steps required to perform the data fusion have been developed with an eye toward efficiency, so that the algorithm can perform in real time on a UGV.

Of course, there are many implementation details which this high-level overview of the algorithm does not cover; the rest of this section will describe in detail how we arrive at the uncertainty models for the sensors, how we discretize that model, what equations we use to implement the Kalman filter, and how we attempt to solve the "disappearing obstacle" problem.

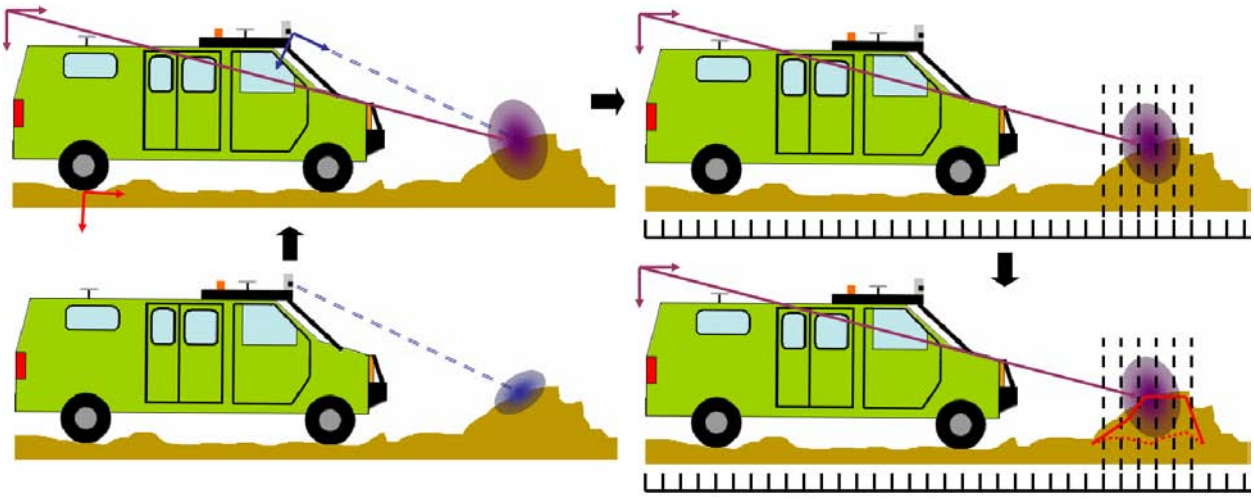

Figure 2.1: A diagram of our algorithm – processing proceeds clockwise from the bottom-left. First, a measurement is taken using a sensor and some uncertainty is associated with it using an error model for that sensor. Second, the measurement is tranformed into the global coordinate frame, as indicated by the purple axes. (The red axes indicate the intermediary vehicle coordinate frame, as described in the text.) Third, the uncertainty is discretized, as indicated by the dashed lines. Finally, the existing estimate of the elevation in the map, the dashed red line, is fused with the measurement to give a new estimate, the solid red line.

Before we continue, we make note here of the different coordinate systems and notational conventions used throughout the rest of this thesis. In particular, three coordinate frames are used throughout this thesis (figure 2.2):

1. The sensor coordinate frame, which has its origin at a given range sensor. This frame varies from sensor to sensor.

2. The vehicle coordinate frame, which has its origin at ground-level beneath the rear axle of the vehicle, and is oriented such that the x-axis points forward toward the point on ground level beneath the front axle, the y-axis points to the right of the vehicle toward where the right-rear tire touches ground-level, and the z-axis points downward into the ground. This frame moves along with the vehicle.

3. The global coordinate frame, an Earth-fixed reference frame. In this thesis, we make use of the standard UTM (Universal Transverse Mercator) coordinate system, in which the x-axis points north, the y-axis points east, and the z-axis once again points downward into the ground.

To specify in which frame a given variable is being referenced, we frequently use subscripts of the form "descriptor, frame." So, for example, the x-coordinate of a measurement in the vehicle frame would be referred to by "$x_{m,v}$", where the m stands for "measurement" and the v stands for "vehicle frame."

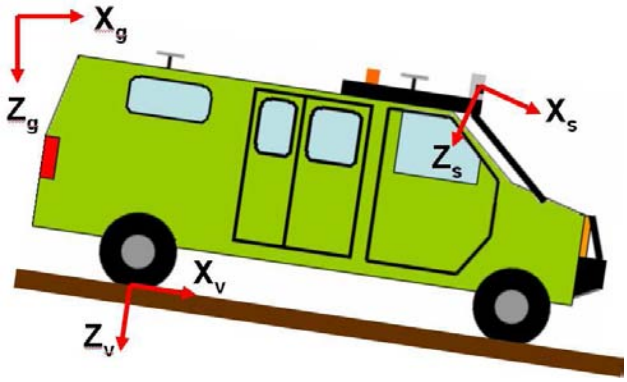

Figure 2.2: The different coordinate frames used throughout this paper, simplified to two dimensions for easier demonstration. For each of the coordinate frames, the Y axis points out of the page toward the reader.

For notation, in section 2.1, actual values are denoted with hats (*e.g.* $\hat{\rho}$) and Gaussian errors are denoted with tildes (*e.g.* $\tilde{\rho}$). In sections 2.2 and 2.3, hats are used to denote an estimate of an actual value (*e.g.* $\hat{z}_{i,j}$).

2.1 Sensor Uncertainty Models

As was mentioned above, to probabilistically fuse the range measurements from multiple sensors, it is necessary to have both the measurements from the range sensors and the uncertainty estimates associated with each range measurement. We will generate these uncertainty estimates by using statistical error models of the sensors that predict how the sensor's measurements are affected by noise in the system. We have chosen to present error models for two common types of range sensors used on UGV's, LADAR (LAser Detection And Ranging) and stereovision. Because the measurements are transformed into the global coordinate system, we also need to take into account the error model of the state sensors, whose output is used to perform the coordinate transformation from the vehicle frame to the global frame.

2.1.1 LADAR Uncertainty Model

LADAR (sometimes referred to as LIDAR) is a name given to a broad category of range sensors which use lasers to precisely determine distances to objects in the sensor's field of view. Here, we cover LADAR that works on a time-of-flight principle: at a precise time t_0, the sensor emits an infrared laser pulse which is reflected off of a spinning mirror into the environment; that pulse is then reflected by an object back to the sensor, which measures the time t_1 at which the pulse returned. The range is then simply $\rho = c(t_1 - t_0)/2$ (where c is the speed of light). (Figure 2.3.) This process is repeated at distinct intervals as the mirror spins, resulting in range measurements at regularly spaced angles $\theta \in [\theta_{min}, \theta_{max}]$.

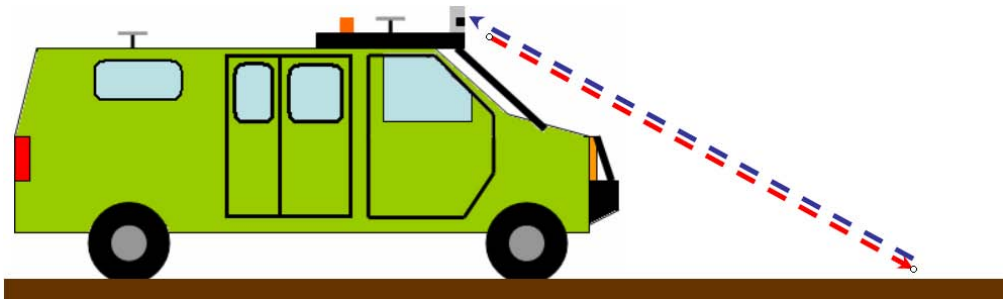

Figure 2.3: A diagram of how LADAR works: a pulse of light is emitted from the LADAR. The light then bounces off an object in the environment, and reflects back to a sensor in the LADAR. The time it takes for the pulse to be emitted and then reflected is used to report the range. This procedure is done dozens of times a second as a mirror within the LADAR spins, allowing measurements to be taken in an arc in the robot's direction of travel.

If we assume that the noise on both the range and the angle is Gaussian, our error model becomes:

$$\rho = \hat{\rho} + \tilde{\rho} \tag{2.1}$$
$$\theta = \hat{\theta} + \tilde{\theta} \tag{2.2}$$

where ρ and θ are the reported range and angle measurements, respectively, $\hat{\rho}$ and $\hat{\theta}$ are the true range and angle, and $\tilde{\rho}$ and $\tilde{\theta}$ are Gaussian zero-mean random variables that introduce noise into the measurement. This is similar to the model described by Ye and Borenstein (2002), who found an error model for the range of SICK laser scanners (a specific brand and model of LADAR) of the form

$$\rho = k\hat{\rho} + b + \tilde{\rho} \tag{2.3}$$

where k and b are parameters that account for the fact that the reported measurement is not precisely the same as the actual range, but is instead linearly related. In their experiment, they determined that $k = 1.0002$ and $b = 3.6$mm. Because these values are negligible compared to the ranges we are measuring[1], we felt that for our purposes it was safe to neglect these terms and use our original error model. For an analysis of how well our model fits the actual LADARs used in the experiment, see section 3.2.

2.1.2 Stereovision Uncertainty Model

In the case of stereovision the error model is somewhat more complex. Stereovision estimates range by making use of two cameras, each mounted on approximately the same backplane a distance b apart from each other, such that the scan lines of each camera are as close to

[1] At the longest possible range reportable by the LADAR units we used in our experiment (81m, according to their specifications), the difference between our error model and that of Ye and Borenstein is 1.98cm. At the longest reported range actually measured during the experiment (∼30m), the difference is 0.96cm.

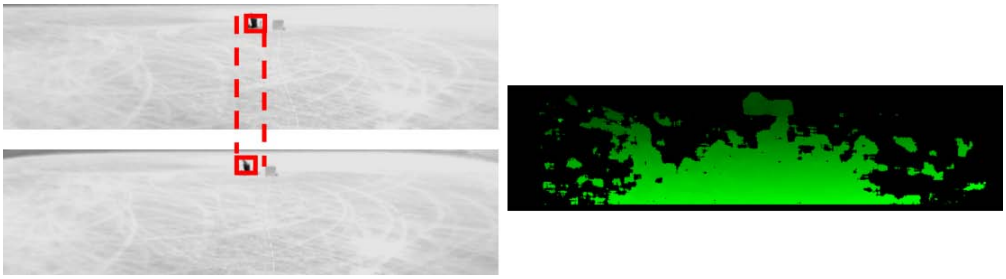

Figure 2.4: On the left are two images from a stereovision pair, taken of the same scene simultaneously. The boxes in each image highlight corresponding regions of pixels, and the dashed lines illustrate the different positions the same object has in the two images. By analyzing these differences a disparity image can be created, as shown on the right. In this image, lighter green pixels are closer to the camera, and darker green are further away. Black areas indicate pixels where the algorithm could not confidently make a match, and thus no range was estimated.

parallel as possible. The cameras are then synchronized so that they take images simultaneously; these images are then fed to a stereovision algorithm, which first warps the images so that their scan lines are exactly parallel using an *a priori* mapping determined offline using calibration data. The system then analyzes the two images to find similar, or corresponding, regions of pixels. (The methods used to solve this correspondence problem are beyond the scope of this thesis — for a review of such algorithms, see Scharstein and Szeliski (2002).) Once a match is found, the algorithm takes the x coordinate of the central pixel of the matching region in each image, x_l and x_r, and computes the disparity $d = x_l - x_r$. (Figure 2.4.) Through some simple geometry, it can be seen that the range will then be given by

$$\rho = \frac{b \times f}{d}, d = \{1, \ldots, N\} \tag{2.4}$$

where again, b is the baseline (the distance between the two cameras), f is the focal length of the cameras (in pixels), and N is the maximum number of disparities searched. Additional analysis (see Matthies (1992)) shows that the uncertainty inherent in stereovision is fundamentally different from that of LADAR — error in stereovision is proportional to the square of the range and the difference in disparity, and inversely proportional to the focal length and baseline, *e.g.* if the range is given by

$$\rho = \hat{\rho} + \tilde{\rho} \tag{2.5}$$

then the error is:

$$\tilde{\rho} \propto \frac{\hat{\rho}^2 \Delta d}{b \times f} \tag{2.6}$$

Thus we can model the Gaussian noise $\tilde{\rho}$ as having a variance

$$\frac{\hat{\rho}^2 \sigma_d}{b \times f} \tag{2.7}$$

where σ_d is the variance in disparity.

For simplcity we can treat range measurements from stereovision as identical to those from LADAR, despite these differences. We simply treat each scan line from stereovision as we would an individual scan from a LADAR unit, except the pitch of the sensor is now a function of which scan line the measurement comes from. Additionally, we must adjust the error model for each measurement to depend on the reported distance - instead of being a Gaussian of fixed variance regardless of the range reported (as in the LADAR case), we now adjust the variance based on the equation for the error given above. Thus we have:

$$\rho = \hat{\rho} + \tilde{\rho} \tag{2.8}$$
$$\theta = \hat{\theta} + \tilde{\theta} \tag{2.9}$$
$$\phi = \hat{\phi} + \tilde{\phi} \tag{2.10}$$

where ρ, θ, and ϕ are the reported range and angle measurements, respectively, $\hat{\rho}$, $\hat{\theta}$, and $\hat{\phi}$ are the true range and angles, and $\tilde{\rho}$, $\tilde{\theta}$, and $\tilde{\phi}$ are Gaussian zero-mean random variables that introduce noise into the measurement.

2.1.3 State Estimation Uncertainty Model

To make use of the range information provided by the sensors, we must transform the range measurements into a global coordinate frame so that they can be fused into a global map. This transformation is performed in two steps: first, the range measurement is transformed into the vehicle coordinate frame using calibration data taken offline about the position and orientation of the sensor on the vehicle; second, the range measurement is transformed into the global frame by using position and orientation data provided online by the vehicle's state estimation system, as in figure 2.5.

If we apply the small angle approximation and neglect second-order terms in the measurement noise, the measurement's location in the vehicle frame ($\begin{bmatrix} x_{M,v} & y_{M,v} & z_{M,v} \end{bmatrix}^{\mathrm{T}}$) is:

$$\begin{bmatrix} x_{M,v} \\ y_{M,v} \\ z_{M,v} \end{bmatrix} = \begin{bmatrix} x_{S,v} \\ y_{S,v} \\ z_{S,v} \end{bmatrix} + \begin{bmatrix} \hat{\rho}\cos\hat{\theta}\cos\hat{\phi} \\ \hat{\rho}\sin\hat{\theta}\cos\hat{\phi} \\ -\hat{\rho}\sin\hat{\phi} \end{bmatrix} + \begin{bmatrix} \cos\hat{\theta}\cos\hat{\phi} & -\hat{\rho}\sin\hat{\theta}\cos\hat{\phi} & -\hat{\rho}\cos\hat{\theta}\sin\hat{\phi} \\ \sin\hat{\theta}\cos\hat{\phi} & \hat{\rho}\cos\hat{\theta}\cos\hat{\phi} & -\hat{\rho}\sin\hat{\theta}\sin\hat{\phi} \\ -\sin\hat{\phi} & 0 & -\hat{\rho}\cos\hat{\phi} \end{bmatrix} \begin{bmatrix} \tilde{\rho} \\ \tilde{\theta} \\ \tilde{\phi} \end{bmatrix} \tag{2.11}$$

As before, in this expression $\hat{\phi}$ is the pitch angle of the measurement with respect to horizontal; for LADAR units, this number is a constant for all measurements, while for stereovision systems, it is a function of the scan line of the range measurement. $\begin{bmatrix} x_{S,v} & y_{S,v} & z_{S,v} \end{bmatrix}^{\mathrm{T}}$ is the location of the sensor in the vehicle coordinate frame.

Finally, we must include the uncertainty that is added by translating from the vehicle coordinate frame to a global coordinate frame (*e.g.* the uncertainty in our state estimate

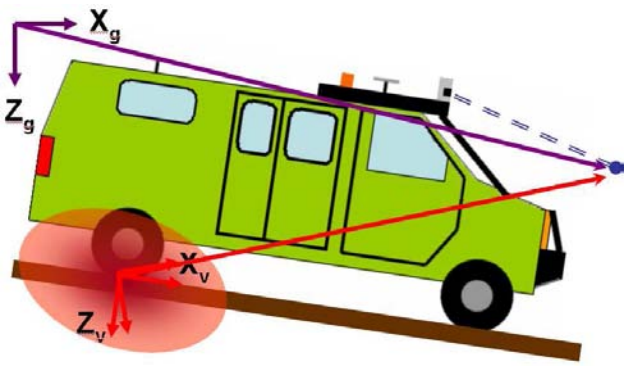

Figure 2.5: To transform a measurement (the blue dot) from the vehicle frame (the red line pointing to the measurement) to the global frame (the purple line pointing to the measurement), we must make use of the vehicle's state estimate to determine where the vehicle is located in the global frame. However, due to errors in the signals from the GPS and IMU, that state estimate is accompanied by some uncertainty (the faded oval cloud centered on the vehicle frame's origin, or the offset in the angle between the estimated and actual vehicle frame).

that results from noisy measurements coming from the GPS and IMU). We model these errors as

$$\begin{bmatrix} x_{V,g} \\ y_{V,g} \\ z_{V,g} \end{bmatrix} = \begin{bmatrix} \hat{x}_{V,g} \\ \hat{y}_{V,g} \\ \hat{z}_{V,g} \end{bmatrix} + \begin{bmatrix} \tilde{x}_{V,g} \\ \tilde{y}_{V,g} \\ \tilde{z}_{V,g} \end{bmatrix} \tag{2.12}$$

and

$$\begin{bmatrix} p_{V,g} \\ h_{V,g} \\ r_{V,g} \end{bmatrix} = \begin{bmatrix} \hat{p}_{V,g} \\ \hat{h}_{V,g} \\ \hat{r}_{V,g} \end{bmatrix} + \begin{bmatrix} \tilde{p}_{V,g} \\ \tilde{h}_{V,g} \\ \tilde{r}_{V,g} \end{bmatrix} \tag{2.13}$$

where $\begin{bmatrix} x_{V,g} & y_{V,g} & z_{V,g} \end{bmatrix}^{\mathrm{T}}$ is the vehicle's reported position (as determined by the state-estimation software) in the global frame (in which we remind the reader that the x-axis points North, the y-axis points East, and the positive z-axis points downward), and $\begin{bmatrix} p_{V,g} & h_{V,g} & r_{V,g} \end{bmatrix}^{\mathrm{T}}$ is the vehicle's reported pitch, yaw (or heading), and roll with respect to the global coordinate frame. As before, $\begin{bmatrix} \hat{x}_{V,g} & \hat{y}_{V,g} & \hat{z}_{V,g} \end{bmatrix}^{\mathrm{T}}$ and $\begin{bmatrix} \hat{p}_{V,g} & \hat{h}_{V,g} & \hat{r}_{V,g} \end{bmatrix}^{\mathrm{T}}$ correspond to the actual values of the corresponding positions and angles, and $\begin{bmatrix} \tilde{x}_{V,g} & \tilde{y}_{V,g} & \tilde{z}_{V,g} \end{bmatrix}^{\mathrm{T}}$ and $\begin{bmatrix} \tilde{p}_{V,g} & \tilde{h}_{V,g} & \tilde{r}_{V,g} \end{bmatrix}^{\mathrm{T}}$ correspond to Gaussian zero-mean random variables that introduce noise into the state estimate.

In general, to transform a measurement from the vehicle coordinate frame to the global coordinate frame, we have:

$$\begin{bmatrix} x_{M,g} \\ y_{M,g} \\ z_{M,g} \end{bmatrix} = R(p_{V,g}, h_{V,g}, r_{V,g}) \begin{bmatrix} x_{M,v} \\ y_{M,v} \\ z_{M,v} \end{bmatrix} + \begin{bmatrix} x_{V,g} \\ y_{V,g} \\ z_{V,g} \end{bmatrix} \tag{2.14}$$

where $\begin{bmatrix} x_{M,g} & y_{M,g} & z_{M,g} \end{bmatrix}^{\mathrm{T}}$ is the measurement's location in the global coordinate frame, and the rotation matrix is defined by:

$$R(p,h,r) = \begin{bmatrix} \cos h \cos p & -\cos r \sin h + \cos h \sin r \sin p & \sin r \sin h + \cos r \cos h \sin p \\ \sin h \cos p & \cos r \cos h + \sin r \sin h \sin p & -\cos h \sin r + \cos r \sin h \sin p \\ -\sin p & \cos p \sin r & \cos r \cos p \end{bmatrix} \tag{2.15}$$

Using these equations, it becomes possible (although algebraically messy) to finally transform the measurements and their covariance matrices into the global coordinate frame, taking into account all the uncertainties. Although we omit the actual equations here for brevity, the procedure is outlined as follows:

1. Transform the noisy measurement into the global coordinate frame, to get a vector $\vec{r}$ in $\Re^3$.
2. Subtract away what would be the true measurement $\hat{\vec{r}}$ (*e.g.* $\vec{r}$ but with all the noise terms set to zero), to get $\tilde{\vec{r}} = \vec{r} - \hat{\vec{r}}$.
3. The full 3x3 covariance matrix is then $C = \tilde{\vec{r}}\,\tilde{\vec{r}}^{\mathrm{T}}$.
4. To reduce calculation time, we then apply the small-angle approximation in all off the noise terms (since we assume them to be small) and neglect higher-order noise cross terms, giving us $\tilde{C}$.

The result is a 3x3 matrix which describes the uncertainty ellipse associated with the measurement.

2.2 The Kalman Filter Framework

The next step in the algorithm is to fuse the transformed data with existing measurements using a Kalman filter, the details of which are described below.

2.2.1 Cell Update Equations

The basic premise of our Kalman filter framework is that the state we are estimating in each cell is the elevation of that cell, $z_{i,j}$, where the subscript i,j indicates the location of the cell in our discrete map. We begin by assuming that the state can be modeled as a standard discrete linear system of the form:

$$z_{i,j}(k+1) = Az_{i,j}(k) + Bu_{i,j}(k) + w(k) \tag{2.16}$$

$$\tag{2.17}$$

with a measurement of the form

$$y_{i,j}(k+1) = Cz_{i,j}(k+1) + v(k) \tag{2.18}$$

where $z_{i,j}(k)$ is the state of the system at time k, the matrices A, B, and C represent the linear model of the plant and the measurements, $u_{i,j}(k)$ is the input to the plant at time k, $y_{i,j}(k)$ is the measurement taken at time k, and $w(k)$ and $v(k)$ are independent, zero mean, white, Gaussian random variables that represent the process and measurement noise (respectively), with covariance matrices Q and R.

For this system, the general propagation equations for a discrete Kalman filter are (Welch and Bishop, 2004):

$$\hat{z}_{i,j}(k+1|k) = A\hat{z}_{i,j}(k|k) + Bu_{i,j}(k) \tag{2.19}$$
$$P_{i,j}(k+1|k) = AP_{i,j}(k|k)A^{\mathrm{T}} + Q \tag{2.20}$$

where as is standard, $\hat{z}_{i,j}(k+1|k)$ is the estimate of the state at time $k+1$ given the first k measurements, $\hat{z}_{i,j}(k|k)$ is the estimate of the state at time k given all of the measurements up to and including those at time k, $P_{i,j}(k+1|k)$ is the estimate error covariance at time $k+1$ given the first k measurements, and $P_{i,j}(k|k)$ is the estimate error covariance at time k given all of the measurements up to and including those at time k.

At this point, it would be possible to make assumptions about the relationship between the elevations of neighboring cells. For example, one could use the dynamic model of the elevation (*i.e.* the A or B matrices) to apply some sort of smoothness constraint, by relating the elevations of neighboring cells. Within the types of terrain we have encountered experimentally, however, we do not believe that any sort of smoothness condition applies; various natural as well as man-made obstacles (such as cliff faces, fence posts, or signs) introduce sudden large discontinuities in elevation which make it difficult to relate the elevation of a given cell to that of its neighbors. (Of course, once a measurement is taken, it is possible to extract information about multiple cells simultaneously - this point is addressed in section 2.2.2.)

Since we have ruled out any relationship between neighboring cells, and since we assume that the environment is static (and since it is not changing, not subject to noise itself) it follows that $A = 1$, $Q = 0$, and $B = 0$. Thus the propagation equations are simply:

$$\hat{z}_{i,j}(k+1|k) = A\hat{z}_{i,j}(k|k) \tag{2.21}$$
$$P_{i,j}(k+1|k) = P_{i,j}(k|k) \tag{2.22}$$

and so we do not need to perform any time updates. (This is fortunate, because performing some sort of time update to **every** cell in the map could be extremely computationally expensive, rendering the current algorithm useless in real-time applications.)

We next turn our attention to the measurement update equations. For the system we have described, the general update equations for a discrete Kalman filter are given by (Welch and Bishop, 2004):

$$\hat{z}_{i,j}(k+1|k+1) = \hat{z}_{i,j}(k+1|k) + K(k)\left(z_m - C\hat{z}_{i,j}\right) \tag{2.23}$$
$$P_{i,j}(k+1|k+1) = \left(I - K(k)C\right)P_{i,j}(k+1|k) \tag{2.24}$$

where z_m corresponds to a new elevation measurement for the cell (i, j), and the Kalman filter gain K is given by:

$$K = P_{i,j}(k+1|k)C^{\mathrm{T}}\left(CP_{i,j}(k+1|k)C^{\mathrm{T}} + R\right)^{-1} \tag{2.25}$$

As with the time update equations, the measurement update equations can be simplified. To begin with, since the coordinate transformations we perform give us immediate access to the elevation of the measurement, we have $C = 1$. Thus we can write:

$$\begin{aligned}
\hat{z}_{i,j}(k+1|k+1) &= \hat{z}_{i,j}(k+1|k) + \frac{P_{i,j}(k+1|k)\left(z_m - \hat{z}_{i,j}\right)}{P_{i,j}(k+1|k) + R} \\
&= \frac{\hat{z}_{i,j}(k+1|k)\left(P_{i,j}(k+1|k) + R\right) + P_{i,j}(k+1|k)\left(z_m - \hat{z}_{i,j}\right)}{P_{i,j}(k+1|k) + R} \\
&= \frac{R\hat{z}_{i,j}(k+1|k) + P_{i,j}(k+1|k)z_m}{P_{i,j}(k+1|k) + R}
\end{aligned} \tag{2.26}$$

and

$$\begin{aligned}
P_{i,j}(k+1|k+1) &= P_{i,j}(k+1|k) - \frac{P_{i,j}(k+1|k)P_{i,j}(k+1|k)}{P_{i,j}(k+1|k) + R} \\
&= \frac{P_{i,j}(k+1|k)^2 + P_{i,j}(k+1|k)R - P_{i,j}(k+1|k)^2}{P_{i,j}(k+1|k) + R} \\
&= \frac{P_{i,j}(k+1|k)R}{P_{i,j}(k+1|k) + R}
\end{aligned} \tag{2.27}$$

where we remind the reader that R is the error covariance of the new elevation measurement. In the next section, we explain how we determine R based on the 3D uncertainty ellipsoid of the measurement, $\tilde{C}$.

2.2.2 Measurement Discretization

Now that we have determined both the measurement mean and the appropriate covariance matrix, as well as how to update a cell given this information, we must choose which cells to update (and how to transform the 3D covariance matrix $\tilde{C}$ into a 1D variance R for use in the Kalman filter). As was indicated in the previous section, for each measurement we will update the estimate of the elevation of multiple cells by taking advantage of the fact that the uncertainty ellipsoid of a single measurement extends over several cells. There are essentially three methods one could use to perform this update.

1. Simply update all the cells in the map with the new measurement.

2. Update only the cells that fall within a fixed geometric pattern centered around the mean of the measurement (e.g. a square of a fixed side length or a circle of a fixed radius) regardless of the orientation and size of the uncertainty ellipsoid surrounding the measurement.

3. Update cells based on the probability that the measurement came from that cell, *i.e.* if the probability that a measurement came from a cell (as determined by the uncertainty ellipsoid generated by the error model) is greater than a given threshold, then we update that cell with the corresponding measurement.

For our implementation we dismissed the first method due to the latency it would introduce by requiring an update to every map cell for every single measurement. Instead, we chose to use an update algorithm that combines the second and third methods. In our update algorithm, we calculate which cells have centers that fall within the 95% confidence ellipsoid, and we then update only those cells whose distance from the mean of the measurement falls below a given threshold. For our experiment, this threshold was set to be 2m, which effectively limited the number of cells updated such that the algorithm could perform in real time on our computing system.

Because we are not estimating a single parameter (like the location of an object in some coordinate system), but are instead discretizing a single measurement so as to estimate several parameters (the elevations of the cells we choose to update), we must also find a way to discretize the covariance we use when we update those cells. In particular, we would like our discretization method to reflect the belief that cells further away from the mean of the measurement are less likely to contain whatever object generated that measurement, and thus should not be as heavily influenced by the measurement. This notion is captured by using the probability function $p_{i,j}(z)$, which describes the probability that the measurement actually came from the cell (i, j), from an object of height z, given the uncertainty ellipsoid of that measurement:

$$p_{i,j}(z) \;=\; \int_{C_x(i,j)-\frac{\Delta_x}{2}}^{C_x(i,j)+\frac{\Delta_x}{2}} \int_{C_y(i,j)-\frac{\Delta_y}{2}}^{C_y(i,j)+\frac{\Delta_y}{2}} p(x, y, z)\, dy\, dx \tag{2.28}$$

where $C_x(i, j)$ and $C_y(i, j)$ are the x and y coordinates of the center of cell (i, j), Δ_x and Δ_y are the width of the cell in the x and y directions (both constants, since we are not using a multi-scale map), and $p(x, y, z)$ is the model of the uncertainty ellipse surrounding the measurement, given by:

$$p(\mathbf{x}) \;=\; \frac{1}{(2\pi)^{3/2}\sqrt{\det \tilde{C}}} \exp\left[\frac{1}{2}(\mathbf{x}-\mu)^{\mathrm{T}}\, \tilde{C}^{-1}\, (\mathbf{x}-\mu)\right] \tag{2.29}$$

where $\mathbf{x} = [x\ y\ z]^{\mathrm{T}}$, and $\mu = [x_{M,g}\ y_{M,g}\ z_{M,g}]^{\mathrm{T}}$, the measurement itself.

Because equation 2.28 cannot be solved analytically, and would be difficult to solve numerically in real time (given that we must makes thousands of updates to cells per second), we instead approximate it by:

$$p_{i,j}(z) \;\approx\; p(C_x(i, j), C_y(i, j), z)\Delta_x\Delta_y \tag{2.30}$$

To make sure that cells closer to the measurement mean have a lower covariance when we fuse them using the Kalman filter, we simply take the inverse of this probability as our

measurement covariance, *i.e.*:

$$R = \frac{1}{p_{i,j}(z_{M,g})} \tag{2.31}$$

2.3 The Disappearing Obstacle Problem

One final topic we will cover is a solution to what we have termed the "disappearing obstacle" problem. Figure 2.6 is a diagram of this phenomenon; a description follows:

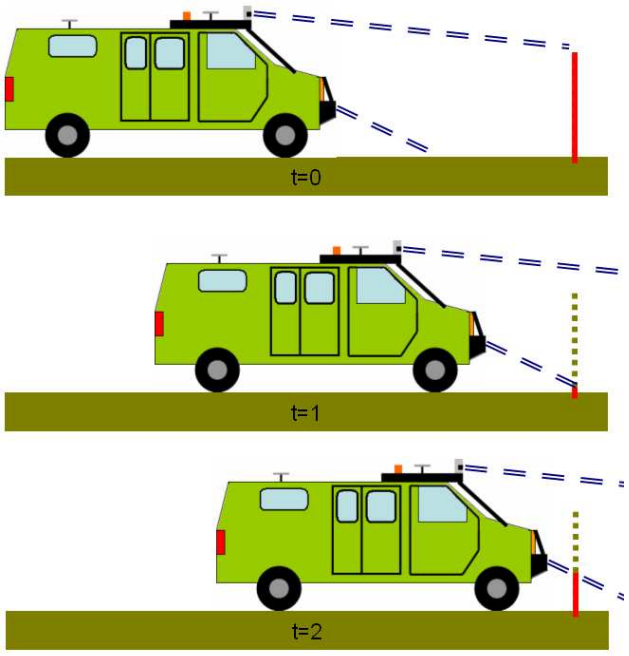

Figure 2.6: The "disappearing obstacle" problem. A long-range sensor detects the obstacle accurately at $t = 0$. A short-range obstacle then incorrectly states that the obstacle is only as high as the solid line at $t = 1$, when it is really as tall as the dotted line. At $t = 2$ the obstacle's height estimate indicates that it will not be traversable, but it is too late for the vehicle to react and choose another path.

1. $t = 0$: A long-range sensor detects an obstacle accurately (although with high covariance), and it is fused into the map as a series of cells with a given height and covariance.

2. $t = 1$: A short-range LADAR detects the lower portions of the obstacle prior to detecting the higher portions. The cells in the map temporarily appear as if the obstacle does not exist (or is simply very short) because the measurements from the short-range LADAR have a lower covariance than those from the long-range sensor, and so "overrule" the existing estimate.

3. $t = 2$: The obstacle's correct height is detected by the short-range LADAR, but it is too late for the vehicle to react in time to choose another path.

This phenomenon can occur in any system where there are vertical obstacles such as posts or trees, and close-range sensors with low covariances on their estimates which scan the terrain in a "push-broom" fashion as do LADARs. And while the phenomenon is temporary and the cells are eventually updated to reflect the obstacle's true height, this correction sometimes does not come quickly enough. In the time it can take for the short-range LADAR to sweep all the way up an obstacle (as the vehicle drives forward), the vehicle's planning algorithms can attempt to plan a path through the area once occupied by the now vanished obstacle. In fact, this behavior was encountered many times over the course of testing for the 2005 DARPA Grand Challenge; the disappearing obstacle problem frequently caused Alice to attempt to take shortcuts over obstacles it thought for a moment were no longer there, only to find out too late that the obstacle was, in fact, still there.

We make use of the probabilistic nature of our new framework to help solve this problem. In particular, we use a sequential probability ratio test (hereafter referred to as an SPRT) to determine whether or not the new measurements coming from the short-range LADAR should be fused into, and in some sense replace, the existing data in a given cell. For every new measurement that comes into a non-empty cell, we test the simple hypothesis that either:

H_0**:** The new data corresponds to the same obstacle previously detected in that cell, and fusing the new data with the old data will not cause the existing obstacle to disappear, or,

H_1**:** The new data does **not** correspond to the same obstacle previously detected in that cell, but it will result in a more accurate estimate of the elevation of that cell (*i.e.* the old data was inaccurate and should be replaced).

In either case, if the hypothesis is proven true then the proper course of action is to fuse in the new data.

Of course, we can determine which of H_0 or H_1 is true simply by examining the values of the new measurements. If they are "close" to the existing value then we can be fairly sure H_0 is true. If they are "far" from the existing value, then the old data is probably inaccurate and H_1 is likely to be true. We will determine whether the measurements are "close" by testing whether or not they correspond to a normal distribution with mean given by the existing estimate, and variance given by the variance in elevation of the measurements that have been fused into that cell, weighted by their covariance. This variance is given by

$$\sigma_{i,j}^2 = \frac{\sum_{k=0}^{n} \left[R_k \left(\hat{z}_{i,j} - z_{M,g_k} \right)^2 \right]}{\sum_{k=0}^{N} R_k} \tag{2.32}$$

where n measurements $(z_{M,g_1}, \ldots, z_{M,g_n})$ have fallen into the cell (i, j), whose current estimate is $z_{i,j}$.

Now that we have the mean ($\hat{z}_{i,j}$) and variance ($\sigma_{i,j}$), we can proceed to describe the specific SPRT we perform. Following Wald (1947), we note that the probability density of a sequence of n measurements of the elevation of the cell $(z_{M,g_1}, \ldots, z_{M,g_n})$ under our hypothesis is:

$$p_{0n} = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma_{i,j}^n} \exp\left(-\frac{1}{2\sigma_{i,j}^2} \sum_{i=1}^{n} (z_{M,g_i} - \hat{z}_{i,j})^2\right) \tag{2.33}$$

According to Wald (1947), we want to compare this probability density to a probability density p_{1n} which is a weighted average of the probability density corresponding to various values of $z_{i,j}$ for which we would reject H_0. As derived in Wald (1947), an optimal choice for p_{1n} is:

$$\begin{aligned} p_{1n} = {} & \frac{1}{2\,(2\pi)^{\frac{n}{2}} \sigma_{i,j}^n} \exp\left(-\frac{1}{2\sigma_{i,j}^2} \sum_{i=1}^{n} (z_{M,g_i} - \hat{z}_{i,j} + \delta\sigma_{i,j})^2\right) + \\ & \frac{1}{2\,(2\pi)^{\frac{n}{2}} \sigma_{i,j}^n} \exp\left(-\frac{1}{2\sigma_{i,j}^2} \sum_{i=1}^{n} (z_{M,g_i} - \hat{z}_{i,j} - \delta\sigma_{i,j})^2\right) \end{aligned} \tag{2.34}$$

which is simply the average of the two density functions corresponding to $z_{i,j} = \hat{z}_{i,j} + \delta\sigma_{i,j}$ and $z_{i,j} = \hat{z}_{i,j} - \delta\sigma_{i,j}$, where $z_{i,j}$ is the actual value of the elevation of the cell (i,j), and δ is given by:

$$\left|\frac{\hat{z}_{i,j} - z_{i,j}}{\sigma_{i,j}}\right| \geq \delta \tag{2.35}$$

where we wish to reject the hypothesis H_0 if this inequality is true. Of course, rejecting H_0 is equivalent to accepting H_1.

Then, for some thresholds A and B, the test would normally be carried out as follows:

1. When a new measurement of a cell comes in, check to see if any data already exists in that cell. If less than two data points have been used to estimate the state of the current cell, fuse in the new measurement immediately. Otherwise:

2. Calculate p_{1n}/p_{0n}, as given by equations 2.34 and 2.33.

   (a) If $p_{1n}/p_{0n} \leq B$, accept H_0, and fuse in all of the data in the current cell's buffer using the Kalman filter procedure described in equations 2.26 and 2.27, and keep track of the variance in elevation using equation 2.32.

   (b) If $p_{1n}/p_{0n} \geq A$, accept H_1, and fuse in all of the data in the current cell's buffer using the Kalman filter procedure described in equations 2.26 and 2.27, and keep track of the variance in elevation using equation 2.32.

   (c) If $B \leq p_{1n}/p_{0n} \leq A$, add the measurement to the buffer and continue taking measurements.

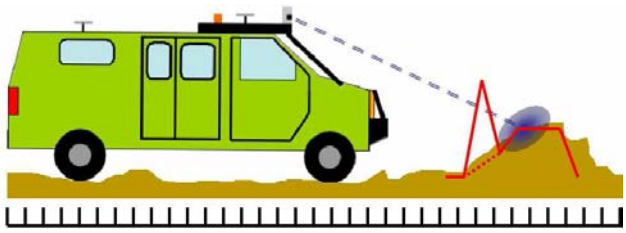

Figure 2.7: Assume the current estimate of the elevation is given by the solid red line, and the new estimate that would result from fusing in new sensor data is given by the dashed red line. If we then receive a LADAR measurement as shown in blue, there is a high probability that the existing sharp peak in elevation is incorrect, and we should proceed with fusing in the new data.

One can see that since the acceptance of either hypothesis leads to the same action, the SPRT essentially serves as a mechanicsm to delay fusion of new range data. One possible drawback to this approach is that it could result in the delayed detection of obstacles in cases where the old data did not detect the obstacle, but the new, delayed data does. Despite this, we believe that this drawback will be less dangerous than the disappearing obstacle problem which the SPRT seeks to solve.

In the next section, we describe our implementation of this algorithm, as well as the Kalman filter framework described above. Before we continue, though, we point out to the reader that there is an additional source of information which we have not tapped — that of which cells were **not** occluded by an obstacle. Figure 2.7 illustrates our point: if a measurement comes in which indicates that an existing elevation estimate is incorrect because it would have caused a sensor to pick up an obstacle at a closer range than was actually detected, then we can take this to be a "null" measurement in some sense. (Of course, this assumption is invalid in the case of partially transparent obstacles.) Unfortunately we were not able to find an efficient way to integrate this information into our framework, especially in a manner that would be generalizable to both LADAR and stereovision. This may be due in some part to the framework itself — a 3D voxel map may be more amenable to "null" measurements indicating which cells should be empty than our 2.5D DEM. Regardless, we believe using these "null" measurements could greatly improve map accuracy, as we will mention in our notes on future work in section 5.

Chapter 3

Implementation and Testing

To test the algorithm we have described, we implemented it within the framework of Team Caltech's entry in the 2005 DARPA Grand Challenge, Alice. In this section, we describe in more detail the various hardware and software used for this implementation; most of this information is also available in Cremean et al. (2006), which is more broadly focused.

3.1 Experimental Platform

Alice, the UGV used for testing, is a Ford E350 van modified by Sportsmobile West of Fresno, CA, for off-road operations. While the hardware modifications that allow Alice to perform autonomously off-road are nontrivial and required a significant amount of effort, for the purposes of this thesis they are not nearly as important as Alice's software architecture. Thus, for more information on the hardware aspects of the testing platform, we direct the reader to Cremean et al. (2006).

The software system used on Alice was designed to be modular and easily adaptable (see figure 3.1). Each sensor has a dedicated software module known as a "feeder," which reads in data from that sensor over a physical connection (such as a USB or IEEE 1394 port), and then broadcasts that data to Alice's internal LAN, where it can be received by any

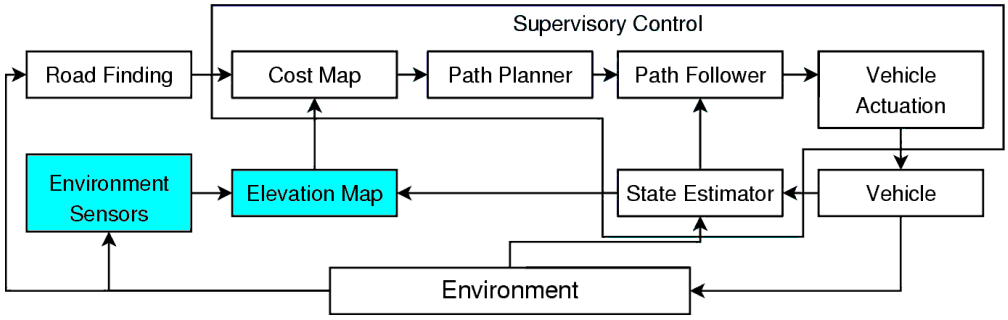

Figure 3.1: Alice's software architecture. The portions relevant to this thesis were the environment sensors (at the bottom left) and the elevation map.

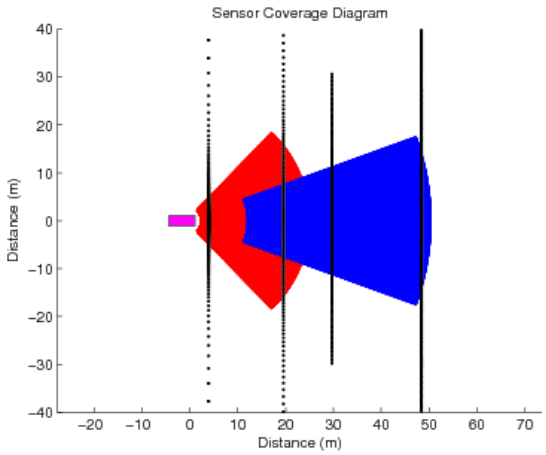

Figure 3.2: Alice's sensor coverage. The wider cone corresponds to the short-range stereovision; the longer, thinner cone corresponds to the long-range stereovision. The black lines correspond to the intersection of the ground plane with the various scan planes of the LADAR units. The small rectangle is Alice, approximately to scale.

other software module running on a computer connected to the network. One such module is the mapping module, which then combines the data from the different sensors into a single cost map of Alice's environment. The cost-map itself, as well as the elevation maps, are implemented using a scrolling framework in which the vehicle is always located at the center of the map, and the edges of cells are oriented along the northing-easting coordinate system. That cost-map is then broadcast to the vehicle's path planning module, which uses the map to compute an appropriate path. Although communications between the various components of the system are based on message passing over a gigabit Ethernet network, we did not encounter any problems with dropped packets or lack of bandwidth. Throughout the experiment, we used maps that were 200m on a side, with cells that were 40cm on a side.

3.2 Sensors Used and Verification of Error Models

Prior to the 2005 DARPA Grand Challenge, members of Team Caltech, with help from the Aluminess company, outfitted Alice with several sets of semi-redundant sensors. For a complete description of the sensors used, see table 3.1; figure 3.2 shows the coverage of the sensors on flat, level terrain; figure 3.3 shows where the sensors are actually located on Alice.

Of course, it was necessary to verify that the sensor error models we presented earlier were accurate, *i.e.*, that they had the statistical properties we assumed. To ensure this, we collected large samples of data points from both types of sensors, and then analyzed their statistical properties. The results, shown in figures 3.4 and 3.5, demonstrate that our assumptions hold.

In particular, in figure 3.4, we can see that the distribution of measurements from a single LADAR scan (averaged over many thousands of scans) are indeed Gaussian.

Sensor Type	Mounting Location	Specifications
LADAR (SICK LMS 221-30206)	Roof	180° FOV, 1° resolution, 75Hz, 80m max range, pointed 20m away
LADAR (SICK LMS 291-S14)	Roof	90° FOV, 0.5° resolution, 75Hz, 80m max range, pointed 35m away
LADAR (Riegl LMS Q120i)	Roof	80° FOV, 0.4° resolution, 50Hz, 120m max range, pointed 50 m away
LADAR (SICK LMS 291-S05)	Bumper	180° FOV, 1° resolution, 80 m max range, pointed 3m away
LADAR (SICK LMS 221-30206)	Bumper	180° FOV, 1° resolution, 80m max range, pointed horizontally
Stereovision Pair (Point Grey Dragonfly)	Roof	1 m baseline, 640x480 resolution, 2.8 mm focal length, 128 disparities
Stereovision Pair (Point Grey Dragonfly)	Roof	1.5 m baseline, 640x480 resolution, 8 mm focal length, 128 disparities
Road-Finding Camera (Point Grey Dragonfly)	Roof	640x480 resolution, 2.8mm focal length
IMU (Northrop Grumman LN-200)	Roof	1–10° gyro bias, 0.3–3 mg acceleration bias, 400 Hz update rate
GPS (Navcom SF-2050)	Roof	0.5 m CEP, 2 Hz update rate
GPS (NovAtel DL-4plus)	Roof	0.4 m CEP, 10 Hz update rate

Table 3.1: The various terrain sensors used on Alice on the day of the Grand Challenge. Not all of these sensors were used at all points during the current experiment.

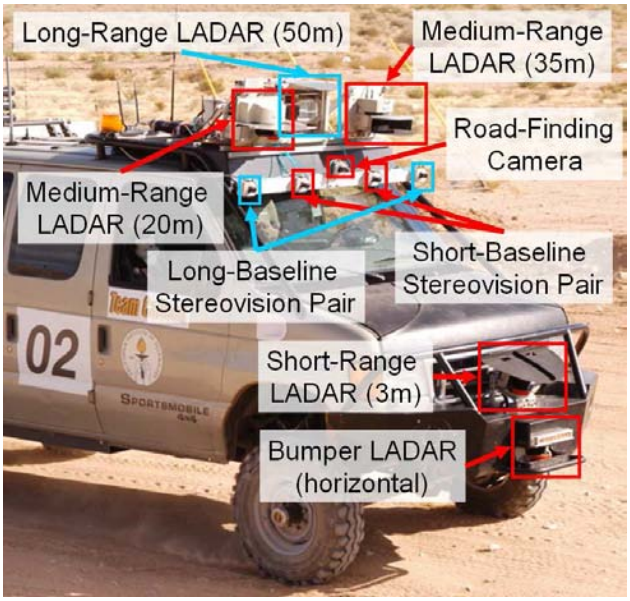

Figure 3.3: Alice, with terrain sensors labeled. (GPS and IMU are not labeled.)

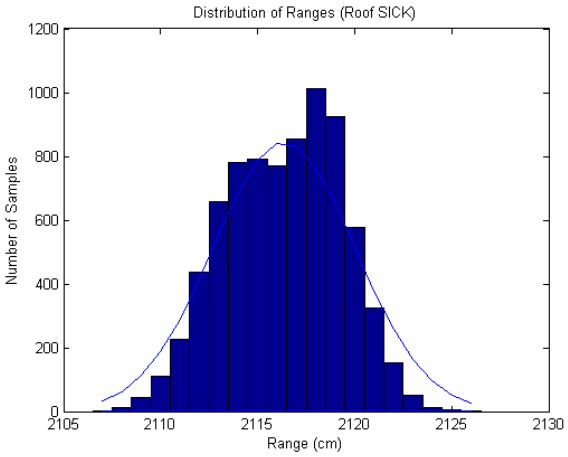

Figure 3.4: Distribution of scan measurements from a single LADAR unit, in this case the roof-mounted SICK LMS 221-30206, and the corresponding Gaussian error model.

In figure 3.5, we have plotted the mean and standard deviations of the disparity (averaged over more than one hundred images). The overall variance of the disparity for the pixels in the image was $\sigma_d = 0.7256$.

3.3 Experimental Procedure

To test the final system, we drove the UGV manually through two different types of terrain, as depicted in figure 3.6. The first type of terrain was a flat dry lakebed, populated by obstacles comprised of simple geometric shapes where ground-truth was approximately known, so that we could perform calibrated tests on the accuracy of our system's ability to properly estimate elevation. The second type of terrain was a more realistic desert environment which was chosen so as to verify that our algorithm would successfully achieve three goals:

1. Comparable accuracy in elevation to the old algorithm, along with reasonable covariance values.

2. Reduced areas of no data in comparison to the old algorithm.

3. Elimination of the disappearing obstacle problem.

For this terrain, which was composed of a bumpy dirt road along with obstacles such as sage brush and small rocks, ground truth was not known, so only qualitative comparisons were made.

During testing the raw sensor data was timestamped and logged so that it could be post-processed and analyzed at a later date. However, the final algorithm we developed was still able to process the data in real time. The results of that analysis are the topic of the next section.

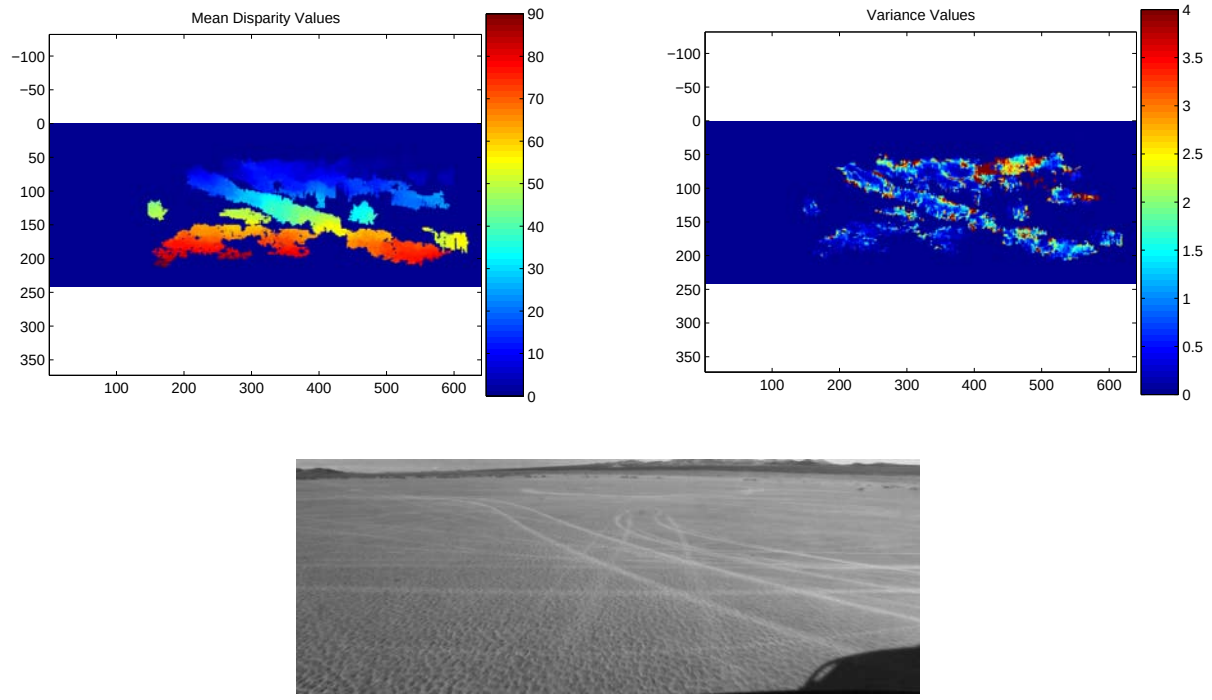

Figure 3.5: On the left are mean disparity measurements from a set of stereovision images taken by the short-range (1m baseline) pair while the vehicle was stationary. The disparity of each pixel is indicated by its color, according to the colorbar on the right. On the right is the variance in disparity at each pixel. Again, the value of the variance at each pixel is indicated by its color, according to the colorbar on the right. In both cases, dark blue pixels at the bottom of the scale indicate no disparity was estimated by the stereovision algorithm due to lack of texture in the image. At the bottom is a sample image from the sequence used during this test.

Figure 3.6: Two representative images of the types of desert terrain in which the system was tested. On the left is a sample desert road, lined with small rocks and brush. On the right is a sample dry lakebed, the elevation of which we assumed was completely flat so as to approximate ground-truth.

Chapter 4

Results and Discussion

Our major results are fourfold, and are described below. First, in section 4.1, we show that our algorithm generates maps with the elevation and covariance values we would expect given the different sensors that have filled in the map. Second, in section 4.2, we show that our algorithm dramatically reduces the "no data" problem. Third, in section 4.3 we show that our algorithm successfully solves the "disappearing obstacle" problem. Finally, we close with section 4.4 by mentioning a few unanticipated results of our algorithm.

4.1 Map Accuracy and Covariance Characteristics

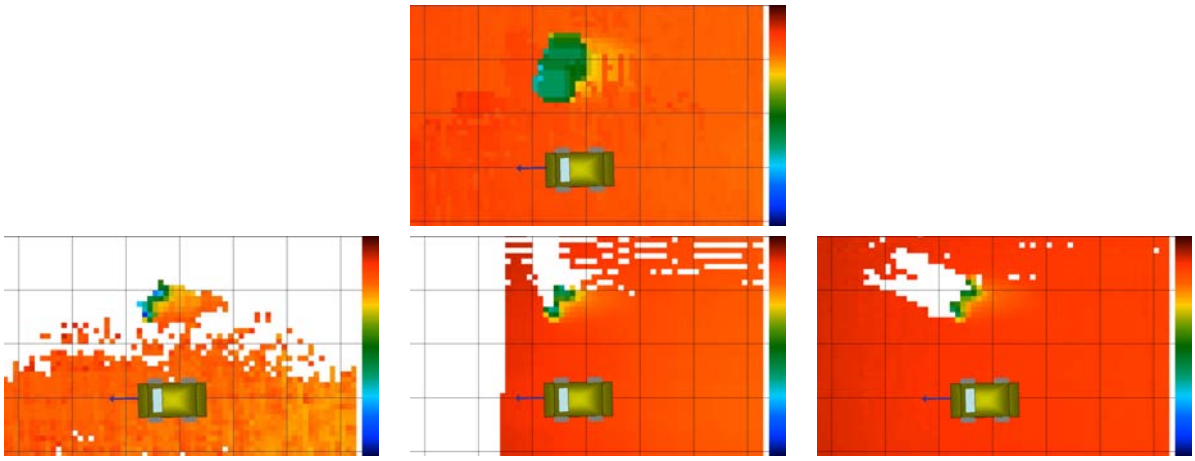

Figure 4.1: A sample elevation map generated by our algorithm (top) and by individual sensors using the old algorithm (bottom) in conditions for which ground truth is approximately known. The ground-plane (a dry lakebed) is assumed to be flat, and the obstacle (the green blob in the maps) was a 1m by 1m flat board. The elevation of a cell is given by its color — white indicates no data, blue indicates higher elevation, and brown indicates lower elevation, with an overall range of approximately 1m. For scale, the gridlines are 4m apart.

Figure 4.1 shows a sample elevation map generated as a result of our algorithm. The estimated height of the obstacle, which is 1m tall, is approximately 50cm. This disparity is to be expected — our algoirhtm fuses together all of the range data from the obstacle (from its top to its bottom) and so the resulting height estimate is close to the obstacle's mean elevation. We also note that the obstacle appears much larger in the map than it actually is — this phenomenon is explained in more detail in section 4.4. The height estimates in the individual elevation maps (shown on the bottom) are comparable, indicating no degradation in performance from the use of our framework.

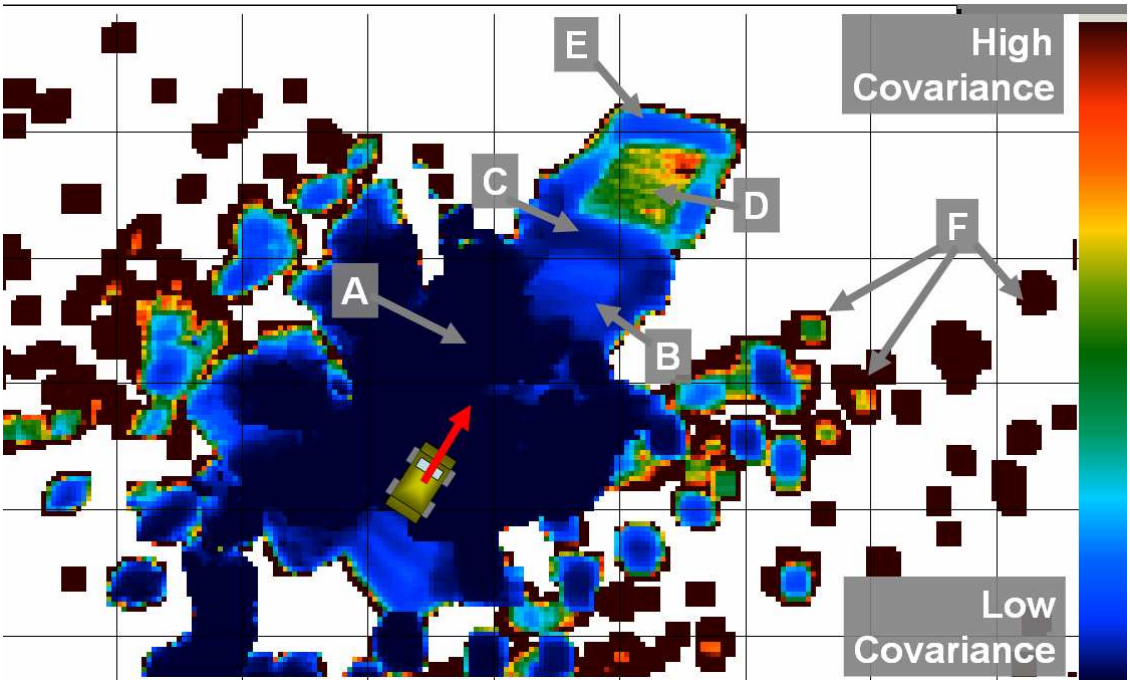

Figure 4.2: A sample covariance map generated by our algorithm. The covariance value of each cell is indicated by its color. Blue indicates lower covariance, while brown indicates higher covariance. The vehicle's approximate position and direction of travel are given by the red arrow. The terrain is a relatively flat desert road, with brush on either side. A description of the significance of each labeled region is given in the text.

Figure 4.2 shows a sample covariance map generated as a result of our algorithm. For this example we used the short-range (1m baseline) stereovision pair, as well as the roof-mounted SICK LMS 221-30206 and SICK LMS 291-S14. Of course, our algorithm is not limited to using only three different sensors — in this example we used only three for simplicity of presentation. In particular, one can pick out which regions were filled in by which sensors based on the values of the covariance in those regions, as described below:

Region A: This darkest region, the region with the lowest covariance, was generated by the combination of all three sensors.

Region B: This light-blue region was generated by coverage of both the short- and long-range LADAR units, but not the stereovision pair. Thus the covariance is slightly higher than that in region A.

Region C: This darker region was generated as a result of multiple scans by the short-range LADAR of the same patch of ground due to the vehicle being stationary. Because more measurements have been taken, the covariance is lower than that of region B.

Region D: This lighter region has been seen only by the long-range LADAR, thus it has the highest covariance of any area in front of the vehicle.

Region E: As with region C, this region is slightly darker than region D (and thus has a lower covariance) because the long-range LADAR was able to take multiple scans of this area due to the vehicle being stationary.

Region F: The spotty regions to either side of the road the vehicle is traveling on are a result of measurements of nearby brush. Because measurements are less dense to the sides of the vehicle than directly in front of the vehicle, the covariance is the highest in these areas.

4.2 Data Smoothing Results

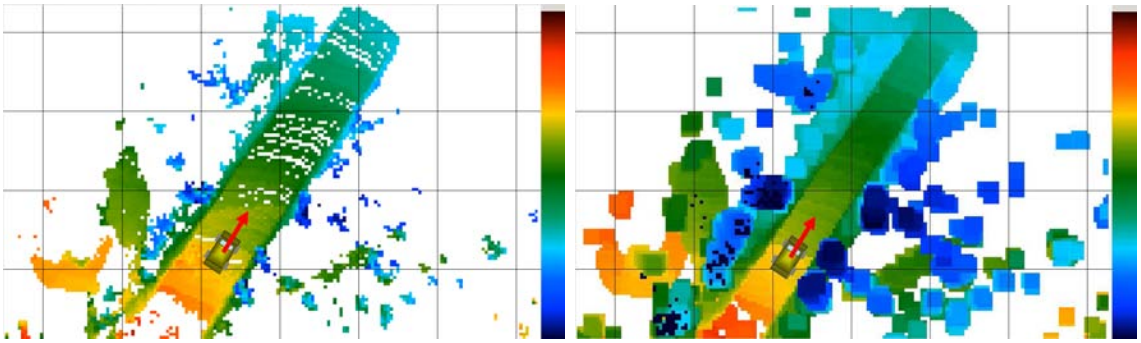

Figure 4.3: On the left is a sample map generated using the old algorithm, which is dotted with cells containing no data. On the right is a sample map generated using our algorithm, which has significantly fewer no-data cells. In these maps, the color of a cell corresponds to the estimate of its elevation, with blue cells being higher, red cells being lower, and white cells indicating no data.

Figure 4.3 demonstrates how our algorithm reduces the no-data problem mentioned in the introduction, even when only one sensor is used. In this example case we use only the SICK LMS 291-S14 to generate the maps. This example shows what is potentially a worst-case scenario for the old algorithm; the old algorithm was capable of doing interpolation across small gaps of no data, which is not shown here. Even so, our algorithm still performs

better in that its "interpolation" is performed probabilistically as a function of the incoming measurements, and makes no assumptions about the smoothness of the map.

4.3 Disappearing Obstacle Problem Results

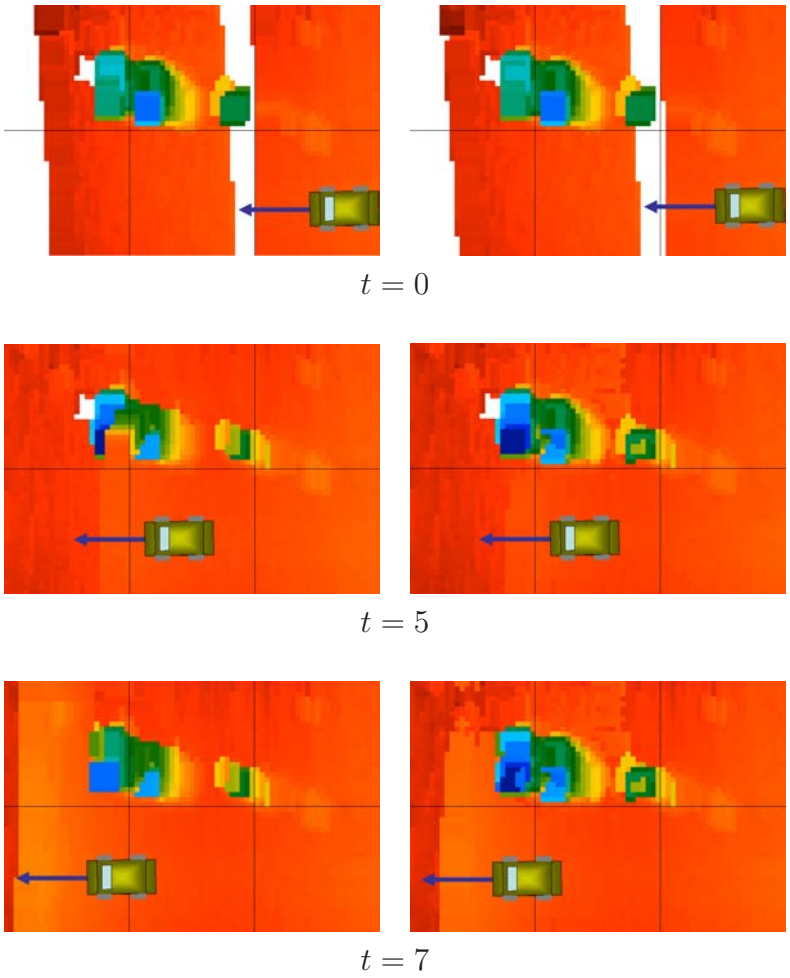

Figure 4.4: On the left is a sequence of elevation maps showing the fused map that results from using the old algorithm, which did not adequately solve the disappearing obstacle problem. On the right is a sequence of maps, taken at the same time steps, illustrating that our method eliminates this problem.

Figure 4.4 demonstrates how our algorithm performs when confronted with data that, in the old algorithm, would have lead to a disappearing obstacle scenario. Both cases start out the same way: at $t = 0$, the obstacle on the left (the blue-green blob, which in reality is a bush) has been detected by the long-range LADAR. At time $t = 5$, however, the short-range LADAR scans over the obstacle, causing parts of it to disappear in the map on the left,

while it is preserved in the map on the right. Finally, at $t = 7$, the vehicle has passed the obstacle and it has reappeared in both maps. Although the algorithm performed correctly on this specific set of data, there were cases for which the obstacle still disappeared (albeit to a smaller extent than before we began using the SPRT). Additionally, a great deal of trial-and-error was involved in choosing values of A, B, and δ which would result in the algorithm performing correctly.

4.4 Unanticipated Results

Finally, we close this section by noting some interesting unanticipated characteristics of our solution to the elevation-fusion problem not mentioned above.

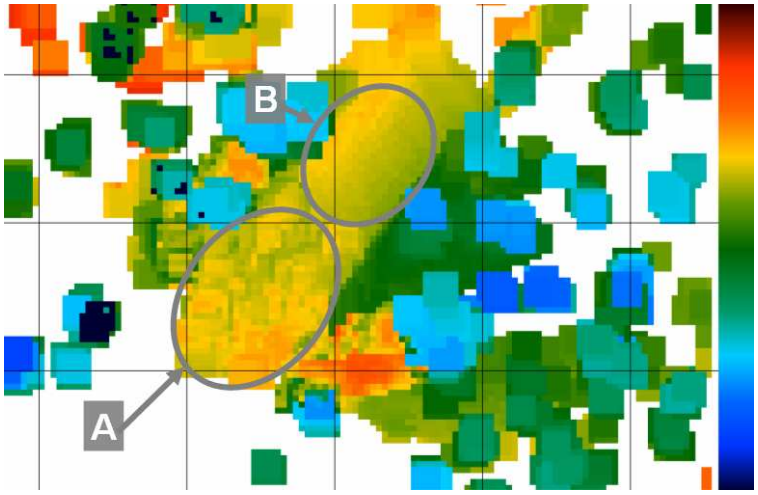

Figure 4.5: An example of data from stereovision overpowering data from a LADAR unit. The region labeled "B" has been seen by only the LADAR unit, and has a smooth elevation (due to the high accuracy of LADAR). In contrast, the region labeled "A" has been seen by both stereovision and LADAR data, but the elevation in this region is very rough due to the more numerous measurements from stereovision overpowering those of the LADAR, as described in more detail in the text.

The first such characteristic is that measurements taken using stereovision tend to "overpower" measurements taken by LADAR units despite the relative inaccuracy of stereovision compared to that of LADAR, as seen in figure 4.5. (By "overpower", we mean that if a stereovision system and a LADAR unit both measure the elevation of the same terrain, the resulting estimate will be more heavily influenced by the measurements generated by the stereovision system than by those generated using the LADAR unit.) This bias towards stereovision is due in large part to the fact that the number of measurements stereovision provides from a single image is an order of magnitude greater than the number of measurements LADAR produces from a single scan. In a sample stereovision image, the number of measurements that fall into a single cell may be in the dozens, while for a LADAR scan

that number is rarely higher than two or three. Additionally, for a given frame the measurements from stereovision that fall into a single cell tend to be highly correlated. (In other words, the Gaussian noise occurs between the disparity values a pixel has in separate images, not between the disparity values individual pixels have within a single image.) Thus, when the fusion is performed, the more numerous and highly correlated measurements from stereovision tend to outweigh the measurements generated via LADAR. While this does not necessarily represent a flaw in the system, it may be desirable to investigate an alternative error model for the stereovision system that takes into account the highly correlated nature of the measurements generated by stereovision for a given image.

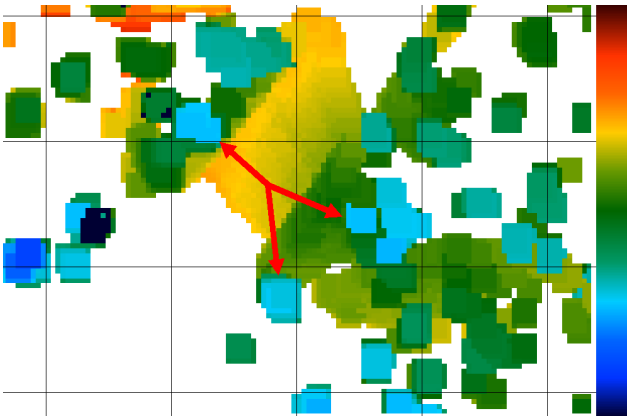

Figure 4.6: An example of the data smoothing/obstacle growing that results from using our algorithm. The growing phenomenon (which is described in the text) makes the obstacles indicated by the red arrows appear larger in the map than they are in reality.

An additional characteristic of our algorithm is the apparent smoothing effect it has on obstacles (*i.e.* vertical terrain features), as in figure 4.6. This phenomenon is generated only by LADAR measurements, and is similar to the previous phenomenon in which stereovision measurements "overpowered" LADAR measurements. In particular, this phenomenon is a result of two aspects of our system. The first is the fact that LADAR units will generate more measurements for a vertical terrain feature than for a horizontal one, since the LADAR scan plane will intersect a vertical feature at multiple elevations as the vehicle drives forward. The second is that we use the uncertainty ellipsoid surrounding the measurement to update multiple cells for a given measurement. As a result, the cells surrounding the measurement of a vertical obstacle are updated more times with data generated by that vertical obstacle than by data coming from the actual terrain in that cell. The result is that obstacles appear to be grown and smoothed within the elevation map, to an extent that varies proportionally to the uncertainty ellipse surrounding the measurement. Thus measurements of obstacles with a larger uncertainty ellipse tend to be grown more, while those with more accurate measurements are grown less. Of course, this is not necessarily a flaw — the designer of the path-planning algorithm, for example, can make use of this "obstacle growing" to treat the vehicle as a point within the map, thereby potentially reducing the computational complexity of the planning algorithms.

Chapter 5

Conclusions and Future Work

In this thesis we have presented a probabilistic framework for performing real-time mapping aboard an unmanned ground vehicle. Our framework has successfully solved some of the problems that plagued the previous heuristic solution used by Team Caltech on its 2005 DARPA Grand Challenge entry. In particular, we have presented error models for two different types of range sensors, and made use of Kalman filters to fuse together elevation measurements from those different sensors using their corresponding error models. Additionally, we have added safeguards that take advantage of the probabilistic nature of our framework to prevent obstacles from temporarily disappearing from the map. The resulting algorithm solves the "disappearing obstacle problem," reduces the number of "no-data" cells in the map, and provides a covariance map which indicates the algorithm's confidence in each cell's estimated elevation.

A problem we have not covered in this thesis, but which is of critical importance within the overall software architecture used by Caltech's vehicle, is the problem of how to convert elevation maps into goodness maps. For the 2005 Grand Challenge, a heuristic method was used to convert elevation maps into goodness maps. However, due to the heuristic nature of the elevation fusion, no confidence data was available for use during this conversion, and so Alice drove at the same speed through terrain seen with its less accurate long-range sensors as it did through terrain seen with its more accurate short-range sensors. In fact, this lack of distinction between which sensors had seen (or not seen) the terrain surrounding Alice was one of the major causes of the Caltech vehicle's failure to complete the 2005 Grand Challenge. By incorporating the covariance values generated by our algorithm (*e.g.*, by setting cells with a high covariance to have a lower goodness, and vice versa) this sort of failure can be avoided in the future, and we strongly recommend that this adaptation be investigated in future research.

There are also other potential extensions to our algorithm which have not yet been explored. As mentioned in section 2.3, one potential extension would be to augment our current range measurements with "null" measurements that indicate areas where there are no obstacles, so as to better eliminate any spurious obstacles that are detected as a result of sensor noise. This extension could be implemented both for stereovision and LADAR, since range measurements from either could be used to determine rays along which no solid

obstacle exists.

Another major such extension would be to use a multi-resolution map for storing the elevation data — that is, a map that does not have a fixed resolution, as ours did, but instead automatically subdivides into smaller cells as new measurements are taken and smaller features are detected. Alternatively, one could abandon the discretized nature of a grid-based map altogether, and attempt to group measurements together into individual obstacles or features that were not restricted to specific grid cells, but whose positions and other characteristics were instead freely estimated using the incoming range measurements. In essence, this framework would model the world in a vectorized manner, as opposed to the rasterized model we have used.

Before we conclude, there is one final issue which we feel needs to be addressed given the overall framework within which the research presented in this thesis fits. In particular, this thesis has been the culmination of several years of work on the 2004 and 2005 DARPA Grand Challenges. Recently, however, DARPA has announced a third Grand Challenge, to be held in November 2007, which will focus primarily on autonomous driving in urban, dynamic environments. It is clear that this urban Grand Challenge (DGC3) will present difficulties that are somewhat different from those of the previous two Grand Challenges (DGC1-2). The most notable difference, of course, is that in DGC1-2, the assumption that the environment would be static was a valid one, whereas in DGC3, the environment will be dynamic, *i.e.*, it will contain moving obstacles such as other vehicles. Because this static assumption was critical for the mapping framework developed for DGC1-2 and presented here, it is clear that additional work will be required for the Team Caltech vehicle to be able to compete in DGC3. In particular, we believe that navigation in an urban environment will require the vehicle to have a great deal more understanding and awareness of its environment. It will no longer be adequate for the vehicle to treat all terrain features the same based solely on their rough geometry, *e.g.*, treating vehicles and large boulders the same, since they have roughly the same overall size and shape in an elevation map. Instead, the vehicle will have to actively recognize objects like vehicles, and attempt to model their dynamics to predict how they will behave. The rasterized mapping framework used for DGC1-2 may not be sufficient for this aspect of DGC3; a framework closer to the vectorized one described in the previous paragraph may be needed.

Despite this fundamental shift in approach, we believe that many of the broader lessons learned over the course of this research will be applicable to future work on DGC3. For example, if both stereovision and LADAR continue to be used as the vehicle's primary range sensors, the error models presented in this thesis will continue to be applicable. Additionally, the new framework that is developed should be designed so as to avoid issues like the disappearing obstacle problem and the no data problem. And of course the benefits of a probabilistic framework are now obvious; we believe that some of the same techniques used in this thesis could be adapted for use in estimating parameters of distinct objects in the vehicle's environment. Regardless of the framework used for DGC3, the completion of this thesis indicates that the torch is being passed to a new team of Caltech students and we wish them luck in their continued development of algorithms for autonomous navigation.

Chapter 6

Acknowledgements

First and foremost, I would like to thank my thesis advisor, Professor Joel Burdick, for being an extremely patient and flexible mentor over the course of this project. I also want to thank Professor Richard Murray for his inspiration and leadership — without his support, this thesis (not to mention a lot of other research projects) never would have happened. Additionally, I want to thank Professor Pietro Perona for his assistance and advice over the course of the DARPA Grand Challenge. Finally, I want to thank all three of these men for all of their advice on matters both inside and outside the technical realm — their insight and support has been invaluable throughout my undergraduate career.

I also wish to thank all of my friends on Team Caltech who helped make Alice (and Bob!) possible, especially: Lars Cremean, Kristo Kriechbaum, Sam Pfister, Alex Stewart, Dima Kogan, and Joe Donovan. Most of all, I want to especially recognize Tony and Sandie Fender, without whose support the team would have collapsed from starvation and heat exhaustion long ago.

And last, but not least, thank you Alice, for never letting us down.

Jeremy Gillula
June 2nd, 2006

Bibliography

Bellutta, P., Manduchi, R., Matthies, L., Owens, K., and Rankin, A. (2000). Terrain perception for DEMO III. In *Proceedings of the 2000 Intelligent Vehicles Conference*, Dearborn, MI.

Cremean, L. B. (2006). *System Architectures and Environment Modeling for High-Speed Autonomous Navigation*. PhD thesis, Caltech.

Cremean, L. B., Foote, T. B., Gillula, J. H., Hines, G. H., Kogan, D., Kriechbaum, K. L., Lamb, J. C., Leibs, J., Lindzey, L., Stewart, A. D., Burdick, J. W., and Murray, R. M. (2006). Alice: An information-rich autonomous vehicle for high-speed desert navigation. *Journal of Field Robotics*.

Goldberg, S., Matthies, L., and Maimone, M. (2002). Stereo vision and rover navigation software for planetary exploration. In *Proceedings of the 2002 IEEE Aerospace Conference*, Big Sky, MT.

Luo, R. and Kay, M. (1989). Multisensor integration and fusion in intelligent systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):901–931.

Matthies, L. (1992). Toward stochastic modeling of obstacle detectability in passive stereo range imagery. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Champaign, IL.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42.

Wald, A. (1947). *Sequential Analysis*. Wiley Publications in Statistics. John Wiley and Sons, Inc., New York.

Welch, G. and Bishop, G. (2004). An introduction to the kalman filter. Technical Report 95-041, University of North Carolina at Chapel Hill.

Ye, C. and Borenstein, J. (2002). Characterization of a 2-D laser scanner for mobile robot obstacle negotiation. In *Proceedings of the IEEE Conference on Robotics and Automation*, Washington, D.C.