2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Gamma-SLAM: Using Stereo Vision and Variance Grid Maps for SLAM in Unstructured Environments

Tim K. Marks, Andrew Howard, Max Bajracharya, Garrison W. Cottrell, and Larry Matthies

Abstract— **We introduce a new method for stereo visual SLAM (simultaneous localization and mapping) that works in unstructured, outdoor environments. Unlike other grid-based SLAM algorithms, which use occupancy grid maps, our algorithm uses a new mapping technique that maintains a posterior distribution over the height variance in each cell. This idea was motivated by our experience with outdoor navigation tasks, which has shown height variance to be a useful measure of traversability. To obtain a joint posterior over poses and maps, we use a Rao-Blackwellized particle filter: the pose distribution is estimated using a particle filter, and each particle has its own map that is obtained through exact filtering conditioned on the particle's pose. Visual odometry provides good proposal distributions for the particle pose. In the analytical (exact) filter for the map, we update the sufficient statistics of a gamma distribution over the precision (inverse variance) of heights in each grid cell. We verify the algorithm's accuracy on two outdoor courses by comparing with ground truth data obtained using electronic surveying equipment. In addition, we solve for the optimal transformation from the SLAM map to georeferenced coordinates, based on a noisy GPS signal. We derive an online version of this alignment process, which can be used to maintain a running estimate of the robot's global position that is much more accurate than the GPS readings.**

I. Introduction

The task of simultaneous localization and mapping (SLAM) is to estimate from a temporal sequence of observations both a map of the environment and the pose (position and orientation) of the observer (the robot) in this map. Like many SLAM systems (e.g., [1; 2; 3]), our system uses a Rao-Blackwellized particle filter [4]: Each particle has a single hypothesis about the robot's pose; based on its pose history, each particle uses an exact filter to obtain a map. However, our algorithm's exact filter and map representation differ from those of existing algorithms.

Most current approaches to SLAM use laser range scans for their observations. Many of these laser-based systems (such as [1]) are *landmark-based*, in that each particle's map consists of a posterior distribution over the locations of a number of salient landmarks. Others (such as [2]) are *grid-based*, meaning that each particle's map is a dense occupancy grid containing the posterior probability that each cell in the grid is occupied. In addition to SLAM algorithms that use laser range measurements, there are a growing number of examples of vision-based slam, a few of which use stereo vision [3; 5; 6]. All of the existing stereo vision SLAM algorithms are landmark-based. Futhermore, most current approaches to SLAM address either indoor environments or structured outdoor environments.

We take a new approach to stereo visual SLAM that can be used in unstructured outdoor environments. In our experience, autonomous navigation in such environments requires the map to contain more than simply the locations of easily-identifiable landmarks. For planning purposes, we need a dense map for estimating the traversability of every location (every cell) in the map. Furthermore, a simple binary occupancy grid is not sufficient for off-road navigation, particularly in vegetated terrain. In such environments, a grid containing the variance of heights in each cell works much better than an occupancy or elevation grid, providing a more useful representation for navigation while not significantly increasing the storage and processing requirements.

Existing grid-based SLAM systems [2] use laser range finders and a binary occupancy grid. Occupancy grids, which are often used in structured indoor environments, are based on an underlying assumption that each cell in the world is either occupied (non-traversable) or free (unoccupied). Based on observations of each cell, the SLAM system [2] infers the posterior probability that the cell is occupied vs. unoccupied, which is a Bernoulli distribution. Another type of grid map, an elevation map, corresponds to a representation of the horizontal surfaces of an environment [7]. Elevation maps are based on an underlying assumption that each cell in the world is a flat surface, in which all samples from a cell have the same elevation, or height, up to additive Gaussian noise. Each cell of an elevation map, which is constructed using the Kalman filter update rule, contains a Gaussian posterior distribution over the height of the surface in that cell. The mean of this Gaussian is the estimate of the height of the surface, and the variance of the Gaussian measures the uncertainty of this height estimate. In recent extensions of elevation maps, which have been used in systems for outdoor localization and for mapping and loop closing [8; 7; 9], a cell is allowed to contain one or more horizontal surfaces or a vertical surface. These extensions still assume that each cell is a planar patch (or multiple levels of planar patches, or a vertical surface), which makes them well suited to structured outdoor environments such as urban roads and overpasses. Map matching in these systems [8; 9] uses an ICP algorithm, and they are currently much too slow for real-time SLAM.

In contrast to these other systems that use grid-based maps, our system uses dense stereo vision, and the maps contain the posterior distribution over the variance of heights in each

Tim K. Marks and Garrison W. Cottrell are at the Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA. {tkmarks, gary}@cs.ucsd.edu

Andrew Howard, Max Bajracharya, and Larry Matthies are at NASA Jet Propulsion Laboratory, Pasadena, CA. {abhoward, maxb, lhm}@robotics.jpl.nasa.gov

cell. Unlike occupancy grids (with the underlying assumption that each cell in the world is either occupied or free) and elevation maps (with the underlying assumption that each cell in the world is a flat surface), our maps are based on the assumption that the points observed from each cell in the world are drawn from a Gaussian height distribution. Rather than estimating the cell's *mean* height, our system estimates the *variance* of the heights. Variance maps are not limited to flat surfaces, which is crucial for unstructured outdoor environments. Our new mapping technique maintains a posterior distribution over the variance of the heights (variance of the elevations) of points in each grid cell. For each grid cell, the posterior distribution over the precision (inverse variance) of the heights is a gamma distribution. (Hence, we call our algorithm *Gamma-SLAM*). Storage requirements are small: each map cell simply contains two scalar values that are the sufficient statistics of this gamma distribution (see Section IV).

In Section VIII, we demonstrate how to use GPS readings that have high inherent error to obtain an accurate alignment of the Gamma-SLAM map to georeferenced coordinates. We derive an online version of this alignment method that is extremely efficient. This enables the robot to leverage the precision of its SLAM maps to essentially obtain a super-resolution GPS reading, by combining information from all of the GPS readings obtained during the map collection.

II. The Mobile Robotic Platform

The algorithm was designed for and tested on the LAGR hHerminator robot [10] (see Fig. 3, top), a differential drive vehicle with 0.75 m wheel separation, 0.16 m wheel radius, and top speed 1.3 m/s. Onboard sensors include an inertial measurement unit (IMU) and two stereo camera pairs that together provide a 140° field of view. Dense stereo range data are generated from each of the two camera pairs, then fused into a cartesian map containing map cell statistics [11]. The incremental vehicle pose is estimated using a visual odometry (VO) algorithm based on [12], applying a corner detector on each frame, matching features using the stereo SAD (sum of absolute differences) scores over a local window, and then estimating the camera motion using the 3D feature positions determined by stereo.

III. The Generative Model

Our Gamma-SLAM system can be conceived as performing inference on a generative model described by the graphical model in Fig. 1. Given the sequence of observations made by the robot from time 1 to t, denoted $\mathbf{z}_{1:t}$, and the sequence of control commands (determined from visual odometry), denoted $\mathbf{u}_{1:t}$, we simultaneously infer a posterior distribution over both the map of the environment, $\mathbf{h}$, and the robot's path (pose sequence) through the environment, $\mathbf{x}_{1:t}$.

The world map is a grid of G square cells with side length 0.16 m. We assume that every point observed in cell g (where $g = 1, \ldots, G$) has its height drawn from a Gaussian distribution, with precision (inverse variance) h_g and unknown mean. The entire world map is $\mathbf{h} = \{h_g\}_{g=1}^{G}$, the collection of all grid cells.

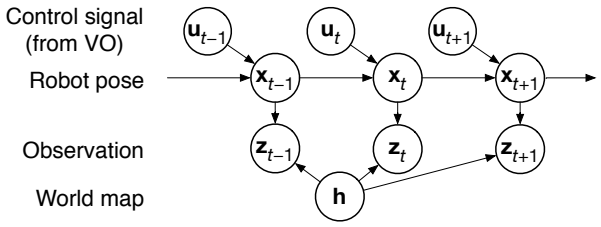

Fig. 1. The goal of Gamma-SLAM is to simultaneously infer $\mathbf{h}$ and $\mathbf{x}_{1:t}$ given a sequence of observations $\mathbf{z}_{1:t}$ and a sequence of control signals $\mathbf{u}_{1:t}$ (obtained from visual odometry).

A. Motion model

The controller for our robotic vehicle maintains a full 6-degree-of-freedom estimate of the robot's pose, but for navigation and planning, a lower-dimensional representation is sufficient. For the purposes of the algorithm described in this paper, the robot pose $\mathbf{x}_t$ consists of a 2D position, $\mathbf{d}_t$, and orientation, θ_t. The control signal, $\mathbf{u}_t$, consists of a robot-centered translation, $\Delta\mathbf{d}_t$, and rotation, $\Delta\theta_t$. The motion model dictates that $\mathbf{x}_t$ is given by the previous pose incremented by the control signal, plus a small amount of Gaussian noise, $\mathbf{q}_t$:

$$\underset{\mathbf{x}_t}{\begin{bmatrix} \mathbf{d}_t \\ \theta_t \end{bmatrix}} = \underset{\mathbf{x}_{t-1}}{\begin{bmatrix} \mathbf{d}_{t-1} \\ \theta_{t-1} \end{bmatrix}} + \underset{\mathbf{u}_t}{\begin{bmatrix} \Delta\mathbf{d}_t \\ \Delta\theta_t \end{bmatrix}} + \underset{\mathbf{q}_t}{\begin{bmatrix} \mathbf{q}_{dt} \\ q_{\theta t} \end{bmatrix}}, \quad (1)$$

where $\mathbf{q}_{dt} \sim N(0, \boldsymbol{\sigma}_d^2)$, $q_{\theta t} \sim N(0, \sigma_\theta^2)$.

B. Observation model

Each observation obtained from dense stereo vision contains a large number of points with locations (x, y, z). For each point observed, we bin the xy-locations into a square grid aligned with the world map. After the observed points are collected into cells, each cell in the grid contains a set of n heights (z-coordinates). For each cell, we assume that these heights are i.i.d. samples, $\{s_1, \ldots, s_n\}$, from a one-dimensional (1D) Gaussian with unknown mean μ_g and precision h_g $\left(\text{where precision} = \frac{1}{\text{variance}}\right)$. (For simplicity, we almost always omit the subscript g and simply call these μ and h.) Note that observations may have different values of n for different cells; for example, cells close to the robot tend to have larger n than faraway cells. We can write the likelihood of drawing these n samples as a function of the unknown population mean and precision, μ and h:

$$\begin{aligned} p(s_1, \ldots, s_n \,|\, \mu, h) &= \prod_{i=1}^{n} p(s_i \,|\, \mu, h) = \prod_{i=1}^{n} f_N(s_i; \mu, h) \\ &= (2\pi)^{-n/2} h^{n/2} e^{-\frac{1}{2}h\sum_{i=1}^{n}(s_i-\mu)^2}, \quad (2) \end{aligned}$$

where $f_N(s_i; \mu, h)$ denotes the 1D Gaussian pdf with mean μ and precision h.

We summarize the collection of data samples $\{s_i\}_{i=1}^{n}$ by its sample mean and variance, m and v:

$$m = \frac{1}{n}\sum_{i=1}^{n} s_i, \quad v = \frac{1}{k}\sum_{i=1}^{n}(s_i - m)^2, \text{ where } k = n - 1. \quad (3)$$

Now with some algebraic manipulation, we can rewrite the likelihood equation (2) in terms of m and v:

$$p(s_1,\ldots,s_n \mid \mu,h) = (2\pi)^{-n/2} e^{-\frac{1}{2}hn(m-\mu)^2} \cdot h^{n/2} e^{-\frac{1}{2}hkv}. \tag{4}$$

Integrating this likelihood over all values of $s_1,\ldots,s_n$ that have the same sufficient statistics m, v gives the likelihood of observing data with mean and variance m, v when we sample n points from a Gaussian distribution whose population mean and precision are μ, h. This likelihood is the product of a normal distribution and a gamma distribution [13]:

$$p(m,v \mid \mu,h;n,k) = f_N(m;\mu,hn) f_{\gamma 2}\big(v;h,k\big) = \tag{5}$$
$$(2\pi)^{-\frac{1}{2}} (hn)^{\frac{1}{2}} e^{-\frac{1}{2}hn(m-\mu)^2} \cdot \frac{1}{\Gamma\big(\frac{1}{2}k\big)} \big(\tfrac{1}{2}hk\big)^{k/2} v^{\frac{1}{2}k-1} e^{-\frac{1}{2}hkv}.$$

As is evident in the equation above, we use $f_{\gamma 2}$ as a convenient parameterization of the gamma function:

$$f_{\gamma 2}(v;h,k) \stackrel{\text{def}}{=} f_\gamma\big(v;\tfrac{1}{2}k,\tfrac{1}{2}hk\big) \stackrel{\text{def}}{=} \frac{1}{\Gamma\big(\frac{1}{2}k\big)} \big(\tfrac{1}{2}hk\big)^{k/2} v^{\frac{1}{2}k-1} e^{-\frac{1}{2}hkv}. \tag{6}$$

Reducing the dimensionality of the hypothesis space for both the robot location and the map makes SLAM algorithms much more efficient. Using the pitch and roll information from the robot's inertial measurement unit (IMU) enables us to reduce the robot's position to 4 degrees of freedom and to record the stereo range data as height values within cells in a horizontal grid map. For the environments we have tested so far, we have been able to reduce the dimensionality of both the pose space and the map further by choosing to omit the z-value (absolute elevation) of the robot and of each map cell. In each observation, we know the relative z-values of every point, but the the evolution of the z-values over time is not strongly constrained by the visual odometry (VO) results, so we have chosen not to maintain an accurate estimate of the robot's elevation over time nor of the mean elevation of the points in each map cell. This $2\frac{1}{2}$-D representation of the world has been sufficient for navigating a wide variety of off-road environments. For this reason, we do not use the mean height statistic m collected from each observation, except to compute the variance statistic, v. We therefore ignore the posterior distribution over μ, computing the marginal posterior distribution over h by integrating m out of (5):

$$p(v \mid h;k) = \int_m f_N(m;\mu,hn) f_{\gamma 2}\big(v;h,k\big) dm \tag{7}$$
$$= f_{\gamma 2}\big(v;h,k\big) = \frac{1}{\Gamma\big(\frac{1}{2}k\big)} \big(\tfrac{1}{2}hk\big)^{k/2} v^{\frac{1}{2}k-1} e^{-\frac{1}{2}hkv}.$$

IV. Bayesian Update of a Map Cell

If we know the precise pose history (path) of the robot, $\mathbf{x}_{1:t}$, as well as the sequence of observations $\mathbf{z}_{1:t}$, then we can infer the posterior distribution over the world map. For each grid cell g, the inferred map contains a distribution over the precision (inverse variance) of the Gaussian distribution of heights in that cell. The map can be thought of as combining all of the observations from time 1 to time t, aligned (rotated and translated) according to the pose at each time step, $\mathbf{x}_{1:t}$. Making the simplifying assumption that the precisions of the grid cells are conditionally independent given the robot's path and observations, we express the map at time t as:

$$\underbrace{p(\mathbf{h} \mid \mathbf{x}_{1:t}, \mathbf{z}_{1:t})}_{\text{map at time } t} = \prod_{g=1}^{G} p(h_g \mid \mathbf{x}_{1:t}, \mathbf{z}_{1:t}). \tag{8}$$

A map is made up of cells, each of which contains the sufficient statistics (v,k) of a gamma distribution. This can be interpreted as the distribution of our beliefs about the precision h of the normal distribution of heights from which that cell's points were sampled. When we collect a new observation (a new set of data points for a map cell), we use the previous gamma distribution for the cell as our prior, then use the new data points to update our beliefs, obtaining a new gamma distribution as our posterior for that map cell.

Given the pose at time t, we know which points from the observation at time t correspond to each grid cell in the map. For any particular cell, at time t we observe some number $n = k+1$ of points that lie in that cell (according to the pose $\mathbf{x}_t$), and we compute their data variance, v, using (3). This section explains how to use this observation (summarized by statistics v, k) to update the corresponding cell in the prior map (summarized by statistics v', k'), thus obtaining a posterior map (a posterior distribution over the precision of heights in this cell, summarized by statistics v'', k'').

A. The gamma prior

When estimating the precision h of the Gaussian distribution of heights in a cell, the conjugate prior is the natural conjugate of the likelihood distribution (7), which is a gamma distribution. We parameterize this gamma prior over h with parameters v' and k':

$$p(h \mid v',k') = f_{\gamma 2}\big(h;v',k'\big)$$
$$= \frac{1}{\Gamma\big(\frac{1}{2}k'\big)} \big(\tfrac{1}{2}k'v'\big)^{k'/2} h^{\frac{1}{2}k'-1} e^{-\frac{1}{2}hk'v'}. \tag{9}$$

This is the prior distribution before we incorporate our current data samples; the sufficient statistics v' and k' are obtained from the data collected in previous time steps.[1]

B. The gamma posterior

Suppose that for a given cell in the map, the prior distribution over h is defined by (9), with sufficient statistics (v',k'). Then we take an observation (we observe $n = k+1$ new points from the cell), and this observation has sufficient statistics (v,k), defined by (3). By Bayes rule, the posterior distribution for the cell in the map is proportional to the

[1] Note that if we wanted to maintain an estimate of the mean height of each cell as well as its variance, we could choose to calculate the posterior distribution over both the mean and precision of heights in each cell; in that case, the natural conjugate distribution would be the normal-gamma distribution [13].

product of the prior distribution (9) and the likelihood (7):

$$p(h \mid v', k', v, k) = \frac{p(h \mid v', k')\, p(v \mid h; k)}{\int_h p(h \mid v', k')\, p(v \mid h; k)\, dh} = \frac{f_{\gamma 2}(h; v', k')\, f_{\gamma 2}(v; h, k)}{\int_h f_{\gamma 2}(h; v', k')\, f_{\gamma 2}(v; h, k)\, dh}. \quad (10)$$

Multiplying (9) by (7) and eliminating terms that do not depend on h, we obtain:

$$p(h \mid v', k', v, k) \propto e^{-\frac{1}{2}h(k'v' + kv)}\, h^{\frac{1}{2}(k'+k)-1}. \quad (11)$$

We can write the posterior more simply by defining new posterior parameters k'' and v'': by letting

$$k'' = k' + k\,, \qquad v'' = \frac{k'v' + kv}{k' + k}, \quad (12)$$

we obtain

$$p(h \mid v', k', v, k) \propto e^{-\frac{1}{2}hk''v''}\, h^{\frac{1}{2}k''-1}. \quad (13)$$

This has the form of a gamma distribution in h. Since the posterior distribution over h is a properly normalized probability distribution, this posterior must be the following gamma distribution:

$$p(h \mid v', k', v, k) = f_{\gamma 2}(h; v'', k''). \quad (14)$$

a) The Uninformative Prior: If no points have yet been observed in a cell, we use the uninformative prior: $k' = 0$, and $v' = 0$ (or indeed let v' equal any finite value). For this prior, the Bayesian update (12) simply produces $k'' = k$ and $v'' = v$. Thus if the incoming data are the first data to go into a cell, the sufficient statistics for the posterior distribution will simply be equal to the data statistics.

C. Likelihood of an observation

Suppose we have our map at time $t-1$, which is $p(\mathbf{h} \mid \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1})$. Each cell of this map, $p(h \mid \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1})$, is a gamma distribution that is summarized by the sufficient statistics v', k'. Suppose we observe k new points from this cell at time t. What is the likelihood that our observation will have a given data variance, v? (We will need the answer later to determine the relative weights of our particles.)

Suppose a cell of the prior map has statistics v', k', and we now observe $n = k + 1$ new points from the same cell. The probability that our current observation will have data variance v is:

$$p(v \mid v', k'; k) = \int_h p(h, v \mid v', k'; k)\, dh = \int_h p(v \mid h; k)\, p(h \mid v', k')\, dh. \quad (15)$$

By noting that this is the denominator of (10), we can use (10) and (14) to obtain the following:

$$p(v \mid v', k'; k) = \frac{f_{\gamma 2}(v; h, k)\, f_{\gamma 2}(h; v', k')}{f_{\gamma 2}(h; v'', k'')} = \frac{\Gamma\left(\frac{1}{2}k''\right)}{\Gamma\left(\frac{1}{2}k\right)\Gamma\left(\frac{1}{2}k'\right)} \cdot \frac{(kv)^{k/2}(k'v')^{k'/2}}{(k''v'')^{k''/2}} \cdot v^{-1}. \quad (16)$$

V. SLAM using Rao-Blackwellized Particle Filtering

The distribution that we estimate at each time step, which we call the target distribution, is the posterior distribution over the path (pose history) of the robot and the corresponding map. Our inputs are the history of observations, $\mathbf{z}_{1:t}$, and the visual odometry (VO) which we treat as the control signal, $\mathbf{u}_{1:t}$. We factorize the target distribution as follows:

$$\underbrace{p(\mathbf{x}_{1:t}, \mathbf{h} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t})}_{\text{target distribution}} = \underbrace{p(\mathbf{x}_{1:t} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t})}_{\text{filtering distribution}}\, \underbrace{p(\mathbf{h} \mid \mathbf{x}_{1:t}, \mathbf{z}_{1:t})}_{\text{map}}. \quad (17)$$

This factorization enables us to estimate the target distribution using Rao-Blackwellized particle filtering [4], which combines the approximate technique of particle filtering for some variables, with exact filtering (based on the values of the particle-filtered variables) for the remaining variables. We use a particle filter to approximate the first term on the right side of (17), the filtering distribution for the particle filter, using discrete samples $\mathbf{x}_{1:t}^{[j]}$. For each of these samples (particles), we use exact filtering to obtain the second term in (17), the map, using the map updates described in Section IV.

By Bayes Rule and the probabilistic dependencies implied by the graphical model (Fig. 1), we can write the filtering distribution for the particle filter as follows:

$$\underbrace{p(\mathbf{x}_{1:t} | \mathbf{z}_{1:t}, \mathbf{u}_{1:t})}_{\text{filtering distribution}} \propto \underbrace{p(\mathbf{z}_t | \mathbf{x}_{1:t}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t})}_{\text{importance factor}}\, \underbrace{p(\mathbf{x}_{1:t} | \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t})}_{\text{proposal distribution}}, \quad (18)$$

where the proportionality constant does not depend on the path $\mathbf{x}_{1:t}$. We further factor the last term:

$$\underbrace{p(\mathbf{x}_{1:t} | \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t})}_{\text{proposal distribution}} = \underbrace{p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)}_{\text{sampling distribution}}\, \underbrace{p(\mathbf{x}_{1:t-1} | \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t-1})}_{\text{prior filtering distribution}}. \quad (19)$$

Equations (18) and (19) express the filtering distribution at time t as a function of the filtering distribution at $t-1$. The sampling distribution in (19) is given by the motion model (1). To compute the importance factor in (18), observe that

$$\underbrace{p(\mathbf{z}_t \mid \mathbf{x}_{1:t}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t})}_{\text{importance factor}} = \int_{\mathbf{h}} p(\mathbf{z}_t, \mathbf{h} \mid \mathbf{x}_{1:t}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}) d\mathbf{h} = \int_{\mathbf{h}} \underbrace{p(\mathbf{z}_t \mid \mathbf{h}, \mathbf{x}_t)}_{\text{likelihood of observation}}\, \underbrace{p(\mathbf{h} \mid \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1})}_{\text{prior map}}\, d\mathbf{h}. \quad (20)$$

Since we have assumed that each cell of the map is independent, we compute the importance factor separately for each grid cell that is nonempty in both the prior map and the current observation, then take the product over all of these grid cells to obtain the importance factor for the entire observation. To see how to compute the importance factor for a grid cell, note that (20) for a single map cell is precisely the observation likelihood (15), so we can compute it using (16).

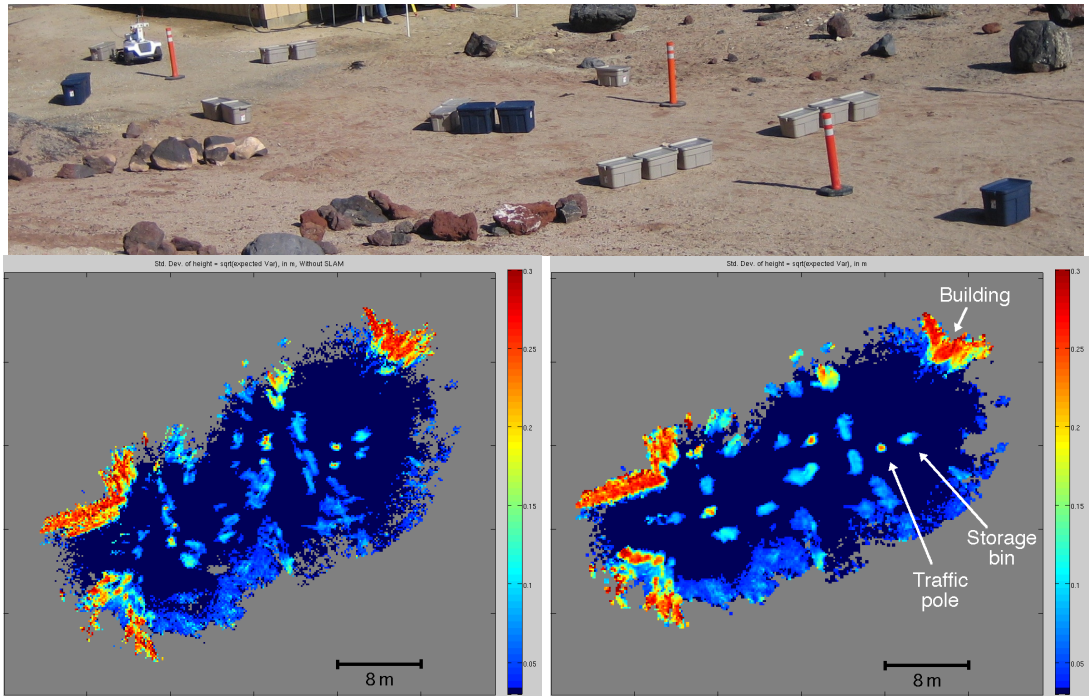

Fig. 2. Course A. **Top:** Photo. **Lower left:** Map made using VO alone. **Lower right:** Map made by Gamma-SLAM with 100 particles. The left and right sides of each map correspond to the left and right sides of the photo, respectively. Color indicates the expected standard deviation of the heights in each grid cell. In the Gamma-SLAM map, the orange traffic poles appear as small yellow circles with orange centers. The plastic storage bins, which are shorter, appear in light blue, while the corners of the tall buildings (at the upper left and upper right of the map) appear in red-orange.

VI. The Gamma-SLAM Algorithm

At time step t, estimate the effective number of particles [14] using the particle weights from all J particles at time $t-1$: $J_{\text{eff}} = \frac{1}{\sum_{i=1}^{J}\left(w_{t-1}^{[i]}\right)^2}$. Then do the following for each particle $j = 1, 2, \ldots, J$.

- If $J_{\text{eff}} \geq \frac{J}{2}$: Let $\tilde{\mathbf{x}}_{1:t-1}^{[j]} = \mathbf{x}_{1:t-1}^{[j]}$, and let $\tilde{w}_{t-1}^{[j]} = w_{t-1}^{[j]}$.
- If $J_{\text{eff}} < \frac{J}{2}$ [Resampling step]: Sample (with replacement) a particle $\tilde{\mathbf{x}}_{1:t-1}^{[j]}$ from the previous time step's particle set, $\{\mathbf{x}_{1:t-1}^{[i]}\}_{i=1}^{J}$, with probability $w_{t-1}^{[i]}$. Set $\tilde{w}_{t-1}^{[j]} = \frac{1}{J}$.
- **Prediction:** Sample a new pose from the proposal distribution, given by the motion model (1):

$$\mathbf{x}_t^{[j]} \sim p(\mathbf{x}_t \,|\, \tilde{\mathbf{x}}_{t-1}^{[j]}, \mathbf{u}_t). \tag{21}$$

  Append the new pose onto the particle's path (pose history): $\mathbf{x}_{1:t}^{[j]} = \{\tilde{\mathbf{x}}_{1:t-1}^{[j]}, \mathbf{x}_t^{[j]}\}$.
- **Measurement update:** Rotate and translate the observation $\mathbf{z}_t$ according to pose $\mathbf{x}_t^{[j]}$, so that the observation aligns with the particle's existing map (so the grid cell boundaries are in the same locations). For each grid cell that is observed, compute the data statistics v, k, then combine this with the prior map for this cell (statistics v', k') by updating the map using (12). This gives the statistics v'', k'', which comprise that cell of the particle's map at time t.
- **Importance weight:** For each grid cell g that is nonempty in both the current observation and the prior map, compute an importance factor, $\lambda_{tg}^{[j]}$, using the logarithm of (16):

$$\begin{aligned}\log \lambda_{tg}^{[j]} &= \log\Gamma\left(\tfrac{1}{2}k''\right) - \log\Gamma\left(\tfrac{1}{2}k\right) - \log\Gamma\left(\tfrac{1}{2}k'\right) \\ &\quad + \tfrac{1}{2}\left[k\log(kv) + k'\log(k'v') - k''\log(k''v'')\right] - \log v.\end{aligned} \tag{22}$$

  To compute the importance factor for the particle, sum the log importance factors of all cells that are nonempty in both the current observation and the prior map, then divide by a constant β times the number $\tilde{G}$ of such cells[2]:

$$\log \lambda_t^{[j]} = \frac{1}{\beta\tilde{G}} \sum_g \log \lambda_{tg}^{[j]}. \tag{23}$$

  The new weight of the particle is equal to the product of its old weight and its importance factor:

$$w_t^{[j]} \propto \tilde{w}_{t-1}^{[j]} \cdot \lambda_t^{[j]}, \tag{24}$$

  where at the end of the time step, the particle weights $w_t^{[j]}$ are normalized so $\sum_j w_t^{[j]} = 1$.

VII. SLAM Results

The robot was driven via remote control through two courses (course A and course B) on uneven sandy ground. On both courses, we ran Gamma-SLAM with 100 particles. To test the accuracy of the Gamma-SLAM system versus visual

[2]If all grid cells were truly independent as we assumed, then we could simply calculate the importance factor $\lambda_t^{[j]}$ using $\log \lambda_t^{[j]} = \sum_g \log \lambda_{tg}^{[j]}$. In practice, however, this equation results in an observation likelihood that is too steep (the best particle has almost all of the weight, whereas its competitors have almost none), which is probably due (at least in part) to the independence assumption. The factor $\frac{1}{\beta\tilde{G}}$ in (23) counteracts this effect.

odometry (VO) alone, we paused the robot several times during the course of the run and measured its ground truth position using a Total Station electronic surveying device. Table I shows the root-mean-squared position errors for each course, using Gamma-SLAM versus using VO alone.

The objects in course A (shown in Fig. 2, top) included orange traffic poles, plastic storage bins, rocks, and the corners of buildings. The robot was driven back and forth over this course three times during the course of the run. Fig. 2 shows the map made using Gamma-SLAM (bottom right), as well as the map made without SLAM using the pose information from VO alone (lower left). We made this VO-only map by applying the same map update rules from Section IV using the pose taken directly from the visual odometry. In the VO-only map, there are two or three copies of each object, because each time it drove through the map, the robot's VO estimate of its position had shifted with respect to the true position. As a result, narrow gaps have been closed off in the map, which would make planning and navigation virtually impossible. In contrast, the Gamma-SLAM map (lower right) provides an accurate map of the terrain. Course B (shown in Fig. 3, top) consists of large loop, which the robot drove around three times during the run. Course B contains four traffic poles (roughly indicating the robot's path through the course) and a few plastic bins, but the course consists mainly of rocks of various sizes, from small rocks to boulders. Fig. 3 (top) shows a photo of the course, with the robot facing one of the orange traffic poles.

By comparing to the course photos (Figs. 2 and 3, top), notice that the Gamma-SLAM maps capture not only the locations of obstacles (as an occupancy grid would), but also information about the heights of objects. The system will observe small height variance in a cell (the map cell will appear more blue) if an object is flatter to the ground. A taller object will have greater height variance (the cell will appear more red). The accompanying video shows Gamma-SLAM in action as the robot drives through course B.

VIII. GPS Alignment

The robotic vehicle we used has a GPS receiver, but the methods we have described so far do not use GPS. The error in the GPS signal is quite large compared to the precision of the SLAM maps, at least for all cases we have tested to date (maps a few tens-of-meters on a side). However, while the SLAM maps provide excellent relative position information, they are not absolutely aligned with the Earth. We can combine Gamma-SLAM's precise measure of the relative position of the robot at each time step with the comparatively imprecise GPS readings to obtain a measure of the robot's position in georeferenced coordinates much more accurate than that provided by the GPS receiver alone.

TABLE I

ERROR FROM GROUND TRUTH ROBOT POSITIONS: GAMMA-SLAM VS. VISUAL ODOMETRY (VO) ALONE.

Course	Distance traveled	RMS error (in m)	
		VO only	Gamma-SLAM
Course A	164 m	1.24	**0.29**
Course B	204 m	1.12	**0.21**

Below, we first describe a batch method for aligning a particle's SLAM map with the Earth (in georeferenced coordinates) using all of the past GPS data and the SLAM position data for the particle. Then we present an incremental (online) version of the algorithm, usable for real-time updates of the georeferenced coordinates of the SLAM map (and the georeferenced position of the robot) for multiple particles.

A. Batch algorithm for GPS alignment

The GPS receiver on the robot provides estimates of latitude, longitude, and horizontal estimated position error (EPE), with typical errors of 3–10 m (WAAS-enabled, unobstructed sky view). For simplicity, we assume that this horizontal error estimate, σ, is proportional to the standard deviation of the error distribution of that GPS reading.

Define the $2 \times t$ matrix $\mathbf{B}$ so that its ith column, $\mathbf{b}_i$, contains the 2D position of the robot inferred by Gamma-SLAM at each time step $i = 1, \ldots, t$. Note that the values in $\mathbf{B}$ (in meters) are in a *local* coordinate system. Let the $2 \times t$ matrix $\mathbf{A}$ contain the GPS reading of the 2D global position of the robot at each of the same time steps, $\mathbf{a}_i$, in the *global* (utm) coordinate system (also in meters). Finally, let σ_i denote the GPS receiver's horizontal estimated position error (EPE) for the reading at time i.

Our goal is to find the 2×1 translation vector, $\boldsymbol{l}$ (in meters), and 2×2 rotation matrix, $\mathbf{R}$, that transform the SLAM map from local coordinates to global coordinates with the least error from the GPS signal. Furthermore, the measure of how egregious a particular error is should be divided by the corresponding horizontal estimated position error (EPE): a difference of 10 m is more worrisome when the EPE reading is 5 m than when the EPE reading is 20 m. Thus, our goal is to find $\mathbf{R}$ and $\boldsymbol{l}$ that minimize the following error function:

$$\epsilon = \sum_{i=1}^{t} \left\| \frac{\mathbf{a}_i - (\mathbf{R}\mathbf{b}_i + \boldsymbol{l})}{\sigma_i} \right\|_2^2. \tag{25}$$

Define the weight $\omega_i = \frac{1}{\sigma_i}$, and let $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ represent the following weighted averages of the GPS positions and the SLAM positions, respectively:

$$\boldsymbol{\mu} = \frac{1}{\sum_{i=1}^{t} \omega_i^2} \sum_{i=1}^{t} \omega_i^2 \mathbf{a}_i \qquad \boldsymbol{\nu} = \frac{1}{\sum_{i=1}^{t} \omega_i^2} \sum_{i=1}^{t} \omega_i^2 \mathbf{b}_i. \tag{26}$$

It can be shown that the rotation and translation that minimize the error function (25) map from $\boldsymbol{\nu}$ to $\boldsymbol{\mu}$. In other words, if we subtract the weighted mean $\boldsymbol{\mu}$ from each $\mathbf{a}_i$ and subtract the weighted mean $\boldsymbol{\nu}$ from each $\mathbf{b}_i$, then the translation from one to the other that minimizes the error function is the zero translation. Denote these mean-subtracted GPS positions and mean-subtracted SLAM positions respectively by

$$\tilde{\mathbf{a}}_i = \mathbf{a}_i - \boldsymbol{\mu}, \qquad \tilde{\mathbf{b}}_i = \mathbf{b}_i - \boldsymbol{\nu}. \tag{27}$$

Define the $2 \times t$ matrix $\tilde{\mathbf{A}}$ so that the ith column is the vector $\tilde{\mathbf{a}}_i$, and similarly define the $2 \times t$ matrix $\tilde{\mathbf{B}}$ so its ith column

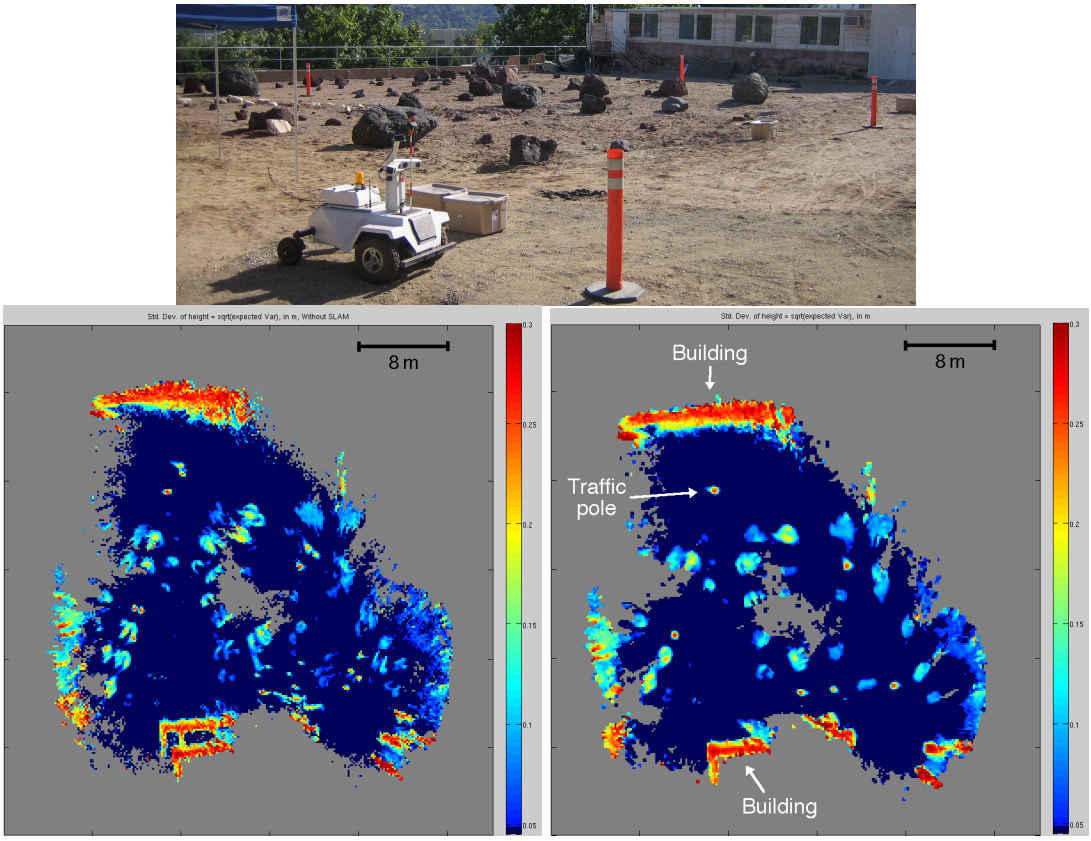

Fig. 3. Course B. **Top:** Photo. **Lower left:** Map made with VO alone. **Lower right:** Map made by Gamma-SLAM with 100 particles. The tall buildings (at the top and bottom of the Gamma-SLAM map) appear red-orange. The four orange traffic poles and the two white tent poles appear as small yellow circles with orange centers. The rocks vary in color depending upon their size: small (short) rocks are dark blue, larger rocks are light blue, and the largest (tallest) boulders are yellow-green.

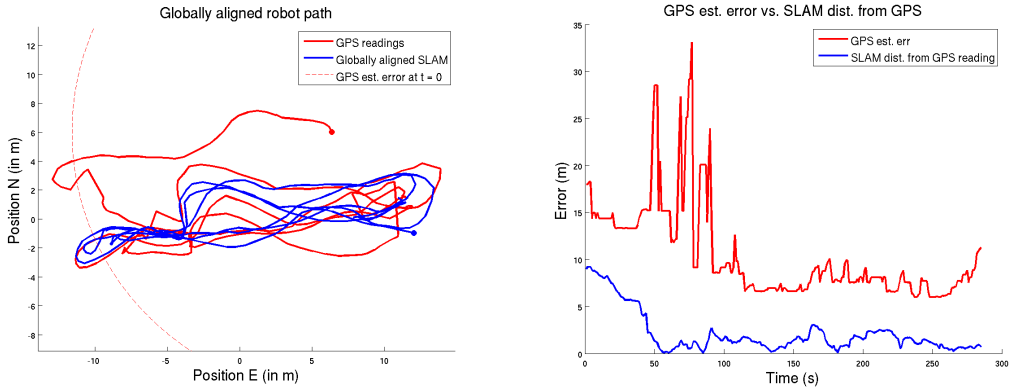

Fig. 4. GPS alignment results on Course A. **Left:** The path of the robot according to the GPS readings is shown in red. (Notice the GPS error artifacts such as the occasional sudden jumps in the GPS reading from one moment to the next. The red dot shows the GPS reading at the start of the run, and the large dashed red circle (centered at the red dot) shows the horizontal estimated position error (EPE) given by the GPS unit at for that reading. In blue is the path of the robot as inferred by Gamma-SLAM, translated and rotated so as to minimize the GPS error, ϵ. The blue dot indicates the position of the robot at the start of the run. **Right:** Comparison of the horizontal EPE given by the GPS unit (in red) to the GPS error (measured as the distance from the GPS reading to the optimally-aligned SLAM result). The SLAM solution is well within the error bounds given by the GPS unit throughout the run.

is $\tilde{\mathbf{b}}_i$. Then we can rewrite the error function (25) as:

$$\epsilon = \sum_{i=1}^{t} \left\| \omega_i \left(\tilde{\mathbf{a}}_i - \mathbf{R}\tilde{\mathbf{b}}_i \right) \right\|_2^2 . \tag{28}$$

Letting $\mathbf{\Omega}$ be the diagonal matrix of the weights, $\mathbf{\Omega} = \mathrm{diag}(\omega_1, \ldots, \omega_t)$, we can rewrite the error in matrix form:

$$\epsilon = \left\| \tilde{\mathbf{A}}\mathbf{\Omega} - \mathbf{R}\tilde{\mathbf{B}}\mathbf{\Omega} \right\|_F^2 , \tag{29}$$

where $\|\cdot\|_F^2$ represents the square of the Frobenius norm (the sum of the squares of all of the elements of a matrix).

Find the rotation matrix, $\mathbf{R}$, that minimizes the error (29), using the orthogonal Procrustes method [15]: Take the singular value decomposition (SVD) of $\tilde{\mathbf{A}}\mathbf{\Omega}(\tilde{\mathbf{B}}\mathbf{\Omega})^T = \tilde{\mathbf{A}}\mathbf{\Omega}^2\tilde{\mathbf{B}}^T$:

$$\tilde{\mathbf{A}}\mathbf{\Omega}^2\tilde{\mathbf{B}}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T , \tag{30}$$

where $\mathbf{S}$ is diagonal $\mathbf{U}$ and $\mathbf{V}$ are unitary (orthonormal) matrices. Then $\mathbf{U}\mathbf{V}^T$ is the unitary transformation matrix

that minimizes the error. In this case, we require a pure rotation (determinant $= 1$). If $\left|\mathbf{U}\mathbf{V}^T\right| = -1$ (which can happen if the robot's path is fairly linear and/or the GPS error is high), there is a simple modification [16]: The rotation matrix that minimizes the error, ϵ, is

$$\mathbf{R} = \mathbf{U}\mathbf{V}^T, \qquad \text{if } \left|\mathbf{U}\mathbf{V}^T\right| = 1; \tag{31}$$

$$\mathbf{R} = \mathbf{U}\left[\begin{smallmatrix}1 & 0\\ 0 & -1\end{smallmatrix}\right]\mathbf{V}^T, \qquad \text{if } \left|\mathbf{U}\mathbf{V}^T\right| = -1. \tag{32}$$

B. Online algorithm for GPS alignment

An online (incremental) version of the algorithm described above requires the running updates of a few values:

$$\boldsymbol{\alpha}_t \stackrel{\text{def}}{=} \sum_{i=1}^{t} \omega_i^2 \mathbf{a}_i, \qquad \boldsymbol{\beta}_t \stackrel{\text{def}}{=} \sum_{i=1}^{t} \omega_i^2 \mathbf{b}_i, \qquad \gamma_t \stackrel{\text{def}}{=} \sum_{i=1}^{t} \omega_i^2. \tag{33}$$

The update equations from time $t-1$ to time t for these sums, as well as for the weighted averages $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ from (26), are:

$$\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1} + \omega_t^2 \mathbf{a}_t, \quad \boldsymbol{\beta}_t = \boldsymbol{\beta}_{t-1} + \omega_t^2 \mathbf{b}_t, \quad \gamma_t = \gamma_{t-1} + \omega_t^2,$$

$$\boldsymbol{\mu}_t = \frac{\boldsymbol{\alpha}_t}{\gamma_t}, \qquad \boldsymbol{\nu}_t = \frac{\boldsymbol{\beta}_t}{\gamma_t}, \tag{34}$$

Using these values, we can find the value at time t of the 2×2 matrix $\tilde{\mathbf{A}}\boldsymbol{\Omega}^2\tilde{\mathbf{B}}^T$ by updating from its value at time $t-1$:

$$\left[\tilde{\mathbf{A}}\boldsymbol{\Omega}^2\tilde{\mathbf{B}}^T\right]_t = \left[\tilde{\mathbf{A}}\boldsymbol{\Omega}^2\tilde{\mathbf{B}}^T\right]_{t-1} + \gamma_{t-1}(\boldsymbol{\mu}_{t-1} - \boldsymbol{\mu}_t)(\boldsymbol{\nu}_{t-1} - \boldsymbol{\nu}_t)^T + \omega_t^2 \tilde{\mathbf{a}}_t \tilde{\mathbf{b}}_t^T. \tag{35}$$

These update equations provide the 2×2 matrix, $\tilde{\mathbf{A}}\boldsymbol{\Omega}^2\tilde{\mathbf{B}}^T$, on which to perform SVD. The optimal transformation from local (SLAM) coordinates to global (georeferenced) coordinates is then simply obtained using equations (30–32).

Since the update equations only require us to maintain and perform a few calculations on a fixed number of 2-vectors and 2×2 matrices, this GPS alignment requires extremely low overhead in terms of both memory and computation time, making it easy to incorporate into a real-time SLAM system. Fig. 4 shows GPS alignment results on Course A.

The ability to georeference maps has two important impacts. First, the robot can correctly fuse knowledge defined in the GPS frame (such as waypoints) with the traversability knowledge stored in the Gamma-SLAM map, thereby supporting autonomous navigation behaviors. Second, maps can be stored and reloaded for later use, since the error in the stored map pose is negligble compared to the error in the current robot GPS position estimate. Thus, the robot can incrementally fuse data that are acquired during multiple runs over the same territory into a single map.

IX. Discussion

We have developed a new approach to SLAM that uses dense stereo vision and infers a map containing a posterior distribution over the variance of heights in each grid cell. Our results show significant improvement over visual odometry alone in an outdoor environment. Our map is unique among SLAM systems due to the information that it contains and the type of filter it uses to update this information. This type of map is well suited to off-road outdoor environments, where occupancy grids often provide insufficient information about a cell's contents, and landmark-based maps can be too sparse to provide information needed for planning and navigation. We also derived an online algorithm for aligning the SLAM maps in georeferenced coordinates, enabling the integration of GPS readings over time for more accurate running estimates of the robot's global position. We are incorporating Gamma-SLAM into a real-time system for autonomous robot navigation in unstructured outdoor environments.

X. Acknowledgments

This work was performed for the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the DARPA LAGR program through an agreement with the National Aeronautics and Space Administration. GWC is supported in part by NSF grant SBE-0542013.

References

[1] Michael Montemerlo and Sebastian Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. In *Proc. ICRA*, 2003.

[2] Dirk Haehnel, Wolfram Burgard, Dieter Fox, and Sebastian Thrun. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *IROS*, 2003.

[3] P. Elinas, R. Sim, and J. J. Little. σSLAM: Stereo vision SLAM using the Rao-Blackwellised particle filter and a novel mixture proposal distribution. In *ICRA*, 2006.

[4] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised particle filtering for dynamic bayesian networks. In *16th Conf. Uncertainty in AI*, pages 176–183, 2000.

[5] Stephen Se, Timothy Barfoot, and Piotr Jasiobedzki. Visual motion estimation and terrain modeling for planetary rovers. In *Proc. ISAIRAS*, 2005.

[6] M. Dailey and M. Parnichkun. Simultaneous localization and mapping with stereo vision. In *Proc. ICARCV*, 2006.

[7] R. Kümmerle, R. Triebel, P. Pfaff, and W. Burgard. Monte carlo localization in outdoor terrains using multi-level surface maps. In *Intl. Conf. Field and Service Robotics (FSR)*, 2007.

[8] R. Triebel, P. Pfaff, and W. Burgard. Multi level surface maps for outdoor terrain mapping and loop closing. In *IROS*, 2006.

[9] P. Pfaff, R. Triebel, and W. Burgard. An efficient extension to elevation maps for outdoor terrain mapping and loop closing. *International Journal of Robotics Research*, 2007.

[10] L. D. Jackel, Eric Krotkov, Michael Perschbacher, Jim Pippine, and Chad Sullivan. The DARPA LAGR program: Goals, challenges, methodology, and phase I results. *Journal of Field Robotics*, 24, 2007.

[11] Andrew Howard, Michael Turmon, Larry Matthies, Benyang Tang, Anelia Angelova, and Eric Mjolsness. Towards learned traversability for robot navigation: From underfoot to the far field. *Journal of Field Robotics*, 24, 2007.

[12] H. Hirschmuller, P.R. Innocent, and J.M. Garibaldi. Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics. In *ICARCV'02*, pages 1099–1104, 2002.

[13] Howard Raiffa and Robert Schlaifer. *Applied Statistical Decision Theory*. Wiley Classics Library, 2000.

[14] A. Doucet, S. J. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10:197–208, 2000.

[15] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1989.

[16] Jos M.F. ten Berge. The rigid orthogonal procrustes rotation problem. *Psychometrika*, 71(1):201–205, 2006.