2005 IEEE/RSJ International Conference on Intelligent Robots and Systems

Online Visual Motion Estimation using FastSLAM with SIFT Features

Timothy D. Barfoot
MDA, Space Missions
9445 Airport Road
Brampton, Ontario L6S 4J3
tim.barfoot@mdacorporation.com
905-790-2800 x 4012

***Abstract*— This paper describes a technique to estimate the 3D motion of a vehicle using odometric sensors and a stereo camera. The algorithm falls into the category of simultaneous localization and mapping as a large database of visual landmarks is created. The algorithm has been field tested online on a rover traversing loose terrain in the presence of obstacles. The resulting position estimation errors are between 0.5% and 4% of distance travelled, a significant improvement over odometry alone.**

I. Introduction

Estimating vehicle motion online is critical to creating truly autonomous systems. For some terrestrial applications, the Global Positioning System (GPS) offers a good solution to this problem. However, there are many scenarios where one cannot rely on GPS, such as planetary exploration by a rover. The 2004 Mars Exploration Rovers experienced considerable difficulties in using odometric sensors alone to estimate their motion. Autonomous land vehicles in areas with foliage and indoor robots in cluttered, non-planar environments might also benefit from visual motion estimation.

In this paper our focus is on developing a technique that works both inside and outside, in three dimensional environments, both man-made and natural. Our approach is to use a stereo camera to track a large number (e.g., $100,000$) of visual landmarks over time. No prior knowledge of these landmarks is assumed. This visual motion estimation has been tested on an outdoor mobile robot (see Figure 1) traversing up to 120 m through an obstacle course to a pre-specified goal location. The tests were conducted in an autonomous manner by also running terrain assessment and motion planning algorithms (which will not be described here). The presence of obstacles in loose terrain forced the motion of the vehicle to be demanding enough to be a reasonable test of our motion estimation. Our autonomous placement errors ranged from 0.5% to 4% of distanced travelled, a considerable improvement over odometric sensors alone.

This paper is organized as follows. First, we describe the visual landmarks we are using and how data association is accomplished. Next, we discuss the estimation algorithm itself. We then present results of experiments, followed by discussion and conclusions.

Fig. 1. Rover used for visual motion estimation experiments.

II. Visual Landmarks

The visual landmarks we use are SIFT (Scale Invariant Feature Transform) features [1], which may be identified fairly uniquely by their descriptors. On the surface, this uniqueness provides a straightforward solution to the data association problem. Note that we do not use the landmarks' 3D position at all in the data association; it is done purely based on SIFT descriptors. Mis-matches between similar-looking features are handled later by outlier detection. For each stereo pair of images, we first find SIFT matches between the left and right images. Each one of these matches is called an *observation*, $\mathbf{z}$, and is of the form $\mathbf{z} \triangleq \begin{bmatrix} c & r & d \end{bmatrix}^T$, where c, r, and d are the column, row, and disparity (difference in column coordinate of feature between left and right images) of the SIFT feature. The k^{th} observation at time, t, is denoted $\mathbf{z}_t^k$.

Next, we take each of these observations and find the best match in our growing database of features. If a feature cannot be found in the database, we add it and assign it an id number, α. For fast access, a kd-tree is built online and matching of observed features to the database (i.e., data association) is carried out by a best-bin-first search [2]. At the moment we

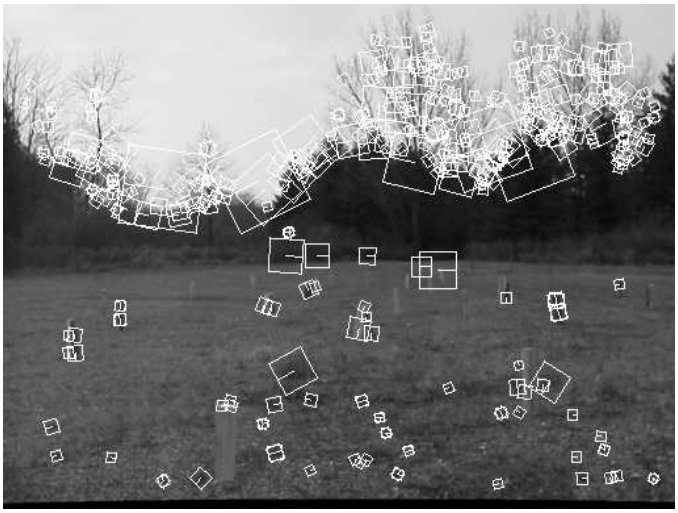

Fig. 2. Example image with SIFT features marked. For this image, some features are found in the foreground but many are found along the skyline.

are not pruning nor rebalancing the kd-tree. This has not been necessary in our experiments with up to $200,000$ landmarks. The output of this step is the data association, α_t^k, for each observation, $\mathbf{z}_t^k$. We also keep track of a measure, β_t^k, of how well each SIFT feature matches to the database. This number is the 'distance' between two SIFT features and is the same metric used to construct the kd-tree and do the best-bin-first search. The lower the number, the better the match.

Our vision system runs on a Pentium-M 1.8 GHz machine and a custom FPGA board [4] to identify SIFT features in each image. For a single 1024×768 image, the FPGA takes approximately 0.15 seconds to identify up to 2000 SIFT features (compared with 1 second when processed in software on the Pentium-M). All of the other steps described above occur in software. However, we parallelize operations wherever possible. For example, while the FPGA is extracting SIFT features in one set of images, the Pentium-M is able to perform the matching for the previous set of images and the rectification for the next set of images. We are able to process pairs of 1024×768 images at 3.0 Hz with databases of up to $200,000$ landmarks.

III. Estimation

Our estimation algorithm is derived from the FastSLAM 2.0 algorithm [6]. Some modifications were necessary to make the algorithm compatible with our scenario. The biggest change to the original algorithm is that we observe a large number, K, of visual landmarks simultaneously (e.g., $K = 500$). We also needed to incorporate outlier detection as some of the visual landmarks are inevitably mismatched. Another major modification was that we reformulated the entire algorithm in terms of unscented transformations [7] (replacing the linearization steps) due to the nonlinearities associated with our stereo camera observation model. Owing to space constraints, we do not include this unscented extension.

We seek to simultaneously estimate the trajectory of a vehicle as well as the states of L landmarks. Our goal is to compute the posterior density:

$$p(\mathbf{s}^t, \mathbf{x}_1, \ldots, \mathbf{x}_L \mid \mathbf{z}^t, \mathbf{u}^t, \alpha^t) = p(\mathbf{s}^t \mid \mathbf{z}^t, \mathbf{u}^t, \alpha^t) \prod_{l=1}^{L} p(\mathbf{x}_l \mid \mathbf{s}^t, \mathbf{z}^t, \mathbf{u}^t, \alpha^t) \quad (1)$$

which we see can be factored into L landmark state-estimators and one vehicle trajectory estimator. The vehicle states, up to time t (a.k.a., its trajectory up to time t), is denoted $\mathbf{s}^t$. The l^{th} landmark state is denoted $\mathbf{x}_l$ (which is assumed to be stationary). The sensor observations, up to time t, are denoted $\mathbf{z}^t$. The control inputs (or odometry measurements), up to time t, are denoted $\mathbf{u}^t$. The data associations, which assign particular observations to particular landmarks, up to time t, are denoted α^t.

As is described in [6], [8], a Rao-Blackwellized particle filter will be used to update the posterior as new observations are gathered. This type of particle filter uses samples to represent uncertainty in the vehicle trajectory. Within each particle (a.k.a., sample), an independent Kalman filter [9] is implemented for each landmark in the map. Thus for each landmark (in each particle) we are estimating a mean and covariance:

$$p(\mathbf{x}_l \mid \mathbf{s}^{(m),t}, \mathbf{z}^t, \mathbf{u}^t, \alpha^t) \sim \mathcal{N}\left(\bar{\mathbf{x}}_{l,t}^{(m)}, \mathbf{C}_{l,t}^{(m)}\right) \quad (2)$$

where $^{(m)}$ is the particle index. This has the advantage of not requiring a monolithic filter to represent the joint density for the vehicle and all the landmarks.

A. Vehicle Trajectory Update

The density for the vehicle trajectory may be rewritten as

$$\underbrace{p(\mathbf{s}^t \mid \mathbf{z}^t, \mathbf{u}^t, \alpha^t)}_{\text{target}} = \eta \int \underbrace{p(\mathbf{z}_t \mid \mathbf{s}^{t-1}, \mathbf{z}^{t-1}, \mathbf{u}^t, \alpha^t)}_{\text{observation}} \times \underbrace{p(\mathbf{s}_t \mid \mathbf{s}^{t-1}, \mathbf{z}^t, \mathbf{u}^t, \alpha^t)}_{\text{motion}} \underbrace{p(\mathbf{s}^{t-1} \mid \mathbf{z}^{t-1}, \mathbf{u}^{t-1}, \alpha^{t-1})}_{\text{old target}} d\mathbf{s}^{t-1} \quad (3)$$

To compute this integral we turn to Monte Carlo integration whereby probability density functions are represented by a finite number of samples. In this case we will represent the target density by M particles (samples), denoted $\mathbf{s}^{(m),t}$, where $m = 1 \ldots M$. Rather than draw samples directly from the 'target' density we factor it as follows:

$$\underbrace{p(\mathbf{s}^t \mid \mathbf{z}^t, \mathbf{u}^t, \alpha^t)}_{\text{target}} = \underbrace{\eta\, p(\mathbf{z}_t \mid \mathbf{s}^{(m),t-1}, \mathbf{z}^{t-1}, \mathbf{u}^t, \alpha^t)}_{\text{weight}} \times \underbrace{p(\mathbf{s}_t \mid \mathbf{s}^{(m),t-1}, \mathbf{z}^t, \mathbf{u}^t, \alpha^t)}_{\text{proposal}} \quad (4)$$

The new samples are drawn from the density labelled 'proposal'. For each of the new samples the 'weight' is computed. A resampling process then occurs in proportion to these weights [10].

Not made explicit in [8], is the expression for the proposal density when K simultaneous measurements are made at time t. To compute this we condition the probability of the k^{th} observation on observations 1 through $k-1$, whereupon the proposal density becomes

$$\begin{aligned} & p\left(\mathbf{s}_t|\mathbf{s}^{(m),t-1},\mathbf{u}^t,\mathbf{z}^t,\alpha^t\right) \\ = \quad & \eta\, p\left(\mathbf{s}_t \mid \mathbf{s}_{t-1}^{(m)},\mathbf{u}_t\right)\prod_{k=1}^{K} p\left(\mathbf{z}_t^k \mid \mathbf{s}_t,\mathbf{s}^{(m),t-1},\mathbf{u}^t,\mathbf{z}^{t-1},\alpha^{t-1}\right) \\ = \eta \;\; & \underbrace{p\left(\mathbf{s}_t \mid \mathbf{s}_{t-1}^{(m)},\mathbf{u}_t\right)}_{\sim\mathcal{N}\left(\mathbf{h}\left(\mathbf{s}_{t-1}^{(m)},\mathbf{u}_t\right),\mathbf{Q}_t\right)}\prod_{k=1}^{K}\int_{\mathbf{x}_{\alpha_t^k}}\underbrace{p\left(\mathbf{z}_t^k \mid \mathbf{x}_{\alpha_t^k},\mathbf{s}_t,\alpha_t^k\right)}_{\sim\mathcal{N}\left(\mathbf{g}\left(\mathbf{s}_t,\mathbf{x}_{\alpha_t^k}\right),\mathbf{R}_t^k\right)} \\ & \times\underbrace{p\left(\mathbf{x}_{\alpha_t^k} \mid \mathbf{s}^{(m),t-1},\mathbf{z}^{t-1},\mathbf{u}^{t-1},\alpha^{t-1}\right)}_{\sim\mathcal{N}\left(\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)},\mathbf{C}_{\alpha_t^k,t-1}^{(m)}\right)} d\mathbf{x}_{\alpha_t^k} \end{aligned} \quad (5)$$

All of the densities that appear on the right side of the equation are Gaussian, with means and covariances indicated below. The nonlinear map, $\mathbf{z} = \mathbf{g}(\mathbf{s},\mathbf{x})$, is called the observation model. The nonlinear map, $\mathbf{s}_t = \mathbf{h}(\mathbf{s}_{t-1},\mathbf{u}_t)$, is called the motion model.

We turn to linearization (of $\mathbf{g}$) to compute the Bayesian inference process whereby

$$\begin{aligned} \mathbf{g}(\mathbf{s}_t,\mathbf{x}_{\alpha_t^k}) &\approx \bar{\mathbf{z}}_t^k + \mathbf{G}_{\mathbf{s}}^k\left(\mathbf{s}_t-\bar{\mathbf{s}}_t\right)+\mathbf{G}_{\mathbf{x}}^k\left(\mathbf{x}_{\alpha_t^k}-\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}\right) \\ \bar{\mathbf{s}}_t &\triangleq \mathbf{h}\left(\mathbf{s}_{t-1}^{(m)},\mathbf{u}_t\right), \quad \bar{\mathbf{z}}_t^k \triangleq \mathbf{g}\left(\bar{\mathbf{s}}_t,\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}\right) \end{aligned} \quad (6)$$

$$\mathbf{G}_{\mathbf{x}}^k \triangleq \left.\frac{\partial\mathbf{g}(\mathbf{s},\mathbf{x})}{\partial\mathbf{x}}\right|_{\mathbf{s}=\bar{\mathbf{s}}_t,\mathbf{x}=\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}}, \quad \mathbf{G}_{\mathbf{s}}^k \triangleq \left.\frac{\partial\mathbf{g}(\mathbf{s},\mathbf{x})}{\partial\mathbf{s}}\right|_{\mathbf{s}=\bar{\mathbf{s}}_t,\mathbf{x}=\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}}$$

Applying the convolution theorem we find that the k^{th} integral in the product in (5) is a Gaussian:

$$\begin{aligned} & p\left(\mathbf{z}_t^k \mid \mathbf{s}_t,\mathbf{s}^{(m),t-1},\mathbf{u}^t,\mathbf{z}^{t-1},\alpha^{t-1}\right) \\ & \quad \sim\mathcal{N}\left(\bar{\mathbf{z}}_t^k+\mathbf{G}_{\mathbf{s}}^k\left(\mathbf{s}_t-\bar{\mathbf{s}}_t\right),\mathbf{R}_t^k+\mathbf{G}_{\mathbf{x}}^k\mathbf{C}_{\alpha_t^k,t-1}^{(m)}\mathbf{G}_{\mathbf{x}}^{k^T}\right) \end{aligned} \quad (7)$$

The proposal density is thus the product of $K+1$ Gaussians (K for the integrals plus 1 for the motion model term). After taking this product of Gaussians, we have for the proposal density that

$$\begin{aligned} p\left(\mathbf{s}_t|\mathbf{s}^{(m),t-1},\mathbf{u}^t,\mathbf{z}^t,\alpha^t\right) &\sim\mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{s}_t},\boldsymbol{\Sigma}_{\mathbf{s}_t}\right) \\ \boldsymbol{\Sigma}_{\mathbf{s}_t}^{-1} &\triangleq \mathbf{Q}_t^{-1}+\sum_{k=1}^{K}\mathbf{G}_{\mathbf{s}}^{k^T}\left(\mathbf{R}_t^k+\mathbf{G}_{\mathbf{x}}^k\mathbf{C}_{\alpha_t^k,t-1}^{(m)}\mathbf{G}_{\mathbf{x}}^{k^T}\right)^{-1}\mathbf{G}_{\mathbf{s}}^k \quad (8) \\ \boldsymbol{\mu}_{\mathbf{s}_t} &\triangleq \bar{\mathbf{s}}_t+\boldsymbol{\Sigma}_{\mathbf{s}_t}\sum_{k=1}^{K}\mathbf{G}_{\mathbf{s}}^{k^T}\left(\mathbf{R}_t^k+\mathbf{G}_{\mathbf{x}}^k\mathbf{C}_{\alpha_t^k,t-1}^{(m)}\mathbf{G}_{\mathbf{x}}^{k^T}\right)^{-1}\left(\mathbf{z}_t^k-\bar{\mathbf{z}}_t^k\right) \end{aligned}$$

where it should be pointed out that $\bar{\mathbf{s}}_t$, $\bar{\mathbf{z}}_t^k$, $\mathbf{G}_{\mathbf{x}}^k$, $\mathbf{G}_{\mathbf{s}}^k$, $\boldsymbol{\mu}_{\mathbf{s}_t}$, and $\boldsymbol{\Sigma}_{\mathbf{s}_t}$ all depend on the particle index, despite the fact that the superscript, $^{(m)}$, does not appear in the notation. The superscript will be reserved for quantities that are actually stored in the particle, not temporary variables such as these.

A important comment that should be made at this point is that we can compute the updates in (8) recursively over the observation index, k. as suggested by [8]. We switch from inverse-covariance form to Kalman form using the Sherman-Morrison-Woodbury identity [11] to avoid too many matrix inversions and to avoid numerical instabilities:

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{s}_t}^0 &= \mathbf{h}\left(\mathbf{s}_{t-1}^{(m)},\mathbf{u}_t\right) \\ \boldsymbol{\Sigma}_{\mathbf{s}_t}^0 &\triangleq \mathbf{Q}_t \\ \bar{\mathbf{z}}_t^k &= \mathbf{g}\left(\boldsymbol{\mu}_{\mathbf{s}_t}^{k-1},\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}\right) \\ \mathbf{Z}_t^k &\triangleq \mathbf{G}_{\mathbf{s}}^k\boldsymbol{\Sigma}_{\mathbf{s}_t}^{k-1}\mathbf{G}_{\mathbf{s}}^{k^T}+\mathbf{G}_{\mathbf{x}}^k\mathbf{C}_{\alpha_t^k,t-1}^{(m)}\mathbf{G}_{\mathbf{x}}^{k^T}+\mathbf{R}_t^k \quad (9) \\ \mathbf{K}_t^k &= \boldsymbol{\Sigma}_{\mathbf{s}_t}^{k-1}\mathbf{G}_{\mathbf{s}}^{k^T}\mathbf{Z}_t^{k^{-1}} \\ \boldsymbol{\mu}_{\mathbf{s}_t}^k &= \boldsymbol{\mu}_{\mathbf{s}_t}^{k-1}+\mathbf{K}_t^k\left(\mathbf{z}_t^k-\bar{\mathbf{z}}_t^k\right) \\ \boldsymbol{\Sigma}_{\mathbf{s}_t}^k &= \left(\mathbf{1}-\mathbf{K}_t^k\mathbf{G}_{\mathbf{s}}^k\right)\boldsymbol{\Sigma}_{\mathbf{s}_t}^{k-1} \end{aligned}$$

which may be recursively computed over k. To improve on this, we may iterate several times, reevaluating the Jacobians with the state as the mean of the proposal density, $\boldsymbol{\mu}_{\mathbf{s}_t}$. With the proposal density in hand, we may now safely draw samples from it so that $\mathbf{s}_t^{(m)} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{s}_t},\boldsymbol{\Sigma}_{\mathbf{s}_t}\right)$. We initialize the particles ($t=0$) by drawing $\mathbf{s}_0^{(m)}$ for $m=1\ldots M$ from some initial density. Typically all particles will start with no landmarks.

We must now compute the importance weights for the new samples. Referring back to equation (4), we define the weights, denoted $w_t^{(m)}$, as

$$\begin{aligned} w_t^{(m)} &\triangleq \eta\, p(\mathbf{z}_t \mid \mathbf{s}^{(m),t-1},\mathbf{z}^{t-1},\mathbf{u}^t,\alpha^t) \\ &= \eta\prod_{k=1}^{K} p(\mathbf{z}_t^k \mid \mathbf{z}_t^1,\ldots,\mathbf{z}_t^{k-1},\mathbf{s}^{(m),t-1},\mathbf{z}^{t-1},\mathbf{u}^t,\alpha^t) \\ &= \eta\prod_{k=1}^{K} w_t^{(m),k} \end{aligned} \quad (10)$$

where

$$\begin{aligned} w_t^{(m),k} &\triangleq p(\mathbf{z}_t^k \mid \mathbf{z}_t^1,\ldots,\mathbf{z}_t^{k-1},\mathbf{s}^{(m),t-1},\mathbf{z}^{t-1},\mathbf{u}^t,\alpha^t) \\ &= \int\underbrace{p\left(\mathbf{s}_t \mid \mathbf{s}^{(m),t-1},\mathbf{z}_t^1,\ldots,\mathbf{z}_t^{k-1},\mathbf{u}_t\right)}_{\sim\mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{s}_t}^{k-1},\boldsymbol{\Sigma}_{\mathbf{s}_t}^{k-1}\right)} \\ &\quad\times\int\underbrace{p\left(\mathbf{z}_t^k \mid \mathbf{x}_{\alpha_t^k},\mathbf{s}_t,\alpha_t^k\right)}_{\sim\mathcal{N}\left(\mathbf{g}\left(\mathbf{s}_t,\mathbf{x}_{\alpha_t^k}\right),\mathbf{R}_t^k\right)} \\ &\quad\times\underbrace{p\left(\mathbf{x}_{\alpha_t^k} \mid \mathbf{s}^{(m),t-1},\mathbf{z}^{t-1},\mathbf{u}^{t-1},\alpha^{t-1}\right)}_{\sim\mathcal{N}\left(\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)},\mathbf{C}_{\alpha_t^k,t-1}^{(m)}\right)} d\mathbf{x}_{\alpha_t^k}\, d\mathbf{s}_t \\ &= \frac{1}{\sqrt{2\pi\det\mathbf{Z}_t^k}}e^{\left(-\frac{1}{2}\left(\mathbf{z}_t^k-\bar{\mathbf{z}}_t^k\right)^T\mathbf{Z}_t^{k^{-1}}\left(\mathbf{z}_t^k-\bar{\mathbf{z}}_t^k\right)\right)} \end{aligned} \quad (11)$$

and we have once again used the convolution theorem to arrive at the final line. Note the important fact that the overall weight is simply the product of the weights for each of the K simultaneous observations.

B. Outliers

On the surface, it may seem that SIFT features implicitly handle the data association problem, allowing for a simpler estimation algorithm. This is true to a certain extent, however, the data association problem does not simply go away. Mismatches occur frequently and outliers must be detected. Fortunately, the percentage of outliers tends to be fairly small when everything is working properly such that throwing away a few observations is tolerable. Our approach to handling outliers has a few steps:

1) Avoidance: We sort the incoming observations based on β_t^k, which tells us how good the SIFT match was and thus how good the data association was. We then begin processing the observations from best to worst, stopping after a predefined number of non-outliers (e.g., 50). This is also a convenient way to limit the filter update time in feature-intensive scenes.

2) Probability: The weight, $w_t^{(m),k}$, is proportional to the probability of the observation, given the current robot pose and old landmark position. Thus, we check that this probability is not too small. If it is, the observation is labelled an outlier.

3) Pose Change: As a final resort, we check how far the proposal density mean moves as a result of incorporating an observation. If it moves too far, the observation is labelled an outlier.

If an outlier is detected we revert to the previous iteration of the proposal density and move to the next observation in the sorted list.

C. Landmark Position Update

The last step is to provide the update equations for the landmarks. The density for the l^{th} landmark can be expressed, using Bayes rule, as

$$\underbrace{p(\mathbf{x}_l \mid \mathbf{s}^{(m),t}, \mathbf{z}^t, \mathbf{u}^t, \alpha^t)}_{\sim\mathcal{N}\left(\bar{\mathbf{x}}_{l,t}^{(m)}, \mathbf{C}_{l,t}^{(m)}\right)} \tag{12}$$

$$= \eta \ \underbrace{p(\mathbf{x}_l \mid \mathbf{s}^{(m),t-1}, \mathbf{z}^{t-1}, \mathbf{u}^{t-1}, \alpha^{t-1})}_{\sim\mathcal{N}\left(\bar{\mathbf{x}}_{l,t-1}^{(m)}, \mathbf{C}_{l,t-1}^{(m)}\right)} \prod_{k=1}^{K} p(\mathbf{z}_t^k \mid \mathbf{x}_l, \mathbf{s}_t^{(m)}, \alpha_t^k)$$

For the density of the observation we have two cases such that

$$p(\mathbf{z}_t^k \mid \mathbf{x}_l, \mathbf{s}_t^{(m)}, \alpha_t^k) \sim \begin{cases} \mathcal{N}\left(\mathbf{g}\left(\mathbf{s}_t^{(m)}, \mathbf{x}_{\alpha_t^k}\right), \mathbf{R}_t^k\right) & \text{if } \alpha_t^k = l \\ \text{uniform} & \text{otherwise} \end{cases}$$

In the first case, the l^{th} landmark is observed (during the k^{th} observation and its density will be altered. In the second case, the landmark is not observed and the density is not altered. It should be pointed out that normally at a given time, t, at most one observation will be associated with a given landmark. However, the equations above do not make this assumption and thus simultaneous observations of the same landmark can be incorporated.

For the first case, the nonlinear observation model again necessitates a linearization process in order to come up with an update for the landmark estimate. We make the approximation

$$\mathbf{g}\left(\mathbf{s}_t^{(m)}, \mathbf{x}_{\alpha_t^k}\right) \approx \tilde{\mathbf{z}}_t^k + \tilde{\mathbf{G}}_{\mathbf{x}}^k \left(\mathbf{x}_{\alpha_t^k} - \bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}\right) \tag{13}$$

$$\tilde{\mathbf{z}}_t^k \triangleq \mathbf{g}\left(\mathbf{s}_t^{(m)}, \bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}\right), \quad \tilde{\mathbf{G}}_{\mathbf{x}}^k \triangleq \left.\frac{\partial \mathbf{g}(\mathbf{s},\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{s}=\mathbf{s}_t^{(m)}, \mathbf{x}=\bar{\mathbf{x}}_{\alpha_t^k,t-1}^{(m)}}$$

where the $\tilde{}$ symbol has been used to make the distinction from the previously computed $\bar{\mathbf{z}}_t^k$ and $\mathbf{G}_{\mathbf{x}}^k$. The $\tilde{}$ quantities are based on the sample, $\mathbf{s}_t^{(m)}$, whereas the old quantities are based on the estimate, $\bar{\mathbf{s}}_t$. We now have a product of Gaussians, which we may combined to form:

$$\mathbf{C}_{l,t}^{(m)^{-1}} = \mathbf{C}_{l,t-1}^{(m)^{-1}} + \sum_{\substack{k=1, \\ l=\alpha_t^k}}^{K} \tilde{\mathbf{G}}_{\mathbf{x}}^{k^T} \mathbf{R}_t^{k^{-1}} \tilde{\mathbf{G}}_{\mathbf{x}}^k \tag{14}$$

$$\mathbf{C}_{l,t}^{(m)^{-1}} \bar{\mathbf{x}}_{l,t}^{(m)} = \mathbf{C}_{l,t-1}^{(m)^{-1}} \bar{\mathbf{x}}_{l,t-1}^{(m)} + \sum_{\substack{k=1, \\ l=\alpha_t^k}}^{K} \tilde{\mathbf{G}}_{\mathbf{x}}^{k^T} \mathbf{R}_t^{k^{-1}} \left(\mathbf{z}_t^k - \tilde{\mathbf{z}}_t^k\right)$$

which we show in inverse covariance form for succinctness. Switching to Kalman form may be accomplished by using the Sherman-Morrison-Woodbury identity [11]. Note we may again iteratively recompute the Jacobian, $\tilde{\mathbf{G}}_{\mathbf{x}}^k$ as well as $\bar{\mathbf{z}}_t^k$ using the new landmark mean until the process converges.

If a landmark is simply not observed at time t, then these equations leave its estimate unaltered. For a new landmark, we assume $\mathbf{C}_{l,t-1}^{(m)^{-1}} \triangleq \mathbf{0}$, which corresponds to no a priori information.

IV. Motion and Observation Models

In our formulation, we are actually estimating the motion of a coordinate frame attached to the right camera of the stereo pair, rather than one attached to the robot base. This is purely a matter of efficiency; it is more efficient to transform a single odometry measurement to the camera frame than K visual landmark observations to the robot frame. It is a simple matter to transform the final camera pose back to the robot frame to provide an estimate of the robot motion.

We represent the six degree-of-freedom pose of the camera as $\mathbf{s} \triangleq \begin{bmatrix} \mathbf{d}^T & \boldsymbol{\epsilon}^T & \eta \end{bmatrix}^T$ where $\mathbf{d} \triangleq \begin{bmatrix} x & y & z \end{bmatrix}^T$ is the position, and $\boldsymbol{\epsilon}, \eta$ are the four Euler parameters (a.k.a., quaternions) representing the rotation of the camera. It is important to note that we must enforce the constraint $\boldsymbol{\epsilon}^T\boldsymbol{\epsilon} + \eta^2 = 1$ at all times. One must be careful here because the constraint implies the space of robot poses is not a vector space. To handle this we must think of the linearization steps as compounding a 'small' rotation vector with a 'large' mean rotation [12]. The details are left to the reader.

The six degree-of-freedom control input (in the camera frame) is denoted $\mathbf{u} \triangleq \begin{bmatrix} \mathbf{v}^T & \boldsymbol{\omega}^T \end{bmatrix}^T$ where $\mathbf{v}$ is the translational velocity and $\boldsymbol{\omega}$ is the rotational velocity. Note that if odometry measurements are used as the 'control inputs', one must carefully transform the associated covariance matrix from the robot frame to the camera frame, particularly if the camera is on a rotating pan-tilt unit, attached to the robot base. If $\boldsymbol{\omega}h$

is small, where h is the time-step size, then we can use the following motion model:

$$\mathbf{h}(\mathbf{s},\mathbf{u}) \triangleq \begin{bmatrix} \mathbf{d} + h\mathbf{C}^T\mathbf{v} \\ \hat{\eta}\boldsymbol{\epsilon} + \eta\hat{\boldsymbol{\epsilon}} - \hat{\boldsymbol{\epsilon}}^{\times}\boldsymbol{\epsilon} \\ \hat{\eta}\eta - \hat{\boldsymbol{\epsilon}}^T\boldsymbol{\epsilon} \end{bmatrix} \tag{15}$$

where

$$\hat{\boldsymbol{\epsilon}} \triangleq \sin\left(\frac{1}{2}\omega h\right)\boldsymbol{\omega}/\omega, \quad \hat{\eta} \triangleq \cos\left(\frac{1}{2}\omega h\right), \quad \omega \triangleq \sqrt{\boldsymbol{\omega}^T\boldsymbol{\omega}}$$

$$\mathbf{C} \triangleq \left(\eta^2 - \boldsymbol{\epsilon}^T\boldsymbol{\epsilon}\right)\mathbf{1} + 2\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T - 2\eta\boldsymbol{\epsilon}^{\times}, \quad \boldsymbol{\epsilon}^{\times} \triangleq \begin{bmatrix} 0 & -\epsilon_3 & \epsilon_2 \\ \epsilon_3 & 0 & -\epsilon_1 \\ -\epsilon_2 & \epsilon_1 & 0 \end{bmatrix}$$

and $\boldsymbol{\epsilon} \triangleq \begin{bmatrix} \epsilon_1 & \epsilon_2 & \epsilon_3 \end{bmatrix}^T$ and $\mathbf{1}$ is the identity matrix.

For the observation model we use a pinhole stereo camera model [13] such that

$$\mathbf{g}(\mathbf{s},\mathbf{x}) \triangleq \frac{f}{z_c}\begin{bmatrix} x_c \\ y_c \\ 2b \end{bmatrix} \tag{16}$$

where f is the focal length and $2b$ is the camera separation. The position, $\mathbf{x}_c \triangleq \begin{bmatrix} x_c & y_c & z_c \end{bmatrix}^T$ is the position of the landmark, in the camera frame. The camera frame has the focal axis in the z direction and the two cameras on the x axis. The relation to the absolute position of the landmark is $\mathbf{x}_c \triangleq \mathbf{C}(\mathbf{x} - \mathbf{d})$, where $\mathbf{C}$ is defined above.

The process of linearizing the motion and observation models is left to the reader.

V. Experiments

Initially we tested our algorithm offline using rover simulations and image sequences from a satellite capture mockup scenario wherein a stereo camera approaches a free-flying satellite. Reasonable motion estimates were obtained for numbers of particles between 1 and 300. However, computation times for large numbers of particles were too high to provide an online estimation technique. Thus, for the moment, our visual motion estimation, implemented on the rover depicted in Figure 1 uses a single particle ($M = 1$). This has the effect of not keeping the error correlations between landmark positions and the robot pose. Still, our experiments revealed promising preliminary results, likely because our data association is reasonably good.

We have field-tested the above algorithm running online both indoors and outdoors. Outdoors we conducted tests with the rover driving on two types of terrain: sand and gravel. The presence of loose terrain was a good test as it often caused odometry to become erroneous.

A. Indoors

Initial testing was conducted indoors with varying success. Figure 3 (above) shows the results of an average traverse in a lab environment. Both the robot path and the resulting map are shown (projected to ground plane). In Figure 3 (bottom) we show an occupancy grid created using a laser rangefinder for comparison. In reality the robot's path was a closed loop but

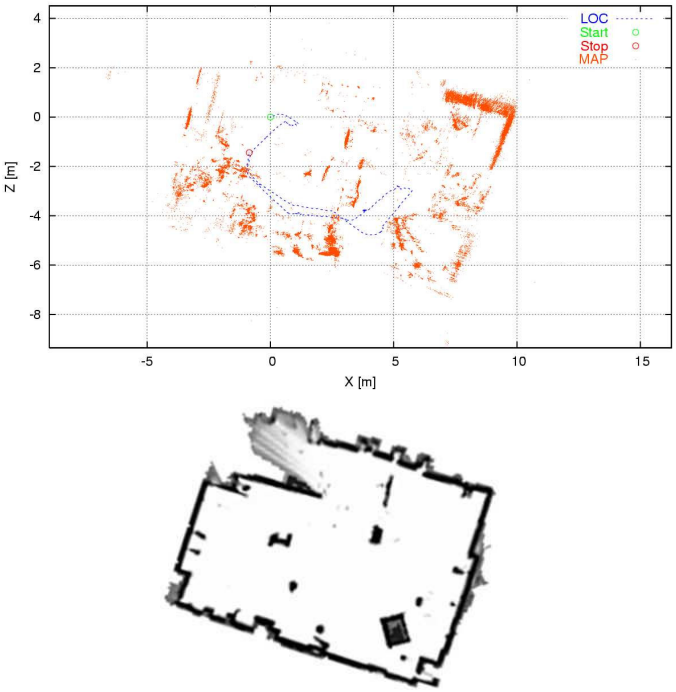

Fig. 3. (above) Estimated rover localization and SIFT map (all projected to ground plane). (below) Occupancy grid created using laser-based technique for comparison.

we see here that there is a large final error. Not surprisingly, there are also errors in the map (even compared to the laser map which itself is not perfect).

Figure 4 shows how the number of visual observations changes throughout the traverse shown in Figure 3. From this plot we can begin to see the difficultly of using this technique in a cluttered, indoor environment. Often the camera got too close to objects to get good stereo matches (due to limits on maximum disparity) and the number of visual observations plummeted to zero. Similarly, if the camera was pointed at a blank wall, the number of visual observations would drop. With no visual observations, our system essentially defaults to simple odometry.

At the other end of the spectrum, if there are too many features, our system can also have trouble. Although we limit the number of observations of previously seen landmarks used in the estimation (typically to 50), we still incorporate all new landmarks into the database. If there are a lot of new landmarks seen, then we are sometimes forced to drop camera frames in order to not let the localization fall too far behind.

We found that most realistic traverses indoors were confronted with periods of few or no features. This was particularly true for hallway traverses with large sections of blank walls.

B. Outdoors, Sand

A set of field trials was conducted on a sandy surface (10 traverses total). Figure 5 shows the results of an approximately 40 m traverse at maximum speed of 5 cm/s on loose sand. During this run, the motion planning software chose to go left around an obstacle (8 m into the traverse). While executing this

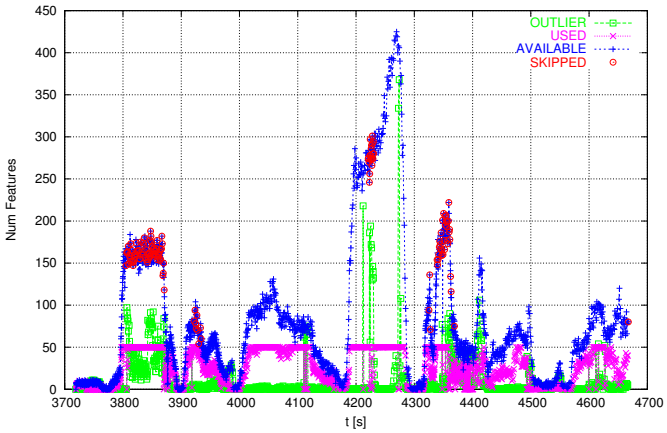

Fig. 4. Statistics on number of visual features observed (blue), used in estimation (purple), outliers (green) during traverse shown in Figure 3. Red is used to indicate frames that had to be dropped to due to processing limitations.

turn, a considerable amount of slip occurred, causing odometry to be quite erroneous in the orientation estimate. Our visual technique did a much better job of estimating the robot path, as can be seen by comparison to GPS. Using a tape measure, the final position of the rover was 37.8 m from the start. Visual motion estimation found 39.4 m and GPS found 38.8 m. The tape measure was taken as ground truth, indicating the visual motion estimation over-predicted the position by 4% of distance travelled. However, it should be noted that most of this positioning error was in the longitudinal direction (along the line joining the start and final positions). Visually, the lateral error was extremely small, indicating that orientation was estimated very well throughout the traverse.

Similar results were found for all the sandy traverses. The repeatability of the system was found to be quite high across trials. Qualitatively, this was demonstrated by the tracks that the robot left in the sand. An example of this can be seen in Figure 5, where there are actually two sets of virtually identical tracks heading to the left (along the marked arrows).

Post-analysis revealed that the final position errors during the sand field tests were in part due to the fact that odometry was slightly over-predicting distance travelled along the path (even in the lab on hard terrain). Odometry does affect the final visual estimate, particularly if the visual features being used are far away (e.g., horizon features). This slight miscalibration does not account for the catastrophic pure-odometry estimation errors of Figure 5. This over-prediction effect was made less severe through further calibration.

C. Outdoors, Gravel

A second set of field trials was conducted on a large gravel area (5 traverses total). Figure 6 shows the results of an approximately 120 m traverse at maximum speed of 10 cm/s on gravel. Here we found that odometry did not experience isolated positioning errors, as was the case on sand, but did experience a systematic error (gradual curve to the left). All of the trials at this site had similar errors, likely due to a slightly higher tire pressure on one side of the rover than the other. This systematic error was not observed prior to the field test;

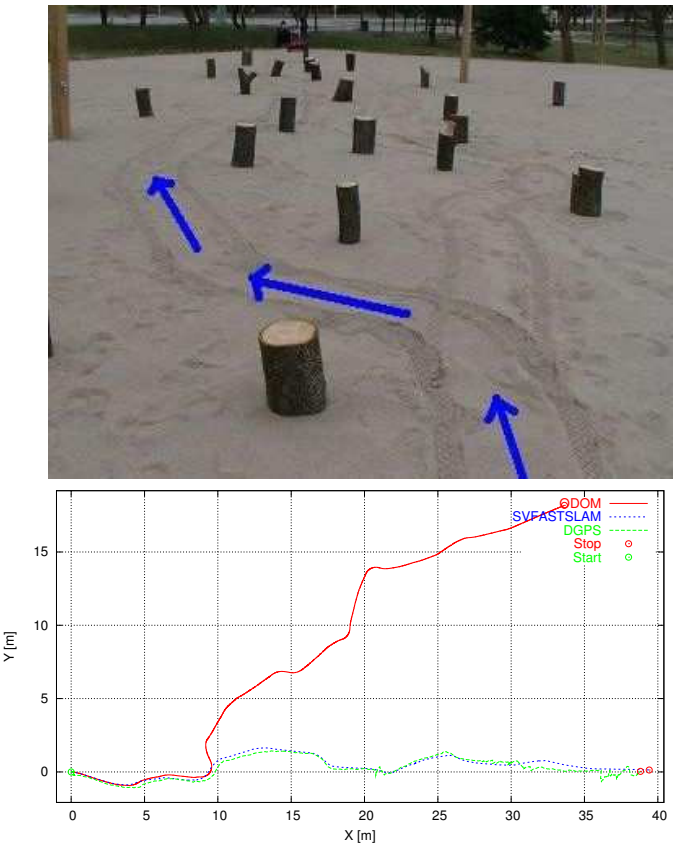

Fig. 5. (above) Sand test site with arrows indicating path taken by rover. (below) Path estimated by FastSLAM (blue), odometry (red), and GPS (green) on loose sand.

it was attributed to changes in tire pressure on the test day and hence miscalibration of odometry.

Using a tape measure, the final position of the rover was 117.4 m from the start. Visual motion estimation found 116.8 m and DGPS found 117.6 m. The tape measure was taken as ground truth, indicated the visual motion estimation under-predicted the position by 0.5% of distance travelled. Again, most of this positioning error was in the longitudinal direction (along the line joining the start and final positions). Visually, the lateral error was extremely small, indicating that orientation was estimated very well throughout the traverse.

The results of other trials at this site were mixed as we tried to push the system to move more quickly and use less images. Increasing the vehicle speed to 20 cm/s, or decreasing the frame-rate to 1.5 Hz, resulted in decreased performance.

VI. Related and Future Work

With only $M = 1$ particle, our algorithm is actually quite similar to [14]. Key algorithmic differences include: a quaternion representation of rotations is used and estimation is done in 3D, unscented transformations replace the linearization steps, the outlier strategy adds more checks and balances. We have also fielded the algorithm outdoors in an online manner, on longer traverses, and with much larger databases of SIFT features. Success in running online is due mostly to the use of a kd-tree with best-bin-first search and hardware acceleration and parallelization of the algorithm. However, efficiency must

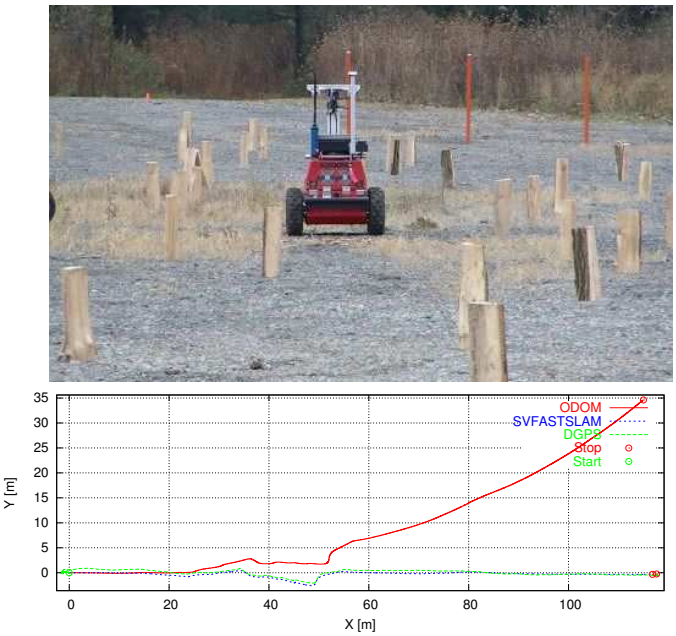

Fig. 6. (above) Gravel test site. (below) Path estimated by FastSLAM (blue), odometry (red), and GPS (green) on loose sand.

be further improved in our implementation if we seek to increase the number of particles and make use of the fact that we have formulated the problem in terms of SLAM. Real-time SLAM results using SIFT features have recently been demonstrated indoors by [3], [5] with a monocular camera and much smaller images than ours. Moreover, similar performance in stereo ego-motion estimation has been achieved elsewhere [15], [16], sometimes at higher frame-rates with fewer computational resources. Our technique differs in that are actually building a database of landmarks.

A future step in our work is to incorporate loop-closure detection and possibly backwards correction. This could be incorporated in our outlier detection scheme as the number of outliers tends to spike when loops are closed. This is because a large number of SIFT matches are made but not in the expected locations. If loop closure can be robustly detected, we could switch to a 'kidnapped robot' scenario to reset the localization and make corrections backwards in time. Work must also be done to prune and rebalance the kd-tree to allow significantly longer operation of the algorithm. We also seek to make our approach more robust to the translation and rotation that can occur between consecutive images. Currently, we can tolerate only 5 cm translations and 2 degree rotations (our cameras have a 40 degree field-of-view). For practical applications we would like to be able to move 1 m in translation and 20 degrees in rotation. This would make the algorithm efficient enough for high-speed terrestrial applications as well as planetary exploration, where computational resources are scarce.

Lastly, we acknowledge that our results thus far are quite preliminary. We plan to test our system on more realistic environments and provide better quantification of performance. As part of this we will be migrating our vision system to a prototype Mars rover chassis and testing the combined system on a planetary-relevant terrain.

VII. Conclusion

We have demonstrated the ability for a rover to use SIFT features as the landmarks for efficient simultaneous localization and mapping. The resulting online visual motion estimation was used for autonomous outdoor rover traverses up to 120 m long on loose terrain. The final positioning errors were 0.5% to 4% of distance travelled, a major improvement over using odometry alone.

Acknowledgment

The author would like to thank Stephen Se and Piotr Jasiobedski for many useful insights on the use of SIFT features and for the FPGA implementation of SIFT feature extraction. Tariq Rafique, Robert Nguyen, and Raymond Oung provided invaluable help with implementation and field testing. MDA, Space Missions would also like to thank David Lowe at the University of British Columbia for providing most of the code related to SIFT features under license.

References

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[2] J. S. Beis and D. G. Lowe, "Indexing without invariants in 3d object recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 1000–1015, 1999.

[3] L. Goncalvez, E. Di Bernardo, D. Benson, M. Svedman, J. Ostrowski, N. Karlsson, P. Prijanian, "A Visual Front-end for Simultaneous Localization and Mapping" Barcelona, Spain: IEEE Int. Conf. on Rob. and Aut. (ICRA), April 2005.

[4] S. Se, H. Ng, P. Jasiobedski, and T. Moyung, "Vision Based Modeling and Localization for Planetary Exploration Rovers" Vancouver, Canada: 55th International Astronautical Congress (IAC), September 2004.

[5] N. Karlsson and E. Di Bernardo, "The vSLAM Algorithm for Robust Localization and Mapping" Barcelona, Spain: IEEE Int. Conf. on Rob. and Aut. (ICRA), April 2005.

[6] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges." Acapulco, Mexico: International Joint Conference on Artificial Intelligence (IJCAI), August 9-15 2003.

[7] S. Julier and J. Uhlmann, "A general method for approximating nonlinear transformations of probability distributions," 1996. [Online]. Available: citeseer.nj.nec.com/julier96general.html

[8] M. Montemerlo, "Fastslam: A factored solution to the simultaneous localization and mapping problem," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2003.

[9] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, J. of Basic Eng.*, vol. 82, pp. 35–45, 1960.

[10] W. F. Madow, "On the theory of systematic sampling, ii," *Annals of Mathematical Statistics*, vol. 30, pp. 333–354, 1949.

[11] J. Sherman and W. J. Morrison, "Adjustment of an inverse matrix corresponding to changes in the elements of a given column or given row of the original matrix," *Ann. Math. Stat.*, vol. 20, p. 621, 1949.

[12] E. Kraft, "A quaternion-based unscented kalman filter for orientation tracking." Cairns, Australia: Proceedings of the 6th International Conference on Information Fusion (ISIF), July 2003.

[13] L. Matthies and S. A. Shafer, "Error modeling in stereo navigation," *IEEE J. of Rob. and Aut.*, vol. RA-3, no. 3, pp. 239–248, 1987.

[14] S. Se, D. G. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *International Journal of Robotics Research*, vol. 21, no. 8, pp. 735–758, 2002.

[15] C. F. Olson, L. H. Matthies, M. Schoppers, and M. W. Maimone, "Rover navigation using stereo ego-motion," *Robotics and Autonomous Systems*, vol. 43, no. 4, pp. 215–229, 2003.

[16] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry." Washington, DC: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), June 2004.