Bayesian Visual Odometry

Julian L. Center, Jr.* and Kevin H. Knuth†,*

*Autonomous Exploration Inc., 385 High Plain Rd., Andover, MA 01810
†University at Albany (SUNY), Albany, NY

Abstract. Visual odometry refers to tracking the motion of a body using an onboard vision system. Practical visual odometry systems combine the complementary accuracy characteristics of vision and inertial measurement units. The Mars Exploration Rovers, Spirit and Opportunity, used this type of visual odometry. The visual odometry algorithms in Spirit and Opportunity were based on Bayesian methods, but a number of simplifying approximations were needed to deal with onboard computer limitations. Furthermore, the allowable motion of the rover had to be severely limited so that computations could keep up.Recent advances in computer technology make it feasible to implement a fully Bayesian approach to visual odometry. This approach combines dense stereo vision, dense optical flow, and inertial measurements. As with all true Bayesian methods, it also determines error bars for all estimates. This approach also offers the possibility of using Micro-Electro Mechanical Systems (MEMS) inertial components, which are more economical, weigh less, and consume less power than conventional inertial components.

Keywords: Navigation, Unmanned Ground Vehicle, Robotics, Machine Vision
PACS: 02.50Tt, 07.05.Bx, 07.07.Tw.

OVERVIEW

Application of Bayesian methods can integrate real-time stereo vision with an Inertial Measurement Unit (IMU) to produce an accurate navigation system for Unmanned Ground Vehicles (UGV). In keeping with current literature, we call this a Bayesian Visual Odometry (VO) system. However, it should be noted that this system determines all six degrees of freedom of vehicle motion, not just distance along track.

The overall objective is to develop a light-weight, low-power, self-contained module that can be interfaced to a wide variety of UGVs. This module will contain stereo cameras, an Inertial Measurement Unit (IMU), and an image-processing/navigation computer.

Micro-Electro-Mechanical Systems (MEMS) inertial sensors are attractive candidates for navigation system components because they are low cost, light weight, and consume little power. Although MEMS gyros and accelerometers can measure high-frequency vehicle motions accurately, they are subject to "pink noise" and long-term drifts. The BVO system uses Bayesian methods to combine dense-stereo vision, optical flow, and feature tracking to mitigate these long-term MEMS IMU errors.

Figure 1 is an overview of the system architecture. A MEMS IMU measures linear accelerations and angular rates. Inertial Navigation Calculations correct these signals for errors and integrate them to keep track of the vehicle velocity, orientation, and position relative to a navigation coordinate system. A Bayesian Information filter combines information derived from both optical flow and feature tracking to determine corrections to send to the Inertial Navigation Calculations. These corrections account for both

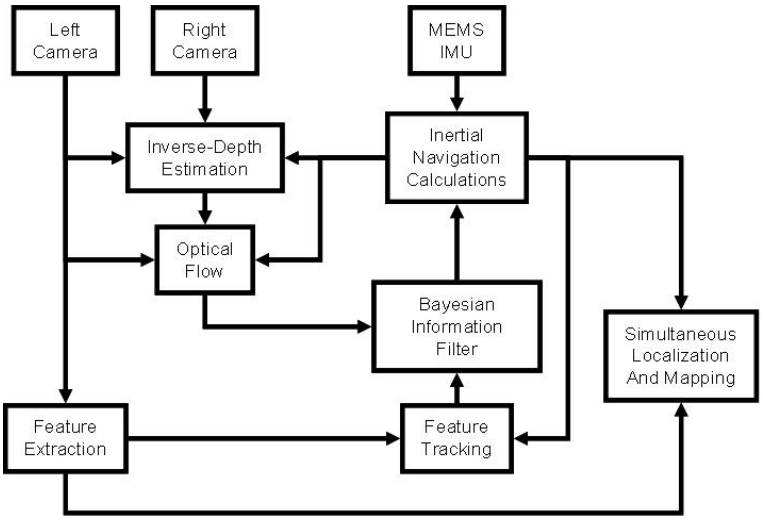

FIGURE 1. Overview of Bayesian Visual Odometry.

IMU errors and accumulated inertial navigation errors. Optical flow uses the linear and angular velocities from the inertial navigation calculations to predict the changes between images gathered a short time apart. Small unexpected changes can be attributed to errors in the linear and angular velocities. Large unexpected changes can be identified as moving objects.

To distinguish linear velocity errors from angular velocity errors, optical flow requires an estimate of the inverse-depth of every point in the image. With stereo camera pair, the inverse depth can be computed by correlating elements of the two images.

Optical flow alone may not be able to eliminate all long-term drifts. To deal with any residual long-term drifts, we incorporate a feature tracking algorithm. Feature tracking works in a manner similar to optical flow, but at a much slower update rate because it is far more computationally intensive.

Visual odometry is essentially an inverse problem: Given a sequence of stereo images and IMU signals, we want to estimate the vehicle movements (ego motion) and infer IMU errors so that corrections can be applied to the IMU outputs. Bayesian methods are noted for their ability to deal with inverse problems such as this.

It has been recognized for some time that Bayesian methods provide a consistent, logical approach to combining and interpreting data from a wide variety of sources. However, until recently, computer limitations have dictated that a number of approximations must be made to process data in a timely manner. While most VO systems use some Bayesian methods, previous work in this area has relied on a number of simplifying assumptions and maximum likelihood approximations [2], [3], [5], [6]. Now, computer capabilities have grown to the point that it is reasonable to consider implementing fully Bayesian methods, with few simplifying assumptions, in a real-time computer on board a small UGV.

Our fully Bayesian approach to VO utilizes more information from the images than previous methods. Previous VO algorithms have used corner detection algorithms (e.g., the Harris detector) to select features to track. Our fully Bayesian optical-flow method

does not explicitly select features to track. Instead it implicitly determines what can be learned from each image pixel and weights the information accordingly. This approach can work with images that have no distinct corners, which can be a significant advantage with low-contrast images from shadowed areas.

As with all true Bayesian methods, our approach keeps track of the uncertainties of its estimates. Therefore, the uncertainty in the vehicle's path will be known and can be used by Simultaneous Localization And Mapping (SLAM) algorithms to construct maps complete with uncertainty "error bars."

To create an efficient, real-time algorithm, we use optimized image-processing algorithms to perform almost all of the required computations. We are evaluating the feasibility of this approach by implementing the algorithms using image-processing algorithms from the Open Computer Vision (OpenCV) library [1]. The OpenCV library is an open-source collection of useful computer-vision functions that is widely used and vetted by the computer-vision research community.

DETAILS

Inertial Navigation System Calculations: Inertial navigation is based on Newton's second law of motion $f = ma$, which is a valid in a inertial (non-rotating, non-accelerating) reference frame. To determine motion in a non-rotating coordinate frame, such an earth-referenced frame, "ficticious" forces must be added. However, the target application for the BVO system is a low-velocity vehicle operating in a small area near the earth's surface and using low accuracy inertial instruments to locate itself relative to a navigation frame fixed to a point on the earth. Therefore, we can neglect ficticious forces, such as Coriolis effects, and the inertial navigation calculations can be comparatively simple.

The preferred design for a low-cost inertial navigation system is a strapdown configuration; that is, the gyros and accelerometers are hard mounted to the vehicle body, instead of on a gimballed platform. The gyros measure angular rates of the body relative to an inertial frame. These angular rates can be integrated to estimate the attitude of the body relative to the inertial frame. The accelerometers measure $\frac{f}{m} - g$, where $-g$ is the force countering gravity. Therefore, to determine acceleration we must use a model of gravity based on the current best estimate of vehicle position. Estimates of the vehicle velocities are produced by integrating the estimated acceleration, and estimates of the vehicle position are determined by integrating the etimated velocities.

Clearly, any instrument errors will accumlate and the navigation accuracy will degrade unless external references can be used to provide corrections, called "fixes" in the navigation literature. Although, the equations describing the propagation of navigation errors are nonlinear, if the errors are kept small, standard extended Kalman filter methods can be used to incorporated the navigation fixes.

We will show below how Bayesian optical flow methods can be used to process stereo video to produce both linear and angular velocity fixes. Visual feature tracking methods can be used to provide position and attitude fixes, but we will not describe these methods in detail here.

Perspective Projection: Prior to system operation, the cameras are calibrated to correct for distortions [1]. Therefore, we can model the imaging process as a perspective

projection [4, p. 49]. We can view perspective projection as the pinhole camera model shown in Figure 2. In this figure the camera's (x,y,z) axes are (temporarily) aligned

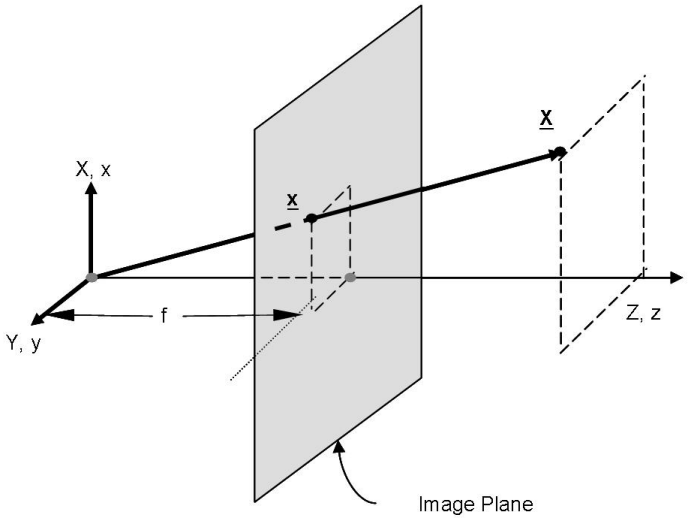

FIGURE 2. Pinhole Camera Model of Perspective Projection.

with the external (X,Y,Z) axes. From this figure, we can see that the point at $\underline{X}=\begin{bmatrix} X & Y & Z \end{bmatrix}^T$ projects to the point $\underline{x}=\begin{bmatrix} x & y & f \end{bmatrix}^T$ on the image plane. Noting the similarity of the triangles formed, we can see that $x = f\rho X$ and $y = f\rho Y$ where $\rho = Z^{-1}$ is the inverse depth.

Dense Stereo Vision: To implement stereo vision, we use two cameras hard-mounted to each other. For concreteness, we label one camera as left and the other as right. In addition to calibrating each camera, we determine the rotation and translation between the two camera-coordinate frames prior to operation. We transform the images so that corresponding points lie along a horizontal line. This process is called equipolar rectification [4]. The separation of the cameras in the new coordinates is the baseline b. When a point X is projected onto the rectified image planes, its images are located at $\underline{x}_L = f\rho\underline{X}$ and $\underline{x}_R = f\rho\left(\underline{X} - b\underline{e}_y\right) = \underline{x}_L - d\underline{e}_y$ where $\underline{e}_y$ is the unit vector in the y direction. Here $d = bf\rho$ is called the *disparity*.

Since we assume that the reflectance of the object does not change with the relatively small baseline shift, we can get a measurement of the disparity by shifting the right image, differencing it with the left image, and applying a function $c()$ that determines the matching cost. Thus, at a fixed time t, we form a three-dimensional array $C_o(d,x,y,t) = c[I_L(x,y,t) - I_R(x,y-d,t)]$. Scharstein and Szeliski [7] called this the initial disparity space image. Note that we have explicitly included the time index in this equation, for reasons that will become clear later.

We view the disparity space image as a stack of two-dimensional images that range over x and y, one image for each value of d. Generally, d will range over integer multiples of a pixel width. However, if we need sub-pixel resolution in d, we can interpolate the right image before differencing.

A variety of choices are available for the cost function $c()$. Popular choices are the square and the absolute value. To limit the effects of outliers, some researchers have

used cost functions that saturate at some maximum value.

Note that we can use standard image-processing methods to determine each layer of C_0. Thus we can use well-tested, optimized methods for these computations.

Because noise is usually present in the images, some kind of smoothness assumption is usually invoked to reduce the effects of noise. Here we will focus on methods that implicitly introduce a smoothness assumption by convolving the initial disparity function with a window function $w_d(x,y)$ to aggregate the cost in a region around each point. This convolution produces the disparity space image $C(d,x,y,t) = w_d(x,y) \otimes C_0(d.x,y,t)$, where denotes convolution over x and y.

A common choice of window function is simply a rectangle, which permits a very efficient implementation based on applying the "box filter" image-processing algorithm. If we use a rectangular window and the absolute-value cost function, we get the Sum of Absolute Differences (SAD) algorithm. Using a rectangular window with the square cost function gives the Sum of Squared Differences (SSD) algorithm [7].

If we model the image noise as Gaussian, we can use the SSD algorithm to arrive at a log likelihood function for the inverse depth at time t. To implement this, we can compute the log likelihood function by $\lambda(\rho|x,y,t) = -\frac{1}{4\sigma^2}C(bf\rho,x,y,t)$ .We can compute a similar log likelihood function if we use the SAD cost function. Once again, we can use standard image-processing algorithms to perform these calculations for each disparity layer.

For a Bayesian implementation, we compute the log of the posterior probability by combining the log likelihood function at time t with $\pi(\rho|x,y,t-\Delta t)$, the log of the probability distribution based on information up through time $t-\Delta t$. The log posterior probability is $\pi(\rho|x,y,t) = \pi(\rho|x,y,t-\Delta t) + \lambda(\rho|x,y,t) + const$.

The result of these calculations is a three-dimensional array of log probabilities that we can view as a stack of two-dimensional images. Each 2-D image corresponds to a particular value of the inverse depth. The value stored at (x,y) in that image corresponds to $\pi(\rho|x,y,t)$.

Optical Flow: In this section, we assume that the scene is static and the only change in the image is due to vehicle motion (ego motion). We also assume that the brightness of a point does not vary with a small change in viewing angle. In other words, the object at that location exhibits Lambertian reflectance. This assumption is valid for most natural surfaces, but is not valid for reflective surfaces, such as water or ice.

Optical-flow calculations view the image as moving relative to the camera coordinate frame. We can relate the relative motion of a point at $\underline{X}$ to the change in its projection by $\underline{u} \triangleq \underline{\dot{x}} = \frac{\dot{\rho}}{\rho}x + f\rho\underline{\dot{X}}$ where $u = \begin{bmatrix} \dot{x} & \dot{y} & 0 \end{bmatrix}^T$ is the change in the projection of the point. We can account for the relative motion of the point by $\underline{\dot{X}} = -\underline{\omega} \times \underline{X} - \underline{v}$ where $\underline{\omega}$ is the angular velocity of the vehicle and $\underline{v}$ is its linear velocity. Putting these equations together gives $\underline{u} = -\underline{\omega} \times \underline{x} - f\rho\underline{v} + \frac{\dot{\rho}}{\rho}\underline{x}$.

To see how to simplify this equation, we take its dot product with the unit vector $\underline{e_z} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$. The perspective projection determines that $\underline{e_z} \bullet \underline{u} = 0$ and $e_z \bullet x = f$. Therefore, we have

$$\frac{\dot{\rho}}{\rho} = \left(\underline{e_z} \bullet \underline{\omega} \times \underline{x}\right)/f + + \rho\underline{e_z} \bullet \underline{v} = (\omega_x y - \omega_y x)/f + \rho v_z$$

This yields $\underline{u} = -\underline{\omega} \times x - f\rho v + \left(\frac{1}{f} e_z \bullet \omega \times x + \rho e_z \bullet v\right) x$ or

$$u = \begin{bmatrix} (xy/f)\,\omega_x - \left(f + x^2/f\right)\omega_y + y\omega_z - f\rho v_x + x\rho v_z \\ \left(f + y^2/f\right)\omega_x - (xy/f)\,\omega_y - x\omega_z - f\rho v_y + y\rho v_z \\ 0 \end{bmatrix}$$

The basis of optical flow is the *brightness constancy constraint* [4, p.85]. This constraint asserts that the time derivative of the image intensity at a pixel location is related to the image motion by $I_t(\underline{x}) + \nabla I(\underline{x}) \bullet \underline{u}$. Here $I(\underline{x})$ represents the image intensity at image location $\underline{x}$, $I_t(\underline{x}) = \frac{\partial I(\underline{x})}{\partial t}$, and $\nabla I(\underline{x}) = \begin{bmatrix} \frac{\partial I(\underline{x})}{\partial x} & \frac{\partial I(\underline{x})}{\partial y} & 0 \end{bmatrix}^T$. In other words, the only change in the image is due to relative motion as seen in the image frame. To simplify the notation, we will not explicitly show the dependence on $\underline{x}$ and will merely write $I_t + \nabla I \bullet \underline{u} = 0$ with the understanding that this equation applies at every pixel location.

Now we can see how optical flow can provide an observation of ego motion. Combining the brightness constancy constraint with the equation from the previous section, we can write

$$I_t = \left(-\frac{y}{f} J - f I_y\right)\omega_x + \left(\frac{x}{f} J - f I_x\right)\omega_y + (x I_y - y I_x)\,\omega_z + f\rho I_x v_x + f\rho I_y v_y - \rho J v_z$$

with $J = x I_x + y I_y$. We can approximate the time derivative of the image intensity by differencing two images ∇t seconds apart and dividing by ∇t.

This is a linear equation in the angular velocity, but a bilinear equation in the inverse depth and the linear velocity, since it involves their product. We will assume that we have reasonably good estimates of the angular and linear velocities, $\hat{\underline{\omega}}$ and $\hat{\underline{v}}$, and we want to learn the errors in these estimates, $\delta\underline{\omega}$ and $\delta\underline{v}$. For the inverse depth, we have the estimate $\hat{\rho}$, and we want to learn the error, $\delta\rho$. We can linearize the equation for the illumination change by neglecting terms involving the products of errors to arrive at an equivalent measurement

$$\begin{aligned} z &= \frac{I(t) - I(t - \Delta t)}{\Delta t} - \underline{g}_\omega^T \hat{\underline{\omega}} - \underline{g}_v^T \hat{\underline{v}} - g_\rho \hat{\rho} \\ &\approx \underline{g}_\omega^T \delta\underline{\omega} + \underline{g}_v^T \delta\underline{v} + g_\rho \delta\rho + n \end{aligned}$$

Here

$$\begin{aligned} \underline{g}_\omega^T &= \begin{bmatrix} \left(-\frac{y}{f} J - f I_y\right) & \left(\frac{x}{f} J + f I_x\right) & (x I_y - y I_x) \end{bmatrix} \\ \underline{g}_v^T &= \begin{bmatrix} \hat{\rho} f I_x & \hat{\rho} f I_y & -\hat{\rho} J \end{bmatrix} \\ g_\rho &= f I_x \hat{v}_x + f I_y \hat{v}_y - J \hat{v}_z \end{aligned}$$

and n, which accounts for imaging noise and approximation errors, is assumed to be zero-mean Gaussian and uncorrelated among image locations. If we have good stereo vision and the linear velocity is relatively small, it is reasonable to account for any residual error in the inverse depth by increasing the measurement noise variance R. Then

we can write the measurements as

$$z(\underline{x}) = \left[\begin{array}{cc} \underline{g}_{\omega}^{T}(\underline{x}) & \underline{g}_{v}^{T}(\underline{x}) \end{array} \right] \left[\begin{array}{c} \delta\underline{\omega} \\ \delta\underline{v} \end{array} \right] + n(\underline{x}) = \underline{g}^{T}(\underline{x}) \left[\begin{array}{c} \delta\underline{\omega} \\ \delta\underline{v} \end{array} \right] + n(\underline{x})$$

Here we have restated the dependency on the location in the image to emphasize that this equation applies at each image location with every term. Therefore, we can view the change in the image intensity at each pixel as providing an independent measurement of a combination of the errors in linear and angular velocities. This means that the collection of measurements represented by $z(\underline{x})$ above can be viewed as an image, and we can process all of these measurements at the same time using image-processing methods.

We can combine all of the independent measurements into an equivalent measurement of the linear and angular velocity errors. Since we assumed that the measurement noises are independent and Gaussian, we can write the joint log likelihood function corresponding to all of the optical flow measurements as

$$\begin{aligned} \log p(z(\underline{x}) \,|\, \delta\underline{\omega}, \delta\underline{v}) &= -\frac{1}{2} \sum_{\underline{x}} \left\{ z(\underline{x}) - \underline{g}^{T}(\underline{x}) \left[\begin{array}{c} \delta\underline{\omega} \\ \delta\underline{v} \end{array} \right] \right\}^{2} R^{-1}(\underline{x}) + const \\ &= -\frac{1}{2} \left\{ \underline{m} - \Omega \left[\begin{array}{c} \delta\underline{\omega} \\ \delta\underline{v} \end{array} \right] \right\}^{T} \Omega^{\dagger} \left\{ \underline{m} - \Omega \left[\begin{array}{c} \delta\underline{\omega} \\ \delta\underline{v} \end{array} \right] \right\} + const \end{aligned}$$

where $\underline{m} = \sum_{\underline{x}} \underline{g}(\underline{x}) R^{-1}(\underline{x}) z(\underline{x})$ is a 6-dimensional vector representing the equivalent measurement, $\Omega = \sum_{\underline{x}} \underline{g}(\underline{x}) R^{-1}(\underline{x}) \underline{g}^{T}(x)$ is the 6 by 6 the covariance matrix of an equivalent noise $\underline{n}'$, and the symbol † represents the pseudo-inverse of the matrix. This equivalent measurement can be used to update the extended Kalman filter by noting that $\underline{m}$ is related to the filter state $\underline{s}$ by $\underline{m} = \Omega L \underline{s} + \underline{n}'$, where the matrix L relates filter state vector to the errors in the angular and linear velocities as seen in the camera coordinates.

Thus, we can view $\underline{m}$ as an equivalent linear, Gaussian measurement of the navigation error states, with corresponding measurement matrix Ω and noise covariance Ω. The pixel-by-pixel products and the summations needed to compute the vectors and matrices above can be implemented using standard image processing methods. Computing the terms in $\underline{g}(\underline{x})$ above requires computing the image gradients I_x and I_y, which can also be performed using well understood and highly optimized algorithms in the OpenCV library. Furthermore, we can pre-compute and store images that represent $\underline{x}$ and $\underline{y}$ in the equations above. We can then compute products such as $I_x\underline{x}$ using element-by-element multiplication of the image I_x by the stored $\underline{x}$ image. This is also a standard image-processing algorithm that has been optimized in the OpenCV library. In summary, all of the calculations needed to set up Bayesian optical flow can be efficiently computed using standard image-processing algorithms.

Note that areas of the image that have more detail will have a larger H and, therefore, will have more influence on the estimate. Thus the Bayesian method automatically weights features according to their value for estimation.

Dealing with Moving Objects: An advantage of combining inertial information with vision is the ability to distinguish moving objects in the field of view from ego motion.

Moving objects can potentially introduce errors into the optical flow calculations. However, because we have the benefit of the INS, we can distinguish ego motion from a moving object. Vision alone cannot make this distinction without making assumptions about the size and locations of moving objects. Recall that we expect INS errors to be relatively small and low frequency. Therefore, any area in the image that moves faster than expected can be identified as a moving object.

We can use this information to reduce the effects of moving objects on the updates to the information filter. We do this by changing the model of the measurement noise. Recall that we are assuming that the measurement noises are uncorrelated with each other and, therefore, their joint covariance matrix R is diagonal. Furthermore, we only use the inverse of R in the equivalent measurement calculations. Therefore, we can account for the possibility of a moving object at a particular image location by multiplying the corresponding element of R^{-1} by $p(static|z(\underline{x})) = [1 - p(moving|z(\underline{x}))]$. If $p(static|z)$ is zero, the effect of that pixel location on the equivalent measurement will be zero.

This method is also suitable for dealing with the vehicle's own shadow. Since the shadow moves with the vehicle, it will appear to be a moving object, moving at the same speed as the vehicle. Therefore, the motion of its image will not match the predictions from the INS, and it will be identified as a moving object.

CONCLUSION

Using very few assumptions, we derived equations for implementing a fully Bayesian approach to combining machine vision and inertial measurements to track the motion of a ground vehicle. These equations can produce an equivalent measurement of linear and angular velocity error as well as the uncertainty in this equivalent measurement. They are structured to take advantage of modern computer hardware that is designed to perform image processing functions efficiently. An ongoing project is investigating the accuracy of this approach under a variety of conditions.

REFERENCES

1. G. Bradski and Adrian Kaebler, Learning OpenCV: Computer Vision and the OpenCV Library, O'Reilly, 2004.
2. Y. Cheng, M. Maimone, L. Matthies, "Visual Odometry on the Mars Exploration Rovers," IEEE Conference on Systems, Man and Cybernetics, The Big Island, Hawaii, October 2005.
3. P. Corke, J. Lobo, and J. Dias, "An Introduction to Inertial and Visual Sensing," The International Journal of Robotics Research, No. 26, pp. 519-535, 2007.
4. Y. Ma, S. Soatto, J. Kosecka, S. Sastry, An Invitation to 3-D Vision, Springer, 2006.
5. M. W. Maimone, P. C. Leger, and J. J. Biesiadecki, "Overview of the Mars Exploration Rovers' Autonomous Mobility and Vision Capabilities," IEEE International Conference on Robotics and Automation (ICRA) Space Robotics Workshop, Roma, Italy, 14 April 2007.
6. D. Nistér, O. Naroditsky, and J. Bergen, "Visual Odometry for Ground Vehicle Applications," Journal of Field Robotics, Volume 23, Number 1, January 2006.
7. D. Scharstein and R. Szeliski, A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithm, MSR-TR-2001-81, Microsoft Research, Microsoft Corporation, November 2001.