Planar Surface Tracking Using Direct Stereo

Jason Corso, Greg Hager
Computational Interaction and Robotics Laboratory
The Johns Hopkins University
3400 North Charles Street
Baltimore, Maryland, 21218, USA
{jcorso,hager}@cs.jhu.edu

Abstract

We present a new method for tracking planar surfaces based on the direct update of surface parameters from two stereo images. The plane tracking algorithm is posed as an optimization problem and maintains an iteratively re-weighted least squares guess of the plane's orientation using direct pixel-measurements. The algorithm is discussed and analyzed, and an application to augmented reality is presented.

1 Introduction

Inferring properties of the real-world through one or many images is fundamental to the field of computer vision. Often it is beneficial to have knowledge of a significant surface or set of surfaces during the inference process. One such surface is the planar surface.

Many methods have been proposed to solve this problem for the monocular [10, 11] and binocular [17, 15], calibrated and uncalibrated cameras (similar to using sequences of images [3, 2, 5, 19]) .

A common solution involves a disparity map computation. A disparity map is a matrix of correspondence offsets between two images [20]. The disparity map calculation employs expensive neighborhood correlation routines that often yield sparse maps for typical scenes. However, the method makes no assumptions about the environment and has been widely used for the case of general stereo vision.

In contrast to the general solution discussed above, our method exploits a property that planar surfaces exhibit when viewed through a non-verged stereo camera. A well known transformation, called a homography, can be employed to warp the plane in one image onto the plane in another image [4]. When using rectified images[1], the epipolar geometry reduces the homography to a linear function that yields the disparity value on the plane between the images. We use a direct method to perform the tracking. Direct methods use quantities that are calculated directly from image intensity values [9, 17]. Our

[1]Rectification is the application of a linear transformation to the image pair that forces collinearity amongst conjugate epipolar lines and aligns them with the horizontal image axis. The transformation is pre-computed during calibration from the epipolar geometry. [20, 4]

method has descended from earlier image registration and enhancement techniques [14, 12] and from visual tracking [6]. Similarly, it poses the tracking problem as one of objective function minimization. Doing so incorporates a *vote* from many pixels in the image and computes a least-squares estimate of the global motion parameters in question to sub-pixel levels.

In the remainder of this paper we motivate and discuss our algorithm. Then, we present an analysis of its capabilities, our implementation, and some experimental results. Finally, we conclude with an application to augmented reality and some possible extensions to the algorithm is presented.

2 Planar Disparities

To efficiently track a plane, we can make use of a property that planar surfaces exhibit when viewed in non-verged stereo images. In general, there is a well-known homography that maps the image plane of one image to the plane of another image. In the following discussion, let (u,v,w) be image plane coordinates, (x,y,z) be world coordinates, (α,β) correspond to an arbitrary image pair, and s be an arbitrary scale factor. We call the homographic matrix $\{H : h_1 \dots h_9\}$.

$$s\begin{bmatrix} u_\alpha \\ v_\alpha \\ w_\alpha \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} u_\beta \\ v_\beta \\ w_\beta \end{bmatrix} \tag{1}$$

It would suffice to track the nine parameters of H. However, one can simplify the tracking problem because of the rectified pair of images. Since conjugate epipolar lines are collinear and horizontal, the mapping becomes a simple linear function in three variables (see Figure 1 for a graphical depiction of this property).

$$s\begin{bmatrix} u_\alpha \\ v \\ w \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_\beta \\ v \\ w \end{bmatrix} \tag{2}$$

Thus, the disparity from this function is $u_\beta - u_\alpha$.

2.1 Algebraic Derivation

In this section, we show an algebraic derivation of (2). Here, we assume that the image plane lies at $w = 1$ without loss of generality. If we first take the equation of a plane in three-dimensional space,

$$Ax + By + Cz + E = 0, \tag{3}$$

we can manipulate the equation so that it becomes a linear map. In the following equations, f is the focal length, s_x is the pixel size scale factor, T is the baseline of the stereo system, and D is the disparity.

$$A\frac{x}{z} + B\frac{y}{z} + C = -\frac{E}{z} \tag{4}$$

$$\frac{Tf}{s_x E}[Au + Bv + C] = -D \tag{5}$$

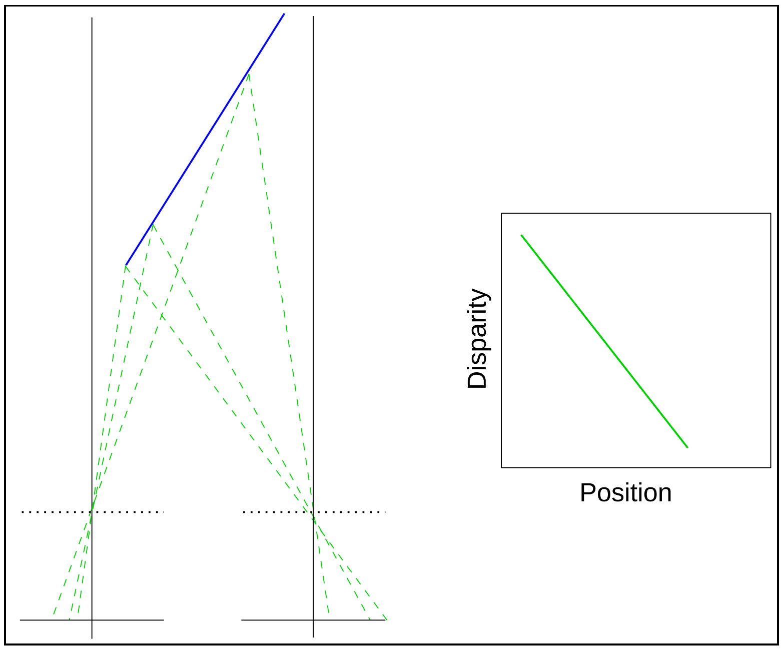

Figure 1. When viewed through non-verged stereo, the disparity of a planar surface can be represented as a linear function. Here, a 2d representation is shown for clarity.

$$
\begin{aligned}
u &= \frac{x}{z} \\
v &= \frac{y}{z} \\
z &= \frac{Tf}{s_x D}
\end{aligned}
$$

It becomes apparent in Equation 5 that, because we are viewing a planar surface in a calibrated stereo camera, we can make use of the known T and remove the z from the equation. The resulting map (Equation 6) is expressed in image coordinates centered about the known optical center.

$$
\begin{aligned}
h_1 u + h_2 v + h_3 &= -D \qquad (6) \\
h_1 &= \frac{TA}{E} \\
h_2 &= \frac{TB}{E} \\
h_3 &= \frac{TfC}{s_x E}
\end{aligned}
$$

Thus, if we can compute and maintain a good guess of the coefficients in this equation, then we are effectively tracking the orientation of the planar surface. From Equation 6, we can easily compute the coefficients of the three-dimensional plane.

$$\begin{aligned}
A &= \frac{Eh_1}{T}; \\
B &= \frac{Eh_2}{T}; \\
C &= \frac{Eh_3 s_x}{Tf}; \\
E &= \frac{T}{\|h_1 h_2 \frac{s_x h_3}{f}\|}
\end{aligned}$$

3 Planar Tracking

From the observation made in Section 2, we see that tracking the parameters $\mathbf{h} = h_1, h_2, h_3$ of the linear map (6) is equivalent to tracking the planar surface. Thus, assuming inter-frame motion is relatively small and both brightness and contrast shifts can be removed, we pose this problem as one of optimization.

3.1 Optimization Formulation

Consider a rectified pair of stereo images, L and R. Based on (6), we can relate the images with the following formula. Let $D(u,v) = h_1 u + h_2 v + h_3$.

$$L(u,v) = R(u - D(u,v), v) \tag{7}$$

Traditionally, similar approaches make use of the so-called *brightness constancy constraint* to enforce (7) [8]. Since we use a stereo pair of images taken with different cameras, we enforce the brightness constancy constraint by performing a zero-mean process on the images. Given an image I, $\overline{I} = I - \sum I$. The planar parameters for the current pair are estimated through a minimization of the following least-squares objective function.

$$E(\mathbf{h}) = \sum (\overline{L}(u,v) - \overline{R}(u - D(u,v), v))^2 \tag{8}$$

Let $\delta\mathbf{h}$ represent the set of vector offsets. Assuming a small magnitude for $\delta\mathbf{h}$ we can solve the minimization by linearizing the expression through a Taylor expansion about $\mathbf{h}$.

$$E(\delta\mathbf{h}) \approx \sum (\overline{L}(u,v) - \overline{R}(u - D(u,v), v) + uI_x\delta h_1 + vI_x\delta h_2 + I_x\delta h_3)^2 \tag{9}$$

Here, I_x refers to the spatial gradient of the right image. We neglect the higher order terms of the Taylor series. This method uses quantities directly measurable from pixel values in contrast to feature based methods; it is a so-called *direct method*. For a thorough discussion of *direct methods*, the reader is encouraged to peruse [9]. Other *direct method* algorithms are [6, 16, 17, 19].

We solve the system using Singular-Value Decomposition (SVD) [18]. It is first convenient to define the error term: $e(u,v) = \overline{L}(u,v) - \overline{R}(u - D(u,v), v)$.

$$\begin{bmatrix} e(u_1,v_1) \\ e(u_1,v_2) \\ \cdots\cdots \\ e(u_m,v_n) \end{bmatrix}_{mn \text{ x } 1} = \begin{bmatrix} u_1 I_{x_{1,1}} & v_1 I_{x_{1,1}} & I_{x_{1,1}} \\ u_1 I_{x_{1,2}} & v_2 I_{x_{1,2}} & I_{x_{1,2}} \\ \cdots & \cdots & \cdots \\ u_m I_{x_{m,n}} & v_n I_{x_{m,n}} & I_{x_{m,n}} \end{bmatrix}_{mn \text{ x } 3} \begin{bmatrix} \delta h_1 \\ \delta h_2 \\ \delta h_3 \end{bmatrix}_{3 \text{ x } 1} \tag{10}$$

3.2 The Weighting Matrix

To increase the solution's robustness and reduce computation time, we incorporate a binary weighting scheme into (9). We define a weighting matrix, W, with an entry per pixel. Such a mask removes inaccurate and poor pixel matches from the SVD solution which decreases its processing demand and increases its stability. Equation (9) is extended to (11) incorporating such a mask; W(u,v) is the value of the mask at (u,v).

$$E(\delta\mathbf{h}) \approx \sum \delta_{W(u,v)=1}[(\overline{L}(u,v) - \overline{R}(u - D(u,v),v) + uI_x\delta h_1 + vI_x\delta h_2 + I_x\delta h_3)^2] \tag{11}$$

In each frame, we compute a normalized cross-correlation ($\odot$) measure to fully recompute the mask matrix based on the current parameters. By fully recomputing the mask each frame in this manner, the algorithm tracks the plane under arbitrary planar motion and is robust to occlusion.

$$\eta_{u,v} = \overline{L}(u,v) \odot \overline{R}(u - D(u,v),v) \tag{12}$$

Since (11) is only sensitive to pixels that demonstrate horizontal gradients, we also mask pixels with low horizontal variance by sampling the correlation response along the epipolar line.

$$\begin{aligned} \alpha_{u,v} &= \frac{\overline{L}(u,v) \odot \overline{R}(u - D(u,v) + \Delta, v)}{\eta_{u,v}} \\ \beta_{u,v} &= \frac{\overline{L}(u,v) \odot \overline{R}(u - D(u,v) - \Delta, v)}{\eta_{u,v}} \end{aligned} \tag{13}$$

$$W(u,v) = (\eta_{u,v} > \tau) \wedge (\alpha_{u,v} > \varepsilon) \wedge (\beta_{u,v} > \varepsilon) \tag{14}$$

In (14), $0 < \tau < 1$ and $\varepsilon > 1$. In (13,14), the selection of Δ,τ, and ε is dependent on the imaging properties of the system and the scene. To increase the robustness of the mask generated in this manner, we also perform a morphological dilation followed by an erosion [6]. This has the effect of removing noisy correlation responses and joining contiguous regions.

3.3 The Tracking Algorithm

From the previous section, we see that it is possible refine a guess at the parameters of the planar map. Since we are interested in tracking the plane through time, we may also use the formulation of (9) to track a plane. Doing so efficiently will yield small inter-iteration magnitudes of $\delta\mathbf{h}$ which are needed to guarantee convergence. We use the following algorithm to track in the image.

ALGORITHM PLANAR_TRACK

```
1 Compute an initial guess of h
2 Build a pixel-mask based on this guess
3 For each subsequent frame
4         Zero-Mean the images
5         Convolve with a gaussian and decimate in half
6         Compute the derivative of the right image
7         Build linear system for all pixels that satisfy the mask
8         Solve the system using SVD
9         Update the mask
```

The tracking begins with a guess for the planar parameters, $\mathbf{h}$. Because of the assumptions made in (9), the algorithm has a relatively small convergence radius (± 1 disparity). Thus, a good initial guess is required; otherwise, an expensive global optimization step would need to be solved. In this case, a closed-form solution for the initial guess is possible through the use of an initial disparity map, $\hat{D}$. Since the result of this computation is crucial to the success of the algorithm (convergence), we compute a least-squares estimate and employ only the most confident of disparities.

$$\begin{bmatrix} u & v & 1 \\ & \cdots\cdots & \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \hat{D}_{u,v} \\ \cdots \end{bmatrix} \tag{15}$$

One might use a different approach to compute the seed by computing a least-squares fit of a plane to a cloud of reconstructed plane-points. The initial pixel mask, W^0, is created by verifying the disparity in $\hat{D}$ against the planar parameters, h_1, h_2, h_3.

4 Analysis

4.1 Theoretical Convergence Radius

We analyze the algorithm for maximum guaranteed convergence. Equation (9) was derived under the assumption that inter-frame parameter offsets ($\delta\mathbf{h}$) would be small in magnitude. In discrete terms, we are restricted to changes of ± 1 disparity. Figure (2) depicts the inverse relationship between the depth and disparity ($D = \frac{Bf}{s_x z}$). To facilitate analysis, we assume an infinite plane.

Under lateral translation, the depth and the disparity are constant and the algorithm is guaranteed to converge. However, under orthogonal translation, the convergence radius is dependent on the current depth; the closer to the camera, the smaller the convergence radius. Let ζ be the change in depth for guaranteed convergence.

$$\frac{Bf}{s_x D \pm 1} - z = \zeta \tag{16}$$

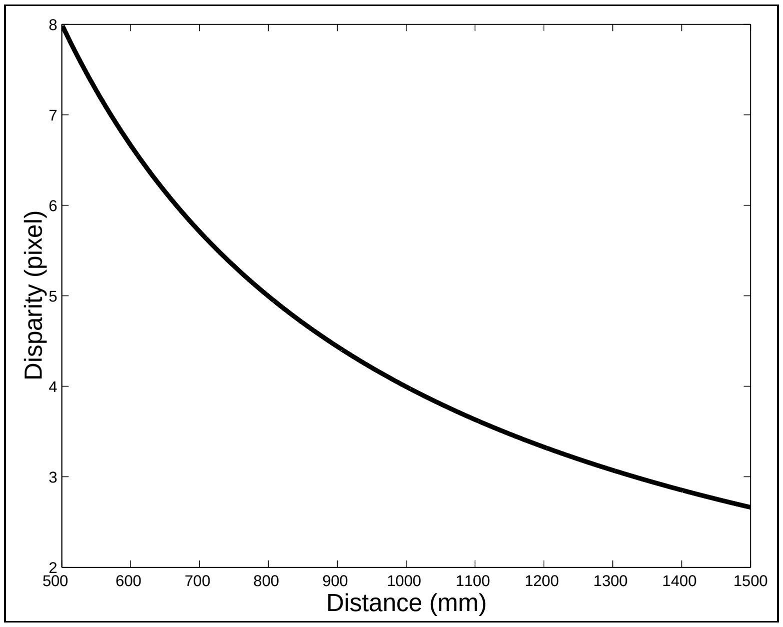

Figure 2. Disparity is inversely proportional to depth.

Rotations in the plane are also dependent on the current depth. However, for planes at normal depths (greater than a few centimeters), we make the observation that to cause a disparity change of one for a given pixel, the plane would have to rotate almost ninety degrees. We can approximate the rotation angle that would cause a disparity change of one by the following formula:

$$\theta = \arctan\left(\frac{B\Delta}{D \pm 1} - \frac{z}{\Delta}\right) \tag{17}$$

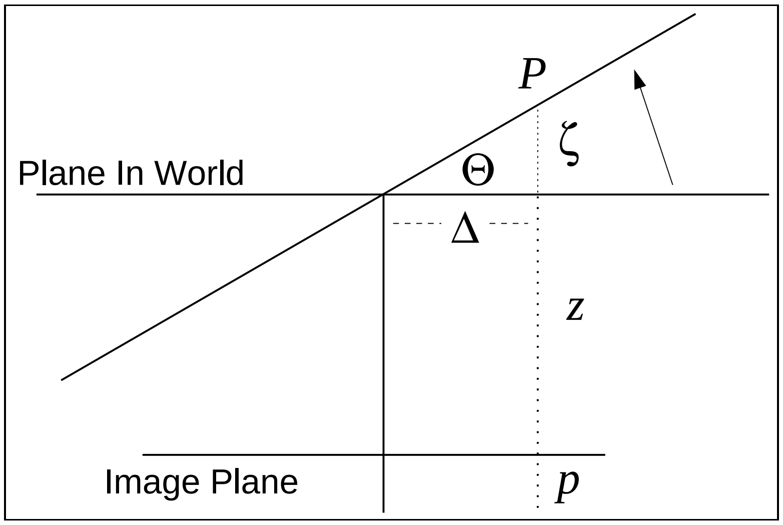

Figure 3. Depiction of the formula for convergence radius under rotation.

In Figure (3), we see a depiction of this formula. We can better simplify (17) for a change of one pixel off the optical center to

$$\theta = \arctan\left(\frac{1}{D \pm 1} - \frac{1}{D}\right). \tag{18}$$

In (18), we see that for typical disparities we'll be taking the arctan of a small number resulting in angles near ninety degrees. Thus, theoretically, the algorithm will converge under large interframe rotations because change in the disparity is small. However, this analysis assumes the plane is directly down the optical axis. As we increase our distance off the optical axis the arctan value will increase resulting in higher disparity shift for the same angle.

4.2 Algorithmic Analysis

The frequency at which we perform an iteration of the algorithm is crucial to its operating well. If the period of an iteration is too long, the algorithm is impractical for real-time applications because too little motion will be permitted. The convergence speed is increased if we can perform multiple inner iterations (steps 7 through 9).

Our algorithm's complexity is dominated by the SVD solution (step 8), $O(n^3)$. All other steps are at most linear in the number of pixels. While the SVD solution is expensive, in actuality, the size of the matrix on which we are computing SVD is relatively small: at most $3xN$, where N is the number of overlapping pixels in the decimated images. In practice only a fraction of these pixels satisfy our constraints (14) and are included in the computation. Since each extra pixel included in this SVD solution is only contributing to our solution, it is also possible to limit the number of pixels included in the computation to those in which we hold the highest confidence.

In contrast, if we compare our approach to an algorithm that employs a general disparity map computation, we see that our execution time is smaller even though we employ the SVD. A disparity map is computed through a correlation process. Once the image pair has been rectified, each pixel in the left image is matched to a pixel in the right image on the same scan-line. This search is limited to a search window, typically 32 or 64 pixels. The larger the window the larger the horopter of disparities. Often to increase confidence the same search is repeated from the right to left. The time complexity of this process is $O(n^2 w)$, where w is the size of the search window. Disparity map computation is an expensive procedure.

5 Implementation and Experiment Results

The algorithm has been implemented on a Pentium III 700MHz PC running Linux (rev. 2.4). We use a Videre TMMEGA-D Megapixel Digital Stereo Head for the stereo camera. The SVS library is used for calibration and the computation of the disparity map that seeds our algorithm. The software library we use for video-acquisition, image-processing, and gui display is the XVision2 library [7, 13]. We also use the Intel Integrated Performance Primitives library for accelerated image-processing.

For the experiments discussed in this section, we are using a stereo head with 5.18mm lenses, a 92mm baseline, and square pixels 0.06mm wide. The plane being observed, unless otherwise specified, is roughly orthogonal to the viewing axis and at a depth of one-meter.

In the following sections, experiment results are presented and discussed.

5.1 Convergence Radius

For a controlled environment with a stationary plane and camera, we calculated a initial guess for the plane parameters and then varied this guess to test the robustness of the tracking algorithm.

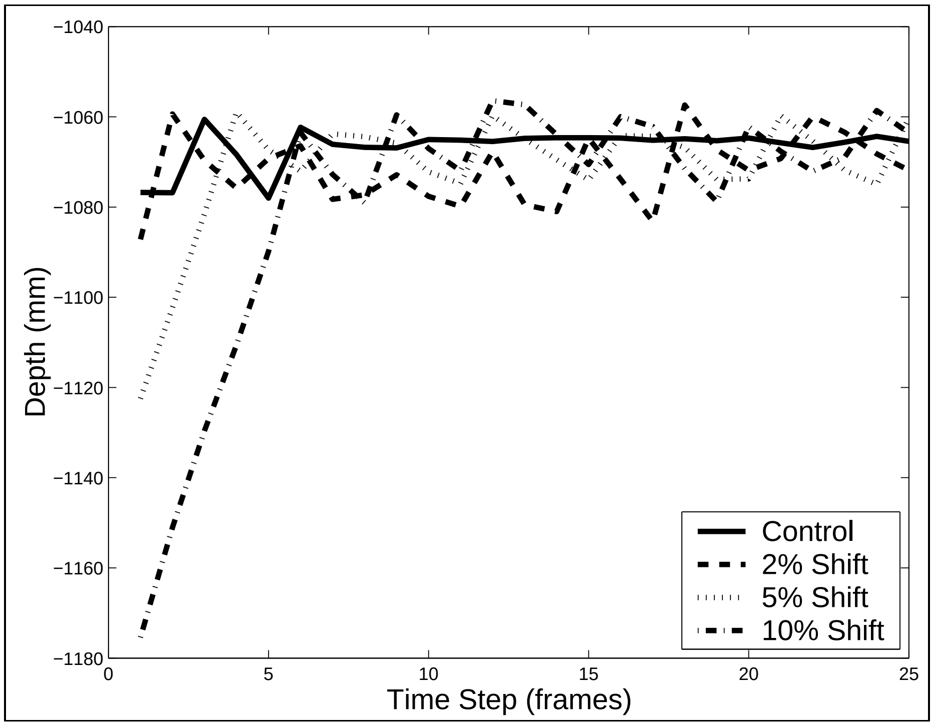

Figure 4. Graph for convergence while introducing error into the seed's depth by 2, 5 and 10 percent toward the camera.

In Figure (4), we show the time to convergence when we shift the seed's depth closer to the camera at varying levels. The convergence speed is directly proportional to the magnitude of the introduced error. We note that the convergence speed is only about 5 frames for an error of 10%. In Figure (5), we show the convergence speed while altering the plane's normal about a single axis. We see similar convergence rates to those observed while altering seed depth.

5.2 Accuracy of Parameter Estimation

Assuming a suitably textured scene, the algorithm estimates a plane's parameters with sub-pixel accuracy. However, this estimation accuracy varies with the depth of the plane being tracked (see Figure 2). Because the depth-per-disparity increases as the distance to the plane increases, the estimation accuracy degrades with depth. Table 1 shows the statistics for the plane.

For the following experiments, we mounted the camera atop mobile robot and used the robot's internal odometry as the ground truth for the rotation and translation of the camera. Figure (6) shows the robot performing oscillatory rotations in front of a plane (700 mm distance), and figure (7) show the robot orthogonally translating in a back-and-forth motion. We see that the algorithm performs extremely well for the rotational motion. The estimated orientation lags minimally behind the odometric values;

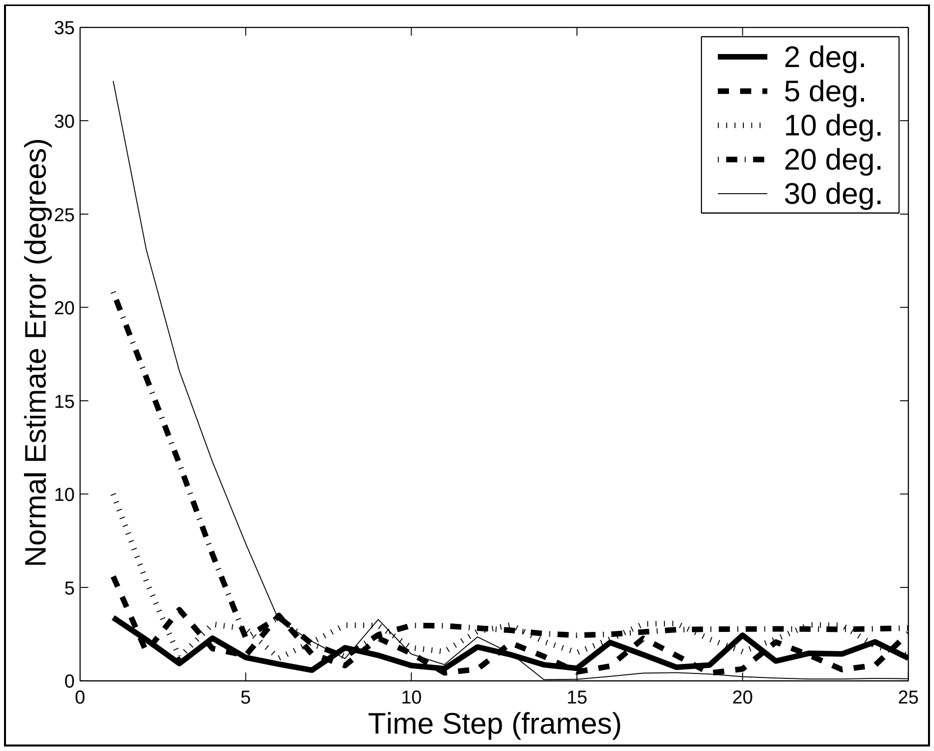

Figure 5. Convergence rates for error in seed normal.

	Z Mean	Z Std Dev	Normal Error Std Dev
1	1064.8mm	2.2359mm	0.2947°
2	1065.3mm	1.7368mm	0.2673°
3	1065.2mm	1.5958mm	0.2258°

Table 1. Parameter estimation accuracy for a plane one meter away.

the length of the lag is proportional to the convergence speed. During the translational motion, the algorithm oscillates about the control value during motion. As expected, it is apparent from the figure that the tracking performs better when the robot is closer to the plane.

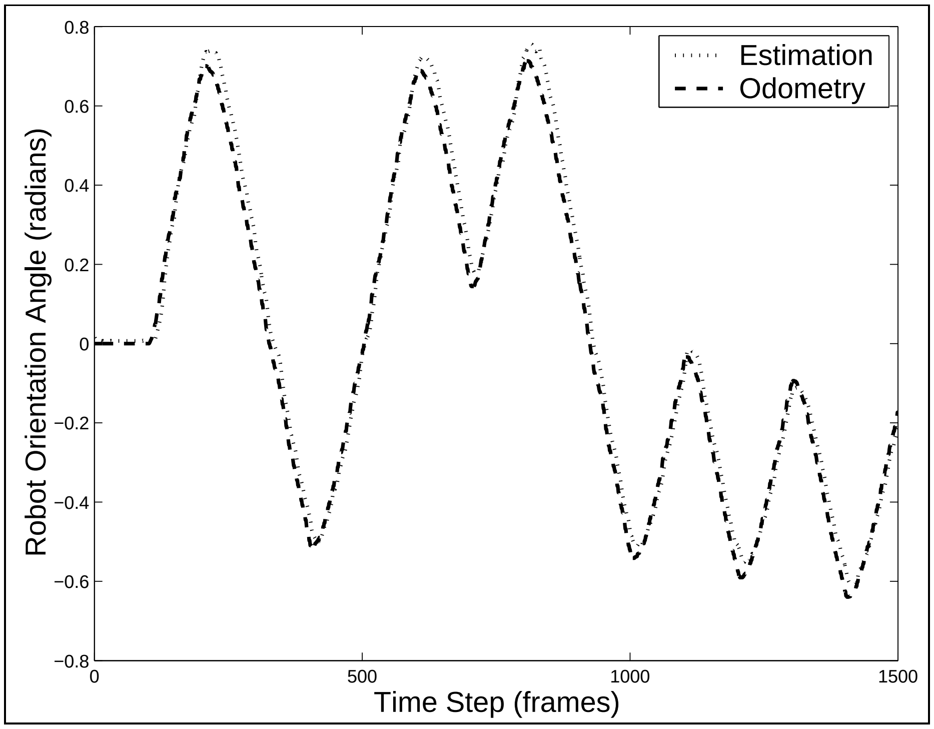

Figure 6. Accuracy of estimated orientation.

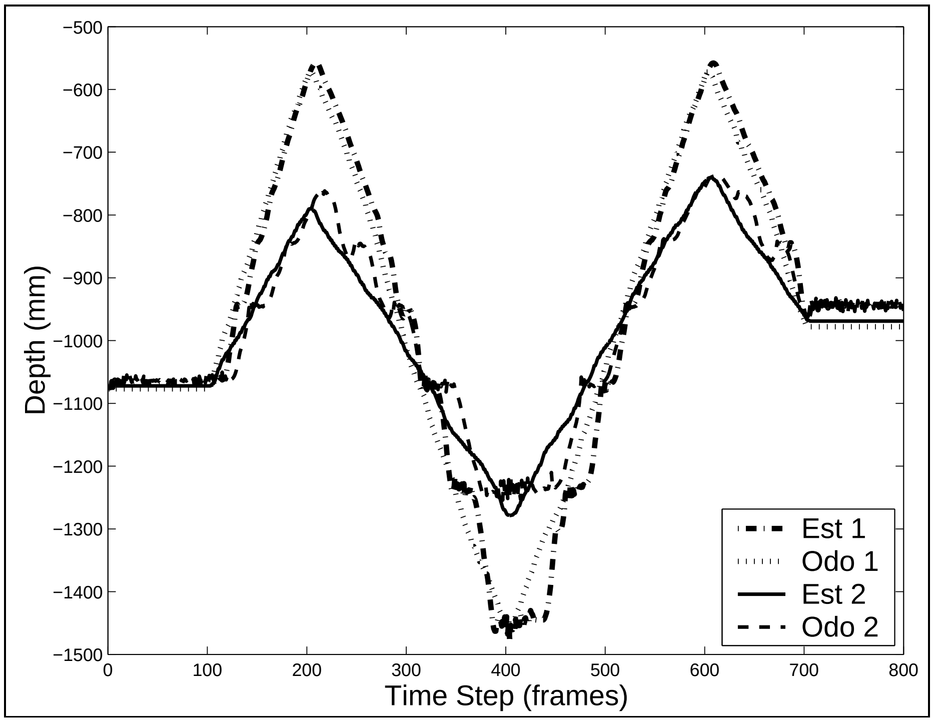

Figure 7. Accuracy of estimated depth.

6 Application To Augmented Reality

We are very interested in vision-based interaction and its implications in the future state-of-the-art in human-computer interation. One typical augmented reality mode is video-see-through augmented reality (VSTAR) [1]. In VSTAR, the user wears a closed head-mounted display (HMD) much like in typical virtual reality applications. Atop the HMD is a pair of video-cameras that feed live streams of video through the computer and then into the displays for the HMD. An image of our setup appears in Figure 8.

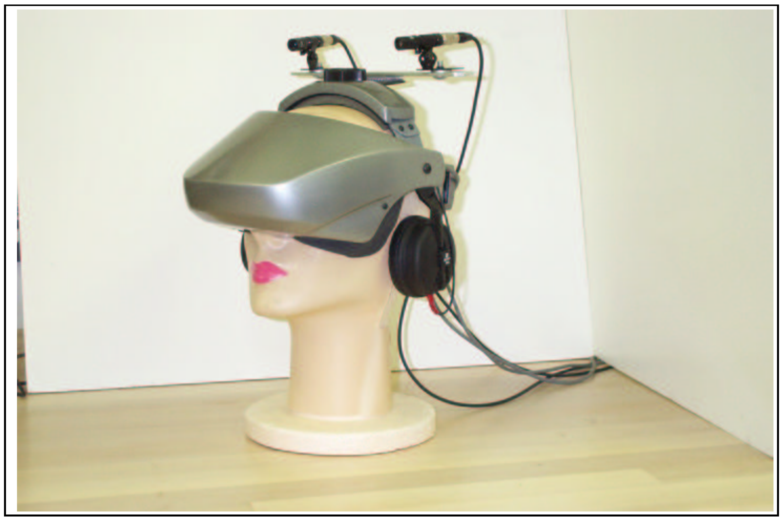

Figure 8. Video See-Through Augmented Reality with a 5DT HMD and two Sony Lipstick cameras.

One important component of VSTAR is the ability to process the video during the pipeline (Figure 9). At this step we can measure important quantities about the environment that enable us to render artificial objects into to the user's view. We wish to augment an ordinary desktop environment where the user dons the HMD and works at an ordinary desk. In order to make the augmented reality configuration similar to ordinary activity we must be able to render new objects on the desk plane for the user. For instance, we may wish to render an artificial *TODO* list onto the desktop, a set of artificial buttons for the user, or perhaps a piano onto the desktop.

To accomplish such tasks we must know where to render objects in the left and right video streams such that they lie on the plane; i.e. we must know the map relating the plane in the two images. Conveniently, our algorithm tracks precisely this map. Below (10) is a screenshot where a quadrilateral is rendered onto plane in the video streams.

However, since we are only tracking the orientation of the plane, our approach would have to be supplemented with knowledge of some origin on the plane. Then, whenever we wish to draw something on the plane, it would be drawn with respect to the current location of the origin. One simple way to allow for this functionality is to place a fiducial, perhaps a colored blob or a small template, on the plane and track its location; to accomodate occlusion multiple fiducials could be placed on the surface even though only one is thereotically needed.

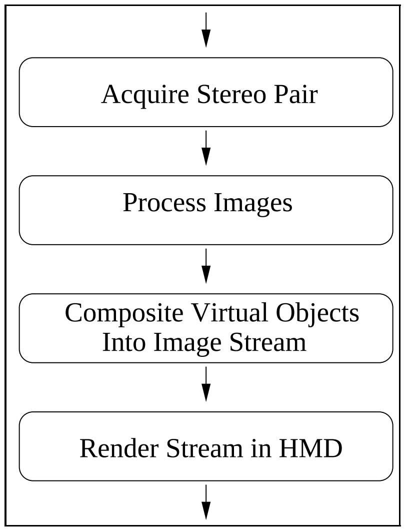

Figure 9. VSTAR Processing Pipeline.

7 Conclusion and Future Work

In this paper, we presented a new algorithm that employs objective function minimization to track the parameters of a plane. The algorithm is a direct method in contrast to more common feature-based methods; using the disparity map, for instance, relies on pixel correspondences. As is widely understood [9], direct methods are robust to outliers and incorporate sub-pixel accuracy by the nature of its process. Our implementation of the approach typically executes at or near frame rate and maintains accurate parameter estimates. The implementation executes two iterations of the optimization routine every frame. This operating rate is in contrast to a disparity calculation based system which typically runs at half frame-rate on the same system.

While the topic of this paper is an algorithm to track the parameters of a planar surface, the algorithm is extendable to more complex surfaces; in fact, the algorithm could be termed an *algebraic surface tracker*. One imagines using it to track the parameters of a quadratic (similar in spirit to [16]) or cubic surface; even a parametric model, eg. NURBS, could be incorporated into the framework with little modification. It is the underlying principle of using quantities directly measurable from the images instead of features that allows such extensions to be proposed with little restraint.

Accurate estimation is highly dependent on the optics of the imaging system and the environment. These dependencies are fundamental to the set of approaches employed in stereo vision, however, not specifically our approach. Currently the parameters of our algorithm are chosen on an environment basis. For instance, the Δ in (13) is dependent on the texture frequency. However, the imaged texture frequency may vary from one plane to another and will vary with large depth changes. Such an operation mode is considered passive by nature. The exploration of algorithms that take more active approaches (i.e. projecting a suitable pattern depending on the current environment) or more intelligent approaches (i.e. altering operating parameters and thresholds autonomously by analyzing important scene qualities) are needed for further advancement of the field.

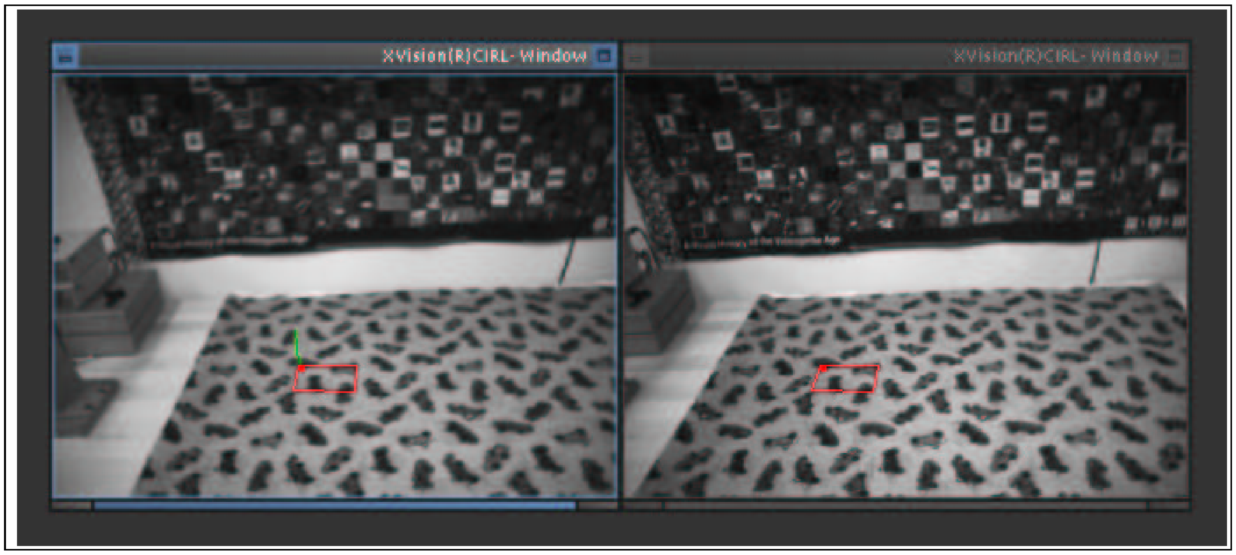

Figure 10. A quadrilateral drawn on the plane in the augmented desktop.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 0112882. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments 6*, pages 355–385, 1997.

[2] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun. Structure from motion without correspondence. Technical Report CMU-RI-TR-99-44, Carnegie Mellon University, 1999.

[3] F. Dellaert, C. Thorpe, and S. Thrun. Super-resolved texture tracking of planar surface patches. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robotic Systems*, 1998.

[4] O. Faugeras and Q. Luong. *The Geometry of Multiple Images*. The MIT Press, 2001.

[5] V. Ferrari, T. Tuytelaars, and L. V. Gool. Real-time affine region tracking and coplanar grouping. In *IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, 2001.

[6] G. Hager and P. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 20(10):1125–1139, 1998.

[7] G. D. Hager and K. Toyama. X vision: A portable substrate for real-time vision applications. *Computer Vision and Image Understanding: CVIU*, 69(1):023–037, 1998.

[8] B. Horn. *Robot Vision*. The MIT Press, 1986.

[9] M. Irani and P. Anandan. All about direct methods. Technical report, http://link.springer.de/link/services/series/0558/bibs/1883/18830267.htm, 2002.

[10] K. Kanatani. Detection of surface orientation and motion from texture by stereological technique. *Artificial Intelligence*, 24:213–237, 1984.

[11] K. Kanatani. Tracing planar surface motion from a projection without knowing the correspondence. *Computer Vision, Graphics, And Image Processing*, 29:1–12, 1985.

[12] D. Keren, S. Peleg, and R. Brada. Image sequence enhancement using sub-pixel displacements. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1988.

[13] S. Lang. Xvision2 - a framework for dynamic vision. Master's thesis, The Johns Hopkins University, 2001.

[14] B. Lucas and T. Kanade. An iterative image registratoin technique with an application to stereo vision. In *Proceedings DARPA Image Understanding Workshop*, 1981.

[15] S. Pei and L. Liou. Tracking a planar patch in three-dimensional space by affine transformation in monocular and binocular vision. *Pattern Recognition*, 26(1):23–31, 1993.

[16] A. Shashua and Y. Wexler. Q-warping: Direct computation of quadratic reference surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):920–925, 2001.

[17] G. Stein and A. Shashua. Direct estimation of motion and extended scene structure from a moving stereo rig. *IEEE Conference on Computer Vision and Pattern Recognition*, 1998.

[18] G. Strang. *Linear Algebra and Its Applications*. Saunders HBJ, 1988.

[19] R. Szeliski and S. Kang. Direct methods for visual scene reconstruction. In *IEEE Workshop on Representation of Visual Scenes*, 1995.

[20] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.