# Predicting Disparity Windows for Real-Time Stereo

Jane Mulligan and Kostas Daniilidis

GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA,
{janem|kostas}@grip.cis.upenn.edu,
WWW home page: http://www.cis.upenn.edu/~janem

**Abstract.** New applications in fields such as augmented or virtualized reality have created a demand for dense, accurate real-time stereo reconstruction. Our goal is to reconstruct a user and her office environment for networked tele-immersion, which requires accurate depth values in a relatively large workspace. In order to cope with the combinatorics of stereo correspondence we can exploit the temporal coherence of image sequences by using coarse optical flow estimates to bound disparity search ranges at the next iteration. We use a simple flood fill segmentation method to cluster similar disparity values into overlapping windows and predict their motion over time using a single optical flow calculation per window. We assume that a contiguous region of disparity represents a single smooth surface which allows us to restrict our search to a narrow disparity range. The values in the range may vary over time as objects move nearer or farther away in $Z$, but we can limit the number of disparities to a feasible search size per window. Further, the disparity search and optical flow calculation are independent for each window, and allow natural distribution over a multi-processor architecture.
We have examined the relative complexity of stereo correspondence on full images versus our proposed window system and found that, depending on the number of frames in time used to estimate optical flow, the window-based system requires about half the time of standard correlation stereo. Experimental comparison to full image correspondence search shows our window-based reconstructions compare favourably to those generated by the full algorithm, even after several frames of propagation via estimated optical flow. The result is a system twice as fast as conventional dense correspondence without significant degradation of extracted depth values.

## 1 Introduction

The difficulty of creating and rendering the geometry of virtual worlds by hand has led to considerable work on using images of the real world to construct realistic virtual environments [15,18,19,10]. As a result the ability to reconstruct or virtualize environments in real-time has become an important consideration in work on stereo reconstruction. Unfortunately, to accurately reconstruct reasonably large volumes, we must search large ranges of disparity for correspondences.

To achieve both large, accurate reconstructions and real-time performance we have to exploit every option to reduce the amount of calculation required, while maintaining the the best accuracy possible to allow interaction with live users.

One possible avenue to improve the temporal performance of reconstruction on an image sequence is to take advantage of temporal coherence. Since the same objects tend to be visible from frame to frame we can use knowledge from earlier frames when processing new ones. There is however a very real and complicated tradeoff between added calculation for exploiting temporal coherence, and its advantages in simplifying the stereo correspondence problem.

In this paper we propose a segmentation of the image based on an initial calculation of the full disparity map. We use a simple flood fill method to extract windows containing points in a narrow disparity range, and then use a local linear differential technique to calculate a single optical flow value for each window. The flow value allows us to predict the location of the disparity windows in a future frame, where we need only consider the updated disparity range for each window. Essentially we are making the assumption that a contiguous region with similar disparity will belong to a single surface and will thus exhibit similar motion (generated flow), to simplify our calculations and speed up our algorithm.

Several real-time stereo systems have become available in recent years including the Triclops vision system by Point Grey Research (*www.ptgrey.com*). Triclops uses three strongly calibrated cameras and rectification, Pentium/MMX processors and Matrox Meteor II/MC frame grabbers. Its reported performance is about 5 dense disparity frames per second (fps) for a $320 \times 240$ image and 32 disparities. The system's performance degrades however when additional accuracy options such as subpixel interpolation of disparities are required. The SRI Small Vision System achieves 12 fps on Pentium II, for similar image size and disparity range, apparently by careful use of system cache while performing correlation operations [9]. The CMU Video Rate Stereo Machine [8] uses special purpose hardware including a parallel array of convolvers and a network of 8 TI C40's to achieve rates of 15 fps on $200 \times 200$ pixel images for 30 disparities. In the image pipeline however, calculations are performed on 4-5 bit integers, and the authors offer no specific analysis of how this affects the accuracy and resolution of matching available. Neither Triclops nor the CMU machine yet exploit temporal coherence in the loop to improve the efficiency of their calculations or the accuracy of their results.

Sarnoff's Visual Front End (VFE) [12] is also a special purpose parallel hardware pipeline for image processing, specifically designed for autonomous vehicles. It can perform some low level functions such as image pyramid construction, image registration and correlation at frame rates or better. The VFE-200 is reported to perform optical flow and stereo calculations at 30 fps. The tasks described are horopter-based stereo [5] obstacle detection applications, which use knowledge of task and environment to reduce complexity. Registration to a ground plane horopter allows the correspondence search to be limited to a band around this plane.

The complimentary nature of optical flow and stereo calculations is well known. Stereo correspondence suffers from the combinatorics of searching across a range of disparities, and from occlusion and surface discontinuities. Structure from motion calculations are generally second order and sensitive to noise, as well as being unable to resolve a scale factor. Exploiting the temporal coherence of depth and flow measurements can take two forms: it can be used to improve the quality or accuracy of computed values of depth and 3D motion [14,23,2] or, as in our case, can be used as means of optimizing computations to achieve real-time performance. Obviously approaches which compute accurate 3D models, using iterative approaches such as linear programming are unlikely to be useful for real-time applications such as ours. Other proposed methods for autonomous robots, restrict or otherwise depend on relative motion [3,20] which cannot be controlled for a freely moving human subject. Tucakov and Lowe [22] proposed exploiting uncertain odometry knowledge of the camera's motion in a static environment to reduce the disparity range at each pixel. Knowledge of the full (though inaccurate) 3D motion is a stronger constraint than the coarse optical flow values for unknown rapid motion we use.

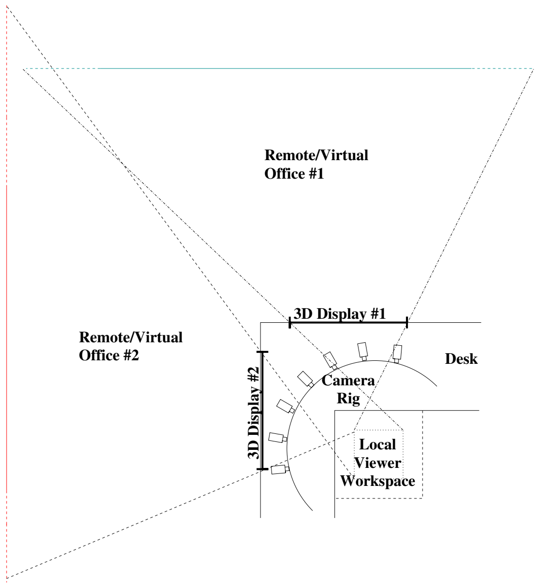


**Fig. 1.** Tele-cubicle Camera configuration

Our interest in real-time reconstruction stems from the goal of generating an immersive office collaboration environment. *Tele-immersion* is a form of networked virtual or augmented reality which is intended to push the bandwidth of Internet2. It entails stereo reconstruction of people and objects at remote sites, Internet transmission, and rendering of the resulting 3-D representations. When projected on special 3D displays these models will generate an immersive experience in the local office environment. The goal is to seem to be sitting across the desk from collaborators who may be half a world away.

For our stereo algorithm this proscribes a number of strong performance requirements. The reconstructions must be dense in order to allow for meshing and rendering, and they must offer fine resolution in depth to be able to distinguish features on the human face and hands. The method must operate in real-time on a sufficiently large image ($320 \times 240$) to provide this level of density and resolution. Resolution in depth naturally depends on resolution in disparities. Figure 2 illustrates a camera pair from the tele-cubicle in Figure 1. For a workspace depth $w = 1$ $m$, the disparity ranges from $d = -185$ pixels at point $A$, 35cm from the cameras to $d = 50$ pixels at $B$, 135cm from the cameras. Clearly a disparity range of $50-(-185) = 235$ is prohibitive for an exhaustive correspondence search. Finally the human subject is of course non-rigid and non-planar, and can move at high velocity inside the workspace.

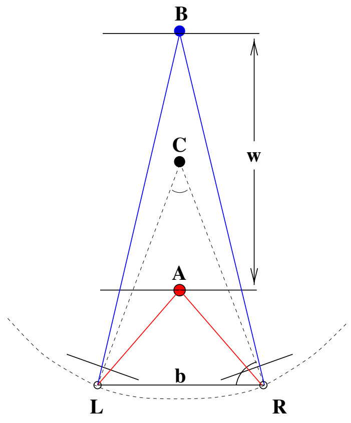


**Fig. 2.** Verged Stereo pair configuration for work volume $w$.

The current tele-cubicle configuration includes a pair of strongly calibrated monochrome cameras and a colour camera for texture mapping. A correlation stereo algorithm running on a Pentium II determines disparities, from which a depth map can be calculated for triangulation and rendering. Finally this triangulation is transmitted via TCP/IP to a 3D display server. The next generation of the tele-cubicle, illustrated in plan view in Figure 1, will have a semi-circular array of 7 colour cameras attached by threes to 5 quad processor Pentium III's. Despite this increase in processing power, we need to exploit any method which will improve our real-time response while not sacrificing the accuracy or work-volume of the tele-cubicle.

In the following sections we will describe the disparity window prediction techniques we propose to enhance the real-time capability of our stereo reconstructor. We describe a number of experiments which look at using only the previous disparity and flow to predict disparities in the next frame, as well as using predicted window locations to search shorter disparity ranges in the next image in a sequence. We examine the accuracy of the methods relative to our existing full image algorithm as well as the quality of window based reconstructions over several frames.

## 2 Predicting Disparity Windows

Our method for integrating disparity segmentation and optical flow can be summarized in the following steps:

**Step 1:** Bootstrap by calculating a full disparity map for the first stereo pair of the sequence.
**Step 2:** Use flood-fill to segment the disparity map into rectangular windows containing a narrow range of disparities.
**Step 3:** Calculate optical flow per window for left and right smoothed, rectified image sequences of intervening frames.
**Step 4:** Adjust disparity window positions, and disparity ranges according to estimated flow.
**Step 5:** Search windows for correspondence using assigned disparity range, selecting 'best' correlation value over all windows and disparities associated with each pixel location.
**Step 6:** Goto Step 2.

In the following sections we will discuss in some detail our existing full frame correlation stereo algorithm, our flood fill segmentation technique, optical flow approximation, window update and regional correspondence techniques.

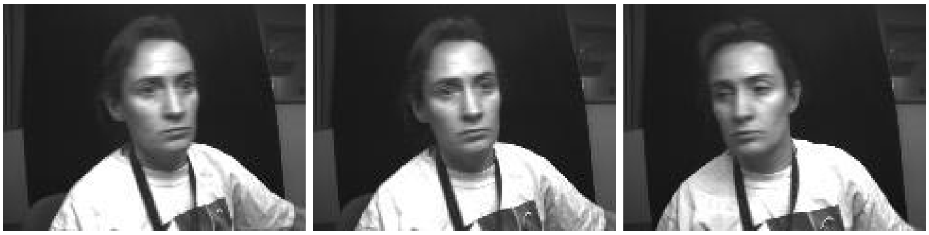


**Fig. 3.** Frames 10, 12 and 18 from the left image sequence. The subject translates and rotates from right to left in the image.

*Correlation Stereo.* In order to use stereo depth maps for tele-immersion or other interactive virtual worlds, they must be accurate and they must be updated quickly, as people or objects move about the environment. To date our work has focused on the accuracy of our reconstructions.

The stereo algorithm we use is a classic area-based correlation approach. These methods compute dense 3-D information, which allows extraction of higher order surface descriptions. In general our stereo system operates on a static set of cameras which are fixed and strongly calibrated. Originally the system was created to generate highly precise surface reconstructions.

Our full image implementation of the reconstruction algorithm begins by grabbing images from 2 strongly calibrated monochrome cameras. The system rectifies the images so that their epipolar lines lie along the horizontal image rows [1] to reduce the search space for correspondences and so that corresponding points lie on the same image lines. Computing depth values from stereo images of course requires finding correspondences, and our system measures the degree of correspondence by a modified normalized cross-correlation (MNCC),

$$c(I_L, I_R) = \frac{2\,\mathrm{cov}(I_L, I_R)}{\sigma^2(I_L) \,+\, \sigma^2(I_R)}. \tag{1}$$

where $I_L$ and $I_R$ are the left and right rectified images over the selected correlation windows. For each pixel $(u, v)$ in the left image, the matching produces a correlation profile $c(u, v, d)$ where disparity $d$ ranges over acceptable integer values.

All peaks in the correlation profile satisfying a relative neighbourhood threshold, are collected in a disparity volume structure. Peaks designated as matches are selected using visibility, ordering and disparity gradient constraints. A simple interpolation is applied to the values in the resulting integer disparity map to obtain subpixel disparities. From the image location, disparity and camera calibration parameters we can now compute a dense map of 3-D depth points based on our matches. We can also easily colour these depth points with the greyscale (or colour) values at the same locations in the left rectified image.

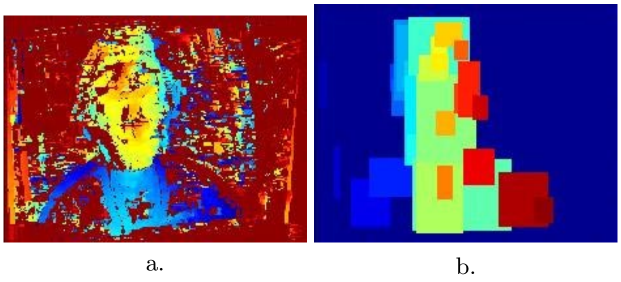


**Fig. 4.** Disparity map (a) and extracted windows of similar disparity (b) (frame 12).

*Flood-fill Segmentation.* It is more common to use flow fields to provide coarse segmentation than to use similar disparity [7,13], but our existing stereo system provides dense disparity maps, whereas most fast optical flow techniques provide relatively sparse flow values. Restricting the change in disparity per window essentially divides the underlying surfaces into patches where depth is nearly constant. The image of a curved surface for example will be broken into a number adjacent windows, as will a flat surface angled steeply away from the cameras. Essentially these windows are small quasi-frontal planar patches on the surface. Rather than apply a fixed bound to the range of disparities as we do, one could use a more natural constraint such as a disparity gradient limit [6]. This tends to create windows with large disparity ranges when smooth slanted surfaces are present in the scene however, which is what we are trying to avoid.

Any efficient region growing method could be applied to cluster the disparities into regions. Since our constraint is a threshold, and we allow regions to overlap we have chosen to use flood fill or seed fill [17, pp. 137-141], a simple polygon filling algorithm from computer graphics. We have implemented a scan-line version which pops a seed pixel location inside a polygon to be filled, then finds the right and left connected boundary pixels on the current scan line, ‘filling’ those pixels between. Pixels in the same x-range in the lines above and below are then examined. The rightmost pixel in any unfilled, non-boundary span on these lines in this range is pushed on the seed stack and the loop is repeated. When the stack is empty the polygon is filled.

We have modified this process slightly so that boundary is defined by whether the current pixel/disparity value falls within a threshold (+/ − 5) of the first seeded pixel. We start with a mask of valid disparity locations in the disparity image. For our purposes filling is marking locations in the mask which have been included in some disparity region, and updating the upper left and lower right pixel coordinates of the current window bounding box. When there are no more pixels adjacent to the current region which fall within the disparity range of the original seed, the next unfilled pixel from the mask is used to seed a new window. Once all of the pixel locations in the mask are set the segmentation is complete.

The disparity map for pair 12 of our test image sequence (Figure 3) is illustrated in Figure 4, along with the disparity windows extracted by the flood-fill segmentation. Twenty-nine regions were extracted, with mean disparity range width of 14 pixels. We maintain only rectangular image windows rather than a convex hull or more complicated structure, because it is generally faster to apply operations to a larger rectangular window than to manage a more complicated region structure. A window can cover pixels which are not connected to the current region being filled (for example a rectangular bounding box for an 'L'-shaped region will cover many pixels that are not explicitly in the disparity range) and therefore the windows extracted overlap. This is an advantage when change in disparity signals a depth discontinuity, because if a previously occluded region becomes visible from behind another surface, the region will be tested for both disparity ranges.

As a final step small regions (< MIN-REG pixels) are attributed to noise and deleted. Nearby or overlapping windows are merged when the corner locations bounding window $W_i$ expanded by a threshold NEAR-WIN, fall within window $W_j$, and the difference between the region mean disparities satisfies:

$$\left| \frac{\sum_{R_i} I_D(x_l, y_l)}{N_i} - \frac{\sum_{R_j} I_D(x_k, y_k)}{N_j} \right| < \text{NEAR-DISP},$$

where $R_i$ and $R_j$ are the set of pixels in two disparity regions, with $N_i$ and $N_j$ elements respectively. In section 3.2 we examine the sensitivity of window properties (number, size and disparity range) to variation in the NEAR-WIN and NEAR-DISP thresholds.

*Flow per Window* Optical flow calculations approximate the motion field of objects moving relative to the cameras, based on the familiar image brightness constancy equation: $I_x v_x + I_y v_y + I_t = 0$, where $I$ is the image brightness and $I_x$, $I_y$ and $I_t$ are the partial derivatives of $I$ with respect to $x$, $y$ and $t$, and $v = [v_x, v_y]$ is the image velocity. We use a standard local weighted least square algorithm [11,21] to calculate values for $v$ based on minimizing

$$e = \sum_{W_i} (I_x v_x + I_y v_y + I_t)^2$$

for the pixels in the current window $W_i$. We do not apply an affine flow assumption because of the increased complexity of solving for 6 parameters rather than just two components of image velocity [4].

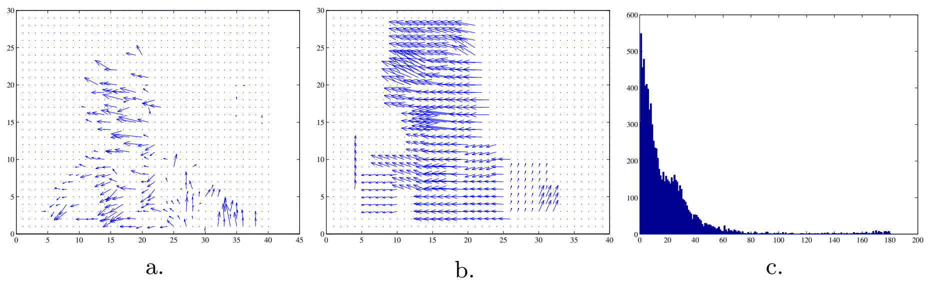


**Fig. 5.** Flow fields computed for (a) full image and (b) segmented windows (left frames 10-16), (c) histogram of absolute angular flow differences (degrees).

For each disparity window we assume the motion field is constant across the region $W_i$, and calculate a single value for the centre pixel. We use weights $w(x, y)$ to reduce the contribution of pixels farthest from this centre location.

$$w(x,y) = \frac{1}{2\pi \frac{n_x}{2} \frac{n_y}{2}} e^{-\left(x^2\left(\frac{n_y}{2}\right)^2 + y^2\left(\frac{n_x}{2}\right)^2\right)\frac{1}{2\left(\frac{n_x}{2}\right)^2\left(\frac{n_y}{2}\right)^2}}$$

where $N_i = n_{xi} \times n_{yi}$ are the current window dimensions.

We construct our linear system for $I_x v_x + I_y v_y = -I_t$ as follows:

$$A = \begin{bmatrix} w(x_1, y_1)I_x(x_1, y_1) & w(x_1, y_1)I_y(x_1, y_1) \\ \vdots & \vdots \\ w(x_{N_i}, y_{N_i})I_x(x_{N_i}, y_{N_i}) & w(x_{N_i}, y_{N_i})I_y(x_{N_i}, y_{N_i}) \end{bmatrix}$$

$$b = -\begin{bmatrix} w(x_1, y_1)I_t(x_1, y_1) \\ \vdots \\ w(x_{N_i}, y_{N_i})I_t(x_{N_i}, y_{N_i}) \end{bmatrix}$$

where locations $(x_1, y_1)...(x_{N_i}, y_{N_i})$ are the pixels contained in window $W_i$. We can then calculate the least squares solution $Av - b = 0$ using one of several forms of factorization [16].

Only one optical flow value is estimated per window. Figure 5 shows the comparison between flow estimates for $5 \times 5$ windows across the full image and values computed for our segmented windows (depicted by the same vector at each window location) for the left image sequence frames 10-16. The figure also includes a histogram of the absolute angle between the flow vectors at each point where a valid flow exists for both the full frame and window flows. The angle difference is clustered around zero degrees so that the dominant flow in the region is on average reasonably represented. A better estimate might be achieved by maintaining the centroid and standard deviation in $x$ and $y$ of the pixels included in a region by the segmentation. Computing optical flow on a window $W_{fi} = (C_{xi} - \sigma_{xi}, C_{yi} - \sigma_{yi}, C_{xi} + \sigma_{xi}, C_{yi} + \sigma_{yi})$ would focus the calculation on the region nominally extracted.

*Window Flow Adjustment.* For each window represented by its upper left and lower right corner locations $W(t) = [(x_{ul}, y_{ul}), (x_{lr}, y_{lr})]$, we must now adjust its location according to our estimated flow for the right and left images $v_l = [v_{xl}, v_{yl}]$ and $v_r = [v_{xr}, v_{yr}]$.

The window corner adjustment for $dt$, the time since the last disparity, basically takes $W(t)$'s upper left and lower right window coordinates and adds $v_l dt$ and $v_r dt$ to each in turn. The upper left corner is updated to the minimum $x$ and $y$ coordinates from $W_{ul}(t)$, $W_{ul}(t) + v_l dt$ and $W_{ul}(t) + v_r dt$. Similarly the lower right coordinate is updated to the maximum $x$ and $y$ coordinates from $W_{lr}(t)$, $W_{lr}(t) + v_l dt$ and $W_{lr}(t) + v_r dt$. This process is illustrated in Figure 6. Basically we force the window to expand rather than actually moving it, if the left coordinate of the window is predicted to move up or left by the right or left flow, then the window is enlarged to the left. If the right coordinate is predicted to move down or right the window is enlarged accordingly. Checks are also made to ensure the window falls within the image bounds (1, maxc), (1, maxr).

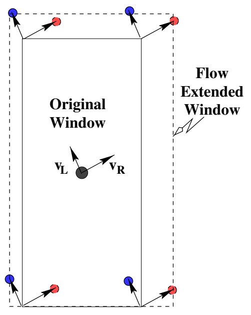


**Fig. 6.** Update expands window in flow directions.

Since the windows have moved as a consequence of objects moving in depth, we must also adjust the disparity range $D(t) = [d_{min}, d_{max}]$ for each window:

$$D(t+dt) = [\min(d_{min} + v_{xl}dt - v_{xr}dt, d_{min}), \max(d_{max} + v_{xl}dt - v_{xr}dt, d_{max})].$$

*Windowed Correspondence.* Window based correspondence proceeds much as described for the full image, except for the necessary manipulation of windows. Calculation of MNCC using Equation 1 allows overall calculation of the terms $\sigma^2(I_L)$, $\sigma^2(I_R)$, and $\mu(I_L)$ and $\mu(I_R)$ on a once per image pair basis. For $\text{cov}(I_L, I_R) = \mu(I_L I_R) - \mu(I_L)\mu(I_R)$ however, $\mu(I_L I_R)$ and the product $\mu(I_L)\mu(I_R)$ must be recalculated for each disparity tested. In the case of our disparity windows, each window can be of arbitrary size, but will have relatively few disparities to check. Because our images are rectified to align the epipolar lines with the scanlines, the windows will have the same $y$ coordinates in the right and left images. Given the disparity range we can extract the desired window from the right image given $x_r = x_l - d$. Correlation matching and assigning valid matches to the disparity volume proceeds as described for the full image method.

The general method of extracting and tracking disparity windows using optical flow does not depend on the specific correlation methods described above. Most real-time stereo algorithms do use some form of dense correlation matching [9], and these will benefit as long as the expense of propagating the windows via optical flow calculations is less than the resulting savings over the full image/full disparity match calculation.

## 3 Results

### 3.1 Complexity Issues

The first question we need to ask is whether the calculation of window based correlation and optical flow, is actually less expensive in practice than the full image correlation search over $(d_{min}, d_{max})$. For images of size $(n_x \times n_y)$ let us consider the operation of convolving with a mask $g$ of size $n_g$. We currently do the same per pair calculations $\sigma^2(I_L)$, $\sigma^2(I_R)$, $\mu(I_L)$ and $\mu(I_R)$ for the full and regional matching, so we will discount these in our comparison. The term $\mu(I_L)\mu(I_R)$ over $(d_{max} - d_{min})$ disparities requires $n_x n_y n_g (d_{max} - d_{min})$ multiplications for the full image case and $\sum_W n_{xi} n_{yi} n_g (d_{max_i} - d_{min_i})$ multiplications for the set of extracted windows $W$, where $W_i$ has dimensions $(n_{xi}, n_{yi})$. Similarly calculating $\mu(I_L I_R)$ will require $n_x n_y n_g (d_{max} - d_{min})$ versus $\sum_W n_{xi} n_{yi} n_g (d_{max_i} - d_{min_i})$ multiplications. We have to weigh this saving in the covariance calculation against the smoothing and least squares calculation per window of the optical flow prediction process.

For temporal estimates over $n_t$ images in a sequence we have to smooth and calculate derivatives for the images in the sequence in $x$, $y$ and $t$. We currently do this calculation over the entire rectified image which requires $(2 \times 3) n_g n_x n_y n_t$ multiplications for each of 2 (right and left) image sequences. To solve $Av = b$, using for example QR decomposition and back substitution requires approximately $(12)(n_{xi} n_{yi})$ flops per window.

Finally for the flood-fill segmentation each pixel may be visited up to 4 times (once when considering each of its neighbours), but probably much fewer. The only calculations performed are comparisons to update the window corners and disparity range as well as a running sum of the pixel values in each region. The cost is small compared to full image correlations, so we will disregard it here.

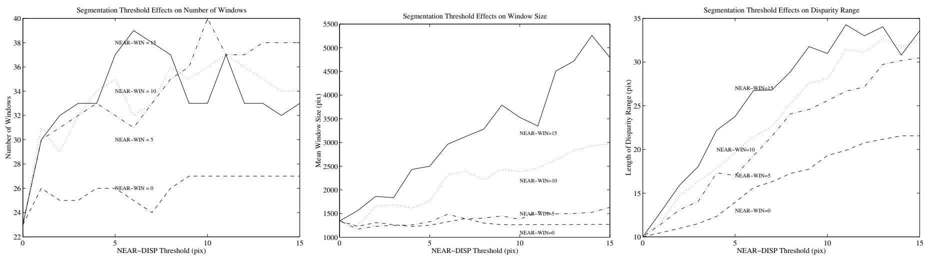
The window based correspondence will be faster if the following comparison is true:

$$\begin{gathered}(2) n_x n_y n_g (d_{max} - d_{min}) > \\ (2 \times 2 \times 3) n_g n_x n_y n_t + (2) \textstyle\sum_W \left[ n_{xi} n_{yi} n_g (d_{max_i} - d_{min_i}) \right] \\ + \textstyle\sum_W \left[ (12)(n_{xi} n_{yi}) \right] .\end{gathered} \tag{2}$$

For the examples demonstrated in this paper the the ratio of window based calculations to full image calculations is about 0.55, which is a significant saving. This saving is largely dependent on the number of frames in time $n_t$ which are used to estimate derivatives for the optical flow calculation.

### 3.2 Experiments

*Segmentation Thresholds.* Having established this complexity relationship we can see that the number and size of window regions and their disparity ranges is critical to the amount of speedup achieved. These properties are to some extent controlled by the thresholds in the flood fill segmentation which determine when nearby windows can be merged based on their disparity range and proximity.



a. Number of Windows b. Window Size c. Disparity Range

**Fig. 7.** Variation in window properties as a result of varying the NEAR-DISP and NEAR-WIN thresholds in the segmentation algorithm. The $x$ axis represents values (in pixels) of the NEAR-DISP threshold, the 4 curves represent values of NEAR-WIN (0,5,10, and 15 pixels)

For the purpose of demonstration we will consider the sequence of images in Figure 3. The full disparity calculation was performed for frame 12, and the resulting disparity map is illustrated in Figure 4, along with the windows extracted by the flood fill algorithm. The plots in Figure 7 illustrate the effect on the number of windows (a), their size in pixels (b) and the length of disparity range (c) associated with varying the thresholds for window merging. Initially the number of windows increases as small windows, which would otherwise be discarded, are merged such that together they are large enough to be retained. Eventually the number of windows levels off and slowly decreases, until in the limit one would have only one large window. The plots for threshold values versus window size in pixels and length of disparity range indicate that larger thresholds result in larger windows and larger disparity ranges, as we would expect.

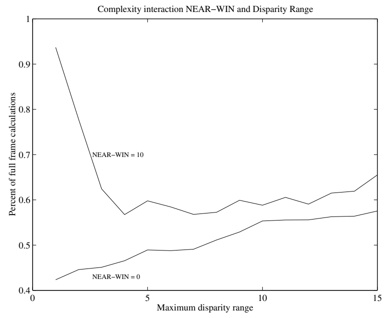


**Fig. 8.** Window algorithm calculation as a proportion of the full frame complexity for NEAR-WIN=0 and NEAR-WIN=10, for disparity bounds from 1 to 15.

Another interesting question is just how small we can make the disparity range, since decreasing the correspondence search space was our aim when we started this work. Generally there will be a tradeoff between the size of the disparity range and the number of windows we have to search in. The size and number of the windows will also be affected by the NEAR-WIN threshold as illustrated by the two curves in Figure 8. These plot region based complexity as a proportion of the full frame correspondence calculation, with respect to disparity bounds ranging from 1 to 15 pixels. When no neighbour merging is done, the smaller the disparity range the lower the complexity. For NEAR-WIN=10 merging many relatively distant small windows with similar very narrow (1-4 pixel) disparity range increases window overlap, and steeply increases complexity. However, this calculation is a little simplistic because it suggests that a

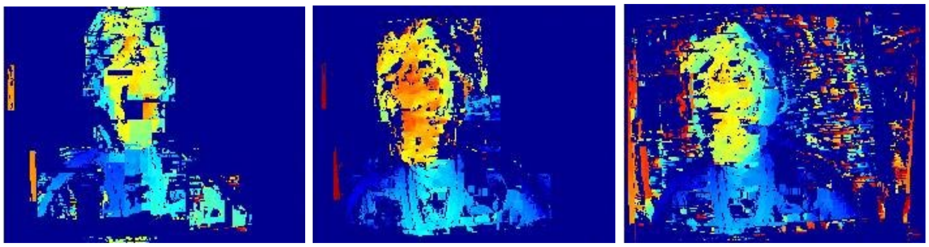


**Fig. 9.** Predicted, region-based and full correspondence disparity maps (frame 18).

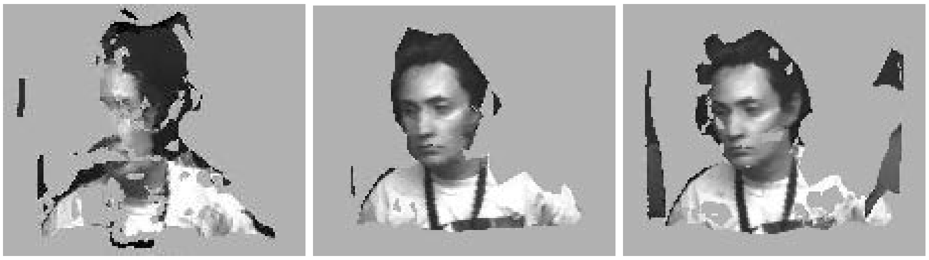


**Fig. 10.** Triangulations of depth points for predicted, region-based and full correspondence (frame 18).

window with disparity range of 1 can simply be propagated forward in time using our window flow value. As we will see below, pure prediction is a very poor way of determining disparities for the next frame, and search is always necessary to compensate for errors in our model.

*Accuracy of Disparity Estimates.* A critical issue when introducing a compromise such as our windowing technique is how accurate the results will be. We have tested two methods for estimating disparities for the current time step: a pure prediction approach simply using the flow and disparity value for each window location to calculate the new disparity according to:

$$d(t+1) = d(t) + \Delta t v_{lx} - \Delta t v_{rx}.$$

Second is the approach which actually calculates the new disparity via correlation on the predicted window location. The windows extracted for frame 12 are shown in Figure 4; the calculated disparity maps for the prediction, region and full image methods for frame 18 in the sequence are illustrated in Figure 9 (brighter pixels indicate higher disparity values). Eliminating small windows in the segmentation obviously acts as a filter, eliminating the background noise seen in the full correlation. We can also examine the histograms of error in disparity in Figure 11. These take the full image calculation as correct and compare the prediction and region-based correlation to it ($d_{full} - d_{est}$). Of course even the full image calculation may have correspondence errors, but in the absence of ground truth we will accept it as the best available estimate.

The pure prediction fares poorly with a mean error of 3.5 pixels, and a standard deviation of 17.9. The region-based correlation however has a mean of -0.4 pixels and a standard deviation of only 5.0 pixels. We can also examine the rendered versions of the data in Figure 10. These views are extracted from our 3D viewer and show the triangulated reconstructions rotated by 30° in $X$ and $-30°$ in $Y$. The pure prediction reconstruction is clearly wrong. It is assigning depth values to regions that are essentially empty in the new image, apparently because of an underestimate of the optical flow for some windows, or because the flow values do not accurately reflect rotation in depth.

The region-based method performs much better. It misses some points matched by the full correspondence, also probably because of an underestimate of flow values. However it fills in some holes visible in the full reconstruction, probably because the full image method finds a stronger (but erroneous) correlation, outside the disparity range of the windows which fall on this region. The triangulation method used to render the reconstructions deletes triangles with long legs, hence the holes left in the full frame triangulation.

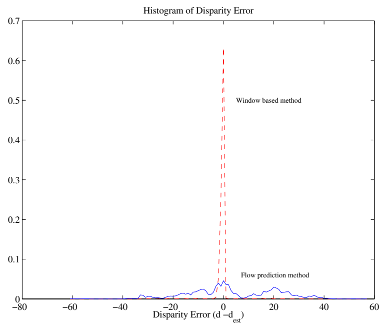


**Fig. 11.** Histogram of disparity error versus full correspondence.

**Table 1.** Sequence performance statistics for regional correspondence.

| frame | % full calc | % unmatched | mean err in $d$ | $\sigma$ err in $d$ |
|---|---|---|---|---|
| 5 | 53.3 | 0.16 | -0.49 | 3.50 |
| 9 | 67.7 | 0.18 | -0.14 | 4.36 |
| 13 | 54.0 | 0.24 | -0.58 | 3.62 |
| 17 | 54.1 | 0.28 | -0.57 | 4.38 |

*Extended Sequence.* Our final experiment is to observe the effect of processing only extracted windows over several frames as proposed by our algorithm. Can the system actually 'lock-on' to objects defined by the segmentation over time, or will it lose them? We ran the algorithm on frames 1-17 (561 ms) of the image sequence. The full disparity map was calculated for the first frame as a starting point, and windows were extracted. Optical flow per window was computed over 4 frame sequences (2-5, 6-9, 10-13, and 14-17), and the region-based disparities were calculated for frames 5, 9, 13 and 17. The images and their corresponding regional disparity maps are illustrated in Figures 12 and 13. Table 1 further breaks down the performance over time. The mean and standard deviation of $(d_{full} - d_{est})$ do not increase significantly, but the percentage of points for which the full calculation finds a match but the regional method does not steadily increases (although relatively small at 0.28%). This is what one would expect, since only motion increases the window size over time, and once a portion of an object is 'lost' there is no guarantee it will eventually fall in a window with

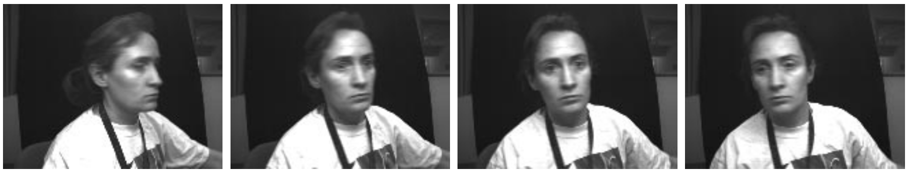


**Fig. 12.** Left image sequence frames 5, 9, 13 and 17.

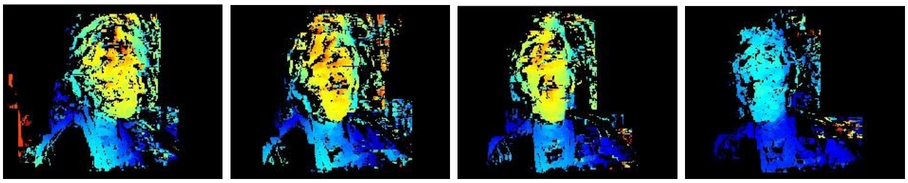


**Fig. 13.** Region-based correspondence disparity maps (frames 5, 9, 13, and 17).

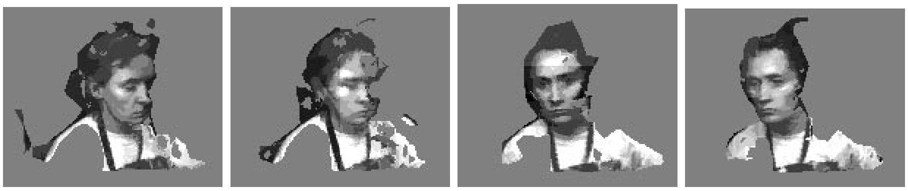


**Fig. 14.** Triangulations of depth points region-based correspondence (frames 5, 9, 13 and 17), views are rotated by 30° in $X$ and $-30°$ in $Y$.

appropriate disparity range again. The percentage of the full image calculation based on Equation 2, is about 55%.

## 4 Conclusions and Future Work

Providing dense accurate 3D reconstructions for virtual or augmented reality systems, in real-time is a challenge for conventional correlation stereo. One way to meet this challenge and to deal with the combinatorics of the long disparity ranges required, is to exploit temporal coherence in binocular image sequences. This requires a tradeoff between the benefit from motion calculations integrated into the stereo system, and the added cost of making these calculations.

In this paper we have proposed a simple method for decreasing the cost of dense stereo correspondence with the goal of reconstructing a person in an office environment in real-time for use in an augmented reality tele-immersion system. We start by segmenting an initial dense disparity map into overlapping rectangular windows using a flood fill algorithm which bounds regions by limiting the range of disparities they contain. We predict the new window locations and disparity ranges in an image sequence via a single optical flow estimate per window. Future reconstructions are based on correspondence in the predicted windows over the predicted disparities.

We have examined the relative complexity of stereo correspondence on full images versus our proposed window system and found that depending on the number of frames in time used to estimate optical flow the window-based system requires about half the time of standard correlation stereo. We have demonstrated experimentally that our window-based reconstructions compare favourably to those generated by the full algorithm even after several frames of propagation via estimated optical flow. The observed mean differences in computed disparities were less than 1 pixel and the maximum standard deviation was 4.4 pixels.

Obviously there is much more we can exploit in the stereo-flow relationship. We plan to examine probabilistic fusion over time, which will allow us to associate window regions with similar motions and treat them as a single object. This 3D segmentation can then guide our meshing and rendering as well as improve the predictions for position and disparity range for our algorithm. In order to catch any new objects entering the scene we have to track changes on the boundary, the simplest way to achieve this is to maintain image differences to detect any new motion, and expand existing windows or generate new ones with a large disparity range.

For our specific application in tele-immersion we plan to expand our methods to a polynocular stereo configuration as illustrated in Figure 1. The problem of combining and verifying correspondence from multiple camera pairs can be restricted in a similar way by projecting window volumes in $(x, y, d)$ from a master pair into the images of other pairs. These windows could be tracked in the auxiliary pairs in much the same way we have proposed here, providing additional constraint and evidence for our reconstructed models.

**Acknowledgment**

This work was supported by Advanced Network and Services (National Teleimmersion Initiative), and by grants ARO/MURI DAAH04-96-1-0007, DARPA-ITO-MARS DABT63-99-1-0017, and NSF CISE CDS-97-03220.

## References

1. Nicholas Ayache. *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception.* The MIT Press, Cambridge, MA, 1991.
2. Poornima Balasubramanyam. The p-field: A computational model for binocular motion processing. In *1991 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'91)*, pages 115–120, 1991.
3. P.A. Beardsley, J.M. Brady, and D.W. Murray. Prediction of stereo disparity using optical flow. In *British Machine Vision Conf.*, pages 259–264, 1990.
4. S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM Computing Surveys*, 27(3):433–467, 1995.
5. Peter Burt, Lambert Wixson, and Garbis Salgian. Electronically directed "focal" stereo. In *5th Intl. Conf. on Computer Vision (ICCV95)*, pages 94–101, Cambridge, Mass., June 1995.
6. U. Dhond and J. Aggarwal. Structure from stereo – a review. *IEEE Trans. on Systems, Man and Cybernetics (SMC)*, 19(6):1489–1510, Nov/Dec 1989.

7. Michal Irani, Benny Rousso, and Shmuel Peleg. Computing occluding transparent motions. *Intl. Journal of Computer Vision*, 12(1):5–16, January 1994.
8. Takeo Kanade, Atsuchi Yoshida, Kazuo Oda, Hiroshi Kano, and Masaya Tanaka. A stereo machine for video-rate dense depth mapping and its new applications. In *1996 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'96)*, pages 196–202, San Francisco, CA, June 1996.
9. Kurt Konolige. Small vision systems: Hardware and implmentation. In *Eighth Intl. Symposium on Robotics Research (Robotics Research 8)*, pages 203–212, Hayama, Japan, Oct. 1997.
10. Stephane Laveau and Olivier Faugeras. 3-d scene representation as a collection of images and fundamental matrices. Technical Report 2205, INRIA, February 1994.
11. B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *7th Intl. Joint Conf. on Artificial Intelligence (IJCAI'81)*, pages 674–679, Vancouver, BC, 1981.
12. R. Mandelbaum, M. Hansen, P. Burt, and S. Baten. Vision for autonomous mobility: Image processing on the vfe-200. In *IEEE Intl. Symposium on Intelligent Control (ISIC), Intl. Symposium on Computational Intelligence in Robotics and Automation (CIRA), and Intelligent Systems and Semiotics (ISAS)*, Gaithersburg, MD, September 1998.
13. Francois Meyer and Patrick Bouthemy. Region-based tracking in an image sequence. In G. Sandini, editor, *Computer Vision– ECCV'92: Second European Conf. on Computer Vision*, volume 588 of *Lecture Notes in Computer Science*, pages 476–484, Santa Margherita Ligure, Italy, May 1992. Springer-Verlag.
14. Hans-Hellmut Nagel. Dynamic stereo vision in a robot feedback loop based on the evaluation of multiple interconnected displacement vector fields. In *Robotics Research: The Third Intl. Symposium*, pages 49–55, Gouvieux, France, Oct. 1985.
15. P. J. Narayanan and Takeo Kanade. Virtual worlds using computer vision. In *IEEE/ATR Workshop on Comp. Vis. for VR based Human Comm. (CVVRHC'98)*, pages 2–13, Bombay, India, January 1998.
16. William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, Cambridge, UK, second edition, 1992.
17. David F. Rogers. *Procedural Elements for Computer Graphics.* WCB/McGraw-Hill, Boston, MA, second edition, 1998.
18. Steven M. Seitz and Charles R. Dyer. Toward image-based scene representation using view morphing. In *13th IEEE Conf. on Computer Vision and Pattern Recognition*, Vienna, Austria, 1996.
19. Amnon Shashua. Algebraic functions for recognition. *EEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 17(8):779–789, 1995.
20. M. Tistarelli, E. Grosso, and G. Sandini. Dynamic stereo in visual navigation. In *1991 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'91)*, pages 186–193, 1991.
21. Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision.* Prentice Hall Inc., Upper Saddle River, NJ, 1998.
22. Vladimir Tucakov and David. G. Lowe. Temporally coherent stereo: Improving performance through knowledge of motion. In *1997 IEEE Intl. Conf. on Robotics and Automation (ICRA'97)*, pages 1999–2006, Albuquerque, NM, April 1997.
23. Allen M. Waxman and James H. Duncan. Binocular image flows: Steps toward stereo-motion fusion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-8(6):715–729, November 1986.