International Journal of Computer Vision. vol.47, no.1/2/3, pp.99-117, May 2002.

Fast Stereo Matching Using Rectangular Subregioning and 3D Maximum-Surface Techniques

Changming Sun

CSIRO Mathematical and Information Sciences
Locked Bag 17, North Ryde, NSW 1670, Australia
changming.sun@csiro.au

Abstract

This paper presents a fast and reliable stereo matching algorithm which produces a dense disparity map by using fast cross correlation, rectangular subregioning (RSR) and 3D maximum-surface techniques in a coarse-to-fine scheme. Fast correlation is achieved by using the box-filtering technique whose speed is invariant to the size of the correlation window and by segmenting the stereo images into rectangular subimages at different levels of the pyramid. By working with rectangular subimages, not only can the speed of the correlation be further increased, the intermediate memory storage requirement can also be reduced. The disparity map for the stereo images is found in the 3D correlation coefficient volume by obtaining the global 3D maximum-surface rather than simply choosing the position that gives the local maximum correlation coefficient value for each pixel. The 3D maximum-surface is obtained using our new two-stage dynamic programming (TSDP) technique. There are two original contributions in this paper: (1) development of the RSR technique for fast similarity measure; and (2) development of the TSDP technique for efficiently obtaining 3D maximum-surface in a 3D volume. Typical running time of our algorithm implemented in the C language on a 512×512 image is in the order of a few seconds on a 500MHz PC. A variety of synthetic and real images have been tested, and good results have been obtained.

Keywords: *Rectangular subregioning (RSR), Fast cross-correlation, Similarity measure, Stereo matching, Coarse-to-fine, Pyramid, 3D Maximum-Surface, Two-stage dynamic programming (TSDP), Sub-pixel accuracy.*

1 Introduction

The correspondence problem in stereo vision and photogrammetry concerns the matching of points or other kinds of primitives such as edges and regions in two or more images (in this paper, we just use two images) such that the matched image points are the projections of the same point in the scene. The disparity map obtained from the matching stage may then be used to compute the 3D positions of the scene points given the imaging geometry.

Because of factors such as noise, lighting variation, occlusion and perspective distortion, the appearances of the corresponding points will differ in the two images. For a particular feature or a local window in one image, there are usually several matching candidates in the other image. It is usually necessary to use additional information or constraints to assist in obtaining the correct match. Some of the commonly used constraints are: (1) Epipolar constraint: Under this constraint, the matching points must lie on the corresponding epipolar lines of the two images. For epipolar rectified images, the matching points lie on the same image scanlines of a stereo pair; (2) Uniqueness constraint: Matching should be unique between the two images; (3) Smoothness constraint: Local regions of the disparity map should be relatively smooth apart from regions with occlusion or disparity discontinuity; and (4) Ordering constraint or monotonicity constraint: For points along the epipolar line in one image of the image pair, the corresponding points have to occur in the same order on the corresponding epipolar line in the other image. In this paper, we assume that we work on the epipolar rectified stereo images so we essentially used the epipolar constraint. Other constraints mentioned will be used in the dynamic programming stage when obtaining the disparity map from the correlation coefficient volume.

Matching techniques can be divided broadly into

area-based and feature-based image matching, or a combination of them. Area-based methods have been applied successfully to aerial images where the surfaces vary smoothly (O'Neill and Denos, 1996) and to other applications where stereo images have good textures. They have the advantage of directly generating dense disparity maps but they tend to breakdown where there is lack of texture or where depth discontinuities occur (Cochran and Medioni, 1992). The feature-based approaches match more abstract features, rather than matching texture regions in the two images (Medioni and Nevatia, 1985; Ayache and Faverjon, 1985; Grimson, 1985). Feature-based methods provide more precise positioning for the matching results. They are also more reliable than area-based matching. Because of the sparse and irregularly distributed nature of the features, the matching results must be augmented by an interpolation step if a dense map of the scene is desired. If a feature-based method is used, an extra stage is needed for feature detection in the two images, which will no doubtly increase the computational cost. Other types of stereo matching methods such as pixel-based (Birchfield and Tomasi, 1999), diffusion-based (Scharstein and Szeliski, 1998), wavelet-based (Kim et al., 1997), phase-based (Porr et al., 1998), and filter-based (Jones and Malik, 1992) have also been developed.

Lotti and Giraudon used a correlation based algorithm with an adaptive window-size that is constrained by an edge map extracted from the image (Lotti and Giraudon, 1994a; Lotti and Giraudon, 1994b). They presented results on aerial images. Intille and Bobick (Intille and Bobick, 1994; Bobick and Intille, 1999) presented a stereo algorithm that incorporates the detection of the occlusion regions directly into the matching process. They developed a dynamic programming solution that obeys the occlusion and ordering constraints to find a best path through the disparity-space image. They also used ground control points to eliminate sensitivity to occlusion cost. Xiong *et al* (Xiong et al., 1996) presented a stereo matching approach which integrates area-based and feature-based processes. Wei *et al* proposed an intensity- and gradient-based stereo matching using hierarchical Gaussian basis functions (Wei et al., 1998). Fua (Fua, 1993) described a correlation based multi-resolution algorithm which is followed by interpolation. Anandan (Anandan, 1987) described a hierarchical computational framework for the determination of dense motion fields from a pair of images. A number of researchers have used dynamic programming to solve the matching problem globally along a pair of epipolar scan lines (Lloyd, 1985; Gimel'farb et al., 1992; Baldwin et al., 1990; Rojas et al., 1997; Bensrhair et al., 1996; Geiger et al., 1995). There are other algorithms which perform fast stereo matching (Fusiello et al., 1997; Kolesnik, 1993; Banks et al., 1999). Sun (Sun, 1997) described a fast stereo matching method using fast cross correlation and dynamic programming techniques in a coarse-to-fine scheme. The dynamic programming was applied to the correlation coefficients matrix along the corresponding epipolar lines. All the methods mentioned above did not consider the continuity of neighbouring epipolar lines. Ohta and Kanade used dynamic programming to match epipolar scanlines first and then improve the solutions iteratively using edges (Ohta and Kanade, 1985). Cox *et al* presented a stereo matching algorithm using the dynamic programming technique considering the inter-scanline constraints (Cox et al., 1996). The method needs small number of iteration and only approximates the global solutions. Belhumeur presented a Bayesian approach to stereo matching (Belhumeur, 1996). In his 2D implementation, he first carried out dynamic programming along epipolar lines to obtain initial etsimates of the disparity. Then he performed an iterative step for vertical smoothing.

Roy (Roy, 1999) and Roy & Cox (Roy and Cox, 1998) developed an algorithm for solving the N-camera stereo correspondence problem by transforming it into a maximum-flow problem. The minimum-cut associated to the maximum-flow yielded a disparity surface for the whole image at once. The *preflow-push lift-to-front* algorithm was used when they calculate the maximum-flow. The average computational complexity for Roy and Cox's algorithm was $O((MN)^{1.2}D^{1.3})$ (with image row and column numbers M, N and depth resolution D) (Roy and Cox, 1998). Ishikawa and Geiger also implemented the maximum-flow approach for stereo matching that models occlusions, discontinuities and epiplar line interaction (Ishikawa and Geiger, 1998). Chen and Medioni (Chen and Medioni, 1998) presented a propagation type of algorithm similar to Ohta and Kanade's method (Ohta and Kanade, 1985). The techniques they used included non-maxima suppression, seed voxel selection and surface tracing. There was no mentioning about the speed issues in (Chen and Medioni, 1998). Yang and Yuille proposed a non-linear filter for detecting disparity surface in a 3D volume (Yang and Yuille, 1995). They first apply the filter to the 3D volume and then simply use maximum-picking. Zitnick and Kanade presented a volumetric iterative algorithm for stereo matching (Zitnick and Kanade, 1998; Zitnick and Kanade, 2000). The algorithm updates the match likelihood values by diffusing support among neighbouring values and in-

hibiting others.

In this paper we will present two new techniques for fast stereo matching. The first technique is for segmenting stereo images into rectangular subimages through a rectangular subregioning (RSR) process for fast similarity calculation; and the second is for obtaining a 3D maximum-surface in a 3D correlation coefficient volume for selecting the disparity. The disparity is obtained from a 3D correlation coefficient volume by using a two-stage dynamic programming (TSDP) technique considering the continuity of the neighbouring epipolar scan lines. The 3D maximum-surface obtained using the TSDP technique can give global maximum summation of the correlation coefficients along the 3D surface. Because the TSDP algorithm uses a two-stage dynamic programming technique, the algorithm's complexity is linear with respect to the size of the 3D volume MND. Furthermore, because the algorithm works in the coarse-to-fine scheme, the complexity of the disparity selection stage by using TSDP is only $O(MND')$, where D' is much smaller than D. We will also address some of the other efficient and reliable techniques for fast stereo matching. The rest of the paper is organised as follows: Section 2 proposes our new RSR method for fast calculation of similarity measure. Section 3 presents our new method for fast stereo matching by finding the maximum-surface in the 3D correlation volume using the TSDP technique. The detailed matching method is described in Section 4. Section 5 shows the experimental results obtained using our fast stereo matching method applied to a variety of images. Section 6 discusses the reliability and computation speed issues of our algorithm. Section 7 gives concluding remarks.

2 Rectangular Subregioning for Fast Similarity Measures

Similarity or dissimilarity is the guiding principle for solving the stereo matching or correspondence problem. Corresponding features or areas should be similar in the two images. Different similarity measures have been used in the literature for matching, and their performance and computation costs vary (Rechsteiner et al., 1994; Aschwanden and Guggenbühl, 1992). The most commonly used similarity measure is the cross correlation coefficient. It is popular because it corresponds to optimal signal-to-noise ratio estimation (Rosenfeld and Kak, 1982). The sum of absolute differences (SAD) and the sum of square differences (SSD), both dissimilarity measures, have also been used. Their usage is usually justified on the ground that they are easy to implement and use less computing power, especially when they are used in the fast sequential similarity detection algorithm (Wu, 1995; Barnea and Silverman, 1972) and when programming in the MMX instruction sets. Barnea and Silverman (Barnea and Silverman, 1972) introduced a class of sequential algorithms for fast image registration. They were designed to reduce computation cost in matching procedures using minimum dissimilarity measures like the sum of the absolute differences (SAD). Konecny and Pape (Konecny and Pape, 1981) reviewed image correlation techniques according to photogrammetric and mathematical fundamentals. It has also been shown that the zero mean normalized cross correlation (ZNCC) and the zero mean sum of squared differences tend to give better results (Wu et al., 1995; Faugeras et al., 1993; Rechsteiner et al., 1994; Aschwanden and Guggenbühl, 1992). The ZNCC estimate is independent of differences in brightness and contrast due to the normalization with respect to mean and standard deviation. We will use the zero mean normalized cross correlation coefficient as the measure of similarity between the candidate matching areas in this paper. But direct calculation of ZNCC is computationally expensive. Faugeras *et al* (Faugeras et al., 1993) developed a recursive technique to calculate the correlation coefficients which are invariant to the correlation window size. Sun (Sun, 1997; Sun, 1998) used box-filtering technique for fast cross correlation. The following subsections describe methods for achieving fast correlation on the whole image and our new technique of using rectangular subregioning for fast similarity measure calculation.

2.1 Fast Cross-Correlation on the Whole Images: Review

Let $f_{m,n}$ be the intensity value of an $M \times N$ image f at position (m, n), where f is to be locally averaged into $\bar{f}$, i.e. obtaining the local mean of the original image within a box. We also have a similar definition for a second image g. The zero mean normalized cross correlation of two local windows can be written as follows:

$$C(i,j,d) = \frac{cov_{ij,d}(f,g)}{\sqrt{var_{ij}(f)} \times \sqrt{var_{ij,d}(g)}} \qquad (1)$$

where

$$cov_{ij,d}(f,g) = \sum_{m=i-K}^{i+K} \sum_{n=j-L}^{j+L} (f_{m,n} - \bar{f}_{i,j})(g_{m+d,n} - \bar{g}_{i+d,j}) \qquad (2)$$

$$var_{ij}(f) = \sum_{m=i-K}^{i+K} \sum_{n=j-L}^{j+L} (f_{m,n} - \bar{f}_{i,j})^2 \quad (3)$$

$$var_{ij,d}(g) = \sum_{m=i-K}^{i+K} \sum_{n=j-L}^{j+L} (g_{m+d,n} - \bar{g}_{i+d,j})^2 \quad (4)$$

and i, j are the image row and column indices. d is the shift of the window in the right image along epipolar lines, and it indicates possible disparity values; K and L define the correlation window size. $\bar{f}$ and $\bar{g}$ are the mean values within the local windows. From Eqs. (2)-(4) it can be seen that to achieve fast calculation of Eq. (1), one needs to have fast ways to obtain the mean and variance of a window and cross covariance values of two local windows in two input images. Fast calculation of local mean and variances can be achieved using the box-filtering technique.

Rewriting Eq. (2), we have:

$$cov_{ij,d}(f, g) = \sum_{m=i-K}^{i+K} \sum_{n=j-L}^{j+L} f_{m,n} g_{m+d,n} - W \bar{f}_{i,j} \bar{g}_{i+d,j} \quad (5)$$

W is the size of the correlation window which equals $(2K + 1)(2L + 1)$. Eq. (5) is the numerator of Eq. (1). Direct calculation of Eq. (5) for every points on the images has nearly $(2K + 1)(2L + 1)$ redundancies. Cross-covariance of two images can be obtained using only a few multiplications by exploiting techniques similar to the fast calculation for the mean and variance by using box-filtering. The first term of the r.h.s. of Eq. (5) is the sum of the pixel multiplications over the correlation window with the right image shifted d pixels in the x- direction. This operation can also be performed using the same box-filtering idea to achieve fast computation speed. The cross covariance calculation uses the multiplication term $f_{m,n}g_{m+d,n}$ instead of $f^2_{m,n}$ when calculating the variance. The second term of the r.h.s. of Eq. (5) is a straight-forward calculation using the available local mean values.

The correlation of two windows in the two images is performed along the same horizontal scan line. For any point in the left image, if the search window is assumed to be within $[-w, +w]$ in the right image, then the value of d in Eq. (5) varies from $-w$ to $+w$. The traditional way of obtaining the cross correlation is to fix a point in the left image and vary d within $[-w, +w]$ in the right image to calculate the correlation coefficients. In an algorithm for fast correlation calculation, one first fixes on one particular d for all the points in the left image and calculates the local cross correlation between the whole left image and the whole shifted right image of the amount d using box-filtering technique. After this, for every point on the left image we have a local cross correlation value for the shift of d. Then we increase the number of d by 1, and repeat the process of correlation calculation until the value of d has gone through $[-w, +w]$. For each d, a plane of correlation coefficients is produced. Putting all of these planes together we have a 3D correlation volume. The size of the volume depends upon the image row and column numbers M, N and the maximum disparity search range $D(=2w + 1)$ as shown in Figure 1. The complexity of the algorithm is $O(MND)$. The storage space needed for the correlation coefficients is in the order of $4MND$ bytes if float data type is used.

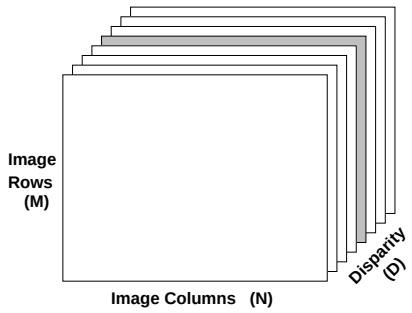


Figure 1: *An illustration of the 3D correlation coefficient volume obtained after using the fast correlation method. The grey plane in the middle of the volume corresponds to the coefficients when d = 0. The size of the volume is MND.*

The fast algorithm described earlier can be easily adopted to obtain the SAD and SSD measures efficiently. By using the following equation rather than using Eq. (1), the efficient SAD measure can be obtained:

$$SAD_{ij,d}(f, g) = \sum_{m=i-K}^{i+K} \sum_{n=j-L}^{j+L} |f_{m,n} - g_{m+d,n}| \quad (6)$$

Similarly, the fast SSD measure can be calculated using the following equation:

$$SSD_{ij,d}(f, g) = \sum_{m=i-K}^{i+K} \sum_{n=j-L}^{j+L} (f_{m,n} - g_{m+d,n})^2 \quad (7)$$

2.2 Rectangular Subregioning (RSR)

Rather than work with the whole image during the fast image correlation stage as described in the previous subsection, we could work with subimages to speed up the correlation calculation further and reduce the memory space for storing the correlation coefficients.

As mentioned earlier, the computation complexity for the fast image correlation step is $O(MND)$ if we work with the whole image.

If the input image is divided into R subimages or rectangular subregions, the computation complexity will be $\sum_{i=0}^{R-1}(M_i N_i D_i)$, where M_i, N_i are the row and column numbers for the ith subimage or region, and D_i is the disparity search range over this subimage. We call the process of segmenting the input images into rectangular subimages as rectangular subregioning (RSR). Because the disparity search range D_i is obtained in a much smaller region ($M_i N_i$), D_i is expected to be smaller than D, which is obtained for the whole image. Even when it is not much smaller, the size of this region ($M_i N_i$) is much smaller than the input image. It is anticipated that $\sum_{i=0}^{R-1}(M_i N_i D_i)$ will be smaller than MND, especially when the disparity changes a lot within the whole image.

Although there are some overheads when working with subimages, such as region segmentation and house-keeping, the time saved during the correlation stage is far greater than the time spent for region segmentation and house-keeping. Another advantage for working with subimages is its smaller memory usage. As mentioned earlier, some memory space is needed to store the correlation coefficients. In the case of working with one whole image, the memory space needed is in the order of $4MND$ bytes. While in the case of working with subimages, the memory space needed is in the order of $\max_i(4M_i N D_i^s)$, because the memory for each subregion is dynamically allocated and freed, where D_i^s is the disparity search range for a particular horizontal stripe starting from the left to the right of the image.

2.3 Rectangular Subregioning Process

Now we will describe our fast method in more detail for segmenting an image into rectangular subregions for fast similarity calculation. Because the shapes of the search window and the input images are all rectangular, the regions of subimages need to be rectangular for efficient operation.

The method that we developed for segmenting an image into rectangles are in the line of region split-merge techniques. The input for this segmentation step is the intermediate disparity map obtained by projecting and interpolating the result from the previous pyramid levels. If the current level is at the top of the pyramid, the current disparity map can be set to zero if there is no prior disparity information. If there is an initial disparity map, it can be used as the input for this segmentation step. The coordinate of the disparity map is the same as the left image if the left image is taken as the reference; otherwise, the coordinate of the disparity map is the same as that of the right image.

The input image is first divided into thin horizontal stripes. Each stripe contains the properties such as stripe corner positions, the minimum and maximum disparity values. Then these thin horizontal stripes are merged according the criteria that the overall computing complexity is minimum taking the computational overhead into account. At each iteration of the merging process, only one pair of neighbouring stripes are merged. The properties of the merged stripe such as the corner positions and the minimum and maximum disparity values need to be updated. After each iteration, the number of horizontal stripes decreases by one. The iteration stops when there is no neighbouring stripes look similar. Different merged horizontal stripes usually have different width and disparity ranges.

After the image has been segmented into horizontal stripes, each such stripe can then be cut into smaller rectangular regions by vertical lines. The steps are similar to those for segmenting images into horizontal stripes. The objective is to obtain large regions with small disparity range and small regions with large disparity range so that the overall cost $\sum_{i=0}^{R-1}(M_i N_i D_i)$ is smaller. Figure 2 illustrates the rectangular subregioning precess. Figure 2(a) gives the initial horizontal stripes of the input image. Figure 2(b) shows the merged horizontal stripes obtained from Figure 2(a). Figure 2(c) is the initial vertical cuttings for each obtained horizontal stripe. Figure 2(d) shows the result of merging the small rectangular regions within each of the horizontal stripe. The resultant segmentation of an input image can be in the form as shown in Figure 3. For example, the segmented image contains horizontal stripe $AA'B'B$, and this stripe is then cut into smaller rectangles. One of these smaller rectangles is $PQST$. Fast correlation is performed on each of these smaller rectangle images, and the obtained correlation coefficients are put together to form 3D cubes. Figure 4(a) shows a disparity map which is used for rectangular subregioning at one level of the pyramid. Figure 4(b) shows the rectangular regions obtained. Most of the regions have small disparity ranges.

2.4 Corresponding Regions in Right Image

In the case when the left image is taken as the reference image, the subregions obtained on the disparity map also correspond to the subregions in the left image. Otherwise, the subregions obtained on the disparity map will correspond to the subregions in the

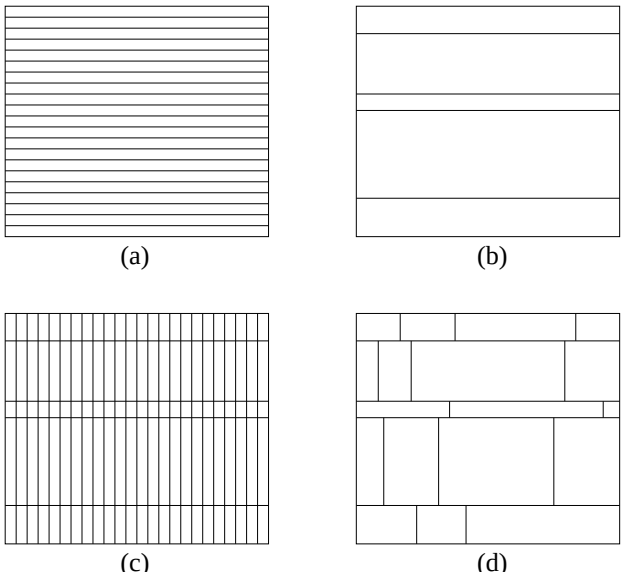


Figure 2: *Rectangular subregioning through merging thin rectangles. (a) shows the initial horizontal stripes for the input disparity map. (b) illustrates some horizontal regions after the horizontal stripe merging process. (c) shows the initial vertical stripes for each of the horizontal regions shown in (b); and (d) is the final rectangular subregions obtained.*

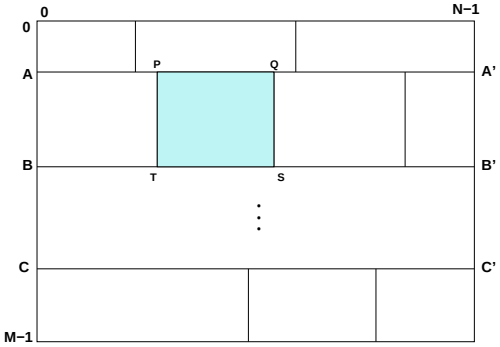


Figure 3: *Sub-dividing the whole image into subimages. Fast correlation is performed on each pair of the corresponding rectangular subregions.*

right image. After obtaining the rectangular subimages for the reference image, one then needs to obtain the corresponding regions in the other image so that fast similarity measures can be made for the corresponding subregions. When calculating the corresponding positions of a subregion in the right image after knowing the position of a rectangular region in the left image, the disparity information of this region in the disparity map will be used. If the disparity search range for a subregion (between $y1, y2$ and $x1L, x2L$) in the left

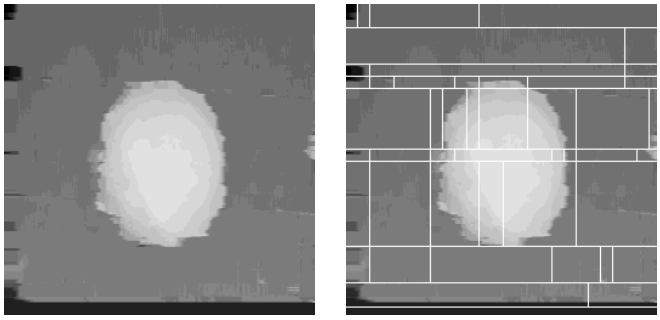


Figure 4: *An example result of sub-dividing the whole image into subimages based on intermediate disparity map in the image pyramid. (a) a disparity map at a particular image pyramid; (b) the disparity map shown in (a) overlaid with the rectangles obtained. Each of these rectangles will be used for running the fast correlation algorithm described earlier.*

image is within d_{min} and d_{max}, the x position of the left side of the corresponding region in the right image $x1R$ should be a position between $x1L + d_{min}$ and $x1L + d_{max}$ as shown in Figure 5. Similarly, the x position of the right side of the corresponding region in the right image $x2R$ should be a position between $x2L + d_{min}$ and $x2L + d_{max}$. The approach we used here for rectangular subregioning may not be the global optimal, but it is fast and simple and serves our purpose for fast processing. When actually performing fast correlation calculation for each pair of the subregions, certain size of region overlapping as shown in Figure 5 by the shaded region needs to be considered in order to eliminate the boundary effect. It is also necessary to allow some overlapping between successive horizontal stripes. The amount of overlapping depends on the size of the correlation window used.

After the fast correlation for each of the rectangular regions, we have a smaller 3D volume for each pair of the corresponding subregions. The 3D volume for the whole image can be constructed from each of the smaller 3D volumes together with the disparity range information. Figure 6 shows the 3D volume size and shape obtained by using the RSR process. One can see the different shapes of Figure 6 compared with that of Figure 1 which is obtained using fast correlation on the whole images without using the RSR.

2.5 Algorithm Steps for Rectangular Subregioning

The steps for performing the rectangular subregioning can be summarised as:

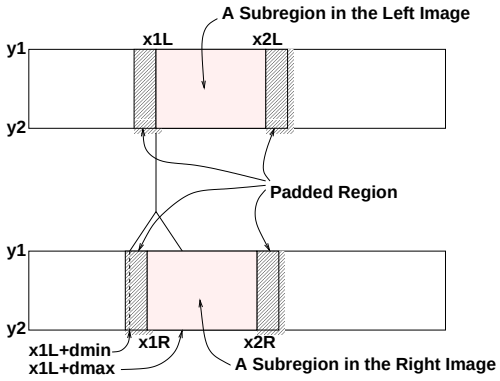


Figure 5: *Obtaining the corresponding region in the right image based on the position of the region in the left image. The position of a corresponding region in the right image also depends on the local disparity information. To remove the boundary effect, a small overlapping region is needed between successive subregions within a horizontal stripe.*

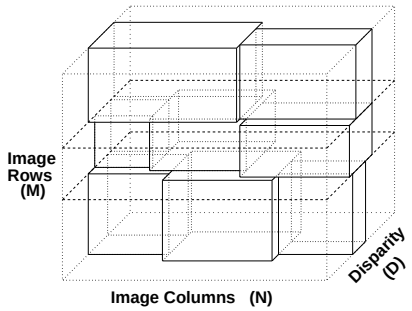


Figure 6: *Size and shape of the 3D correlation volume obtained by stacking together many smaller 3D volumes after correlation calculation using RSR.*

0. Inputs of the algorithm:
   (a) Left/right images for the current level of the pyramid;
   (b) Initial disparity map. This may be zero at the top of a pyramid if there is no prior disparity input;
   (c) Initial width of each of the stripes;
   (d) Penalty threshold.
1. Segment the images into horizontal stripes:
   (a) Divide the disparity map into horizontal stripes with certain width, and obtain the disparity search range for each stripe;
   (b) Recursively merge neighbouring stripes until no neighbouring stripes are similar enough to be merged. At each iteration of the merging process, the two neighbouring stripes with the minimum disparity difference are merged and only one merge occurs at each iteration.
2. For each horizontal stripe, segment it into rectangular regions:
   (a) Divide each horizontal stripe into vertical stripes with certain width, and obtain the disparity search range for each substripe;
   (b) Recursively merge vertical stripes using similar merging technique as in merging horizontal stripes;
   (c) Update the disparity search range information for each region.
3. Obtain the corresponding regions in the right image using the disparity information:
   (a) Use the disparity search range information for each rectangular regions obtained to calculate the corresponding region in the right image.

3 Maximum-Surface in the Volume

From the previous section, we have obtained a 3D cross correlation coefficient volume as shown in Figure 6 using fast cross correlation working on rectangular subregions.

In this section, we will approach the issue of obtaining disparity map from the 3D correlation coefficient volume using dynamic programming techniques, which is computationally efficient. As mentioned in Section 1, there are methods that take information from neighbouring epipolar lines, but they do not provide optimal solutions. The graph-cut framework is usually computationally slow. In this paper, we try to develop a fast algorithm which provides optimal solution for obtaining the disparity map from the 3D volume. We developed a new method to obtain a maximum-surface from a 3D volume using a two-stage dynamic programming (TSDP) technique. Because of the use of TSDP, the algorithm is very fast. This maximum-surface cuts through the 3D volume from the top to the bottom or other directions as illustrated in Figure 7. The maximum-surface gives the global maximum summation of the correlation coefficients along the surface when certain constraints are imposed.

Now we describe our new algorithm for the maximum-surface extraction in a 3D volume of size MND using our fast TSDP method. The first stage of the algorithm is to obtain an accumulated intermediate 3D volume in the vertical direction for each vertical j slice. Assume $C(i, j, d)$ is the correlation coefficient value in the input 3D volume at position (i, j, d), where

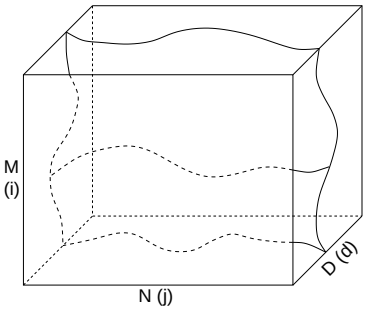


Figure 7: *The illustration of the 3D maximum-surface which give the maximum accumulation of values in the 3D cross correlation coefficient volume.*

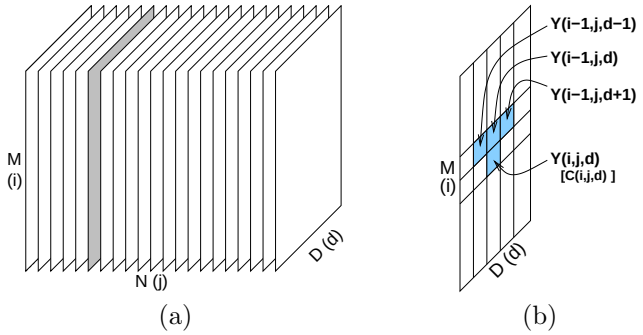


Figure 8: *Obtaining the $Y(i,j,d)$ volume. (a) shows the 3D volume Y with a vertical slice in grey; (b) illustrates the positions of the Y values at each iteration.*

$0 \leq i < M, 0 \leq j < N$, and $0 \leq d < D$. We create an intermediate array $Y(i,j,d)$ which contains the accumulated values of the maximum cross correlation coefficients for each vertical j slice of the same 3D volume using dynamic programming techniques say from top to bottom, i.e. when i changes from 0 to M-1. We may also call the Y volume as the distance function. The Y volume is obtained by working on each individual vertical slice for a particular j. For those values in the top horizontal slice of the volume, i.e. when $i = 0$,

$$Y(0,j,d) = C(0,j,d) \tag{8}$$

i.e. the top (horizontal) slice of Y is a copy of the top slice of C. For the remaining horizontal slices of the volume, the Y values at each position is obtained by using the following recursion which is a typical dynamic programming formula:

$$Y(i,j,d) = C(i,j,d) + \max_{t:|t|\leq p} Y(i-1,j,d+t) \tag{9}$$

where p determines the number of local values that need to be checked. If $p = 1$, only three neighbouring values in Y need to be evaluated. The three values are $Y(i-1,j,d-1), Y(i-1,j,d)$ and $Y(i-1,j,d+1)$. The recursion in Eq. (9) only happens in the (i,d) plane for each particular j as shown in Figure 8. Figure 8(a) is the 3D correlation coefficient volume with one j slice in grey; and Figure 8(b) shows the positions of neighbouring Y values for one plane during the recursion. In this paper we will just use $p = 1$. Other values of p such as 2 or 3 can also be used. But larger p values will increase the computation cost of the above recursion. Other types of smoothness constraints can also be built in at this stage.

After the recursion of the first stage dynamic programming described in the previous paragraph, $Y(i,j,d)$ contains the maximum summation of $C(i,j,d)$ in the vertical direction for each slice from top to bottom of the 3D volume. We now move to a second stage of the TSDP algorithm using volume Y to obtain the disparity map for the input stereo images. In this stage, we work in the horizontal direction. Starting from the bottom of the 3D volume Y where maximum values have been accumulated, we select the 2D horizontal slice with $i = M$-1, i.e. the bottom slice for disparity estimation. From this 2D matrix of size ND, a shortest-path from left to right or from right to left is obtained using dynamic programming techniques as illustrated by the thick line inside the shaded region in Figure 9. The sum of the values along this path gives the maximum value which is also the maximum summation value along the whole 3D surface. This obtained path is related to the disparities for the last or bottom row of the input image. The distance of each point along this path to the middle dashed line in Figure 9 is the obtained disparity for the same x-positioned point of the input image.

We then move from the bottom slice of Y upwards. When calculating the disparity for row number i-1, we use the result obtained for row number i. We now select the 2D horizontal slice number i-1 of the 3D volume Y, and mask out those values which are more than p position away from the shortest-path (the dotted path) obtained from row number i, as shown in Figure 10. Then a new shortest-path (the black curve) is obtained in this 2D matrix from left to right which are constrained to lie inside this grey region. This process of obtaining shortest-path is repeated until the shortest-path for the first row of the image is obtained. When obtaining the shortest-path for each horizontal slice, discontinuity of the path can be factored in.

In the above description, the processing goes from top of the volume to the bottom when building the Y volume; the horizontal dynamic programming step

goes from left to right; and the processing for each horizontal 2D slice goes from the bottom to the top. Other possible ways of obtaining the disparity map is to generate the Y volume from bottom to top; select the top slice of the Y volume and using dynamic programming to find a best path from right to left for obtaining the disparity estimate; move down to the next horizontal 2D slice to obtain disparity. The Y volume can also be built in the horizontal direction for the first stage. In this case, the second stage will be working on each vertical slice of the Y volume.

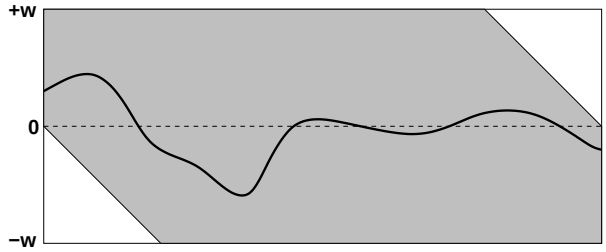


Figure 9: *Shortest-path obtained for the bottom slice of the $Y(i, j, d)$ volume using dynamic programming technique. The bottom-left and the top-right corners of the matrix do not contain valid correlation values.*

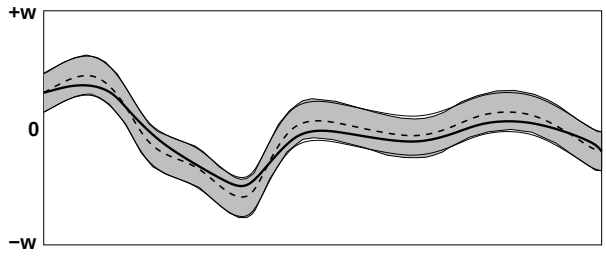


Figure 10: *An illustration of the shortest-path obtained for each horizontal slice of the $Y(i, j, d)$ volume except the bottom one. The dotted line shows the path position obtained from the previous horizontal slice. The grey region include those points that are within p pixels from the dotted line. The dark line is the new path obtained for the current horizontal slice inside the grey band.*

Putting all the shortest-paths for each of the scan line together forms a 3D surface within the 3D volume of Y. Because successive shortest-path for each scan line is obtained in the neighbourhood of the previous path position, the 3D maximum-surface technique gives more consistent disparities. The current implementation does not consider big disparity jumps. Therefore, blurring effect may occur at depth discontinuity regions. The complexity of the TSDP algorithm is linear with respect to the size of the 3D volume, i.e. $O(MND)$.

For a better understanding of the TSDP algorithm which works on a 3D volume, we make an analogy to the ordinary shortest-path extraction on a 2D matrix using dynamic programming. In the case the 2D shortest-path extraction using dynamic programming, the first step is to obtain the distance function from one side of the matrix to the other side. The second step is to backtrack for finding the best path. In our TSDP algorithm for finding the maximum-surface in a 3D volume, the first stage of obtaining the Y volume is similar to the 2D case for calculating the distance function. The second stage of the TSDP algorithm is similar to the backtracking step in the 2D case. The backtracking in 2D matrix is to find a point position along the shortest-path. The backtracking for 3D surface is to find a 2D path along the 3D surface. However, the backtracking stage in the 3D case involves a series of 2D shortest-path extraction for each slice of the 3D volume. The shortest-paths in the neighbouring slices should not be too different from each other.

The steps of our TSDP algorithm for fast stereo matching are:

1. Input: the 3D volume of similarity measurement.
2. Stage One: vertical dynamic programming to obtain the intermediate 3D volume Y which contains the maximum accumulation of correlation values:
   (a) Select each vertical slice of the input 3D volume;
   (b) Update the Y values based on Eq. (9) as shown in Figure 8.
3. Stage Two: horizontal dynamic programming to obtain the disparity map:
   (a) Select the bottom horizontal slice from the Y volume;
   (b) Use dynamic programming technique to find a shortest-path in this 2D slice. This path is related to the disparity estimation for the bottom line of the input image;
   (c) Select next horizontal slice up in the Y volume, mask out pixels which are more than p pixels away from the shortest-path obtained from the previous slice, and find a new shortest-path for this slice. Check to see whether all the slices have been processed, if not, go to Step 3c; otherwise go to Step 3d;
   (d) Put all the shortest-paths from each horizontal slice together to form a disparity map.

4 Matching Strategy

4.1 Coarse-to-fine Scheme

It has been shown that a multi-resolution or coarse-to-fine approach to stereo matching is faster than one without multi-resolution (Kumar and Desai, 1994), as the search range in each level of the pyramid is small. Besides fast computation, a more reliable disparity map can also be obtained by exploiting the multi-resolution data structure. The upper levels of the pyramids are ideal to get an overview of the imaged scene. The details can be found down the pyramid at a higher resolution. There are three useful properties for the coarse-to-fine scheme (Ackermann and Hahn, 1991): (a) the pull-in range or search range can be increased, because at a coarser pyramidal level only rough initial values are needed; (b) the convergence speed can be improved as only the neighbourhood of the previous result needs to be searched; and (c) the reliability of finding correct matches can be increased.

In the current implementation, a lower resolution image is obtained by simply taking the average value of the corresponding $r \times r$ pixels in a higher resolution image in the previous level for its simplicity, where r is the reduction ratio used when building the image pyramid. During the process of projecting the disparity map from the current level of the pyramid to the next (if current level is not level 0, or the highest image resolution), the disparity image size was scaled up by the value of r, and the disparity value was scaled up by the same r. A commonly used value for r is 2. Note that other values of r such as 3 can also be used. The disparity value where the position (i, j) of the next level image is not a multiple of r was obtained by bilinear interpolation.

The size of the 3D volume is small in this coarse-to-fine framework as the disparity search is only necessary in the neighbourhood of the disparity obtained in the previous level. In the coarse-to-fine matching scheme, the computation complexity for the step of obtaining the disparity map from the 3D volume is only $O(MND')$, where $D' = 3$ in the lower levels of the image pyramid. This is due to the fact that the disparity search is local to the initial estimates.

Our new RSR technique which uses the intermediate disparity map for obtaining the smaller rectangular regions works in the coarse-to-fine scheme.

4.2 Sub-pixel Accuracy

Sub-pixel accuracy can be obtained by fitting a second degree curve to the correlation coefficients in the neighbourhood of the disparity and the extrema of the curve can be obtained analytically. The general form of the second degree curve (parabola) is: $f(x) = a + b \cdot x + c \cdot x^2$. The maximum can be found where the slope is zero in the quadratic function. The sub-pixel position can be found at $x = -b/2c$. If only three correlation values at and around the position d are used, e.g. values at the points $d-1, d$, and $d+1$, the sub-pixel position of the disparity can be calculated using the following formula (Anandan, 1989):

$$x = d + \frac{1}{2} \times \frac{C(d-1) - C(d+1)}{C(d-1) - 2C(d) + C(d+1)} \qquad (10)$$

where $C(d)$ is the correlation value in the 2D matrix at position d, and x is the sub-pixel disparity obtained. If five correlation values are used, e.g. the values at five points $d-2, d-1, d, d+1$, and $d+2$, we derive the following equation for the calculation of sub-pixel position:

$$x = d + \frac{7}{20} \times \frac{2C(d-2) + C(d-1) - C(d+1) - 2C(d+2)}{2C(d-2) - C(d-1) - 2C(d) - C(d+1) + 2C(d+2)} \qquad (11)$$

4.3 Algorithm Steps

The steps of our proposed algorithm, which uses the combination of RSR and TSDP, i.e. RSR+TSDP, for fast stereo matching are:

1. Build image pyramids with P levels (from 0 to $P-1$), with the reduction ratio of r, from the original left and right images; The images at upper or coarse resolution levels are obtained by averaging the corresponding $r \times r$ pixels in the lower or finer resolution level images.
2. Initialize the disparity map as zero for level $k = P - 1$ and start stereo matching at this level.
3. Perform stereo matching using the method described in Sections 2-4 which includes:
   (a) Segment images into rectangular subregions based on the current disparity map;
   (b) Perform fast zero mean normalised correlation to obtain the correlation coefficients for each subregions and build a 3D correlation coefficient volume for the whole image;
   (c) Use the two-stage dynamic programming technique to find the 3D maximum-surface, which will then give the disparity map as described in Section 3.
4. If $k \neq 0$, propagate the disparity map to the next level in the pyramid using bilinear interpolation, set $k = k - 1$ and then go back to Step 3; if $k = 0$, go to Step 5.

5. Fit curves to obtain sub-pixel accuracy using Eq. (10) or Eq. (11) if necessary.
6. Display disparity map.

5 Experimental Results

This section shows some of the results obtained using our new RSR+TSDP algorithm described in this paper. A variety of images have been tested, including synthetic images and different types of real images. The input left and right images are assumed to be rectified epipolar images. Therefore, matching points lie on the same horizontal scan line. The positions of the input left and right images have been swapped in the figures so that cross eye viewing becomes easier. Implementations of the Roy's (Roy, 1999), Cox's (Cox et al., 1996) and Sun's (Sun, 1997) methods are used for comparison. The codes for Roy's and Cox's methods are downloaded from their web pages.

Synthetic Images

Figure 11 gives the results of algorithms in (Sun, 1997) and our new algorithms running on two pairs of synthetic images. The two columns on the left show the input left and right images. The third column is the results obtained using our earlier method presented in (Sun, 1997). The last column shows the results using the method described in this paper. Shown in the top row are images of a sphere on a table. The size of this pair of images is 256×256. The bottom row shows images of a corridor. The size of the corridor images is 512×512. It can be seen from these figures that our new 3D maximum-surface method using TSDP gives better results, especially for the sphere image.

Random Dot Stereogram (RDS)

A pair of random dot stereogram are shown in Figure 12(a,b). The stereo matching results for our new method, Roy's, Cox's and Sun'97 methods are given in Figure 12(c,d,e,f). Among these four methods, only Cox's method explicitly formulated stereo occlusions. The results in (c) and (f) are very similar, but the running times as will be shown later are different. The main differences of the disparity estimates using this pair of RDS images for these four methods are at the discontinuity boundaries.

Real Images

Figures 13-15 show some results using real images.

Park meter: The input images shown in Figure 13(a,b) are the frames 2 and 14 of the park meter sequence. The matching results for our new method, Roy's, Cox's and Sun'97 methods are given in Figure 13(c,d,e,f).

Pentagon: The input images are shown in Figure 14(a,b). The matching results for our new method, Roy's, Cox's and Sun'97 methods are given in Figure 14(c,d,e,f).

Fruit scene: The input images are shown in Figure 15(a,b). The matching results for our new method, Roy's, Cox's and Sun'97 methods are given in Figure 15(c,d,e,f).

From the results shown in Figures 13, 14 and 15, it can be seen that our new RSR+TSDP method gives more consistent results than the other three methods. Many other types of real images have also been tested, and good results have been obtained. Due to limitation of space, only small portion of the tested images were shown here. Figure 16 gives some more results obtained by using our new RSR+TSDP method described in this paper. The image shown in Figure 16(a) is a picture of softball on newspaper. Figure 16(b) shows a bent circuit board. Figure 16(c) shows an aerial photo with houses in the image.

Running Times

The computer used is a 500MHz Pentium III running Linux. The algorithm was implemented in the C language without using any hardware supports or assembly languages. The typical running time for the algorithm on a 256×256 image with about 30 pixels disparity is in the order of several hundred milliseconds. Note that the timings reported in this paper do not include the sub-pixel calculation step.

Table 1 gives some of the typical running times of the algorithm on different sizes of images with different disparities using whole image correlation and the RSR methods. The size of the correlation window used for the images shown in the table is 9×9. The reduction ratio r used in the pyramid generation process is 2. The time shown in the table includes the time for the pyramid building process and the time for image reading and writing. For example, for the "`ball`" image of size 256×256 as shown in Figure 16(a), the program only takes 0.32 seconds. It only takes 1.39 seconds to obtain the disparity map for the 512×512 pixel "`pentagon`" image.

The time shown for "User time1" is obtained without using the RSR method as described in Section 2.2, while the time shown for "User time2" is obtained by using the RSR method. It can be seen that the time spent by the algorithm using RSR method is almost half of the time without using the RSR method. The amount of time saved depends on the shape of the objects. Interested readers could try our algorithm using their own images by accessing the web page given in Section 8.

Table 2 gives some of the typical running times of the 2D matrix and 3D maximum-surface algorithms on different size of images. Other parameters used such as

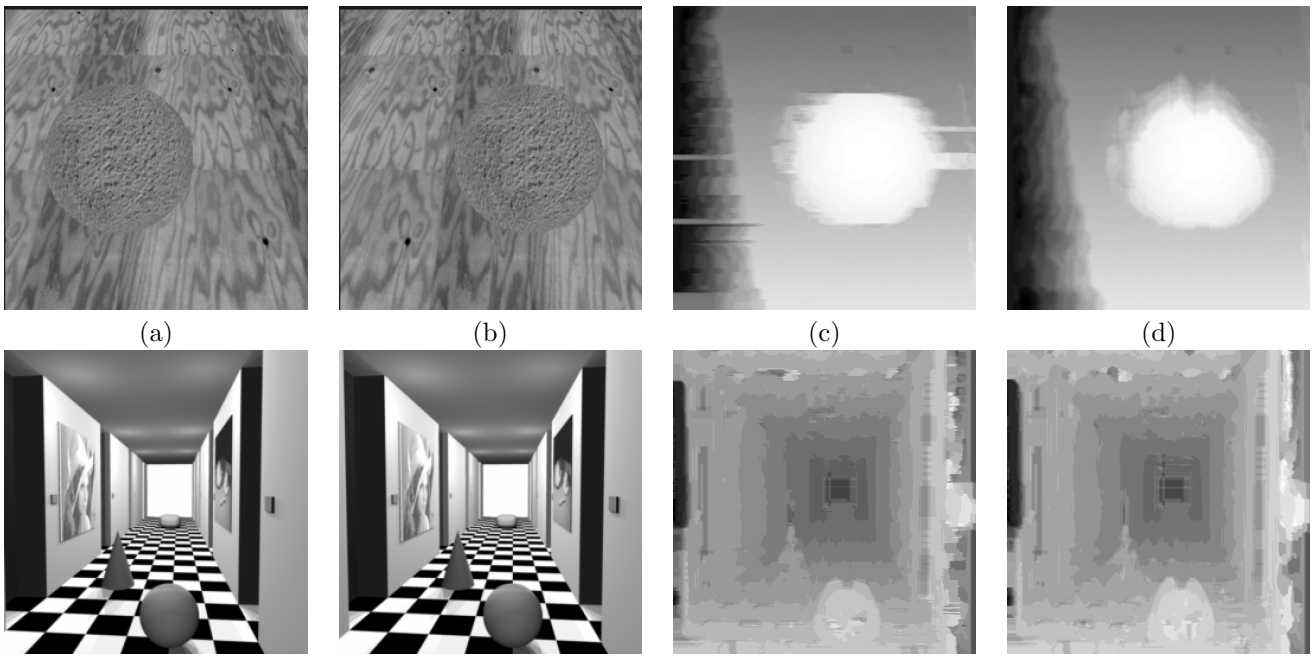


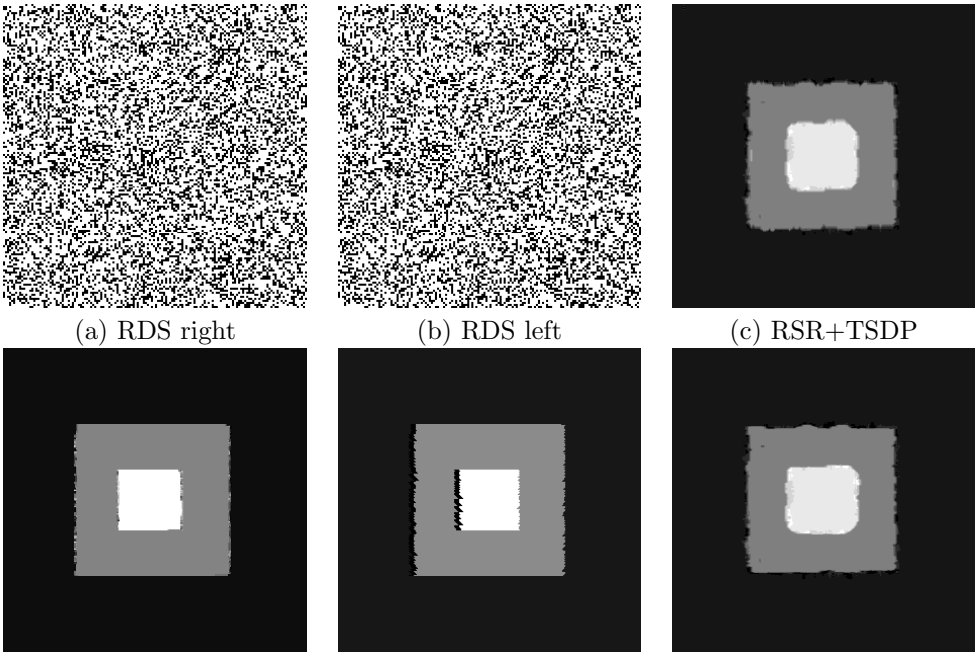
Figure 11: *The matching results for two pairs of synthetic images. The image size for the top row is 256×256. The image size for the bottom row is 512×512. The top row gives the images of a sphere on a table. The bottom row shows the images of a corridor. (a,e) right images; (b,f) left images; (c,g) the disparity maps recovered using method in (Sun, 1997); and (d,h) the disparity maps recovered using our new RSR+TSDP method. (Images (a,b) courtesy of Bill Hoff at the University of Illinois (Hoff and Ahuja, 1989). Images (e,f) courtesy of Computer Vision Group, Computer Science III, University of Bonn.)*

Table 1: *Running times of the whole image correlation and the RSR algorithms on different images. The dynamic programming stage of this test runs on 2D matrix. The size of the correlation window is 9×9. The reduction ratio r used in the pyramid generation process is 2. The* `ball`, `circuit` *and* `flat` *images are shown in Figure 16. The* `pentagon` *image is shown in Figure 14(a)(b).*

Image name	**Image size**	**Pyramid levels**	**Search range**	**Disparity range**	**User time1**	**User time2**
`ball`	256×256	3	[-4,4]	[-19,7]	0.53s	0.32s
`pentagon`	512×512	3	[-2,2]	[-10,10]	2.42s	1.39s
`circuit`	512×512	3	[-5,5]	[-21,23]	3.36s	1.59s
`flat`	1000×1000	4	[-3,3]	[-31,23]	16.86s	7.51s

pyramid levels, disparity search ranges, image sizes, are given in the table. The last two columns in the table show the timings of the algorithm described in (Sun, 1997) (Method 2D path) and the algorithm described in this paper (Method 3D surface). There is not much difference in the speed of the two algorithms. One might expect to see that the execution time for our new algorithm will be much longer than that of the 2D path method because of the need for the extraction of 3D maximum-surface. The computation time for the 3D surface method is only slightly longer than that of the 2D path method.

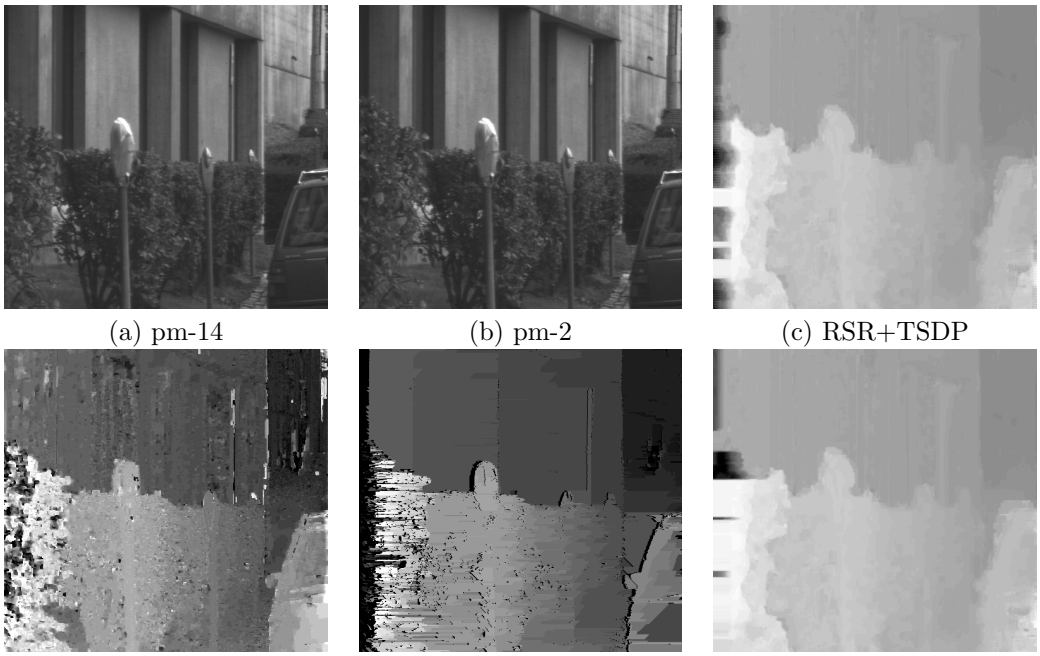
Table 3 shows the computation times for Roy, Cox and our algorithms on three pair of images. Roy's algorithm takes much longer to finish compared with other two algorithms. Our method is also much quicker than Cox's method. The reason that our new algorithm can achieve fast computational speed will be discussed in the next section. There are stereo vision systems that are able to perform stereo match-



(a) RDS right (b) RDS left (c) RSR+TSDP

(d) Roy's method (e) Cox's method (f) Sun'97

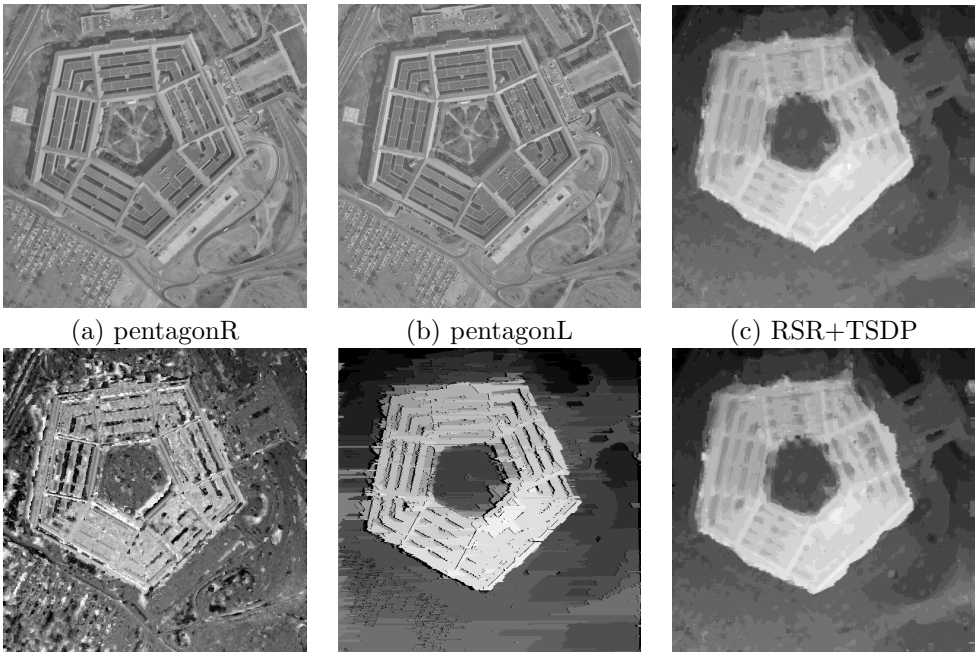
Figure 12: *Synthetic random dot stereogram. (a) and (b) are the right and left input images. (c) shows our result using the RSR+TSDP algorithm described in this paper; (d) result obtained using Roy's method; (e) result obtained using Cox's method; and (f) result obtained using Sun's method described in (Sun, 1997).*



(a) pm-14 (b) pm-2 (c) RSR+TSDP

(d) Roy's method (e) Cox's method (f) Sun'97

Figure 13: *Park meter scene. (a) and (b) are the right and left input images. (c) Results obtained using our new method (RSR+TSDP). (d) Results obtained using Roy's method. (e) Results obtained using Cox's method. (f) The matching results using the method described in (Sun, 1997). (Images (a,b) courtesy of Carnegie Mellon University).*



(a) pentagonR (b) pentagonL (c) RSR+TSDP

(d) Roy's method (e) Cox's method (f) Sun'97

Figure 14: *Pentagon stereo. (a) and (b) are the right and left input images. (c) Results obtained using our new method (RSR+TSDP). (d) Results obtained using Roy's method. (e) Results obtained using Cox's method. (f) Results obtained using the method described in (Sun, 1997). (Images (a,b) courtesy of Bill Hoff at the Univ. of Illinois (Hoff and Ahuja, 1989)).*

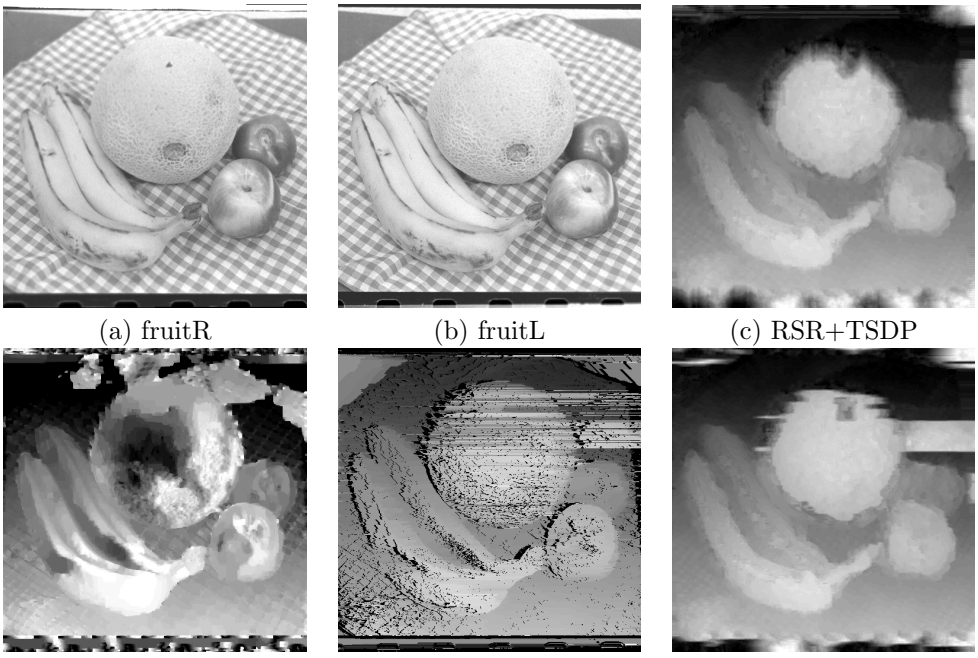
Table 2: *Running times of our algorithms on different images. The size of the correlation window is 9×9. The reduction ratio r used in the pyramid generation process is 2. Both of these algorithms use RSR.*

Image name	**Image size**	**Pyramid levels**	**Search range**	**Disparity range**	**Method 2D path**	**Method 3D surface**
`ball`	256×256	3	[-4,4]	[-19,7]	0.32s	0.37s
`pentagon`	512×512	3	[-2,2]	[-10,10]	1.39s	1.50s
`circuit`	512×512	3	[-5,5]	[-21,23]	1.59s	1.82s
`flat`	1000×1000	4	[-3,3]	[-31,23]	7.51s	7.53s

ing in video rate (Point Grey Research, ; CMU Video-rate Stereo Machine, ; SRI Stereo Engine,). But all these systems have hardware or assembly language supports and some have multiple cameras.

6 Discussion on Reliability and Computational Speed

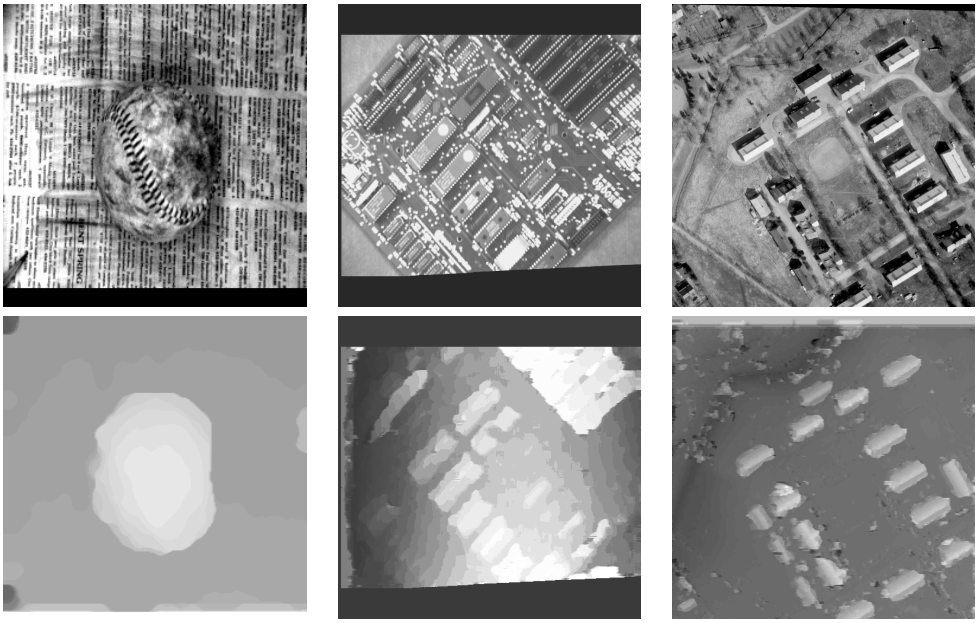
The reliable results of our algorithm are achieved by applying the combination of the following techniques: (1) Coarse-to-fine strategy is used. As mentioned in Section 4.1, the upper levels of the pyramids are ideal to get an overview of the imaged scene. Therefore the matching in the upper levels will have a more global effect. The details can be found down the pyramid at higher resolution. (2) The zero mean normalized cross correlation similarity measure is used, which is independent of differences in brightness and contrast due to the normalization with respect to mean and standard deviation. The similarity measure using SAD or SSD, which is relatively cheap computationally, is not independent of differences in brightness and contrast. (3) The correlation coefficient value is used as input to the dynamic programming stage. A number of approaches that use dynamic programming method just use the intensity value along the left and right epipolar lines. These approaches do not take the neighbourhood



(a) fruitR (b) fruitL (c) RSR+TSDP

(d) Roy's method (e) Cox's method (f) Sun'97

Figure 15: *Fruit stereo. (a) and (b) are the right and left input images. (c) Results obtained using our new method (RSR+TSDP). (d) Results obtained using Roy's method. (e) Results obtained using Cox's method. (f) Results obtained using the method described in (Sun, 1997). (Images (a,b) courtesy of Bill Hoff at the University of Illinois (Hoff and Ahuja, 1989)).*



(a) ball (b) circuit (c) flat

Figure 16: *Three images (only left) and the disparity maps obtained. The top row gives the left images of stereo pairs. The bottom row shows the disparity map. The image size in (a) is 256×256; The image size in (b) is 512×512; The image size in (c) is 1000×1000. (Input images (a,b) courtesy of Bill Hoff at the University of Illinois (Hoff and Ahuja, 1989); input images (c) courtesy of Stuttgart ISPRS Image Understanding datasets).*

Table 3: *Running times of different algorithms. The* RDS *image is shown in Figure 12(a,b). The* pm *image is shown in Figure 13(a,b). The* pentagon *image is shown in Figure 14(a,b).*

Image name	Image size	Disparity search range	Roy's method	Cox's method	Our method
RDS	300×300	10	300.63s	1.44s	0.45s
pm	512×480	25	374.83s	4.28s	1.75s
pentagon	512×512	25	462.47s	5.43s	1.62s

information from the successive scan lines into account. In our approach, information from different scan lines have been used by using the correlation coefficient value which is obtained from a local window during the cross correlation step. (4) Dynamic programming technique is used to find a 3D maximum-surface in the correlation volume. By using the two-stage dynamic programming technique on the input correlation coefficient volume, one will obtain a more smooth surface within the volume. The 3D maximum-surface method takes all the information into account, rather than work individually for each of the epipolar lines. As most of the other correlation based matching methods, if there is a large area in the image contains little texture, the correlation coefficients for a number of points in this area may be undefined. The result of matching may not be very pleasing. However, if this area is not too large, the effect of using the dynamic programming technique to find a surface will have the effect of filling the holes where undefined correlation values exist. The algorithm presented in this paper does not explicitly model occlusion. Also because it is an area-based method, blurring effect at depth discontinuity regions may occur. The coarse-to-fine strategy may also encounter problems at depth discontinuity regions.

The fast computational speed of our algorithm is achieved in conjunction with some of the aspects mentioned above for achieving reliability of the algorithm. Some of the aspects are: (1) Fast zero mean normalized cross correlation is used. The original idea of box-filtering for calculating image mean was used for fast calculation of image variance at the same time when one calculates the image mean. The fast cross correlation between two images are achieved by fixing one shift for every points on the left image and calculating the cross correlation in the way similar to that when one calculates the image variance. This way the redundant computation is eliminated and fast computation is achieved. (2) We have used a rectangular subregioning technique for fast computation of correlation coefficients. Rather than working with the whole image when perform cross correlation, the input images are sub-divided into rectangular subregions depending on the current disparity map in a certain level of the pyramid. These regions tend to have the property that when the disparity range is small the size of the region is large, and when the disparity range is large the size of the region is small. The end effect of these are the reduced computation cost. (3) Apart from having the advantages of increasing the reliability, the coarse-to-fine approach is also faster than one without using it. The smaller image size on the upper levels of the pyramid gives a rough estimate of the disparity. In the larger image on the lower levels of the pyramid, one can use the initial estimate from the previous level to refine the disparity map in a reduced search range. At the stage of obtaining the maximum-surface from the 3D correlation coefficient volume, the complexity of the algorithm is $O(MND')$. Because a coarse-to-fine scheme is used, the disparity search range in the lower levels of the pyramid is only around the neighbourhood of the result obtained in the previous levels. Therefore, D' is usually very small. (4) A two-stage dynamic programming technique is used to find a maximum-surface in the 3D correlation volume. Rather than using the methods described in (Roy and Cox, 1998; Chen and Medioni, 1998), a dynamic programming technique is used which is computationally efficient.

7 Conclusions

We have developed a fast and reliable stereo matching method using rectangular subregioning, fast correlation and 3D maximum-surface techniques in the coarse-to-fine framework. The algorithm produces a reliable dense disparity map: for each point in the image, a disparity value is obtained. The fast cross correlation method was developed from the box-filtering idea. The time spent in the stage for obtaining the normalized cross correlation is almost invariant to the search window size. The processing speed is further improved by segmenting the input image into subimages and work with the smaller images which tend to have smaller disparity ranges. This new subregioning technique is also helpful to reduce the memory stor-

age space. The 3D maximum-surface is obtained from the 3D correlation volume using a new two-stage dynamic programming technique. There are two original contributions in this paper. The first is the rectangular subregioning (RSR) method for further speeding up the correlation calculation. The second is the two-stage dynamic programming (TSDP) method for 3D maximum-surface extraction for disparity estimation. The typical running time for a 512×512 image is in the order of a few seconds. The algorithm is implemented in the C language on standard computers, and no special hardware or assembly language is used. The algorithm was shown to be fast and reliable by testing on several different types of images: both synthetic and real images.

8 Web Demo

There is a web page setup to allow interested readers to run our fast stereo matching algorithm using their own stereo images. The web demo address is at:
http://extra.cmis.csiro.au/IA/changs/stereo/

Acknowledgement

The author is grateful to the author or owners of the images used in this paper. We thank the colleagues at CSIRO Mathematical and Information Sciences for many useful discussions and suggestions. We also thank anonymous reviewers for their valuable comments.

References

Ackermann, F. and Hahn, M. (1991). Image pyramids for digital photogrammetry. In Ebner, H., Fritsch, D., and Heipke, C., editors, *Digital Photogrammetric Systems*, pages 43–58. Wichmann.

Anandan, P. (1987). A computational framework and an algorithm for the measurement of visual motion. Technical Report 87-73, Computer and Information Science, University of Massachusetts at Amherst.

Anandan, P. (1989). A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310.

Aschwanden, P. and Guggenbühl, W. (1992). Experimental results from a comparative study on correlation-type registration algorithms. In Förstner, W. and Ruwiedel, S., editors, *Robust Computer Vision*, pages 268–289. Wichmann.

Ayache, N. and Faverjon, B. (1985). Fast stereo matching of edge segments using prediction and verification of hypotheses. In *Proceedings of Computer Vision and Pattern Recognition*, pages 662–664.

Baldwin, R., Yamada, H., and Yamamoto, K. (1990). Disparity space and dynamic programming for automatic production of very dense range maps. In Gruen, A. and Baltsavias, E., editors, *Close-Range Photogrammetry Meets Machine Vision*, volume 1395, pages 217–225, Zurich, Switzerland. SPIE.

Banks, J., Bennammed, M., and Corke, P. (1999). Fast and robust stereo matching algorithms for mining automation. *Digital Signal Processing*, 9:137–148.

Barnea, D. I. and Silverman, H. F. (1972). A class of algorithms for fast digital image registration. *IEEE Transactions on Computers*, C-21:179–186.

Belhumeur, P. N. (1996). A Bayesian approach to binocular stereopsis. *International Journal of Computer Vision*, 19(3):237–262.

Bensrhair, A., Miché, P., and Debrie, R. (1996). Fast and automatic stereo vision matching algorithm based on dynamic programming method. *Pattern Recognition Letters*, 17:457–466.

Birchfield, S. and Tomasi, C. (1999). Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision*, 35(3):269–293.

Bobick, A. F. and Intille, S. S. (1999). Large occlusion stereo. *International Journal of Computer Vision*, 33(3):181–200.

Chen, Q. and Medioni, G. (1998). Building human face models from two images. In *Multimedia Signal Processing*, pages 117–122, Redonda Beach, CA.

CMU Video-rate Stereo Machine. http://www.ri.cmu.edu/projects/project_53.html.

Cochran, S. D. and Medioni, G. (1992). 3D surface description from binocular stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):981–994.

Cox, I., Hingorani, S., Rao, S., and Maggs, B. (1996). A maximum likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63(3):542–567.

Faugeras, O., Hotz, B., Mathieu, H., Viéville, T., Zhang, Z., Fua, P., Théron, E., Moll, L., Berry, G., Vuillemin, J., Bertin, P., and Proy, C. (1993). Real time correlation-based stereo: Algorithm, implementations and applications. Technical Report RR-2013, INRIA.

Fua, P. (1993). A parallel stereo algorithm that produce dense depth maps and preserves image features. *Machine Vision Applications*, 6(1):35–49.

Fusiello, A., Roberto, V., and Trucco, E. (1997). Efficient stereo with multiple windowing. In *Proceedings of Computer Vision and Pattern Recognition*, pages 858–863, Puerto Rico. IEEE Computer Society Press.

Geiger, D., Ladendorf, B., and Yuille, A. (1995). Occlusions and binocular stereo. *International Journal of Computer Vision*, 14:211–226.

Gimel'farb, G. L., Krot, V. M., and Grigorenko, M. V. (1992). Experiments with symmetrized intensity-based dynamic programming algorithms for reconstructing digital terrain model. *International Journal of Imaging Systems and Technology*, 4:7–21.

Grimson, W. E. L. (1985). Computational experiments with a feature based stereo algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7:17–34.

Hoff, W. and Ahuja, N. (1989). Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(2):121–136.

Intille, S. and Bobick, A. (1994). Disparity-space images and large occlusion stereo. In *Proceedings of European Conference on Computer Vision*, pages 179–186, Stockholm, Sweden.

Ishikawa, H. and Geiger, D. (1998). Occlusions, discontinuities, and epipolar lines in stereo. In *Proceedings of European Conference on Computer Vision*, pages 232–248, Freiburg, Germany.

Jones, D. G. and Malik, J. (1992). Computational framework for determining stereo correspondence from a set of linear spatial filters. *Image and Vision Computing*, 10(10):699–708.

Kim, Y.-S., Lee, J.-J., and Ha, Y.-H. (1997). Stereo matching algorithm based on modified wavelet decomposition process. *Pattern Recognition*, 30(6):929–952.

Kolesnik, M. I. (1993). Fast algorithm for the stereo pair matching with parallel computation. In Chetverikov, D. and Kropatsch, W. G., editors, *5th International Conference on Computer Analysis of Images and Patterns*, pages 533–537, Budapest, Hungary. Springer-Verlag.

Konecny, C. and Pape, D. (1981). Correlation techniques and devices. *Photogrammetric Engineering and Remote Sensing*, 47(3):323–333.

Kumar, K. S. and Desai, U. B. (1994). New algorithms for 3D surface description from binocular stereo using integration. *Journal of the Franklin Institute*, 331B(5):531–554.

Lloyd, S. A. (1985). A dynamic programming algorithm for binocular stereo vision. *GEC Journal of Research*, 3(1):18–24.

Lotti, J.-L. and Giraudon, G. (1994a). Adaptive window algorithm for aerial image stereo. In *Proceedings of International Conference on Pattern Recognition*, volume A, pages 701–703, Jerusalem, Israel. IEEE Computer Society Press.

Lotti, J.-L. and Giraudon, G. (1994b). Correlation algorithm with adaptive window for aerial image in stereo vision. In *European Symposium on Satellite Remote Sensing (EUROPTO)*, pages 2315–10, Rome, Italy.

Medioni, G. and Nevatia, R. (1985). Segment-based stereo matching. *Computer Vision, Graphics, and Image Processing*, 31:2–18.

Ohta, Y. and Kanade, T. (1985). Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7:139–154.

O'Neill, M. and Denos, M. (1996). Automated system for coarse-to-fine pyramidal area correlation stereo matching. *Image and Vision Computing*, 14(3):225–236.

Point Grey Research. http://www.ptgrey.com/.

Porr, B., Cozzi, A., and Wörgötter, F. (1998). How to 'hear' visual disparities: real-time stereoscopic spatial depth analysis using temporal resonance. *Biological Cybernetics*, 78(5):329–336.

Rechsteiner, M., Schneuwly, B., and Troester, G. (1994). Dynamic workspace monitering. In Ebner, H., Heipke, C., and Eder, K., editors, *International Archives of Photogrammetry and Remote*

Sensing, volume 30, pages 689–696, Munich, Germany.

Rojas, A., Calvo, A., and Muñoz, J. (1997). A dense disparity map of stereo images. *Pattern Recognition Letters*, 18(4):385–393.

Rosenfeld, A. and Kak, A. C. (1982). *Digital Picture Processing*, volume II. Academic Press, New York, second edition.

Roy, S. (1999). Stereo without epipolar lines: A maximum-flow formulation. *International Journal of Computer Vision*, 34(2/3):147–161.

Roy, S. and Cox, I. J. (1998). A maximum-flow formulation of the N-camera stereo correspondence problem. In *Proceedings of International Conference on Computer Vision*, pages 492–499, Bombay, India. IEEE.

Scharstein, D. and Szeliski, R. (1998). Stereo matching with nonlinear diffusion. *International Journal of Computer Vision*, 28(2):155–174.

SRI Stereo Engine. http://www.ai.sri.com/~konolige/svs/.

Sun, C. (1997). A fast stereo matching method. In *Digital Image Computing: Techniques and Applications*, pages 95–100, Massey University, Auckland, New Zealand.

Sun, C. (1998). Multi-resolution rectangular subregioning stereo matching using fast correlation and dynamic programming techniques. Technical Report 98/246, CSIRO Mathematical and Information Sciences, Australia.

Wei, G.-Q., Brauer, W., and Hirzinger, G. (1998). Intensity- and gradient-based stereo matching using hierarchical Gaussian basis functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1143–1160.

Wu, Q. X. (1995). A correlation-relaxation-labeling framework for computing optical flow — template matching from a new perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):843–853.

Wu, Q. X., McNeill, S. J., and Pairman, D. (1995). Fast algorithms for correlation-relaxation technique to determine cloud motion fields? In *Digital Image Computing: Techniques and Applications*, pages 330–335, Brisbane, Australia.

Xiong, Y., Wang, D., and Zhang, G. (1996). Integrated method of stereo matching for computer vision. In Tescher, A. G., editor, *SPIE Proc. Applications of Digital Image Processing XIX*, pages 665–676, Denver, Colorado.

Yang, Y. and Yuille, A. L. (1995). Multilevel enhancement and detection of stereo disparity surfaces. *Artificial Intelligence*, 78(1–2):121–145.

Zitnick, C. and Kanade, T. (1998). A volumetric iterative approach to stereo matching and occlusion detection. Technical Report CMU-RI-TR-98-30, Robotics Institute, Carnegie Mellon University.

Zitnick, C. and Kanade, T. (2000). A cooperative algorithm for stereo matching and occlusion detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):675–684.