IEEE Workshop on Stereo and Multi-Baseline Vision. In Conjunction with IEEE CVPR-2001, pp.44-53, Kauai, Hawaii, December 9-10, 2001.

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

Changming Sun

CSIRO Mathematical and Information Sciences
Locked Bag 17, North Ryde, NSW 1670, Australia
changming.sun@cmis.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 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 correlation window and by segmenting the stereo images at different levels of the pyramid into rectangular subimages. The disparity for the whole image is found in the 3D correlation coefficient volume by obtaining the maximum-surface using our novel two-stage dynamic programming technique. There are two original contributions in this paper: (1) development of a rectangular subregioning (RSR) technique for fast similarity measure; and (2) development of a novel two-stage dynamic programming (STDP) technique for obtaining 3D maximum surface in a 3D volume efficiently. Typical running time of our algorithm implemented in C language on a 512×512 image is in the order of a few seconds. A variety of synthetic and real images have been tested, and good results have been obtained.

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 such that the matched 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 position of the scene points given knowledge about the relative geometry of the two cameras. Matching techniques can be divided mainly into area-based and feature-based image matching, or a combination of them. Other types of techniques such as pixel-based [7], diffusion-based [31], wavelet-based [19], phase-based [25], and filter-based [18] matching methods have also been developed.

Intille and Bobick [17, 8] presented a stereo algorithm that incorporates the detection of the occlusion regions directly into the matching process. Wei *et al* proposed an intensity- and gradient-based stereo matching using hierarchical Gaussian basis functions [35]. Fua [13] described a correlation based multiresolution algorithm which is followed by interpolation. Anandan [1] 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 globally the matching problem [15, 3, 27, 6]. There are other algorithms which perform fast stereo matching [14, 20, 4]. Sun [33] developed 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 [23]. Cox *et al* presented a stereo matching algorithm using dynamic programming technique considering the inter-scanline constraints [11]. The method needs small number of iteration and approximates the global solutions.

Roy [29] and Roy & Cox [30] developed an algorithm for solving the N-camera stereo correspondence problem by transforming it into a maximum-flow problem. The average running time for Roy and Cox's algorithm was $O((MN)^{1.2}D^{1.3})$ (with image size M, N and depth resolution D) [30]. Chen and Medioni [9] presented a propagation type of algorithm similar to [23]. The techniques they used included non-maxima suppression, seed voxel selection and surface tracing. There was no mentioning about the speed issues in [9]. Yang and Yuille proposed a non-linear filter for detecting disparity surface in a 3D volume [37]. 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 [38]. The algorithm updates the match likelihood values by diffusing support among neighbouring values and inhibiting others. There are stereo vision systems that are able to perform stereo matching in video rate [24, 10, 32]. But all these systems have hardware or assembly language supports and some have multiple cameras.

In this paper we will present novel techniques for segmenting stereo images into rectangular subregions (RSR) for fast similarity calculation and for obtaining 3D maximum surface in a 3D correlation coefficient volume for fast stereo matching. We will also address some of the other efficient and reliable implementation aspects of the stereo matching algorithms by using fast correlation and dynamic programming techniques in a multi-resolution scheme, which results in very fast stereo matching. The disparity is obtained from a 3D correlation coefficient volume using a two-stage dynamic programming (TSDP) technique considering the continuity of the neighbouring epipolar scan lines. The rest of the paper is organised as follows: Section 2 proposes our new rectangular subregioning method over the input images for fast calculation of similarity measure. Section 3 presents our new method of stereo matching by finding the maximum surface in the 3D correlation volume by using the two-stage dynamic programming techniques. 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 issue of our algorithm. Section 7 gives concluding remarks.

2 Rectangular Subregioning for Fast Similarity Measures

Different similarity measures have been used in the literature for matching, and their performance and computation costs vary [26, 2]. The most commonly used similarity measure is the cross correlation coefficient. It is popular because it corresponds to optimal signal-to-noise ratio estimation [28]. 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 [36, 5]. Konecny and Pape [21] 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 [12, 26, 2]. 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* [12] developed a recursive technique to calculate the correlation coefficients which are invariant to the correlation window size. Sun [33, 34] used box-filtering technique for fast cross correlation. The following subsections describe our early work in [33, 34] for achieving fast correlation on the whole image and our new technique of using rectangular subregioning for fast similarity measure. The complexity of the algorithm is $O(MND)$. The storage space needed for the correlation coefficients is in the order of $4MND$ bytes. The fast algorithm for ZNCC calculation can be easily adopted to obtain the SAD and SSD measures efficiently.

2.1 Rectangular Subregioning (RSR)

Rather than work with the whole image during the fast correlation stage, we could work with subimages to speed up the correlation calculation further and reduce the memory space for storing the correlation coefficients. If the input image is divided into R rectangular subregions, the computation complexity will be $\sum_{i=0}^{R-1}(M_iN_iD_i)$, 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_iN_i), D_i is expected to be smaller than D. Even when it is not much smaller, the size of this region (M_iN_i) is much smaller than the input image. It is anticipated that $\sum_{i=0}^{R-1}(M_iN_iD_i)$ 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 the overhead. There is another advantage for working with subimages in terms of memory usage. 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 $\max_i(4M_iND_i')$, because the memory for each subregion is dynamically allocated and freed, where D_i' is the disparity search range for a particular horizontal stripe.

2.2 Rectangular Subregioning Process

Now we will describe our fast method for segmenting an image into rectangular subregions for fast stereo matching in more detail. 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 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. 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 property 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 overhead into account. At each step of the merging process, only neighbouring stripes can be merged.

After the image has been segmented into horizontal stripes, each such stripe can then be cut into 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 $\sum_{i=0}^{R-1}(M_iN_iD_i)$ is smaller. Figure 1 illustrates the rectangular subregioning precess. Figure 1(a) gives the initial horizontal stripes of the input image. Figure 1(b) shows the subregions obtained from Figure 1(a). Figure 1(c) is the initial vertical cuttings for each horizontal stripe. Figure 1(d) shows the result of merging the small rectangular regions within the horizontal stripe. Fast correlation is performed on each of these smaller rectangle images, and the obtained correlation coefficients are put together into horizontal stripes or cubes. Figure 2(a) shows a disparity map which is used for rectangular subregioning at one level of the pyramid. Figure 2(b) shows the rectangular regions obtained. Most of the regions have small disparity ranges.

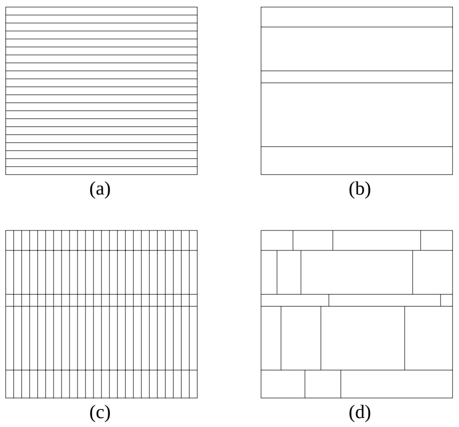

Figure 1: *Rectangular subregioning through merging thin rectangles. (a) shows the initial horizontal stripes for the input disparity map. (b) illustrates some horizontal regions after the horizontal stripe merging process. (c) shows the initial vertical stripes for each of the horizontal regions shown in (b); and (d) is the final rectangular regions obtained.*

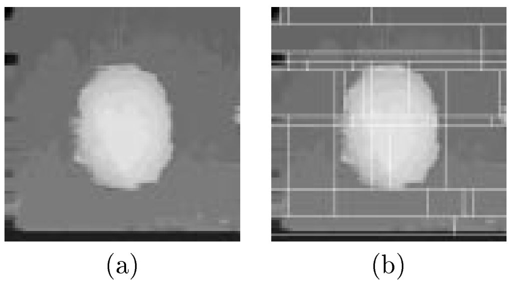

Figure 2: *An example result of sub-dividing the whole image into subimages based on intermediate disparity map in the pyramid. (a) a disparity map at a particular image pyramid; (b) the disparity map shown in (a) overlaid with the rectangles obtained. Each of these rectangles will be used for running the fast correlation algorithm described earlier.*

2.3 Corresponding Regions in Right Image

When actually performing fast correlation calculation for each of the subregions, certain size of region overlapping 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. 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 right image. 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 image is within d_{min} and d_{max}, the x position of the corresponding region in the right image $x1R$ should be a position between $x1L+d_{min}$ and $x1L+d_{max}$. The approach we used here for rectangular subregioning may not be the global minimum, but it is fast and simple and serves our purpose for fast processing.

2.4 Algorithm Steps for Rectangular Subregioning

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

0. Inputs of the algorithm: (1) Current pyramid level stereo images. (2) Initial disparity map.
1. Segment the images into horizontal stripes
   - (a) Divide disparity map into horizontal stripes
   - (b) Recursively merge neighbouring stripes until no neighbouring stripes are similar enough.
2. For each horizontal stripe, segment it into rectangular regions
   - (a) Divide horizontal stripe into vertical stripes
   - (b) Use similar merging technique as in merging horizontal stripes
3. Obtain the corresponding regions in the right image using the disparity information.

3 Maximum-Surface in 3D

From the previous section, we have obtained a 3D cross correlation coefficient volume using fast cross correlation working with 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. 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. The maximum-surface gives the maximum sum 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 method is to obtain an accumulated intermediate volume in the vertical direction. Assume $C(i,j,d)$ is the correlation coefficient value in the input 3D volume at position (i,j,d), where $0 \leq i < M, 0 \leq j < N$, 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 along all the possible surfaces in the same volume using dynamic programming techniques in the vertical direction say from top to bottom, i.e. when i changes from 0 to M-1. For the top horizontal slice of the volume, i.e. when $i = 0$,

$$Y(0,j,d) = C(0,j,d) \tag{1}$$

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:

$$Y(i,j,d) = C(i,j,d) + \max_{t:|t|\leq p} Y(i-1,j,d+t) \tag{2}$$

where p determines the number of local values that need to be checked. If $p = 1$, only three 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. (2) only happens in the (i,d) plane for each particular j as shown in Figure 3. Figure 3(a) is the 3D correlation coefficient volume; and Figure 3(b) shows the positions of neighbouring Y values for one plane during the recursion. 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. In this paper we will just use $p = 1$.

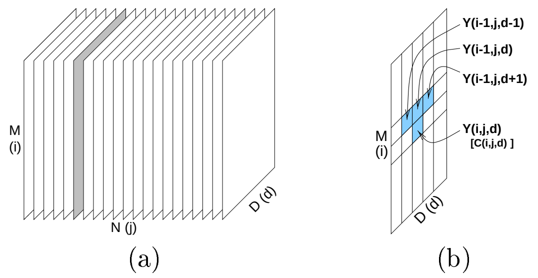

Figure 3: *Obtaining the $Y(i,j,d)$ volume. (a) shows the 3D volume Y with a vertical slice in grey; (b) illustrates the positions of the Y values at each iteration.*

After the recursion described in the previous paragraph, Y contains the maximum sum of C in the vertical direction from top to bottom of the 3D volume. We now move to a second stage of dynamic programming using volume Y to obtain the disparity map for the input stereo images. Starting from the bottom of the 3D volume Y, we select the 2D horizontal slice with $i = M$-1. 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. The sum of the values along this path gives the maximum value. This obtained path is related to the disparities for the last or bottom row 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 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 obtained from row number i. Then a new shortest-path is obtained in this 2D matrix from left to right which are constrained to lie inside this region. This process of obtaining shortest-path is repeated until the disparity for the first row of the image is obtained.

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 surface gives more consistent disparities. The complexity of the STDP algorithm is linear with respect to the size of the 3D volume, i.e. $O(MND)$. The algorithm steps for the STDP matching are:

1. Input: the 3D volume of similarity measurement
2. Stage One: vertical dynamic programming to obtain the intermediate volume Y in the vertical direction:
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 map
   (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, check to see whether all the slices have been matched, if not, go to Step 3b; 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 pyramid data structure approach to stereo matching is faster than one without multi-resolution [22], as the search range in each level 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 image scene. The details can be found down the pyramid at higher resolution.

In the current implementation, the lower resolution image is obtained by simply taking the average value of the corresponding $r \times r$ pixels in the previous higher resolution level for its simplicity. 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 step of obtaining the disparity map from the 3D volume, the computation complexity 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 regions works in the coarse-to-fine scheme.

4.2 Algorithm Steps

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

1. Build image pyramids with K levels (from 0 to $K-1$), with the reduction ratio of r, from the original left and right images; The upper or coarse resolution levels are obtained by averaging the corresponding $r \times r$ pixels in the lower or finer resolution level;
2. Initialize the disparity map as zero for level $k = K-1$ and start stereo matching at this level;
3. Perform image 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 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; otherwise stop.

5 Experimental Results

This section shows some of the results obtained using the method 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 so that cross eye viewing becomes easier. Implementations of the Roy's [29], Cox's [11] and Sun's [33] methods are used for comparison. The codes for Roy's and Cox's methods are downloaded from their web pages.

Synthetic Images

Figure 4 gives the results of algorithms in [33] and our new algorithms running on a pair of synthetic images. The top row shows the input left and right images. Figure 4(c) is the result obtained using our earlier method presented in [33]. Figure 4(d) shows the result using the method described in this paper. Shown in the top row are images of a concrete sphere on a table. The size of this pair of images are 256×256. It can be seen from this figure that our new 3D maximum surface method using TSDP gives better results.

Random dot stereogram (RDS)

A pair of random dot stereogram images are shown in Figure 5(a,b). The stereo matching results for our new method, Roy's, Cox's and Sun'97 methods are given in Figure 5(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 image for these four methods are at the discontinuity boundaries.

Real Images

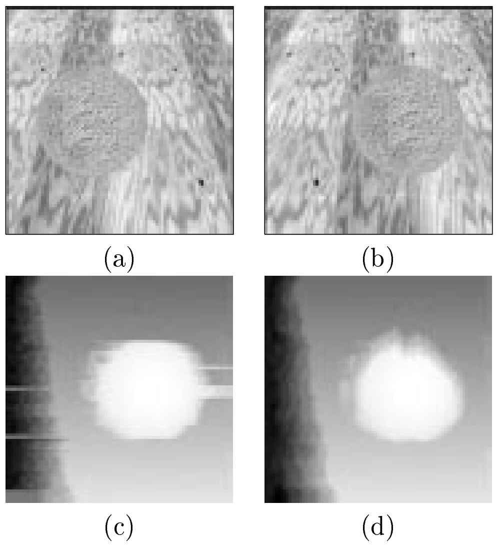

Figure 4: *The matching result for a pair of synthetic images. The image sizes for the first row are 256×256. The top row gives the images of a sphere on a table. (a) right images; (b) left images; (c) the disparity maps recovered using method in [33]; and (d) the disparity maps recovered using our new method. (Images (a,b) courtesy of Bill Hoff at the University of Illinois [16]. Images (e,f) courtesy of Computer Vision Group, Computer Science III, University of Bonn)*

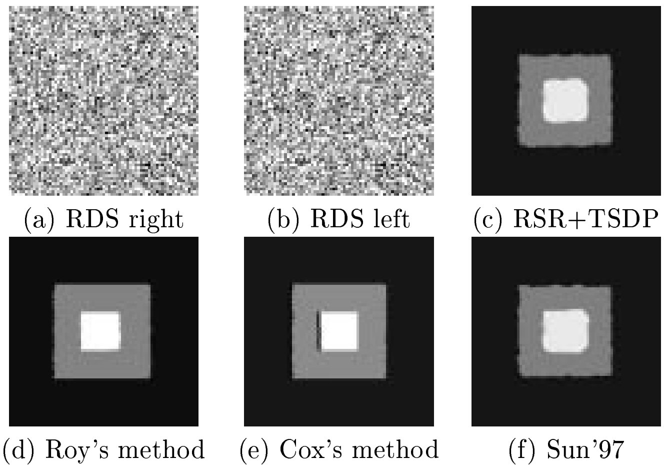

Figure 5: *Synthetic random dot stereogram. (a) and (b) are the right and left input images. (c) shows our result using the algorithm described in this paper; (d) result obtained using Roy's method; (e) result obtained using Cox's method; and (f) result obtained using Sun's method described in [33].*

Figures 6-8 show some results using real images.

Park meter: The input images shown in Figure 6(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 6(c,d,e,f).

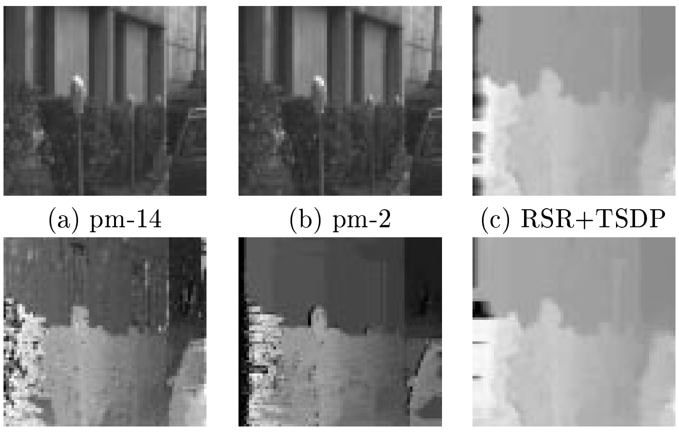

Figure 6: *Park meter scene. (a) and (b) are the right and left input images. (c) Results obtained using our new method (RSR+TSDP). (d) Results obtained using Roy's method. (e) Results obtained using Cox's method. (f) The matching results using the method described in [33]. (Images (a,b) courtesy of CMU).*

Pentagon: The input images are shown in Figure 7(a,b). The matching results for our new method, Roy's, Cox's and Sun'97 methods are given in Figure 7(c,d,e,f).

Fruit scene: The input images are shown in Figure 8(a,b). The matching results for our new method, Roy's, Cox's and Sun'97 methods are given in Figure 8(c,d,e,f).

From the results shown in Figures 6, 7 and 8, it can be seen that our new RSR+TSDP method gives better results than the other three methods. Many other types of real images have also been tested, and good results have been obtained. Figure 9 gives some of the results obtained by using our new RSR+TSDP method described in this paper. The image shown in Figure 9(a) is a picture of softball on newspaper. Figure 9(b) shows a bent circuit board. Figure 9(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 is in the order of several hundred milliseconds. Table 1 gives some of the typical running times of the algorithm on different size of images with different disparities using whole image

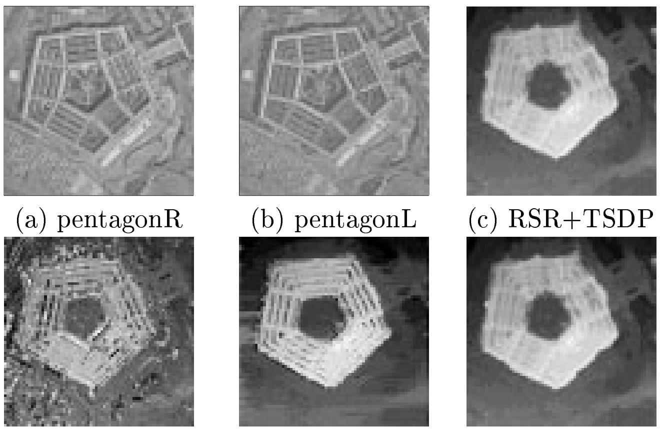

Figure 7: *Pentagon stereo. (a) and (b) are the right and left input images. (c) Results obtained using our new method (RSR+TSDP). (d) Results obtained using Roy's method. (e) Results obtained using Cox's method. (f) The matching results obtained using the method described in [33]. (Images (a,b) courtesy of Bill Hoff at the University of Illinois [16]).*

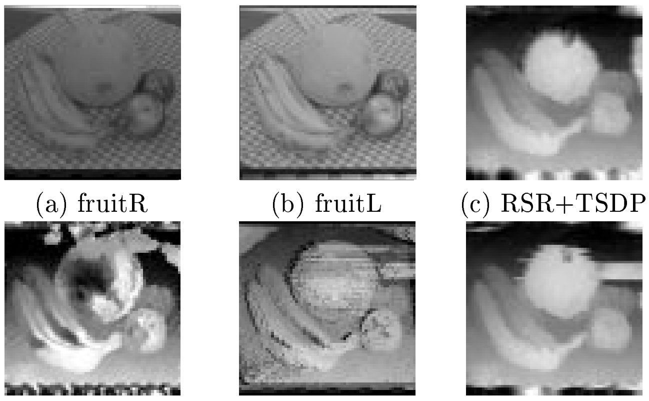

Figure 8: *Fruit stereo. (a) and (b) are the right and left input images. (c) Results obtained using our new method (RSR+TSDP). (d) Results obtained using Roy's method. (e) Results obtained using Cox's method. (f) The matching results obtained using the method described in [33]. (Images (a,b) courtesy of Bill Hoff at the University of Illinois [16]).*

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

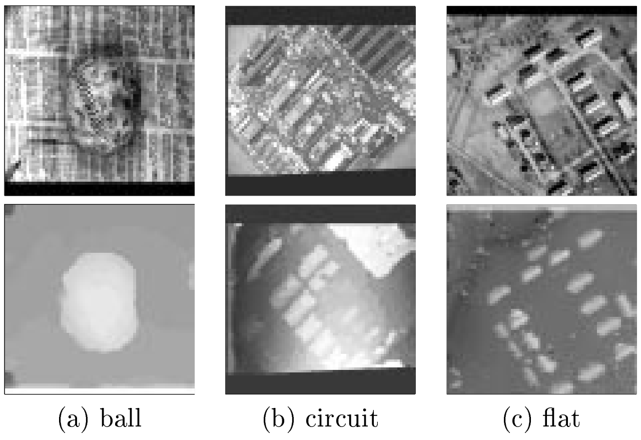
Figure 9: *Three images (only left) and the disparity maps obtained. The top row gives the left images of stereo pairs. The bottom row shows the disparity map. The image size in (a) is 256×256; The image size in (b) is 512×512; The image size in (c) is 1000×1000. (Input images (a,b) courtesy of Bill Hoff at the University of Illinois [16]; input images (c) courtesy of Stuttgart ISPRS Image Understanding datasets).*

Figure 9(a), the program only takes 0.32 seconds. It only takes 1.39 seconds to obtain the disparity for the "pentagon" image of size 512×512.

The time shown for "User time1" is obtained without using the RSR method as described in Section 2.1, 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 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 [33] (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 is 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.

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 pyramid levels used are 3 for the first three images and 4 for the last image. The* `ball`, `circuit` *and* `flat` *images are shown in Figure 9. The* `pentagon` *image is shown in Figure 7(a)(b).*

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

Table 2: *Running times of the algorithm 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. The pyramid levels used are 3 for the first three images and 4 for the last image.*

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

Table 3: *Running times of different algorithms. The* `RDS` *image is shown in Figure 5(a,b). The* `pm` *image is shown in Figure 6(a,b). The* `pentagon` *image is shown in Figure 7(a,b).*

Image name	**Image size**	**Disp. 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

6 Reliability and Speed

The reliable results of our algorithm are achieved by applying the combination of the following techniques: (1) Coarse-to-fine strategy is used. (2) The ZNCC similarity measure is used, which is independent of differences in brightness and contrast. (3) The correlation coefficient value is used as input to the dynamic programming stage. (4) Dynamic programming technique is used to find a 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 maximum surface method takes all the information into account, rather than work individually for each of the epipolar lines.

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 developed further for fast calculation of image variance at the same time when one calculates the image mean. (2) We have used a rectangular subregioning technique for fast computation of correlation coefficients. (3) Apart from having the advantages of increasing the reliability, the coarse-to-fine approach is also faster than one without using it. (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 [30, 9], 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 maximum-surface techniques in the coarse-to-fine framework. 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 storage space. The 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 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 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.

References

[1] P. Anandan. 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, August 1987.

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

[3] R. Baldwin, H. Yamada, and K. Yamamoto. Disparity space and dynamic programming for automatic production of very dense range maps. In A. Gruen and E. Baltsavias, editors, *Close-Range Photogrammetry Meets Machine Vision*, volume 1395, pages 217–225, Zurich, Switzerland, 3-7 September 1990. SPIE.

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

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

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

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

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

[9] Q. Chen and G. Medioni. Building human face models from two images. In *Multimedia Signal Processing*, Redonda Beach, CA, 1998.

[10] http://www.ri.cmu.edu/projects/project_53.html.

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

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

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

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

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

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

[17] S. Intille and A. Bobick. Disparity-space images and large occlusion stereo. In *Proceedings of European Conference on Computer Vision*, Stockholm, Sweden, 1994.

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

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

[20] M. I. Kolesnik. Fast algorithm for the stereo pair matching with parallel computation. In D. Chetverikov and W. G. Kropatsch, editors, *5th International Conference on Computer Analysis of Images and Patterns*, pages 533–537, Budapest, Hungary, September 13-15 1993. Springer-Verlag.

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

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

[23] Y. Ohta and T. Kanade. Stereo by intra- and interscanline search using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7:139–154, March 1985.

[24] http://www.ptgrey.com/.

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

[26] M. Rechsteiner, B. Schneuwly, and G. Troester. Dynamic workspace monitering. In H. Ebner, C. Heipke, and K. Eder, editors, *International Archives of Photogrammetry and Remote Sensing*, volume 30, pages 689–696, Munich, Germany, September 1994.

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

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

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

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

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

[32] http://www.ai.sri.com/~konolige/svs/.

[33] C. Sun. A fast stereo matching method. In *Digital Image Computing: Techniques and Applications*, pages 95–100, Massey University, Auckland, New Zealand, December 10-12 1997.

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

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

[36] Q. X. Wu. 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, September 1995.

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

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