Towards a fast and reliable dense matching algorithm

B. Boufama and K. Jin
SCHOOL OF COMPUTER SCIENCE
University of Windsor
Windsor, Ontario
Canada N9B 3P4

Abstract

In this paper, we present a dense matching algorithm which utilizes the corner and edge features of images to increase the reliability and to speed up the process of dense matching of two uncalibrated images. The major problem of classical area-based dense matching algorithms is the high computational time resulting from intensive correlation calculations for match candidates. Although some methods have attempted to integrate image feature information in the dense matching of uncalibrated images, most of these methods are not practical and are difficult to implement. This paper aims at designing a hybrid matching algorithm that preserves disparity continuity at the object continuous surfaces while discontinuity at object boundaries are treated differently in the matching process. In particular, both CPU-time and likelyhood of mismatches are reduced while the implementation is kept simple and straightforward.

keywords: dense matching, stereo vision

1 Introduction

The problem of establishing correspondences of pixels and other features across images represents a fundamental problem in computer vision. This is a very difficult task due to several reasons, in particular, not every feature in one image has a match in the other one. Although a large amount of work has been carried out over the past two decades on the matching problem [8, 3], dense matching remains unsolved and viewed as the bottleneck for many applications, such 3D reconstruction and view synthesis. Because stereo matching is inherently complicated and noise sensitive, classical approaches either were limited to a sparse matching or made additional assumptions. Sparse matching methods, also called feature-based methods, extract some selected features (e.g., corners, lines) in each image then attempt to establish the matching[11, 9]. The main drawback of these methods is the sparseness of the recovered information. On the other hand, dense matching aims at matching each and every pixel in one image with its correspondence pixel in the other image. Most of these methods are based on correlation techniques[2] to evaluate region similarities. However, because correlations are calculated intensively in the matching process, these methods are too slow. Constraints can be used to reduce the search for a match in the other image. The most popular one is the epipolar constraint resulting from the calculation of the fundamental matrix [6, 4]. Although this constraint reduces the search to one-dimension instead of two, the process is still very slow. The matching process can be made fast if two more constraints, the continuity and order constraints, can be enforced. In that case, the matching will be done pixel by pixel on each epipolar line, where each pixel's disparity will be dependent on its previous neighbor reducing the search to a minimum. However, these two constraints are not always satisfied in stereo images. In particular, the order might be violated with transparent(or overlapping) objects and the continuity does not hold at object physical boundaries. The order constraint is less important since it does not have a significant effect on the CPU-time of the whole matching process. On the other hand, the continuity constraint is far more important and, together with the epipolar constraint, is responsible for reducing most of the CPU-time. The satisfaction of the continuity/discontinuity constraints was addressed by several researchers. In particular, some recent energy-based methods[1, 10] solve the correspondence problem as a minimization problem that preserve discontinuities at object boundaries. However, the process is iterative and is far from straightforward to implement. Furtrhermore, the CPU-time was not addressed as part of the problem.

The method we present in this paper addresses the dense matching problem in the case of uncalibrated stereo images. We make use of the calculated epipolar geometry and the continuity constraints. The latter is used on homogeneous region while it is relaxed at region boundaries. Unlike energy-based methods, our method aims at decreasing CPU-time while it remains simple and straightforwards. Images are segmented into two sets of regions, the boundary regions and the nonboundary regions. The continuity constraint is enforced for the latter and relaxed for the former. In addi-

tion, simple correlation is used as a similarity measurement.

2 Dense matching algorithm

2.1 A basic algorithm

Before designing the targeted version of our matching algorithm, we started with a basic one which consists of three steps and can be summarized as follows. (1) Corners are extracted from both images then matched using ZNCC[1] correlation. Given a corner in the left image, the search for its corresponding corner in the right image was limited to a small set of corners within the search window. Only matched corners with very high correlation values were kept to ensure reliability. (2) This small set of matched corners is used to calculate the epipolar geometry[7]. Although this first step is crucial, its CPU-time was not significant, and can be neglected, compared to the dense matching of the whole image. (3) Finally, for each pixel in the left image we search for its match over its corresponding epipolar line and within some search window.

From our experiments and as predicted, this simple algorithm had two major drawbacks: the matching was too slow and there were too many mismatches. These two problems are indeed related. If the search area is reduced, then both the cpu-time and the likelyhood for a mismatch will be decreased. The search area can be reduced if each pixel in the left image possesses a good guess for its disparity. This disparity guess can simply be the disparity of the closest matched corner since such interest points are more reliable in matching. However, the disparity guess based on image interest points is not always a good guess. For instance, given a pixel located on an object, the closest corner(interest point) to this pixel may be actually located on a different object and therefore, it might have a very different depth in the scene. The search region that will be guessed based on this interest point's disparity might not include the correct match.

2.2 A new hybrid algorithm

Because disparity guesses based only on interest points such as corners proved to be nonefficient, we have considered here using another important type of image features: the edges. Edges have advantages over interest points since the borders of objects in a scene generate edges in most cases. The knowledge of the edge information allows us to locate the area where possible sharp disparity changes may occur. By segmenting the image into edge and non-edge areas, we can apply different matching strategies over these two types of areas for optimal results.

[1]Zero Mean Normalized Cross Correlation

A flow diagram of our approach is shown in Figure 1. First, edges are extracted from the left image which is divided into two sets of regions, edge and non-edge regions. Then, we match all pixels belonging to the edge regions of the left image to their corresponding pixels in the right image without the continuity constraint. Finally, non-edge regions are matched using another algorithm that enforces the continuity constraint.

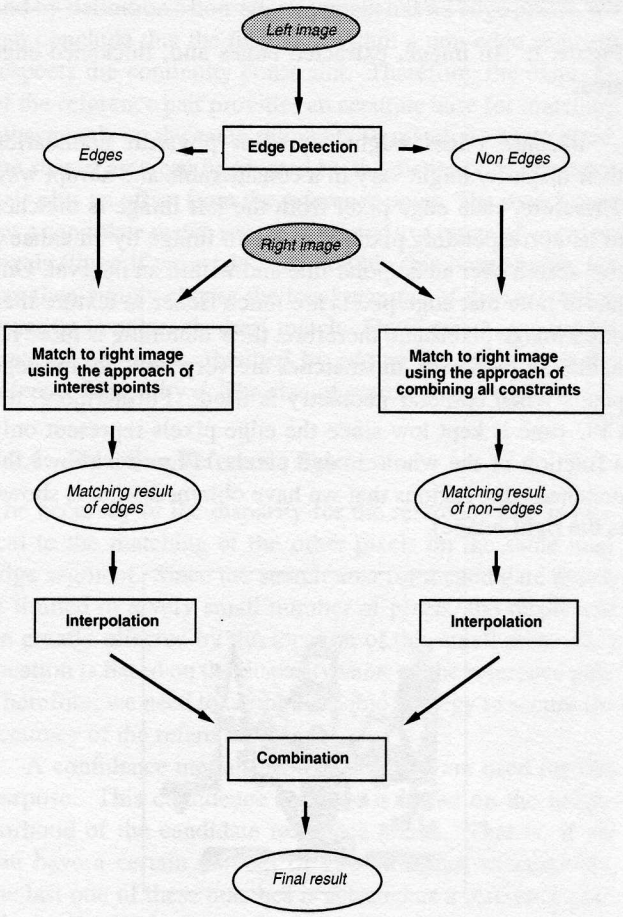


Figure 1: The flow diagram of our hybrid approach.

2.2.1 Matching Edge Area

We have extracted edges from our images using Deriche's edge detector [5], downloaded from the INRIA web site. Because the edge detection process might miss some edge points, we have postprocessed the obtained edges by a simple thickening process. This will ensure that at least all boundary pixels, even the ones missed by the edge detector, are included. It is safer to label a non-edge pixel as an edge pixel than the other way around. Figure 2 shows a test image, the output of Deriche's edge detector and, the thickened edge area we extracted from the image. Note that the CPU-time overhead for edge detection is negligible compared to

the amount of CPU-time for the whole matching process.

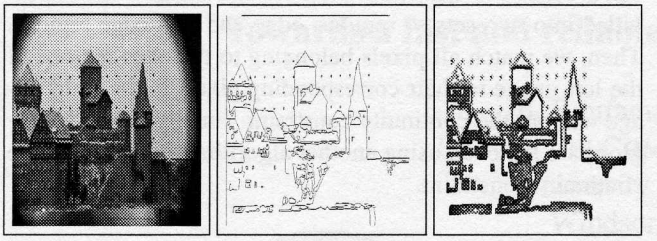


Figure 2: An image, extracted edges and, thickened edge areas.

Because edges might represent physical boundaries, their disparity might vary in a considerable and abrupt way. Therefore, each edge pixel from the left image is matched to its corresponding pixel in the right image by an exhaustive search over an epipolar line and within an interval. One has to note that edge pixels are much richer in texture than other image pixels and therefore, their matching is more reliable. In particular, mismatches are very unlikely for edge pixels when epipolar geometry is used. Furthermore, the CPU-time is kept low since the edge pixels represent only a fraction of the whole image pixels. Figure 3 shows the matched edge regions that we have obtained(what is shown is the right image).

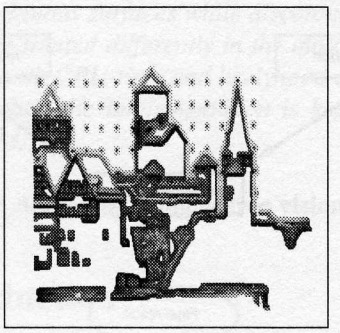


Figure 3: Matching result of the edge regions.

2.2.2 Matching Non-edge Area

The non-edge regions are the remainder of the image, obtained by subtracting edge regions from the original image. To match non-edge regions in the left image, we first need to introduce the term of non-edge segment. Since the matching is carried out along epipolar lines, we define a non-edge segment as a sequence of non-edge pixels on an epipolar line, delimited by two edge areas. As shown on Figure 4, white areas represent the edge regions and gray areas the non-edge regions. Consider the points A, B, C, D, E and G along an epipolar line that crosses the whole image. One can note that we have three non-edge segments; namely the segments AB, CD and, EG.

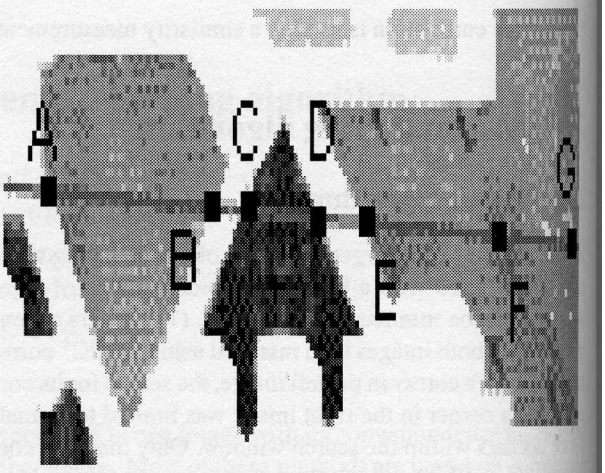


Figure 4: Non-edge segments : definition

Since every pixel from non-edge regions belongs to a certain non-edge segment, the matching of non-edge regions of the image can be viewed as the matching of all non-edge segments on all the epipolar lines. Obviously the matches of the pixels of a non-edge segment must lie on the same corresponding epipolar line in the other image. Furthermore, since non-edge regions consist of smooth surfaces without abrupt disparity changes, we can enforce all the following constraints: order, continuity and uniqueness. As a consequence, the search area for the matching is drastically reduced. In addition, the likelihood of a mismatch becomes extremely small.

2.2.3 Enforcing All Constraints

Consider the matching process of a non-edge segment L of the left image to its corresponding pixels in the right image. Let $L[N]$ be an array of pixels denoting the N pixels of segment L and let R be the corresponding epipolar line in the right image. If we have a matched pair of pixels such as

$$L[i] \longleftrightarrow R[j]$$

where we refer to $L[i]$ as the *reference point*, its match $R[j]$ the *reference match*, and $(L[i], R[j])$ the *reference pair*. We have the following

$$L[i+1] \longleftrightarrow R[j+1].$$

The above says that the reference point's right immediate neighbor on the epipolar line matches the immediate neighbor right of the reference match, while ignoring the scaling that occurs in an image.

As the order constraint states, the order of matching is preserved along the epipolar line, which means that a pixel p located right of the reference point must be matched to a pixel located right of the reference match. Moreover, since

we are matching non-edge regions, where disparity varies smoothly, the disparity of p is almost the same as the disparity of the reference point. Therefore, the match of p should be a neighbor of the reference match on the same side.

However, considering scale changes of the image, we need to add a margin d , so that the matching relationship becomes:

$$L[i+1] \longleftrightarrow R[j+1+d],$$

where $d \in 0, 1, ...k$.

This d represents a small neighborhood around the reference match. We choose $k = 3$ which means that the reference point's immediate neighbor should fall into the region of less than 3 pixel on the same side of the reference match. This is a threshold that depends on how much the second image has stretched or shrunk with respect to the first one.

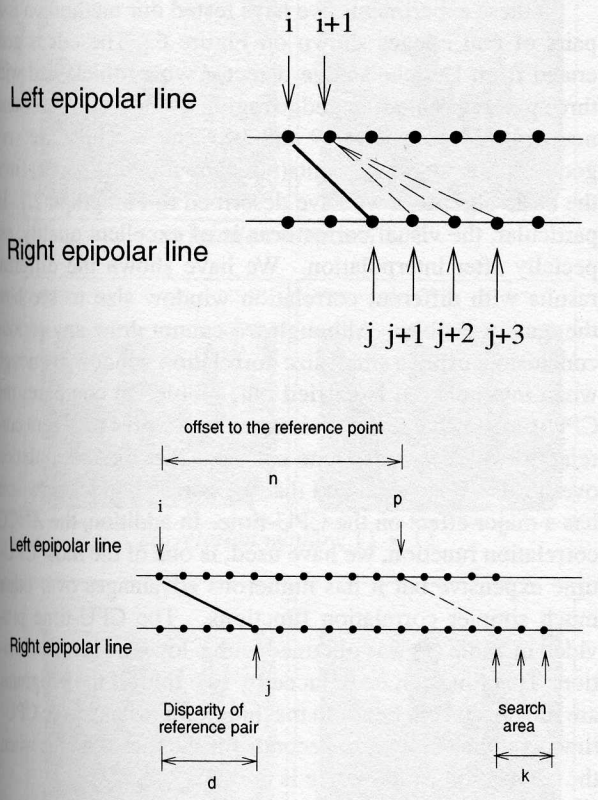


Figure 5: Using the epipolar, continuity and order constraints to match non-edge segments.

As shown in Figure 5, if pixels i and j represent a reference pair, then the candidate match of pixel $i+1$ must be around the right neighborhood of j lying of the epipolar line. Using our threshold $k = 3$, the match of $i+1$ belongs to the three-element set $\{j+1, j+2, j+3\}$.

Now, we can apply this method to the n'th neighbor of the reference point within the same non-edge segment as follows:

Given a pair of matched pixels

$$L[i] \longleftrightarrow R[j],$$

we have

$$L[i+n] \longleftrightarrow R[j+n+d],$$

where $L[i+n], L[i] \in$ the same non-edge segment.

Since the sharp disparity changes only occur at edges and by definition a non-edge segment has no edge pixels, we can conclude that the disparity within a non-edge segment respects the continuity constraint. Therefore, the disparity of the reference pair provides an accurate base for matching other pixels on the same segment. To match a certain pixel, the search area can be located by the disparity of reference pair plus its offset from the reference point. The search area for a candidate match is now restricted to a range of very few pixels (three if we set $k = 3$). Within this limited area, we can then simply choose the local maxima of the correlation function to select the best match. On Figure 5, the search area for point p is obtained by adding the offset n to the reference disparity d. The size of search area is equal to k.

2.2.4 Selecting Reference Pair

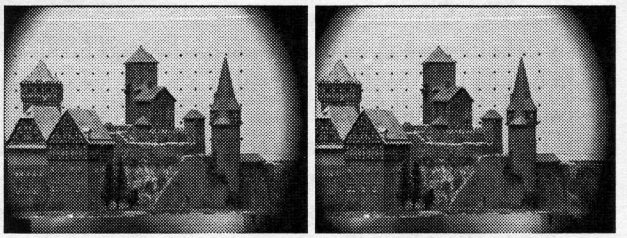
The accuracy of the disparity for the reference pair is critical to the matching of the other pixels on the same non-edge segment. Since the search area for a candidate match is limited to a very small number of pixels, the result will be greatly effected by the location of this small area. This location is based on the disparity value of the reference pair. Therefore, we need to adopt a reliable strategy to secure the accuracy of the reference pair.

A confidence measure and a threshold are used for this purpose. This confidence measure is based on the neighborhood of the candidate reference match. That is, if we can have a certain number of good matches successively, the last one of these matches is selected as a reference pair. The measure of a good match is the threshold value of the correlation score.

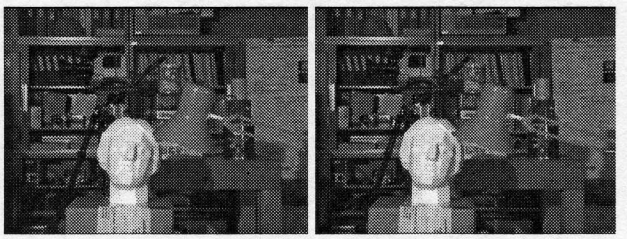
The selection of a reference pair is required in two situations. First at the beginning of each non-edge segment because a reference pair should be determined for matching the following other pixels. However, the reference pair needs to be refreshed to avoid accumulated misalignments. The smoothly changing disparity could be accumulated over a certain range and may cause the search area to be drifted away from the correct match. Therefore, over a wide non-edge segment, we do not use the same reference pair. Instead, a new reference pair is established each time the reference pixel becomes far away from the current pixel to be matched. This distance represents another threshold that we can use to avoid the drifting phenomena. An example is shown in Figure 4. The initial reference point for the non-edge segment EG is E. However, because EG is too long,

we introduced another reference point F. Although F is not the beginning of the non-edge segment, it is used to refresh the reference disparity.

3 Experimental Results



Castle scene.



Head scene.

Tree scene

Meter scene

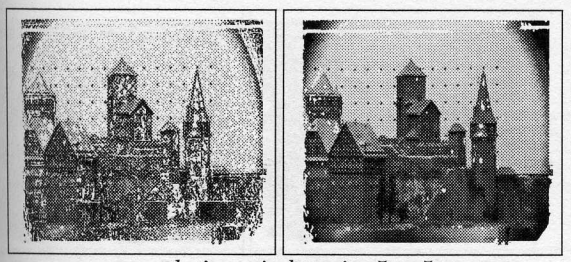
Figure 6: The four test image pairs we have used.

The results of any matching algorithm will be difficult to evaluate in the absence of ground truth. This is always the case when real images are used. In these experiments we have chosen to show the results by re-constructing the right image using the left image pixels, the calculated disparities and the image grey level values. Although this might not be the best way to demonstrate the accuracy of the matching, it is a good way to show the visual correctness of the matching results. In particular, should the calculated matching contain a lot of mismatches, the re-constructed image will not look coherent. Therefore, our results are demonstrated by displaying the re-constructed right image. In addition, we have added an interpolation step to fill-up the gaps for visual purposes only. These gaps resulted from nonmatched pixels because of their low correlation scores or, because of other reasons. The interpolation process we have used was a very simple one. When a pixel in the re-constructed image does not have a match(such a pixel represents a hole) while its immediate neighbors have matches, its grey-level value is interpolated from these immediate neighbors(at least two immediate neighbors are required). This simple interpolation process is repeated a few times until no such pixels remain. Therefore, the results of our matching algorithm are shown with two images. The re-constructed right image and its enhanced version after the interpolation process.

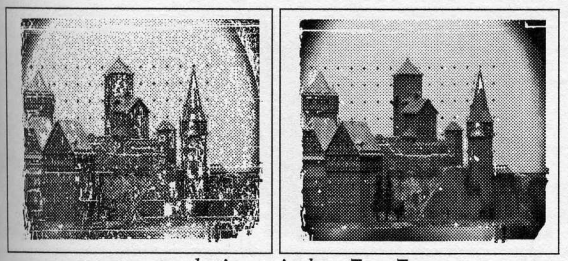
In these experiments, we have tested our method on four pairs of real images shown on Figure 6. The edges generated from Deriche's edge detector where thickened with three pixels to obtain our edge regions. The results are summarized in Figures 7, 8, 9 and 10. These results are very good compared to the results(not shown in this paper) from the basic algorithm we have described in Paragraph 2.1. In particular, the visual correctness is of excellent quality, especially after interpolation. We have shown the different results with different correlation window size to see how the quality evolves. Although we cannot draw any serious conclusion, using a small size correlation window is enough when interpolation is carried out. Table (1) compares the CPU-time for the different images and for the different correlation window sizes. One can note that the interpolation overhead is very small and that the correlation window size has a major effect on the CPU-time. In addition, the ZNCC correlation function, we have used, is one of the most CPU-time expensive but it has numerous advantages over other much simpler correlation functions. The CPU-time provided in Table (1) was obtained with a low-end sun workstation. This time can be reduced by two third if the programs are run on a recent Pentium machine. For instance, the CPU-time would be only three seconds for the **meter** sceme when the correlation window size is 5×5.

4 Conclusion

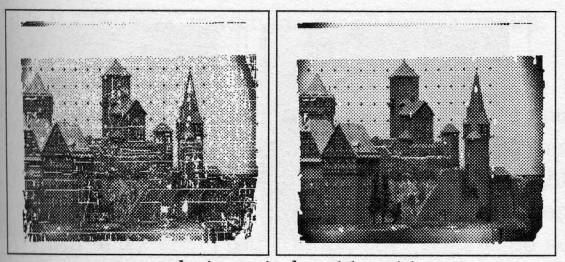
In this paper, we proposed a fast algorithm for dense matching of uncalibrated images. The algorithm utilizes image edge features to increase reliability and to speed-up processing time. Because edges in an image reveal a lot of information on the image structure and indicate locations of possible abrupt disparity changes, they have allowed us to segment an image into two sets of regions: edge regions and non-edge regions. Thus, two different approaches for matching were



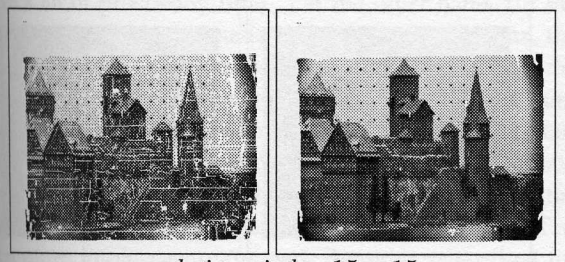
correlation window size 5×5



correlation window 7×7

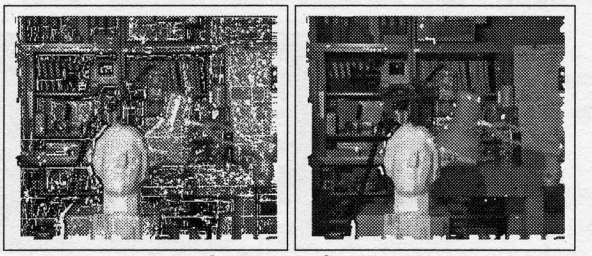


correlation window 11×11

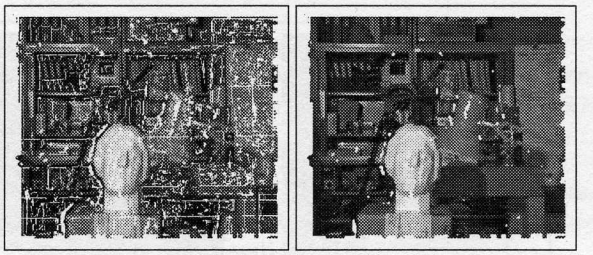


correlation window 15×15

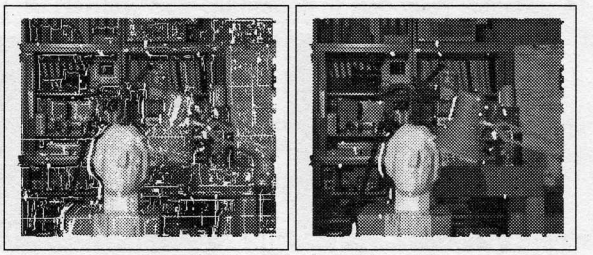
Figure 7: Matching results using hybrid approach for Castle scene; without and with interpolation.



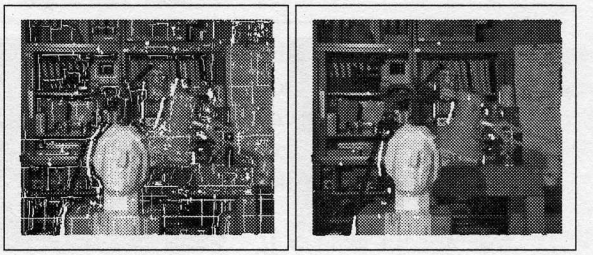
correlation window 5×5



correlation window 7×7

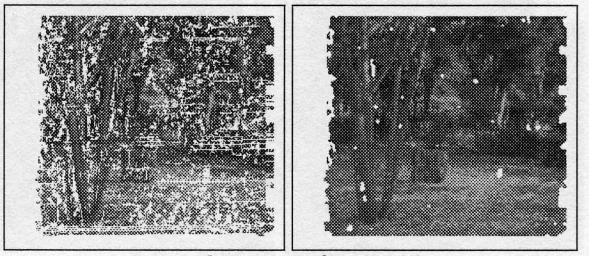


correlation window 11×11

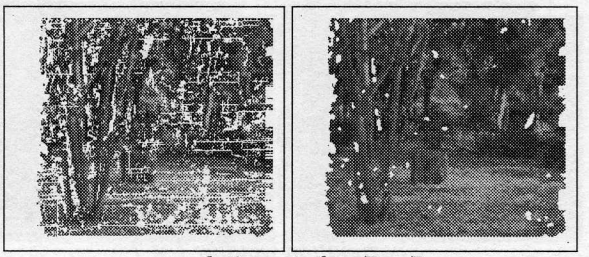


correlation window 15×15

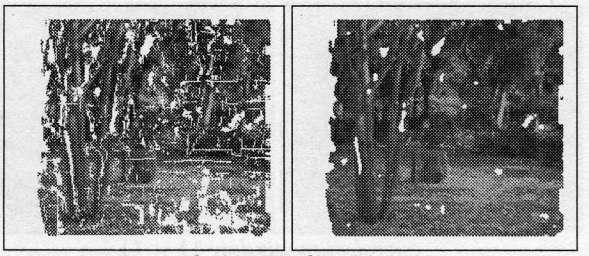
Figure 8: Matching results using hybrid approach for Head scene; without and with interpolation.



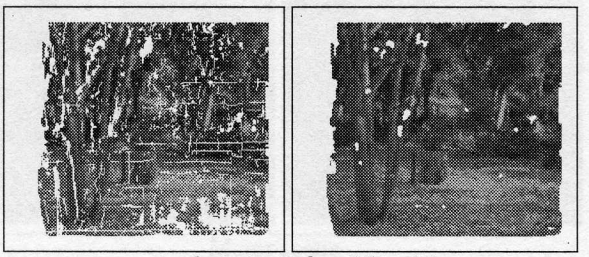
correlation window 5×5



correlation window 7×7

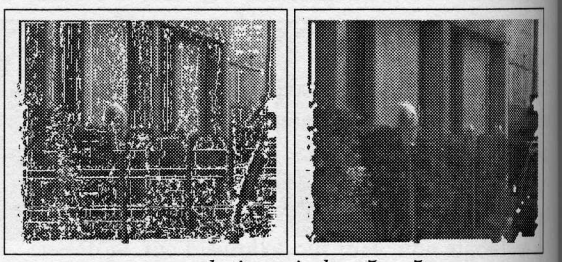


correlation window 11×11

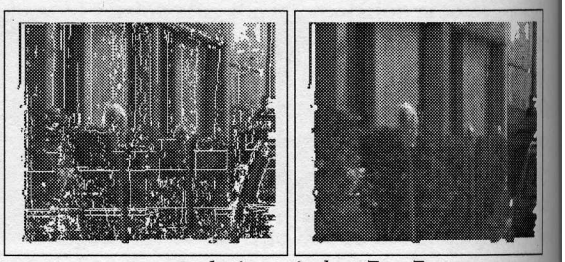


correlation window 15×15

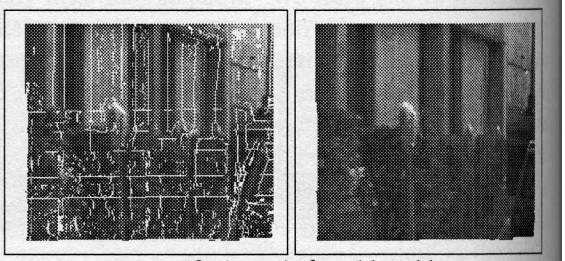
Figure 9: Matching results using hybrid approach for Tree scene; without and with interpolation.



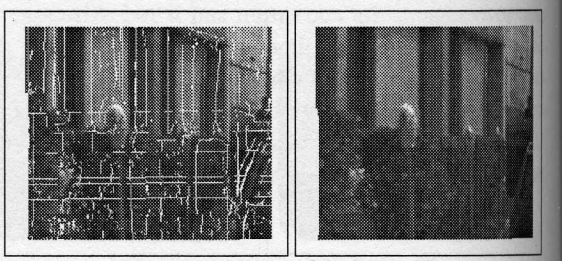
correlation window 5×5



correlation window 7×7



correlation window 11×11



correlation window 15×15

Figure 10: Matching results using hybrid approach for Meter scene; without and with interpolation.

Castle scene (image size 576 × 384)

	correlation window size			
	5 × 5	7 × 7	11 × 11	15 × 15
no interpolation	53s	77s	113s	171s
interpolation	54s	78s	114s	172s

Head scene (image size 384 × 288)

	correlation window size			
	5 × 5	7 × 7	11 × 11	15 × 15
no interpolation	27s	41s	72s	119s
interpolation	27s	41s	73s	120s

Tree scene (image size 256 × 233)

	correlation window size			
	5 × 5	7 × 7	11 × 11	15 × 15
no interpolation	12s	21s	46s	77s
interpolation	12s	21s	47s	78s

Meter scene (image size 256 × 240)

	correlation window size			
	5 × 5	7 × 7	11 × 11	15 × 15
no interpolation	9s	14s	28s	44s
interpolation	9s	14s	29s	45s

Table 1: CPU-time, in seconds, for all four examples.

used for these two different sets of regions. The edge regions were matched using correlations and the epipolar constraint only. This has proven to be sufficient to obtain very reliable matching results on the edge regions because these regions are very rich in texture. Furthermore, edge regions do not represent most of the image pixels and therefore, do not involve a lot of processing time. On the other hand, for the rest of the image that is made up of non-edge regions, the matching approach used correlations together with all the three constraints; the epipolar, continuity and, order constraints. Enforcing all these constraints on non-edge regions allowed us to achieve at least two goals. First, the likelihood of a mismatch to happen was drastically reduced because the search for a match in the other image was reduced to a few pixels(three in our case). Second, the processing time for the search was reduced as well since the correlation scores were calculated on three pixels only, resulting in a fast dense matching process. Furthermore, the algorithm was kept simple and straightforward to implement.

References

[1] L. Alvarez, R. Deriche, J. Sanchez, and J. Weickert. Dense disparity map estimation respecting image discontinuities: A pde and scale-space based approach. Technical Report 3874, INRIA, 2000.

[2] P. Aschwanden and W. Guggenbühl. Experimental results from a comparative study on correlation-type registration algorithms. In Förstner and Ruwiedel, editors, *Robust Computer Vision*, pages 268–282. Wichmann, 1992.

[3] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[4] B. Boufama and R. Mohr. A stable and accurate algorithm for computing epipolar geometry. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(6):817–840, 1998.

[5] R. Deriche. Using Canny's criteria to derive a recursively implemented optimal edge detector. *International Journal of Computer Vision*, 1(2):167–187, 1987.

[6] R. Hartley. In defence of the eight-point algorithm. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593, 1997.

[7] R. Hartley. Self-calibration of stationary cameras. *International Journal of Computer Vision*, 22(1):5–23, 1997.

[8] T. Kanade. Development of a video rate stereo machine. In *Proceedings of the ARPA Image Understanding Workshop*, pages 549–558, November 1994.

[9] J. H. McIntosh and K. M. Mutch. Matching straight lines. *Computer Vision, Graphics and Image Processing*, 43(3), 1998.

[10] L. Robert and R. Deriche. Dense depth map reconstruction: A minimization and regularization approach which preserves discontinuities. In *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 439–451, April 1996.

[11] Z. Zhang, R. Deriche, O.D. Faugeras, and Q.T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1-2):87–119, 1994.