

# Efficient and Accurate Subpixel Path Based Stereo Matching

Arturo Donate<sup>1</sup> Ying Wang<sup>1</sup> Xiuwen Liu<sup>1</sup> Emmanuel Collins<sup>2</sup>

<sup>1</sup>Department of Computer Science <sup>2</sup>FAMU-FSU College of Engineering  
Florida State University Tallahassee, FL 32306  
{donate, ywang, liux}@cs.fsu.edu ecollins@eng.fsu.edu

## Abstract

*This paper presents an efficient algorithm to achieve accurate subpixel matchings for calculating correspondences between stereo images based on a path-based matching algorithm. Compared to point-by-point stereo matching algorithms, path-based algorithms resolve local ambiguities by maximizing the cross correlation (or other measurements) along a path, which can be implemented efficiently using dynamic programming. An effect of the global matching criterion is that the cross correlation at all pixels can contribute to the criterion; since cross correlation can change significantly even with subpixel changes, to achieve subpixel accuracy, it is no longer sufficient to first find the path that maximizes the criterion and then refine to subpixel accuracy. In this paper, by writing bilinear interpolation using integral images, we show that cross correlations at all subpixel locations can be computed efficiently and thus lead to a subpixel accuracy path based matching algorithm. Our results show the feasibility of the method and illustrate the significant improvements over the original path-based matching method.*

## 1. Introduction

The use of stereo images is crucial in computer vision applications where depth perception is required [2]. Stereo vision algorithms rely on the ability to perform accurate point correspondence between the image pair in order for them to be useful [3]. Point correspondence is defined as the problem of finding the accurate location of the same point in a pair of stereo images. Correspondence methods can be divided into region-based ones which rely on matching local windows, and feature-based ones which rely on matching more distinctive features. For dense matching, region-based methods are widely used. A common method is to match the local windows using normalized cross correlation and other matching criteria. To achieve subpixel accuracy, a polynomial, typically a second order one, is used to fit

matching scores in a local neighborhood [1]. One of the intrinsic limitations of point-based matching methods is that it can not resolve local ambiguities effectively. For example, due to low texture variations and other factors, there are often more than one matching candidate in a local region. In addition, local deformations may cause the correct matching one not to be the local maximum, causing many of the algorithms that use local maxima to fail.

To overcome the local ambiguities and achieve more robust matching, a more global matching criterion can be used. In [6], Sun poses the matching problem as an optimization one of the total cross correlation scores over a surface through a 3D cross correlation volume (height, width, and the disparity range of a region) and the matching is solved efficiently using a two-stage dynamic programming algorithm, where cross correlation scores are maximized along paths in the 3D cross correlation volume. Note that by using a global matching criterion, accurate cross correlations are needed at all locations since they affect the optimal path estimation and thus the matching. As cross correlations can change significantly even at subpixel level, in order to achieve optimal matching, subpixel accuracy cross correlations need to be calculated. However, in [6], subpixel accuracy is obtained only in the post processing stage by fitting a quadratic function in a neighborhood. In this paper, we show that subpixel accuracy cross correlations can be computed efficiently using integral images and thus improve the accuracy of the path-based matching significantly, which is supported by experimental results.

The rest of the paper is organized as follows. In Section 2, we summarize the path-based stereo matching algorithm [6] and in Section 3 we show how the subpixel accuracy can be incorporated efficiently using integral images; note that while Sun [6] also performs subpixel matching, it is done after the path matching. As shown by our results in Section 4, our algorithm improves the performance significantly by having more accurate path matching. Section 5 concludes the paper.

## 2. Path-based Stereo Matching

Our work is inspired by an algorithm presented by Sun [6]. In [6], Sun presents a dynamic programming algorithm which uses rectangular subregioning and maximum surface techniques in order to perform path-based matching. The images are first broken down into subregions in order to separate areas with similar disparity ranges. For each subregion, a range of different disparity values is taken into consideration (with the ultimate goal being to determine the optimal disparity value at each location). Using normalized cross correlation (NCC), a 3D correlation coefficient volume of size  $W \times H \times D$  is built for each subregion, where  $W$  and  $H$  are the subregion dimensions and  $D$  is the size of the disparity range.

Given a left image  $f(x, y)$ , a right image  $g(x, y)$ , and a central pixel location in the left image  $(x_0, y_0)$ , we would like to compute the normalized cross correlation using a windows size of  $(2M + 1) \times (2N + 1)$  for a displacement  $(u_0, v_0)$  to be used with the path based matching, proposed by Sun [6]. Here the first dimension (e.g.,  $x, u, x_0, M$ ) refers to the column of the image, and the second dimension (e.g.,  $y, v, y_0, N$ ) refers to the row of the image.

To be specific, let  $\bar{f}(x, y)$  and  $\bar{g}(x, y)$  be the mean of the left and right windows respectively. Similarly, let  $\bar{\bar{f}}(x, y)$  and  $\bar{\bar{g}}(x, y)$  be the variance of the left and right windows. We can define the normalized cross correlation (NCC) between the left image window at  $(x_0, y_0)$  and the right image window at  $(x_0 + u, y_0 + v)$  as  $NCC(x_0, y_0, u, v) =$

$$\frac{\sum_{i,j} \hat{f}(x_0 + i, y_0 + j)g(x_0 + u + i, y_0 + v + j)}{C \sqrt{\bar{\bar{f}}(x_0, y_0)\bar{\bar{g}}(x_0 + u, y_0 + v)}}, \quad (1)$$

where  $\hat{f}(x_0 + i, y_0 + j) = f(x_0 + i, y_0 + j) - \bar{f}(x_0, y_0)$ ,  $C = (2 \times M + 1)(2 \times N + 1)$ , and the summation for  $i$  is from  $-M$  to  $M$  and  $j$  from  $-N$  to  $N$  (also in subsequent equations). As pointed out in [7, 6], for a fixed  $u$  and  $v$ , the summations can be done efficiently using integral images as well as the mean and variance of a local window. For the variance, note that  $\bar{\bar{f}}(x, y) = (1/C)(\sum_{i,j} f^2(x_0 + i, y_0 + j)) - (\bar{f}(x, y))^2$ , and thus it can also be done efficiently using an integral image with pixel values squared.

Given this 3D volume of correlation coefficients, Sun employs a two stage dynamic programming algorithm to find the best surface across the volume and obtain a smooth set of disparity values. The first stage of the dynamic programming algorithm is to separate the volume vertically and calculate an intermediate 3D volume in the vertical direction for each vertical section. Given the original 3D volume of  $NCC$ , the intermediate 3D volume  $Y$  is calculated ac-

ording to the equation

$$Y(x, y, d) = NCC(x, y, x, y + d) + \max_{t:|t| \leq p} Y(x - 1, y, d + t) \quad (2)$$

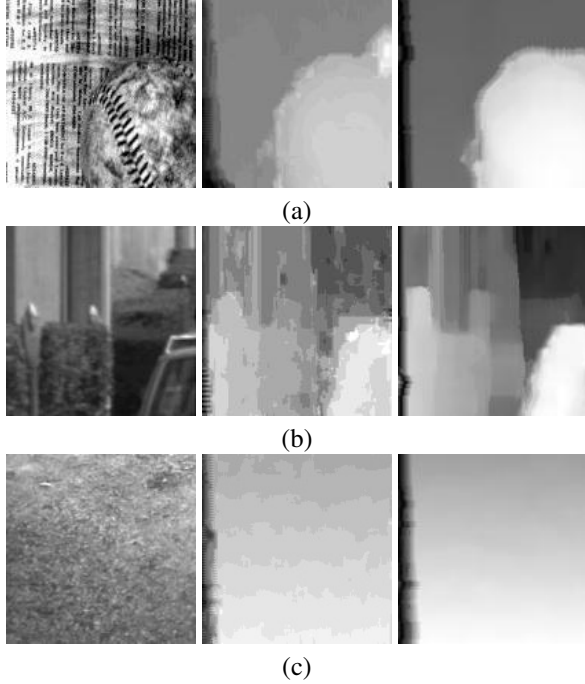
where  $p$  determines the number of local values to be considered (eg., if  $p = 1$ , three locations are considered). Note that the stereo pair is assumed to be rectified, i.e., the corresponding row in the left image matches with that of the right image and thus the disparity is specified by  $d$ . The 3D volume of  $Y$  then contains the maximum summation of correlation coefficients in the vertical direction. The second stage of the algorithm works in the horizontal direction calculating the path from the left side to the right side of the volume that maximizes the summation of  $Y$ 's along the path. The optimal path corresponds to the disparity values for the given row in the disparity image. The second stage can be implemented using a shortest path algorithm similar to the first stage.

The method leads to an efficient path-based matching algorithm. Subpixel accuracy matching is performed as a post processing step by fitting a quadratic function over pixels in a neighborhood. The disparity is solved by finding the maximum of the quadratic function. Experiments demonstrate the improvements of the algorithm over other existing ones. However, this subpixel accuracy matching is not optimal as the obtained paths may not be optimal if we consider subpixel cross correlations. In addition, quadratic functions used are not sufficient for bilinear interpolation.

## 3. Stereo Matching with Subpixel Accuracy

Our goal is to achieve a matching accuracy to the subpixel level between the left and right images. As in [6], we assume that the input stereo pair is rectified, i.e., the matching row is within one row from the corresponding row. As in [6], we also adopt normalized cross correlation as given in (1). For digital images, these values are only defined at integer locations. In order to allow subpixel accuracy, we use bilinear interpolation to obtain values at subpixel locations. Since images are discrete in nature, we define the bilinear interpolation for  $0 \leq s, t \leq 1$  in the right image only. According to the bilinear interpolation, we can express  $g(x + s, y + t)$  as

$$\begin{aligned} &= (g(x, y)(1 - s) + g(x + 1, y)s)(1 - t) \\ &\quad + (g(x, y + 1)(1 - s) + g(x + 1, y + 1)s)t \\ &= g(x, y)(1 - s)(1 - t) + g(x + 1, y)s(1 - t) \\ &\quad + g(x, y + 1)(1 - s)t + g(x + 1, y + 1)st \\ &= [g(x, y) - g(x + 1, y) - g(x, y + 1) \\ &\quad + g(x + 1, y + 1)]st \\ &\quad + [-g(x, y) + g(x + 1, y)]s \\ &\quad + [-g(x, y) + g(x, y + 1)]t + g(x, y). \end{aligned}$$



**Figure 1. Disparity map for stereo pairs (left images) using the method in [6] (middle images) and the proposed method (right images). (a) A baseball pair. (b) A park meter scene. (c) An outdoor ground.**

Since  $NCC(x_0, y_0, u, v)$  can change significantly, even for fractional changes in  $u$  and  $v$ , in order to make the path matching algorithm effective at the subpixel level, we define  $\widehat{NCC}(x_0, y_0, u, v)$  as

$$\begin{aligned} &= \arg \max_{-.5 \leq s \leq .5, -.5 \leq t \leq .5} NCC(x_0, y_0, u + s, v + t) \\ &= \max_{k=1, \dots, 4} \left\{ \widehat{NCC}_k(x_0, y_0, u, v) \right\}, \end{aligned}$$

assuming that  $u$  and  $v$  will be integer values, and

$$\begin{aligned} \widehat{NCC}_1(x_0, y_0, u, v) &= \arg \max_{-.5 \leq s \leq 0, -.5 \leq t \leq 0} NCC(x_0, y_0, u + s, v + t), \\ \widehat{NCC}_2(x_0, y_0, u, v) &= \arg \max_{0 \leq s \leq .5, -.5 \leq t \leq 0} NCC(x_0, y_0, u + s, v + t), \\ \widehat{NCC}_3(x_0, y_0, u, v) &= \arg \max_{-.5 \leq s \leq 0, 0 \leq t \leq .5} NCC(x_0, y_0, u + s, v + t), \\ \widehat{NCC}_4(x_0, y_0, u, v) &= \arg \max_{0.0 \leq s \leq .5, 0 \leq t \leq .5} NCC(x_0, y_0, u + s, v + t). \end{aligned}$$

Here  $NCC(x_0, y_0, u + s, v + t)$  are defined using bilinear interpolation. The central problem is how to compute  $\widehat{NCC}_k(x_0, y_0, u, v)$ ,  $k = 1, \dots, 4$  efficiently; clearly a brute force implementation will be computationally expensive. Here we give details for  $\widehat{NCC}_4(x_0, y_0, u, v)$  and the other three cases can be computed similarly.

As given in (1), to compute  $\widehat{NCC}_4(x_0, y_0, u, v)$ , we need to compute the summation of images with pixel values squared. Let  $x_1 = x_0 + u$ ,  $y_1 = y_0 + v$ ,  $x_2 = x_0 + u + 1$  and  $y_2 = y_0 + v + 1$ . Using bilinear interpolation, we have

$$\begin{aligned} &= \sum_{i,j} g(x_1 + i + s, y_1 + j + t)^2 \\ &= \sum_{i,j} [g(x_1 + i, y_1 + j)(1-s)(1-t) \\ &\quad + g(x_2 + i, y_1 + j)s(1-t) \\ &\quad + g(x_1 + i, y_2 + j)(1-s)t \\ &\quad + g(x_2 + i, y_2 + j)st]^2 \\ &= (1-s)^2(1-t)^2 \left[ \sum_{i,j} g(x_1 + i, y_1 + j)^2 \right] \\ &\quad + s^2(1-t)^2 \left[ \sum_{i,j} g(x_2 + i, y_1 + j)^2 \right] \\ &\quad + (1-s)^2t^2 \left[ \sum_{i,j} g(x_1 + i, y_2 + j)^2 \right] \\ &\quad + s^2t^2 \left[ \sum_{i,j} g(x_2 + i, y_2 + j)^2 \right] \\ &\quad + C_1 \left[ \sum_{i,j} g(x_1 + i, y_1 + j)g(x_2 + i, y_1 + j) \right] \\ &\quad + C_2 \left[ \sum_{i,j} g(x_1 + i, y_1 + j)g(x_1 + i, y_2 + j) \right] \\ &\quad + C_3 \left[ \sum_{i,j} g(x_1 + i, y_1 + j)g(x_2 + i, y_2 + j) \right] \\ &\quad + C_4 \left[ \sum_{i,j} g(x_2 + i, y_1 + j)g(x_1 + i, y_2 + j) \right] \\ &\quad + C_5 \left[ \sum_{i,j} g(x_2 + i, y_1 + j)g(x_2 + i, y_2 + j) \right] \\ &\quad + C_6 \left[ \sum_{i,j} g(x_1 + i, y_2 + j)g(x_2 + i, y_2 + j) \right], \end{aligned} \quad (3)$$

where  $C_1 = 2(1-s)s(1-t)^2$ ,  $C_2 = 2(1-s)^2(1-t)t$ ,  $C_3 = 2(1-s)s(1-t)t$ ,  $C_4 = 2(1-s)s(1-t)t$ ,  $C_5 = 2s^2(1-t)t$ , and  $C_6 = 2(1-s)st^2$ . Note that the  $\sum_{i,j}$  terms can be computed using integral images. Four additional integral images are needed:  $g(x, y)g(x + 1, y)$ ,  $g(x, y)g(x, y + 1)$ ,  $g(x, y)g(x + 1, y + 1)$ , and  $g(x, y + 1)g(x + 1, y + 1)$ .

For  $\bar{g}(x_1 + s, y_1 + t)$ , we have

$$\begin{aligned} &\bar{g}(x_1 + s, y_1 + t) \\ &= (1-s)(1-t)\bar{g}(x_1, y_1) + s(1-t)\bar{g}(x_1 + 1, y_1) \\ &\quad + (1-s)t\bar{g}(x_1, y_1 + 1) + st\bar{g}(x_1 + 1, y_1 + 1). \end{aligned} \quad (4)$$

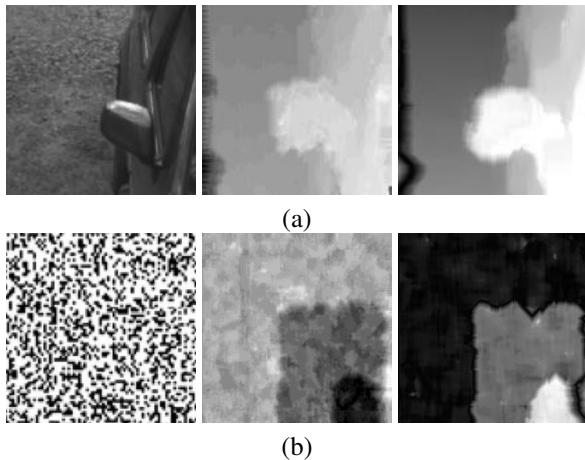
By combining (3) and (4), we can therefore compute the normalized cross correlation with subpixel accuracy efficiently through integral images. As all the integral images only need to be computed once, they will not increase the computational complexity in a significant way.

## 4. Experimental Results

Here we provide several examples to illustrate the results obtained by our algorithm. Since our method is an extension of the work by Sun [6], we provide direct comparisons between his method and ours. The results shown here show the feasibility of our method, as well as illustrate the improvements over the original algorithm.

The first example illustrated in Figure 1(a) presents results using stereo images obtained from the test data in [5].

The left image shows one of the stereo images, a baseball against a noisy background. The second image shows the disparity map obtained by Sun’s algorithm. Notice that the algorithm obtains a good estimate of the disparity. The rightmost image is the disparity image obtained with our algorithm, incorporating our subpixel measurements. Notice that the boundaries of the ball are sharper and better defined than the ones provided by Sun’s algorithm. The second example is illustrated in Figure 1(b). The input images for this test case were images also obtained from the test data in [5]. The disparity image generated by Sun’s algorithm presents an accurate estimate of the scene’s depth, but contains some noise, particularly in areas of low texture. Our disparity image, however, obtained sharper boundaries and smoother disparity areas. The third example illustrated in Figure 1(c) presents an image of the ground taken from a  $45^\circ$  angle such that the top region of the image represents the part of the ground that is furthest from the stereo camera. As before, Sun’s algorithm provides an accurate estimate of the depth in the scene. Our results, however, provide a smoother disparity image with correct depth estimation.



**Figure 2. Two additional disparity map examples. See Fig. 1 for figure caption. (a) Part of a car and ground. (b) A stereogram pair.**

The fourth example illustrated in Figure 2(a) combines the image of the third experiment (illustrated in Figure 1(c)) with the rearview mirror of a car in the foreground. As in the previous cases, Sun’s algorithm does a good job of recovering scene depth. Our disparity image, however, is able to recover better boundaries between the foreground object (car) and the background (ground), as well as provide a smoother disparity map overall. It also appears to perform slightly better in recovering the depth of the ground. The final example presented is illustrated in Figure 2(b). The input images were a stereogram pair obtained from Sun [6]. The results show that although Sun’s method provides a good

disparity map, our results show more accurate boundaries around the edges of the squares, as well as overall smoother measurements inside each square.

## 5. Conclusion

We have presented an efficient subpixel accuracy path-based matching algorithm. Our algorithm provides a normalized cross correlation measurement based on bilinear interpolation between pixel locations in order to provide correlation coefficients accurate to the subpixel level. We apply the subpixel measurements to the cross correlation step, since these values greatly improve the paths obtained by path-based matching [6]. Our experiments show the feasibility of our method. Compared to the algorithm in [6], our subpixel measurements provide smoother and more accurate disparity maps, with sharper and more accurate boundaries. The extra calculations cost roughly twice the amount of work than that of the original algorithm [6], but the time complexity remains linear. As explained in [6], the algorithm’s time complexity is  $O(WHD)$ , where  $W, H$  are the image dimensions, and  $D$  describes the disparity range (essentially,  $W \times H \times D$  is the size of the 3D correlation coefficient volume). Because of the rectangular subregioning process, the disparity range for each subregion can be reduced, therefore reducing the time complexity of the algorithm to  $O(W_i H_i D_i)$ , where  $W_i \leq W$ , and so on. We are currently working on improving the performance of the method by modifying it to work with larger disparity ranges, as done in [4].

## References

- [1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.
- [2] M. Z. Brown, D. Burschka, and G. D. Hager. Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003.
- [3] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, 2000.
- [4] C. Leung, B. Appleton, and C. Sun. Fast stereo matching by iterated dynamic programming and quadtree subregioning. In *British Machine Vision Conference*, pages 97–106, London, UK, 2004.
- [5] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, 2002.
- [6] C. Sun. Fast stereo matching using rectangular subregioning and 3d maximum-surface techniques. *International Journal of Computer Vision*, 47(1-3):99–117, 2002.
- [7] P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.