Efficient Path-Based Stereo Matching With Subpixel Accuracy

Arturo Donate, Xiuwen Liu, and Emmanuel G. Collins, Jr.

***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 with 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 cross correlations at all pixels 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 at integer pixel locations 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 significant improvement over existing path-based matching methods.**

***Index Terms*—Bilinear interpolation, computer vision, disparity, dynamic programming, integral image, normalized cross correlation (NCC), path-based matching, stereo, stereo matching, subpixel accuracy.**

I. Introduction

THE USE of stereo images in computer vision is crucial for applications requiring depth perception [1]. In order to be useful, stereo-vision algorithms rely on the ability to perform accurate point correspondence between an image pair [2], where the correspondence is defined as the problem of finding the accurate location of the same point in the scene in a pair of stereo images. Therefore, corresponding points must describe the same content, although their image coordinates may differ. It is useful to note that these corresponding points may also be used for motion estimation, as well as many other applications.

Stereo correspondence methods can be divided into two main types: region-based and feature-based methods. Region-based methods attempt to find corresponding points by matching intensity values across images. These searches are typically performed within local image windows in both images. In some cases, if the geometry of the cameras is known, the search can be performed across lines, as defined by the epipolar constraint [3]. Feature-based methods typically rely on matching more distinctive but sparse features. Such features are typically assumed to have some higher level meaning (as opposed to image points) and, thus, are assumed to be more stable. Several types of image features, such as Harris corners [4], Canny edges [5], or SIFT points [6], are likely to be easily located in stereo-image pairs.

In this paper, we focus on region-based methods, as they are widely used to generate dense matching. A common approach for solving the correspondence problem is to match the local windows using some matching criterion such as normalized cross correlation (NCC). This region-based method yields results with pixel-level accuracy. In order to achieve accuracy at a subpixel level, typically, a second-order polynomial is used to fit matching scores in a local neighborhood [7]. One of the intrinsic limitations of such region-based matching methods is their inability to resolve local ambiguities effectively. For example, for a given image point, there can often be multiple matching candidate points in the candidate region of the corresponding image. These multiple candidate points often arise from situations with low texture variations, as well as other factors. In addition, local deformations may cause the correct matching point not to be the local maximum, causing many of the algorithms dependent on local maxima to fail.

To overcome the local ambiguities and achieve more robust matching, a global matching criterion may be used. Sun [8] poses the stereo-matching problem as an optimization of the total cross-correlation scores over a surface through a 3-D cross-correlation volume (whose dimensions are given by height, width, and the disparity range of a region). The matching is solved efficiently using a two-stage dynamic programming algorithm. This algorithm attempts to maximize cross-correlation scores along paths in the 3-D cross-correlation volume. It is important to note that by using a global matching criterion, accurate cross-correlation values are needed at all locations since they affect the optimal path estimation (and, thus, the matching). Since cross correlations can change significantly even at the subpixel level, correlation measurements must be calculated with subpixel accuracy in order to achieve an optimal stereo matching.

In [8], Sun performs subpixel measurements as a post-processing stage by fitting a quadratic function in a neighborhood. In this paper, we aim to show that cross-correlation

Manuscript received March 26, 2009; revised December 30, 2009; accepted March 22, 2010. Date of publication June 21, 2010; date of current version January 14, 2011. This work was supported in part by the National Science Foundation Grant CCF-0514743 and Grant DMS-0713012 and in part by the National Institutes of Health (NIH) through the NIH Roadmap for Medical Research (Grant U54 RR021813). This paper was recommended by Associate Editor H. Zhang.

A. Donate and X. Liu are with the Department of Computer Science, Florida State University, Tallahassee, FL 32306 USA (e-mail: donate@cs.fsu.edu; ywang@cs.fsu.edu; liux@cs.fsu.edu).

E. G. Collins, Jr. is with the Department of Mechanical Engineering, Florida A&M University–Florida State University College of Engineering, Tallahassee, FL 32310 USA (e-mail: ecollins@eng.fsu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2010.2049839

values calculated with subpixel accuracy provide a significant improvement to the path-based stereo matching, and this is supported by experimental results. These measurements can be performed efficiently by exploiting certain aspects of the calculations and employing the use of integral images [9], [10].

The rest of this paper is organized as follows. Section II provides a review of the recent literature in the area. In Section III, we summarize the path-based stereo-matching algorithm [8], and in Section IV, we show how the subpixel accuracy can be incorporated efficiently using integral images; note that while Sun [8] also performs subpixel matching, it is done after the path matching. As shown by our results in Section V, our algorithm improves the performance significantly by having more accurate correlation coefficients. Section VI concludes this paper.

II. Related Work

Although stereo matching is one of the classical problems in computer vision, it also appears in other areas such as medical imaging, computer graphics, and image processing. A considerable amount of work has been done on this specific subject over the last 30 years. More specifically, the past decade has brought a plethora of proposed methods for solving the correspondence problem. Even so, the use of subpixel measurements is not common. Typically, subpixel accuracy is done by a separate postprocessing step, calculated after the initial matching. Several methods for calculating subpixel measurements include iterative gradient descent as well as curve fitting on matching costs [11]. This section presents a survey of recent publications dealing with stereo matching, particularly those that incorporate subpixel measurements.

Of all the works discussed here, Sun's method [8] is clearly the most relevant to the proposed method. As mentioned earlier, it uses NCC to determine matching cost coefficients between two images for a given disparity range. These coefficients are arranged into a 3-D volume, and a dynamic programming algorithm is used to determine the shortest path across the volume; the optimal path yields the disparity values. Subpixel measurements are obtained by fitting a second-degree polynomial to the resulting correlation coefficients of pixels around a neighborhood, and the extrema are used as the final disparity. See Section III for further details.

De la Hamette and Tröster [12] present the FingerMouse system, a human–computer interaction framework allowing users to interact with a computer using finger gestures as input. This input is captured by a pair of stereo cameras, and the corresponding images are then used to calculate a 3-D reconstruction of the scene. For greater accuracy, the authors incorporate subpixel measurements into their stereo calculations, where the measurements are performed by averaging the values of several disparity maps of the same view generated at different points in time.

Knoblauch and Kuester [13] propose a system for generating a 3-D avatar of a person for use in tele-immersion environments. After obtaining stereo images from a pair of cameras, the system first performs background extraction and then estimates the best possible disparity value within a given range. This optimal value is defined as the one with the smallest matching cost. The neighboring disparity values are then taken into consideration, and linear interpolation is used in order to calculate the final disparity value.

Sand and Teller [14] introduce a method for achieving a spatiotemporal alignment of two videos recorded at different times. The videos are aligned by using a regression method that matches video frames with large likelihood of correspondence. Given a feature point in the first frame, they perform a local motion optimization on candidate regions of the second frame in order to perform a match. The authors state that the local motion optimization allows for subpixel measurements, which yield better matching results than just simply matching corresponding image features in both frames.

Franke and Joos [15] present a framework for understanding complex traffic scenes. Using stereo vision, they generate a disparity image of the scene observed by a stereo camera. From the scene, the authors perform obstacle detection, space analysis, as well as object recognition. In order to obtain accurate results in real time, the authors propose a method for calculating multiscale disparity images in real time and with subpixel accuracy. Here, subpixel accuracy is obtained by fitting a parabolic function through the correlation coefficients.

Morgan *et al.* [16] propose a stereo-matching technique for determining terrain elevation using stereo images when the images are taken with a relatively small baseline. The method is based on phase correlation techniques to perform the matching. The phase correlation technique, first proposed by Liu and Yan [17], is based on the displacement between two images to be measured in the phase difference of their Fourier transforms. The authors are able to achieve subpixel accuracy by solving the phase difference in the Fourier domain via singular value decomposition.

Szeliski and Scharstein [18] introduce an algorithm that derives a matching cost from reconstructed image signals and then assigns disparity values based on a symmetric matching process that incorporates subpixel information. Interestingly, their experiments show that subpixel measurements sometimes yield higher errors in the stereo matching due to the increased sensitivity to noise in the original image pairs.

Kim and Chung [19] present an algorithm for stereo matching, which successfully handles large depth discontinuities by using variable windows on the images. The correlation between candidate points is determined using NCC, as well as the sum of squared differences. Subpixel measurements are used to reduce occasional local deformations, which may arise due to the variable windows.

Sarkar and Bansal [20] propose a solution to the stereo-matching problem using wavelets, which uses the joint probability density function (pdf) of intensity values between two image windows in order to find the best matching; the pdf allows the method to become more resilient to changes in illumination. Wavelets are used in order to perform the matching hierarchically. Pajares and de la Cruz [21] solve the stereo-matching problem by using edge segments as features. The features are matched locally as well as globally using constraints based on epipolar, similarity, smoothness, ordering, and uniqueness. An optimization-based process incorporating simulated annealing is used to perform global matching. Zhang and Kambhamettu [22] present two systems for multiview matching (using three views). The first system uses a 3-D affine model to represent the motion of a 3×3 image region and incorporates the use of nonlinear least squares to fit the model at each region. The second system is described as an extension to 2-D optical flow. Depth discontinuities are modeled using an image segmentation algorithm. Three-dimensional scene flow is then estimated as a minimization of the optical flow constraints from different views.

Brockers *et al.* [23] pose stereo matching as an optimization problem. For a given pixel, a range of possible disparities is first assigned. Then, the best disparity value is calculated by optimizing a global cost function, taking into account stereoscopic properties as well as similarity measurements, in order to produce dense matching. Kim [24] proposes a stereo-matching algorithm that handles depth discontinuities as well as smooth curved regions using variable windows. Similarity measurements are performed using a correlation function based on voxel components. Rosenberg *et al.* [25] present a system for real-time stereo using programmable graphics hardware. The system employs a matching algorithm originally proposed by Hirscmuller [26], which uses dynamic programming to work across an image in multiple directions; at each pixel, it uses the cost and disparity range to find the shortest path through the disparity range, which gives the final disparity value.

III. Path-Based Stereo Matching

The proposed method is inspired by the work of Sun [8]. There, Sun presents a dynamic programming algorithm that uses rectangular subregioning and maximum surface techniques in order to perform path-based matching. This section describes different parts of his proposed approach. His method assumes that the input images to the system are rectified so that a shift along the epipolar lines corresponds to a shift along the x-axis of the image. This means that, for a given pixel location (i,j) in the left image, the corresponding match in the right image contains the same value for i, while the value for j can vary within some range $[-w,w]$.

The basic outline of the algorithm is as follows. First, a k-level image pyramid is formed for each of the input images. For each level of the pyramid, rectangular subregions are first found using the temporary disparity map calculated at the previous pyramid level. For each subregion, zero-mean NCC is used in order to calculate the correlation coefficient values among pixels. Finally, dynamic programming is used to determine the best path across these correlation coefficients in order to compute a disparity image. The disparity image generated for the last level of the pyramid is the output of the algorithm.

A. Rectangular Subregioning

Initially, the images are segmented into subregions in order to reduce computation time, as well as memory requirements of the system. In a given stereo-image pair, different image points may have disparity values that lie within very different ranges. The aim of the subregioning step is to segment the image in a way such that each point in a given subregion contains a similar range of disparity values.

Prior to the subregioning step, the images are decomposed into pyramids, each with k levels, in order to employ a coarse-to-fine matching strategy. Starting off with the original images, each is subsampled for the next pyramid level by finding the mean intensity value across a window of size $r \times r$.

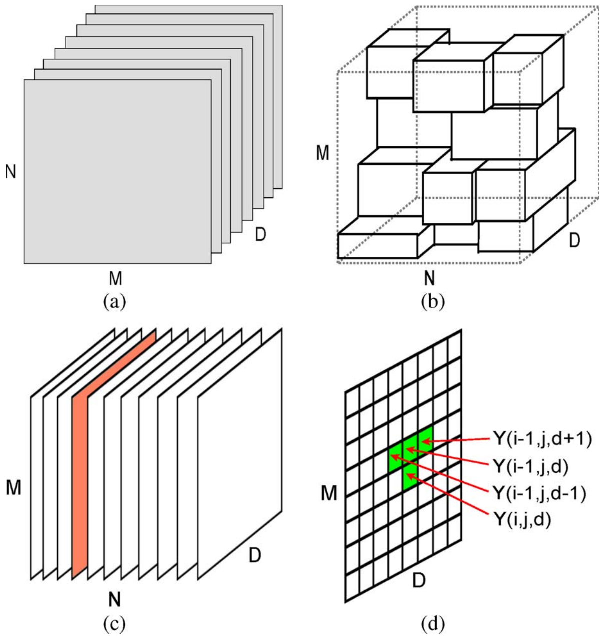


Fig. 1. Illustration of Sun's algorithm. (a) Image volume, composed of an input image with varying disparities. (b) Three-dimensional volume of coefficients with subregioning. (c) Vertical sections. (d) Calculating intermediate volume.

Next, the images are segmented into subregions using the disparity map calculated at the previous pyramid level as input. This input image is first divided evenly into a set number of rows. Adjacent rows are compared and merged using a criterion for minimizing overall computational complexity. Next, each of these resulting regions is divided into a set number of horizontal columns. A similar merging process is repeated until there are no more columns that can be merged. The resulting subregions aim to segment the image in a way such that we end up with large regions containing small disparity ranges and small regions containing large disparity ranges.

It is important to note that for the first level, the disparity map is initialized to zero, so the algorithm disregards the use of subregions at this level. Also, note that by definition, a multiscale image pyramid contains the same image at different scales. In this case, the pyramid contains k levels, each corresponding to a different resolution. For a given level t, the input to the subregioning step is the disparity image calculated at the previous level $t-1$, and, hence, it is at a lower resolution. In order to address this, this input disparity image must first be enlarged by interpolating through the pixel values.

For a given image point location, the goal is now to determine the disparity value within the specified disparity range, which yields the optimal NCC measurement. A 3-D volume of size $W \times H \times D$ can be constructed from the correlation coefficients, where W and H are the dimensions of the original stereo pair, and D is the size of the disparity range. Since the subregioning process segments the images into regions containing similar disparity ranges, each region can then be viewed as a smaller 3-D volume of size $W_i \times H_i \times D_i$, where $W_i \leq W$ and so on. This is illustrated in Fig. 1(b).

B. Matching Algorithm

After the rectangular subregioning process, a 3-D volume of correlation coefficients can be created for every point (i, j, d) in the image volume, where i and j are the row and column indexes obtained from the pixel coordinates, respectively, and d is a possible disparity value in the disparity range (which may vary across different subregions in order to decrease computational costs). In other words, the value at coordinate (i, j, d) within the volume is the zero-mean NCC value between windows in the stereo images f and g using a disparity value of d. Each window is centered at location (i, j). This NCC is defined as

$$C(i,j,d) = \frac{\mathrm{cov}_{ij,d}(f,g)}{\sqrt{\mathrm{var}_{ij}(f)} \times \sqrt{\mathrm{var}_{ij,d}(g)}} \tag{1}$$

where (i, j) are the row and column indexes of the 3-D correlation volume (i.e., pixel location), and d is the possible disparity value. Here, $\mathrm{var}_{ij}(f)$ refers to the variance within a window centered at (i, j) of the image f, and $\mathrm{cov}_{ij,d}(f, g)$ refers to the covariance between the windows of the left and right images. The disparity term d is obtained from the given disparity range in the subregion, corresponding to a shift of the window along the epipolar lines. In order to compute correlation values efficiently, Sun uses box filtering [8], [27], [28].

Given the 3-D 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 3-D volume in the vertical direction for each vertical section [as illustrated in Fig. 1(c)]. Given the original 3-D correlation volume C, the intermediate 3-D volume Y is calculated according to

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

where p determines the number of local values to be considered. For example, when $p = 1$, three locations $d - 1$, d, and $d + 1$ are considered, as shown in Fig. 1(d). In other words, at a given vertical level, only the values that are within distance p from the previous value (in both directions) are considered. The algorithm begins on the highest vertical section and works its way down. In this initial step, $i - 1$ will be undefined, so it is set to zero. Therefore, when $i = 0$

$$Y(0,j,d) = C(0,j,d) \tag{3}$$

so that the very top vertical section is identical to the top section of the 3-D volume C. By the end of the first stage of this dynamic program, the bottom vertical section of the intermediate volume Y contains a summation of the maximum correlation values.

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 3-D 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. First, the algorithm begins by selecting the bottom slice of the intermediate volume Y. It begins at the bottom slice because this is where the correlation values are accumulated.

Using the shortest path algorithm similar to the one used in the first stage, the shortest path along the bottom slice of the intermediate volume Y is calculated (this bottom slice has a dimension $W \times D$). For the next iteration, the algorithm calculates the shortest path in the next level above the current one, until it finds a path at all levels of Y. For each level (not including the bottom level), the shortest path in a given level is constrained to be at a distance no larger than p from the previous shortest path. The value of p was previously defined in (2) and is typically kept at or near 1 in order to keep computational costs low.

This method proposed by Sun [8] leads to an efficient path-based matching algorithm. In order to increase the quality of the matching, subpixel accuracy is performed as a postprocessing step by the use of a quadratic function over pixels in a neighborhood. The author fits a second-degree curve to the correlation coefficients around the neighborhood of a pixel and uses the extrema of the curve to solve for the disparity. This additional step improves the quality of the results over only using pixel values at integer locations. However, this subpixel accuracy matching is not optimal, as the obtained paths may not be optimal if we consider subpixel cross correlations.

IV. Stereo Matching With Subpixel Accuracy

Our goal is to achieve dense matching accurate to the subpixel level while maintaining a low computational cost in order to obtain optimal path matching efficiently. As in the work by Sun [8], we assume that the input stereo pair is rectified, i.e., the matching row is within one row from the corresponding row. Unlike Sun's approach, however, our proposed method incorporates the use of subpixel measurements into the correlation calculation, effectively making the correlation coefficients more accurate and allowing the path-based matching to achieve more precise matching. In order to perform such computations efficiently, we employ the use of integral images. Our main contribution is an improvement in the way Sun's method calculates correlation coefficients. As such, Section IV-A describes in detail our implementation for fast correlation measurements accurate to the subpixel level. The other parts of Sun's algorithm remain unmodified and, hence, are not discussed in detail.

A. Correlation Measurements

As in the work by Sun [8], we also adopt NCC as given in (7). Given a left image f, a right image g, and a pixel location in the left image $f(x_0, y_0)$, the goal is to compute the optimal pixel location in the right image g corresponding to the point $f(x_0, y_0)$ in the left image. In order to achieve this, NCC is used as a similarity measure on a window of size of $(2M + 1) \times (2N + 1)$ on the right image for a displacement (u, v). The value obtained from the similarity measure will then be used with the path-based matching proposed by Sun [8]. 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 define the NCC, we first define the mean of the window in the left image as

$$\overline{f}(x_0, y_0) = \frac{1}{S} \sum_{i=-M}^{i=M} \sum_{j=-N}^{j=N} f(x_0 + i, y_0 + j) \tag{4}$$

and the variance of the window as

$$\overline{\overline{f}}(x_0, y_0) = \left[\frac{1}{S} \sum_{i=-M}^{i=M} \sum_{j=-N}^{j=N} f(x_0 + i, y_0 + j)^2 - \overline{f}(x_0, y_0)^2 \right] \tag{5}$$

where S is defined as

$$S = (2 \times M + 1)(2 \times N + 1). \tag{6}$$

To be specific, let $\overline{f}(x, y)$ and $\overline{g}(x, y)$ be the mean of the left and right windows centered at (x, y) and of size $(2M + 1) \times (2N + 1)$, respectively. Similarly, let $\overline{\overline{f}}(x, y)$ and $\overline{\overline{g}}(x, y)$ be the variance of the left and right windows, respectively. We can define the NCC between the left image window at (x_0, y_0) and the right image window at $(x_0 + u, y_0 + v)$ as

$$\frac{\sum_{i,j} \hat{f}(x_0 + i, y_0 + j)\hat{g}(x_0 + u + i, y_0 + v + j)}{S\sqrt{\overline{\overline{f}}(x_0, y_0)\overline{\overline{g}}(x_0 + u, y_0 + v)}} \tag{7}$$

where $\hat{f}$ is defined as

$$\hat{f}(x_0 + i, y_0 + j) = f(x_0 + i, y_0 + j) - \overline{f}(x_0, y_0) \tag{8}$$

and the summation for i is from $-M$ to M and j from $-N$ to N (also in subsequent equations). As pointed out in [8] and [9], for 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 $\overline{\overline{f}}(x, y)$ can also be done efficiently using an integral image with pixel values squared.

B. Integral Images

Originally proposed by Crow [29], an integral image (also known as a summed-area table) is an intermediate representation of an image that aids in solving certain problems in computer vision. It is essentially an additive representation of an image where the value at a given location is equal to the sum of the pixel values at locations to the left and above the current index location.

Formally, they are defined as

$$I(x, y) = \sum_{i=0}^{x} \sum_{j=0}^{y} O(i, j) \tag{9}$$

where O is the original image, and I is the integral image being calculated. As described by Viola and Jones [9], they can be computed in one pass over the original image. The main benefit for such a data structure is that any given rectangular sum can be calculated from four references to the integral image.

Integral images have been used extensively in solving real-time computer-vision problems. Viola and Jones [9] use them to solve the problem of face recognition in real time. Frintrop *et al.* [30] use integral images to aid in fast image feature computation in their real-time visual attention system. Kisacanin [31] uses integral images in embedded systems to solve complex computations efficiently and describes optimization methods that allow acceptable execution costs in embedded processors by taking advantage of recursion and double buffering techniques. Bay *et al.* [10] use integral images in order to speed up the computation of their robust feature extraction algorithm. In this paper, we will show that integral images can also be used to compute NCC values of stereo images with subpixel accuracy in an efficient manner.

C. Bilinear Interpolation

Due to the nature of digital images, pixel values are only defined at integer locations. Therefore, in order to perform effective subpixel measurements, we propose the use of bilinear interpolation. For a given noninteger image location (x_s, y_s), where $x < x_s < x + 1$ and $y < y_s < y + 1$, we can use the values in the surrounding coordinates at locations (x, y), $(x + 1, y)$, $(x, y + 1)$, and $(x + 1, y + 1)$ to estimate the value at (x_s, y_s) by linearly interpolating in each direction. This allows the estimation of the value at (x_s, y_s) to be computed as a weighted average of the four known points. The interpolation provides a way to estimate the value of an image at subpixel locations (corresponding to noninteger coordinates previously undefined in the image).

Bilinear interpolation has been widely used to solve computer-vision and image-processing problems in the literature. Kim and Kim [32] propose a system that uses image measurements to detect scratches in film and then employs the use of bilinear filtering in order to sample the surface of the image and recover the scratched regions. Van Velden *et al.* [33] propose the use of bilinear filtering for filling in gaps from high-resolution medical scans obtained using 3-D-filtered back-projection strategies. Fahmy [34] proposes an architecture for efficient calculation of bilinear filtering techniques on field-programmable gate arrays.

Since images are discrete in nature, we define the bilinear interpolation for $0 \leq s$ and $t \leq 1$ in the right image only. In other words, given a pixel location in the left image, we would like to find the subpixel location in the right image that gives an optimal match. According to the bilinear interpolation, we can express the right image g as

$$\begin{aligned}
g(x + s, y + t)& \\
&= \left(g(x, y)(1 - s) + g(x + 1, y)s\right)(1 - t) \\
&\quad + \left(g(x, y + 1)(1 - s) + g(x + 1, y + 1)s\right)t \\
&= g(x, y)(1 - s)(1 - t) + g(x + 1, y)s(1 - t) \\
&\quad + g(x, y + 1)(1 - s)t + g(x + 1, y + 1)st \\
&= [g(x, y) - g(x + 1, y) - g(x, y + 1) \\
&\qquad + g(x + 1, y + 1)]\, st \\
&\quad + [-g(x, y) + g(x + 1, y)]\, s \\
&\quad + [-g(x, y) + g(x, y + 1)]\, t + g(x, y).
\end{aligned} \tag{10}$$

D. Correlation Measurements at the Subpixel Level

Using bilinear interpolation as defined by (10) along with NCC as defined by (7), we can compute correlation measurements for points previously undefined at subpixel locations. The correlation value between two such points 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 $\widehat{\mathrm{NCC}}(x_0, y_0, u, v)$ as

$$\begin{aligned} &= \arg\max_{-.5\le s\le .5, -.5\le t\le .5} \mathrm{NCC}(x_0, y_0, u+s, v+t) \\ &= \max_{k=1,\ldots,4}\left\{\widehat{\mathrm{NCC}}_k(x_0, y_0, u, v)\right\} \end{aligned} \tag{11}$$

assuming that u and v will be integer values used to define a pixel offset from the center, but still within the correlation window. To determine the values of s and t, we take discrete samples [10 × 10 samples centered around each pixel are used to achieve (1/10)-pixel accuracy in all the experiments]. Here, s and t correspond to subpixel locations. For each value of k (corresponding to one of the four quadrants around a pixel), we define $\widehat{\mathrm{NCC}}_k$ as

$$\begin{aligned} \widehat{\mathrm{NCC}}_1(x_0, y_0, u, v) &= \arg\max_{-0.5\le s\le 0, -0.5\le t\le 0} \\ &\quad \times \mathrm{NCC}(x_0, y_0, u+s, v+t) \\ \widehat{\mathrm{NCC}}_2(x_0, y_0, u, v) &= \arg\max_{0\le s\le 0.5, -0.5\le t\le 0} \\ &\quad \times \mathrm{NCC}(x_0, y_0, u+s, v+t) \\ \widehat{\mathrm{NCC}}_3(x_0, y_0, u, v) &= \arg\max_{-0.5\le s\le 0, 0\le t\le 0.5} \\ &\quad \times \mathrm{NCC}(x_0, y_0, u+s, v+t) \\ \widehat{\mathrm{NCC}}_4(x_0, y_0, u, v) &= \arg\max_{0.0\le s\le 0.5, 0\le t\le 0.5} \\ &\quad \times \mathrm{NCC}(x_0, y_0, u+s, v+t). \end{aligned} \tag{12}$$

Here $\mathrm{NCC}(x_0, y_0, u+s, v+t)$ are defined using bilinear interpolation. Although the images are rectified, we do not assume that the search for corresponding points between two images is limited to only one dimension; instead, we use bilinear interpolation to interpolate in both the x and y directions, thus allowing for correspondences that do not fit the rectified image exactly. The central problem now is to compute $\widehat{\mathrm{NCC}}_k(x_0, y_0, u, v)$ (for all $k = 1, \ldots, 4$) efficiently. Clearly, a brute force implementation will be computationally expensive and undesirable. Instead, we exploit certain properties of these calculations through the use of integral images in order to compute $\widehat{\mathrm{NCC}}_k$ efficiently for all values of k.

In the following example, we show how the computation can be done for $\widehat{\mathrm{NCC}}_4$, but the same basic algorithm can be used for the three other values of k. As given in (7), in order to compute $\widehat{\mathrm{NCC}}_4(x_0, y_0, u, v)$, we need to compute the summation of images with pixel values squared. Let

$$\begin{aligned} x_1 &= x_0 + u \\ y_1 &= y_0 + v \\ x_2 &= x_0 + u + 1 \\ y_2 &= y_0 + v + 1. \end{aligned} \tag{13}$$

Using bilinear interpolation, we have

$$\begin{aligned} &\sum_{i,j} g(x_1+i+s, y_1+j+t)^2 \\ &\quad = \sum_{i,j} [g(x_1+i, y_1+j)(1-s)(1-t) \\ &\qquad\quad + g(x_2+i, y_1+j)s(1-t) \\ &\qquad\quad + g(x_1+i, y_2+j)(1-s)t \\ &\qquad\quad + g(x_2+i, y_2+j)st]^2 \\ &\quad = (1-s)^2(1-t)^2\left[\sum_{i,j} g(x_1+i, y_1+j)^2\right] \\ &\qquad + s^2(1-t)^2\left[\sum_{i,j} g(x_2+i, y_1+j)^2\right] \\ &\qquad + (1-s)^2t^2\left[\sum_{i,j} g(x_1+i, y_2+j)^2\right] \\ &\qquad + s^2t^2\left[\sum_{i,j} g(x_2+i, y_2+j)^2\right] \\ &\qquad + C_1\left[\sum_{i,j} g(x_1+i, y_1+j)g(x_2+i, y_1+j)\right] \\ &\qquad + C_2\left[\sum_{i,j} g(x_1+i, y_1+j)g(x_1+i, y_2+j)\right] \\ &\qquad + C_3\left[\sum_{i,j} g(x_1+i, y_1+j)g(x_2+i, y_2+j)\right] \\ &\qquad + C_4\left[\sum_{i,j} g(x_2+i, y_1+j)g(x_1+i, y_2+j)\right] \\ &\qquad + C_5\left[\sum_{i,j} g(x_2+i, y_1+j)g(x_2+i, y_2+j)\right] \\ &\qquad + C_6\left[\sum_{i,j} g(x_1+i, y_2+j)g(x_2+i, y_2+j)\right] \end{aligned} \tag{14}$$

where

$$\begin{aligned} C_1 &= 2(1-s)s(1-t)^2 \\ C_2 &= 2(1-s)^2(1-t)t \\ C_3 &= 2(1-s)s(1-t)t \\ C_4 &= 2(1-s)s(1-t)t \\ C_5 &= 2s^2(1-t)t \\ C_6 &= 2(1-s)st^2. \end{aligned} \tag{15}$$

As the derivation of (14) breaks down the original equation into a series of summations, these $\sum_{i,j}$ terms can all be computed very efficiently via integral images, in the same manner as described in Section IV-B.

In order to compute $\widehat{\mathrm{NCC}}_4(x_0, y_0, u, v)$, four additional integral images are required, i.e.,

$$\begin{aligned}&g(x,y)g(x+1,y)\\&g(x,y)g(x,y+1)\\&g(x,y)g(x+1,y+1)\\&g(x,y+1)g(x+1,y+1).\end{aligned}\tag{16}$$

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

$$\begin{aligned}&\overline{g}(x_1+s,y_1+t)\\&\quad=(1-s)(1-t)\overline{g}(x_1,y_1)\\&\qquad+s(1-t)\overline{g}(x_1+1,y_1)\\&\qquad+(1-s)t\overline{g}(x_1,y_1+1)\\&\qquad+st\overline{g}(x_1+1,y_1+1).\end{aligned}\tag{17}$$

By combining (14) and (17), we are able to compute NCC values accurate to the subpixel level. The calculation is done efficiently by solving the summation terms using integral images. As all the integral images only need to be computed once, they will not increase the computational complexity in a significant way for a given stereo pair. It is important to note that the subpixel measurements are used only when calculating correlation coefficients.

At this point, we have one correlation coefficient for each disparity value at each location of the image. For example, if our subregion is of size 24 × 24, and our disparity range is from 0 through 9, then we have a total of $(24 \times 24 \times 10)$ correlation coefficients. This $(X \times Y \times D)$ volume of coefficients is then used as input to the path-based matching in order to find the final disparity values for each pixel pair in the stereo images. In our implementation, this step is performed in the exact same manner as proposed by Sun, which was described in Section III-B.

V. Experimental Results

Here, we provide several examples to illustrate the improved results obtained by our proposed algorithm. Since our method is an extension of the work by Sun [8], we provide direct comparisons between his results and ours. The results presented here show the feasibility of our method, as well as illustrate the improvements over the original algorithm. All data sets were either obtained from Sun's publication [8] and the stereo-image database of Scharstein and Szeliski [11], or generated by us with a Bumblebee stereo camera. For some of the experiments presented here, we do not show the entire image frame in our examples. Rather, we show cropped regions of the results in order to better illustrate the finer differences obtained between the two algorithms. For all the experiments, a window size of 11 × 11 pixels was used. All experiments illustrated here were carried out with relatively small subregions (no larger than 100 × 100 pixels) in order to exploit and better illustrate the advantages of our subpixel correlation measurements and better demonstrate the merits of our approach.

A. Initial Results

This first set of results, shown in Figs. 2 and 3, was initially published in our preliminary paper [35]. The first example illustrated in Fig. 2(a) presents results using stereo images obtained from the test data of Scharstein and Szeliski [11]. The left image shows one of the stereo images, i.e., a baseball against a noisy background. The second image shows the disparity map obtained by Sun's algorithm. Notice that the algorithm obtains good matching of the image points. 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. This example shows a clear advantage of our subpixel measurement approach.

The second example in Fig. 2(b) presents an outdoor scene from the test data of Scharstein and Szeliski [11]. 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 Fig. 2(c) presents an image of the ground taken from a 45° 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. This example illustrates the path-based matching algorithm's ability to find smooth paths along the intermediate 3-D volume of correlation coefficients when these coefficients are obtained with subpixel accuracy. Although the result of Sun's method is smooth, it is still outperformed by our approach.

The fourth example illustrated in Fig. 3(a) combines the image of the third experiment [illustrated in Fig. 2(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 in relation to the car.

The fifth example presented is illustrated in Fig. 3(b). The input images were a stereogram pair obtained from the publication by Sun [8]. 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 region.

B. Accuracy Measurements

As shown in Section V-A, our proposed method of including subpixel measurements into the correlation calculations exhibits clear advantages over the previously published work by Sun [8]. Our results show improvements in finding accurate boundaries between the objects in the scene, but sometimes may sacrifice small image details. Here, we perform a matching on images obtained from the data set provided by Scharstein and Szeliski [11]. We compare results of our method as well as Sun's previously published method against ground-truth images. Distances between the disparity images and the ground-truth images are quantified using well-known error metrics.

This set of experiments contains comparison results for seven different data sets, illustrated in Figs. 4 and 5. For each row in the figures, four images are displayed: The leftmost image

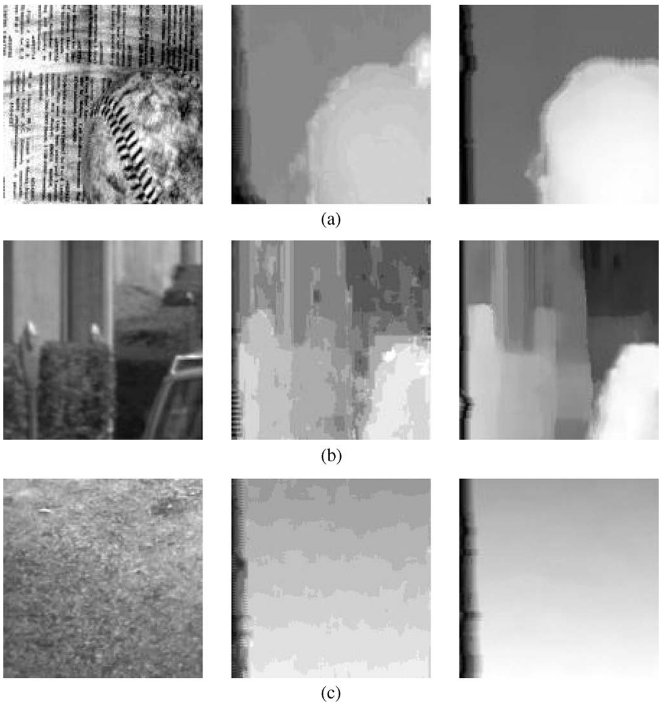


Fig. 2. Disparity map for (left images, only one image shown in each pair) stereo pairs using (middle images) method in [8] and (right images) proposed method. (a) Baseball pair. (b) Park meter scene. (c) Outdoor ground.

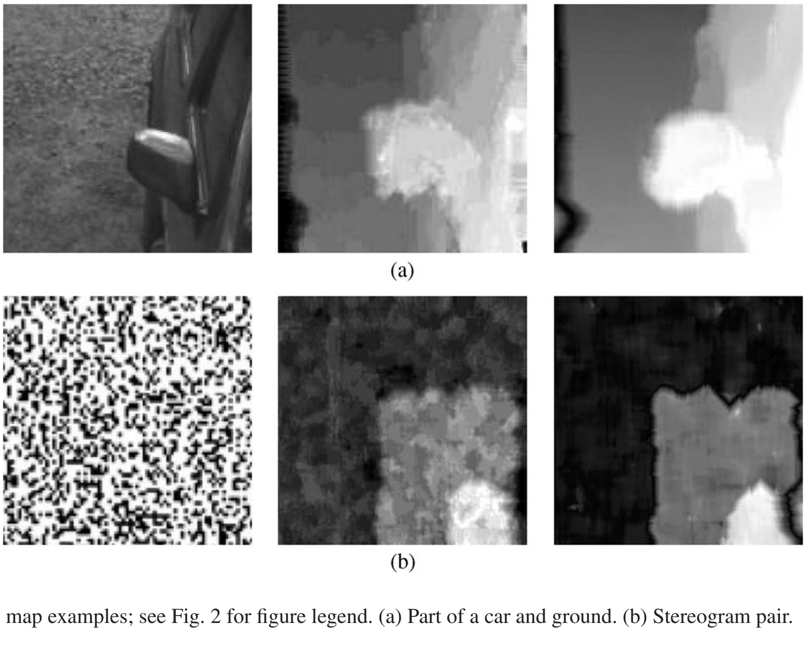


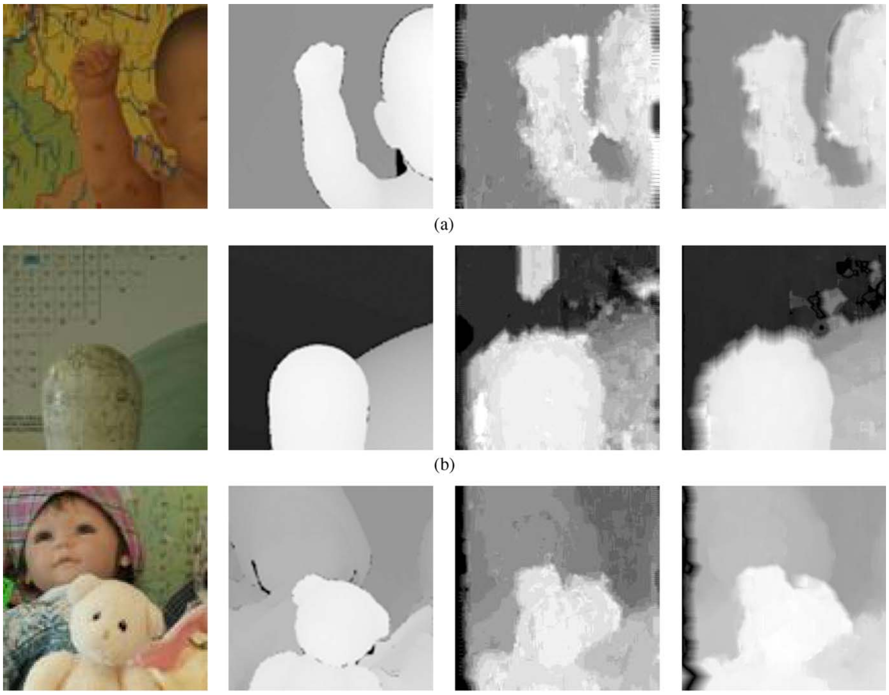
Fig. 3. Two additional disparity map examples; see Fig. 2 for figure legend. (a) Part of a car and ground. (b) Stereogram pair.

is one of the original stereo images; the second image is the ground-truth disparity image; the third image is Sun's disparity matching; finally, the fourth image is our algorithm's result for disparity matching. Each experiment was calculated over an arbitrary image region of pixel size 100 × 100. Only pixels for which occlusion does not occur were taken into consideration (occluded pixels are displayed as black in the ground-truth image). Although the path-based algorithm does a good job of estimating disparity values for regions where occlusion occurs, we decided to only take into account regions for which the ground-truth value is known. Both the disparity images and the ground-truth images were normalized (by dividing the largest disparity value in the corresponding ground-truth image) prior to any comparison.

In order to quantify the comparison results, two different error metrics were used: RMS and bad matching pixels (BMP) [36]. Each is defined as follows:

$$\mathrm{RMS} = \left(\frac{1}{N} \sum_{x,y} |d_I(x,y) - d_T(x,y)|^2 \right)^{\frac{1}{2}}$$

$$\mathrm{BMP} = \frac{1}{N} \sum_{x,y} \left(|d_I(x,y) - d_T(x,y)| > \delta_d \right) \qquad (18)$$



(a)

(b)

(c)

Fig. 4. Disparity maps for stereo pairs. The first column shows one of the original images, the second column shows the ground truth, the third column shows the results using the method by Sun [8], and the fourth column shows the results with the proposed method. (a) Baby doll against textured background. (b) Bowling ball and pin against partially textured background. (c) Several dolls at different distances.

where d_I is the calculated disparity image, d_T is the ground-truth disparity image, and δ_d is a disparity error tolerance (i.e., the allowed error threshold). Note that each disparity image was normalized to have values between 0 and 1. In the following experiments, an error tolerance of $\delta_d = 0.05$ was used.

The quantified results of comparing each of the disparity images against the ground truth are shown in Table I. These error measurements illustrate the distance between each disparity image and its corresponding ground truth; for clarity, these measurements are also illustrated in the graphs of Fig. 6. The vertical axis of the graph represents the error values, and the horizontal axis represents each of the seven experiments in Table I (shown in Figs. 4 and 5). Although the difference between error measurements may vary for a given example, both graphs follow the same general trend and show that our method almost always outperforms Sun's method by generating more accurate disparity maps that contain fewer errors.

Visual analysis of the disparity images illustrates that our results almost always look far more accurate than Sun's original method. Our method is able to provide matching that is closer overall to the ground truth than the matching obtained by Sun's method. The error measurements show that for the first three experiments presented (Fig. 4), we achieve results significantly closer to the ground truth.

This first example in Fig. 4 shows a partial view of a doll against a textured background. The arm of the doll is more clearly visible in our result than Sun's, and the object distances more closely match the ground truth. The second example shows the top of a bowling pin in front of a bowling ball, with a partially textured background. Here, both methods achieve good results for the bowling pin. The ball is a bit more clearly defined in our method. The background is also more accurately defined in our method, whereas Sun's method is unsuccessful in determining the proper disparity. Both methods include an area of the background with incorrect disparity measurements, caused by the complete lack of texture in the region. The final example of Fig. 4 shows a series of dolls at different depths. The bear in the foreground is much more clearly defined in our method. Also, the boundary between the furthest doll and the background is more clearly defined by our method, whereas Sun's approach fails to detect this boundary in a clean manner.

The first example in Fig. 5(a) shows the top corner of a lampshade against a textureless background. Both disparity maps for this image look similar, but our map is able to calculate a more accurate boundary between the foreground and background objects. It is also able to obtain a more accurate disparity value for the foreground object, one closer to the values of the ground-truth image. Interestingly, both methods seem to shift the location of the boundary between the lamp and the background toward the top of the frame. This is due to the shortest path calculation containing small errors due to the ambiguity within each object (since they contain no textures).

The next example in Fig. 5(b) shows the lower corner of a lampshade (in the upper left corner of the image), a section of a pillow, part of a hat, and a textureless background. This example proved challenging for both algorithms, as neither of the methods achieved accurate boundaries for the disparity map. Our method was at least able to calculate relative object distances by exploiting the subpixel accuracy measurements calculated from the image. Sun's method even contains what appears to be a sinusoidal-like pattern across the image, presumably caused by errors in the correlations leading to incorrect shortest path calculations.

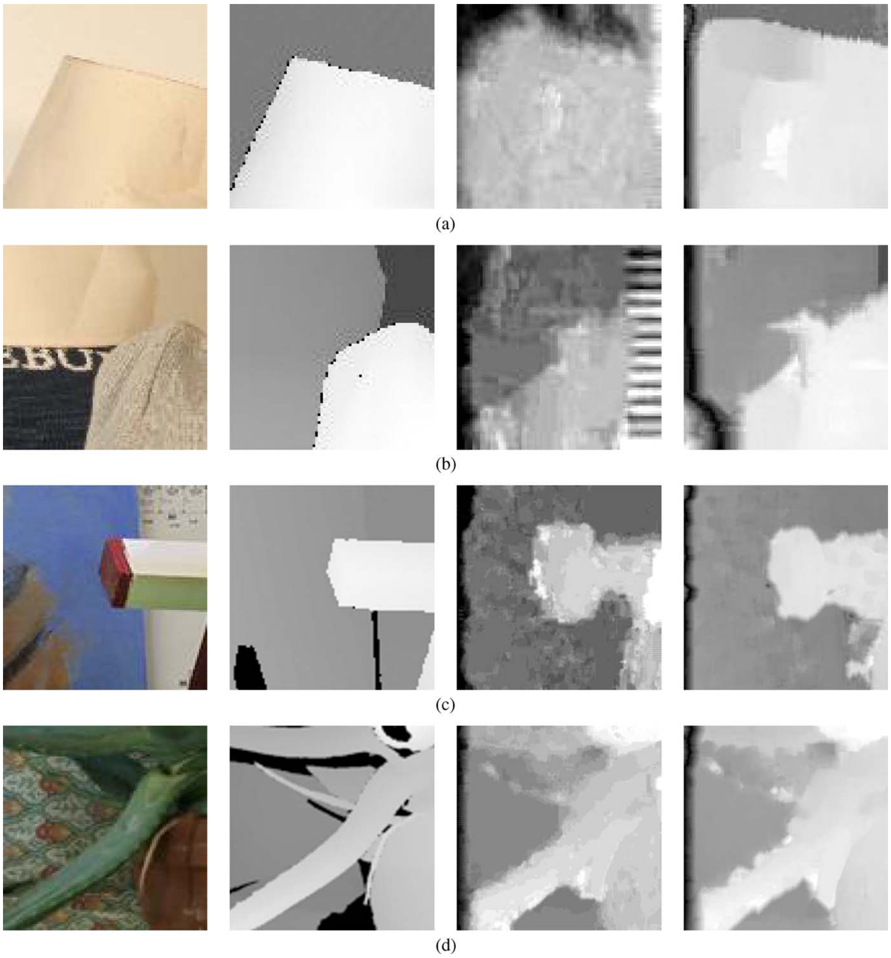


Fig. 5. Disparity maps for stereo pairs. The first column shows one of the original images, the second column shows the ground truth, the third column shows the results using the method by Sun [8], and the fourth column shows the results using the proposed method. (a) Top corner of a lampshade. (b) Part of a hat and textured fabric. (c) Protruding object against a partially textured background. (d) Aloe plant.

TABLE I
ERROR VALUES FOR FIGS. 4 AND 5 DATA

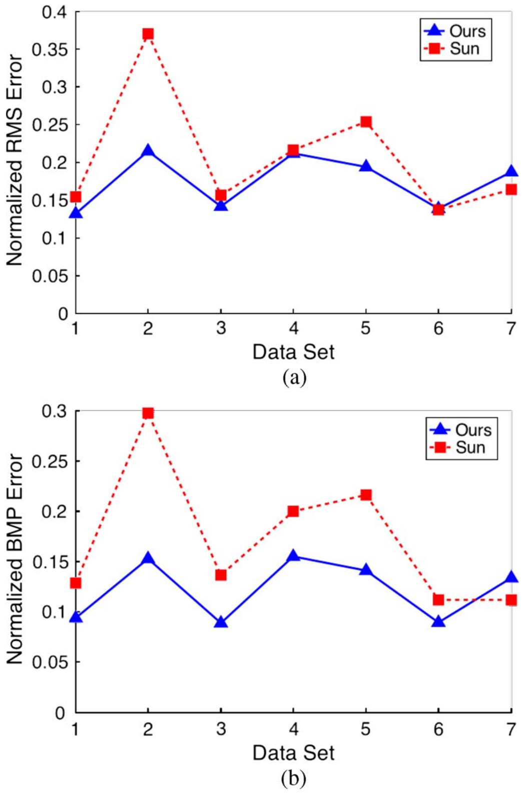
Data	Algorithm	RMS	BMP
1. Baby	Sun	0.1544	0.1286
	Ours	0.1320	0.0941
2. Bowling	Sun	0.3706	0.2975
	Ours	0.2146	0.1528
3. Dolls	Sun	0.1568	0.1368
	Ours	0.1414	0.0891
4. Lamp	Sun	0.2164	0.2001
	Ours	0.2119	0.1549
5. Hat	Sun	0.2538	0.2159
	Ours	0.1942	0.1409
6. Teddy	Sun	0.1375	0.1119
	Ours	0.1385	0.0895
7. Aloe	Sun	0.1638	0.1120
	Ours	0.1874	0.1340

The third example in Fig. 5(c) shows a post coming from the right side of the window, against a partially textured background. Although our disparity map appears visually closer to the ground truth, Sun's method achieved a smaller error value using the RMS metric. Our method achieved a smaller error according to the BMP error, however. In the final example illustrated in Fig. 5(d), Sun's method provided results that are closer to the ground truth according to both error measurements. This example shows that subpixel accuracy does not improve the results 100% of the time. This may be caused by either suboptimal segmentation of regions in the subregioning step or noise in the images. Either of these cases may introduce errors in the correlation process, which, when large enough, may cause the path-based algorithm to calculate an incorrect disparity value. However, from our experiments, we observe that such errors are not a common occurrence.

C. Drawbacks and Limitations

Although our proposed method is capable of obtaining excellent results efficiently, it does not always obtain perfect or acceptable matching. Analyzing the images visually, our method successfully provides smaller errors than Sun's algorithm, as well as better boundaries between the objects in the scene. Our disparity images also have a smoother appearance along surfaces, reducing noise in sections of the disparity images caused by incorrect matches. Our method, however, sometimes sacrifices the smaller visual details of the scene.

At times, textureless regions may cause the path-based matching step to generate errors in the matching. Such an error is illustrated in Fig. 4(b). In our disparity image, near the top of the bowling ball (near the right border of the image), one can see errors in the disparity. It does not always occur,



(a)

(b)

Fig. 6. Error graphs for the data in Figs. 4 and 5. (a) RMS error. (b) BMP error.

but at certain times, the path-based approach will get lost in textureless regions. Notice that Sun's result contains the same error. The error is more prominent in ours due to the subpixel accuracy.

Another drawback of this method is the fact that the path-based algorithm introduces artifacts to the left column of the images due to the indexing going out of bounds in this region. This is a minor drawback, however, that is easily overcome by adding each side of the images, as discussed by Sun [8].

Currently, the method only achieves an accurate disparity map if the objects in each subregion of the scene have a maximum disparity of no more than ten pixels. Note that this limitation is for the maximum disparity within a subregion only, not the entire image. For example, if one subregion contains a region with a disparity range between 1 and 10, and another subregion contains a disparity range from 55 to 65, this will not present a problem for our method since the size of the range within each subregion is no larger than 10 pixels. At times, however, the subregioning step will combine several areas with very different disparities, giving the subregion a very large disparity range. In this case, the algorithm may produce errors in the final disparity images.

Although beyond the scope of this paper, this problem may be overcome by incorporating a more accurate method of calculating the initial subregions. Leung *et al.* [27] address this problem by replacing Sun's original rectangular subregioning method with one based on quad-tree decomposition. Their method should provide better subregioning results that are able to handle large depth discontinuities, as regions are not limited to rectangular ones. It is done by splitting the image into four regions (by employing the use of quad trees) and repeatedly recursing on regions containing large depth discontinuities. Note that these depth discontinuities must be estimated before the algorithm may begin. Combining the quad-tree decomposition with our subpixel correlation metric should provide a powerful and efficient stereo-matching algorithm. Several other methods address the problem using adaptive window sizes on the stereo images (e.g., [19], [24], and [37]).

D. Time Complexity

Our experiments clearly show the advantages of our approach. By using subpixel measurements during the cross-correlation measurements, we are able to achieve accurate matching closer to the ground truth. Compared with the algorithm by Sun [8], our algorithm provides smoother and more accurate disparity maps, with sharper and more refined boundaries. The extra calculations cost roughly twice the amount of work than that of the original algorithm, but the time complexity remains linear. As explained by Sun [8], the time complexity of the algorithm is $O(WHD)$, where W and H are the image dimensions, and D describes the disparity range (essentially, $W \times H \times D$ is the size of the 3-D correlation coefficient volume). Because of the rectangular subregioning process, the disparity range for each subregion can be reduced, thereby reducing the time complexity of the algorithm to $O(W_i H_i D_i)$, where $W_i \leq W$, and so on. Since the integral images can be computed from one pass of the image and need only to be calculated once for every stereo pair, the increased computational cost does not affect computation time in any significant way.

VI. Conclusion

The use of stereo images in computer vision is very popular for applications where the depth among objects in a scene must be estimated. As previously mentioned, one of the key problems in stereo vision is to find accurate point correspondences between the objects in the image pair. We have presented an efficient subpixel accuracy path-based matching algorithm. Our algorithm is based on the previous work by Sun [8]. In order to improve the accuracy of the results while maintaining an efficient computational cost, we propose several major improvements to Sun's original method, including the incorporation of subpixel measurements into the correlation coefficient computation, as well as the use of integral images in order to perform the calculations efficiently.

Our approach incorporates the use of bilinear interpolation between pixel locations combined with an NCC measurement in order to provide correlation coefficients accurate down to the subpixel level. In contrast to Sun's method, which incorporates subpixel accuracy by fitting a quadratic function after the correlation values are computed, our method provides a clear improvement on the final disparity maps.

In order to keep computational costs at a minimum, the correlation measurements employ the use of integral images in order to reduce correlation measurement costs. Since integral images are only required to be computed once for a given stereo pair, the increased amount of work does not have any effect on the time complexity of the algorithm, which remains linear in terms of W, H, and D.

Our experiments illustrate the improvements of our proposed method. We measure the errors using common error metrics, each of which consistently shows that our method is more accurate than the previously proposed approach. Our method achieves smoother disparity regions, with improved boundaries between objects in the scene and more exact distances between objects. Although we limit ourselves to applying our proposed improvement to the path-based matching approach, other methods based on NCC calculations (many of which were discussed in Section II) may benefit from the work presented here.

Acknowledgment

The authors would like to thank Y. Wang for initial implementation and insightful discussions during the development of this paper. Information on the National Centers for Biomedical Computing can be obtained from http://nihroadmap.nih.gov/bioinformatics.

References

[1] M. Z. Brown, D. Burschka, and G. D. Hager, "Advances in computational stereo," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 8, pp. 993–1008, Aug. 2003.

[2] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[3] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Englewood Cliffs, NJ: Prentice-Hall, 2003.

[4] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vis. Conf.*, 1988, pp. 147–151.

[5] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[6] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. IEEE Int. Conf. Comput. Vis.*, 1999, pp. 1150–1157.

[7] P. Anandan, "A computational framework and an algorithm for the measurement of visual motion," *Int. J. Comput. Vis.*, vol. 2, no. 3, pp. 283–310, Jan. 1989.

[8] C. Sun, "Fast stereo matching using rectangular subregioning and 3d maximum-surface techniques," *Int. J. Comput. Vis.*, vol. 47, no. 1–3, pp. 99–117, Apr.–Jun. 2002.

[9] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, May 2004.

[10] H. Bay, A. Ess, T. Tuylelaars, and L. V. Gool, "Surf: Speeded up robust features," *Comput. Vis. Image Underst.*, vol. 11, no. 3, pp. 346–359, 2008.

[11] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, no. 1, pp. 7–42, Apr. 2002.

[12] P. de la Hamette and G. Tröster, "Architecture and applications of the FingerMouse: A smart stereo camera for wearable computing HCI," *Pers. Ubiquitous Comput.*, vol. 12, no. 2, pp. 97–110, Feb. 2008.

[13] D. Knoblauch and F. Kuester, "Virtualizeme: Interactive model reconstruction from stereo video streams," in *Proc. ACM Symp. Virtual Reality Softw. Technol.*, 2008, pp. 193–196.

[14] P. Sand and S. Teller, "Video matching," *ACM Trans. Graph.—Special Interest Group on Graphics and Interactive Techniques*, vol. 23, no. 3, pp. 592–599, Aug. 2004.

[15] U. Franke and A. Joos, "Real-time stereo vision for urban traffic scene understanding," in *Proc. IEEE Intell. Veh. Symp.*, 2000, pp. 273–278.

[16] G. L. Morgan, J. G. Liu, and H. Yan, "Sub-pixel stereo-matching for DEM generation from narrow baseline stereo imagery," in *Proc. Geosci. Remote Sens. Symp.*, 2008, vol. 3, pp. 1284–1287.

[17] J. G. Liu and H. Yan, "Phase correlation pixel-to-pixel image coregistration based on optical flow and median shift propagation," *Int. J. Remote Sens.*, vol. 29, no. 20, pp. 5943–5956, Oct. 2008.

[18] R. Szeliski and D. Scharstein, "Symmetric sub-pixel matching," in *Proc. Eur. Conf. Comput. Vis.*, 2002, pp. 525–540.

[19] G.-B. Kim and S.-C. Chung, "An accurate and robust stereo matching algorithm with variable windows for 3D measurements," *Mechatronics*, vol. 14, no. 6, pp. 715–735, Jul. 2004.

[20] I. Sarkar and M. Bansal, "A wavelet-based multiresolution approach to solve the stereo correspondence problem using mutual information," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 4, pp. 1009–1014, Aug. 2007.

[21] G. Pajares and J. M. de la Cruz, "On combining support vector machines and simulated annealing in stereovision matching," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 4, pp. 1646–1657, Aug. 2004.

[22] Y. Zhang and C. Kambhamettu, "On 3-D scene flow and structure recovery from multiview image sequences," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 33, no. 4, pp. 592–606, Aug. 2003.

[23] B. M. Roland Brockers and M. Hund, "Stereo matching with occlusion detection using cost relaxation," in *Proc. IEEE Int. Conf. Image Process.*, 2005, pp. 389–392.

[24] G.-B. Kim, "A hybrid stereo vision method using a correlation function and variable window based on voxel components," *Int. J. Adv. Manuf. Technol.*, vol. 31, no. 5, pp. 546–555, Dec. 2006.

[25] I. D. Rosenberg, P. L. Davidson, C. M. Muller, and J. Y. Han, "Real-time stereo vision using semi-global matching on programmable graphics hardware," in *Proc. SIGGRAPH*, 2006, p. 89.

[26] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Proc. IEEE Comput. Soc. Conf. CVPR*, 2005, pp. 807–814.

[27] C. Leung, B. Appleton, and C. Sun, "Fast stereo matching by iterated dynamic programming and quadtree subregioning," in *Proc. Brit. Mach. Vis. Conf.*, London, U.K., 2004, pp. 97–106.

[28] C. Sun, "Multi-resolution rectangular subregioning stereo marching using fast correlation and dynamic programming techniques," CSIRO Math. Inf. Sci., Canberra, Australia, Tech. Rep. 98/246, 1998.

[29] F. C. Crow, "Summed-area tables for texture mapping," in *Proc. Special Interest Group Graph. Interactive Tech.*, 1984, pp. 207–212.

[30] S. Frintrop, M. Klodt, and E. Rome, "A real-time visual attention system using integral images," in *Proc. 5th Int. Conf. Comput. Vis. Syst.*, 2007, pp. 124–151.

[31] B. Kisacanin, "Integral image optimizations for embedded vision applications," in *Proc. IEEE Southwest Symp. Image Anal. Interpretation*, 2008, pp. 181–184.

[32] K.-t. Kim and E. y. Kim, "Automatic film line scratch removal system based on spatial information," in *Proc. IEEE ISCE*, 2007, pp. 1–5.

[33] F. H. P. van Velden, R. W. Kloet, B. N. M. van Berckel, C. F. M. Molthoff, A. A. Lammertsma, and R. Boellaard, "Gap filling strategies for 3-D-FBP reconstructions of high-resolution research tomograph scans," *IEEE Trans. Med. Imag.*, vol. 27, no. 7, pp. 934–942, Jul. 2008.

[34] S. A. Fahmy, "Generalised parallel bilinear interpolation architecture for vision systems," in *Proc. Int. Conf. Reconfigurable Comput. FPGAs*, 2008, pp. 331–336.

[35] A. Donate, Y. Wang, X. Liu, and E. Collins, "Efficient and accurate subpixel path based stereo matching," in *Proc. IEEE Int. Conf. Pattern Recog.*, 2008, pp. 1–4.

[36] M. Bleyer and M. Gelautz, "Graph-cut-based stereo matching using image segmentation with symmetrical treatment of occlusions," *Image Commun.*, vol. 22, no. 2, pp. 127–143, Feb. 2007.

[37] H. Kim, S. jun Yang, and K. Sohn, "3D reconstruction of stereo images for interaction between real and virtual worlds," in *Proc. 2nd IEEE/ACM Int. Symp. Mixed Augmented Reality*, 2003, pp. 169–177.

Arturo Donate received the B.S. degree from the University of Miami, Coral Gables, FL, in 2004 and the M.S. degree in computer science from the Florida Institute of Technology, Melbourne, in 2006. He is currently working toward the Ph.D. degree in computer science at Florida State University, Tallahassee.

His current research interests include the areas of computer vision and pattern recognition. More specifically, his research deals with 3-D reconstruction, SLAM algorithms, and video analysis techniques.

Xiuwen Liu received the Ph.D. degree in computer and information science from Ohio State University, Columbus.

Since 2000, he has been with the Department of Computer Science, Florida State University, Tallahassee, where he is currently an Associate Professor. His current research interests include image analysis for high throughput quantitative biology, spatial and temporal organization modeling of chromatin fibers, pattern analysis, and classification for nucleosome positioning, in addition to computational shape analysis, 3-D representations for computer-vision applications, large-scale object detection and recognition via decision trees and deep-learning architectures, energy-efficient real-time computer-vision algorithms and implementations, visual inference modeling, and machine-learning algorithms and architectures.

Emmanuel G. Collins, Jr. received the B.S. degrees from Morehouse College, Atlanta, GA, and the Georgia Institute of Technology, Atlanta, and the Ph.D. degree in aeronautics and astronautics from Purdue University, West Lafayette, IN, in 1987.

He spent seven years in research and development with Harris Corporation. In August 1994, he joined the Department of Mechanical Engineering, Florida A&M University–Florida State University College of Engineering, Tallahassee, as an Associate Professor, where he is currently a John H. Seely Professor of mechanical engineering and the Director of the Center for Intelligent Systems, Control and Robotics. He teaches courses in control, robotics, and dynamics. His current research interests include control and guidance of autonomous vehicles and electric-powered wheelchairs in extreme environments and situations, coordination of teams of heterogeneous agents (including human–robot teams), flow control, and applications of modern control approaches to energy management.