Advances in Computational Stereo

Paper written by:
Myron Z. Brown, Member, IEEE,
Darius Burschka, Member, IEEE
Gregory D. Hager, Senior Member, IEEE

Presentation for:
Graphics and Scientific Visualization (CS 525)

Presented by:
Mahmud Shahriar Hossain
(http://www.cs.montana.edu/~mshossain)

Date: March 9, 2007

1. Introduction:

The computer vision community has been studying the concept of extraction of three-dimensional structure of a scene from stereo images for decades. This involves the fundamentals of image correspondence and stereo geometry. Stereo research has matured significantly throughout the years and many advances in computational stereo continue to be made, allowing stereo to be applied to new and more demanding problems.

The paper of our discussion reviews recent advances in computational stereo, focusing primarily on three important topics:

1. Correspondence methods
2. Methods for occlusion, and
3. Real-time implementations.

The paper also presents distinctions among key ideas and approaches depending on the recent discoveries.

Before we go into some depth, we first concentrate on some preliminary ideas about Computational Stereo. The following section discusses some theories about computational stereo.

2. Computational Stereo:

Computational stereo refers to the problem of determining three-dimensional structure of a scene from two or more images taken from distinct viewpoints. The fundamental basis for stereo is the fact that a single three-dimensional physical location projects to a

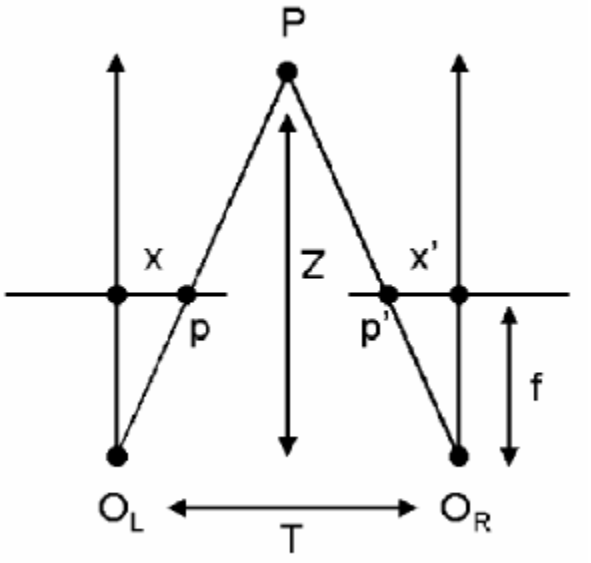

Figure 1: The geometry of nonverged stereo.

unique pair of image locations in two observing cameras. This is shown in Figure 1. As a result, given two camera images, if it is possible to locate the image locations that correspond to the same physical point in space, then it is possible to determine its three-dimensional location.

The problems that computational stereo solves are:

1. Calibration,
2. Correspondence and
3. Reconstruction.

Calibration: It is the process of determining camera system external geometry (the relative positions and orientations of each camera) and internal geometry (focal lengths, optical centers, and lens distortions). Accurate estimates of this geometry are necessary in order to relate image information (expressed in pixels) to an external world coordinate system. There are already some high quality toolkits available for estimating the calibration. e.g., Camera Calibration Toolbox for Matlab. The authors of the paper assume that calibration is already available for the images.

In Figure 1, O_L and O_R are the optical centers. The line O_LO_R is called the baseline, T.

Consider now the camera configuration shown in Figure 1. Both camera coordinates axes are aligned and the baseline is parallel to the camera x coordinate axis. A point in space projects to two locations on the same scan line in the left and right camera images. The resulting displacement of a projected point in one image with respect to the other is termed ***disparity***. The set of all disparities between two images is called a ***disparity map***. Clearly, disparities can only be computed for features visible in both images; features visible in one image but not the other are said to be ***occluded***.

Correspondence: In practice, we are given two images, and, from the information contained in this image, we must compute disparities. The correspondence problem consists of determining the locations in each camera image that are the projection of the same physical point in space. No general solution to the correspondence problem exists, due to ambiguous matches (e.g., due to occlusion, specularities, or lack of texture). Thus, a variety of constraints (e.g., epipolar geometry) and assumptions (e.g., image brightness constancy and surface smoothness) are commonly exploited to make the problem tractable.

Reconstruction: The reconstruction problem consists of determining three dimensional structure from a disparity map, based on known camera geometry. The depth of a point in space P imaged by two cameras with optical centers OL and OR is defined by intersecting the rays from the optical centers through their respective images of P, p, and p'. Given the distance between O_L and O_R, called the baseline T, and the focal length f of the cameras, depth at a given point may be computed by similar triangles as

$$Z = f\frac{T}{d}$$

Where d is the disparity of that point, d=x-x', after being converted to metric units. This process is called triangulation.

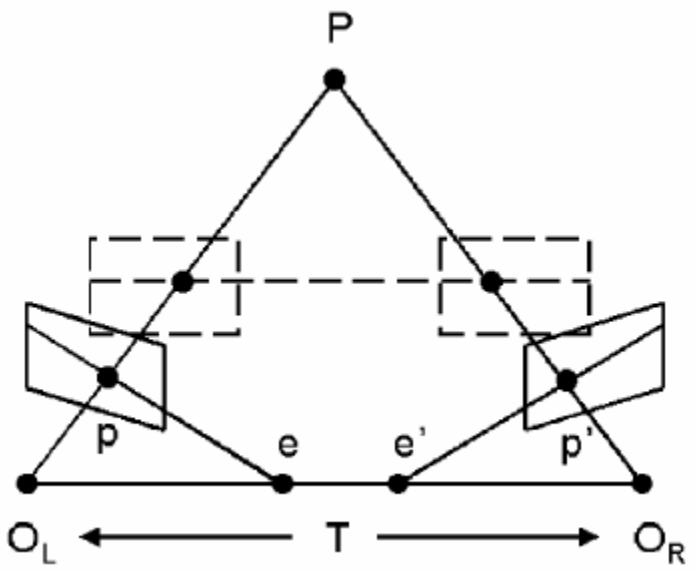

Figure 2: Two arbitrary images of the same scene may be rectified along epipolar lines (solid) to produce collinear scan lines (dashed).

In practice, it is difficult to build stereo systems with nonverged geometry. However, it is well-known that arbitrary stereo image pairs (i.e., with verged geometry) may also be rectified (resampled) to nonverged geometry by exploiting a ***binocular geometric constraint***, commonly referred to as the ***epipolar constraint***. Figure 2 shows the imaging geometry for two cameras with optical centers O_L and O_R. A point P in the scene is imaged by the left and right cameras respectively as points p and p'. The baseline T and optical rays O_L to P andO_R to P define the plane of projection for the point P, called the epipolar plane. This epipolar plane intersects the image planes in lines called epipolar lines. The epipolar line through a point p' is the image of the opposite ray, O_L to P through point p. The point at which an image's epipolar lines intersect the baseline is called the epipole (e and e' for p and p', respectively), and this point corresponds to the image of the opposite camera's optical center as imaged by the corresponding camera. Given this unique geometry, the corresponding point p' of any point p may be found along its respective epipolar line. By rectifying the images such that corresponding epipolar lines lie along horizontal scan-lines, the two-dimensional correspondence search problem is again reduced to a scan-line search, greatly reducing both computational complexity and the likelihood of false matches.

3. Correspondence:

All correspondence methods attempt to match pixels in one image with their corresponding pixels in the other image. For simplicity, we refer to constraints on a small number of pixels surrounding a pixel of interest as ***local constraints***. Similarly, we

Table 1: Stereo Matching Approaches.

APPROACH	BRIEF DESCRIPTION
LOCAL METHODS	
Block Matching	Search for maximum match score or minimum error over small region, typically using variants of cross-correlation or robust rank metrics.
Gradient-Based Optimization	Minimize a functional, typically the sum of squared differences, over a small region.
Feature Matching	Match dependable features rather than intensities themselves.
GLOBAL METHODS	
Dynamic Programming	Determine the disparity surface for a scanline as the best path between two sequences of ordered primitives. Typically, order is defined by the epipolar ordering constraint.
Intrinsic Curves	Map epipolar scanlines to intrinsic curve space to convert the search problem to a nearest-neighbors lookup problem. Ambiguities are resolved using dynamic programming.
Graph Cuts	Determine the disparity surface as the minimum cut of the maximum flow in a graph.
Nonlinear Diffusion	Aggregate support by applying a local diffusion process.
Belief Propagation	Solve for disparities via message passing in a belief network.
Correspondenceless Methods	Deform a model of the scene based on an objective function.

loosely refer to constraints on scan-lines or on the entire image as ***global constraints***. Table 1 outlines the principal methods for exploiting both local and global constraints, excluding methods that rely explicitly on more than two views.

Local methods can be very efficient, but they are sensitive to locally ambiguous regions in images (e.g., occlusion regions or regions with uniform texture). Global methods can be less sensitive to these problems since global constraints provide additional support for regions difficult to match locally. However, these methods are more computationally expensive.

3.1 Local Correspondence Methods:

Local correspondence Methods fall into three broad categories:

1. block matching,
2. gradient methods and
3. feature matching.

Block Matching: Block matching methods seek to estimate disparity at a point in one image by comparing a small region about that point (the template) with a series of small regions extracted from the other image (the search region). The epipolar constraint reduces the search to one dimension. Three classes of metrics are commonly used for block matching: correlation, intensity differences, and rank metrics. Table 2 describes the formulas.

Table 2: Common Block-Matching Methods.

MATCH METRIC	DEFINITION
Normalized Cross-Correlation (NCC)	$\frac{\sum_{u,v}\left(I_1(u,v)-\bar{I}_1\right)\cdot\left(I_2(u+d,v)-\bar{I}_2\right)}{\sqrt{\sum_{u,v}\left(I_1(u,v)-\bar{I}_1\right)^2\cdot\left(I_2(u+d,v)-\bar{I}_2\right)^2}}$
Sum of Squared Differences (SSD)	$\sum_{u,v}\left(I_1(u,v)-I_2(u+d,v)\right)^2$
Normalized SSD	$\sum_{u,v}\left(\frac{\left(I_1(u,v)-\bar{I}_1\right)}{\sqrt{\sum_{u,v}\left(I_1(u,v)-\bar{I}_1\right)^2}}-\frac{\left(I_2(u+d,v)-\bar{I}_2\right)}{\sqrt{\sum_{u,v}\left(I_2(u+d,v)-\bar{I}_2\right)^2}}\right)^2$
Sum of Absolute Differences (SAD)	$\sum_{u,v}\left\lvert I_1(u,v)-I_2(u+d,v)\right\rvert$
Rank	$\sum_{u,v}\left(I_1'(u,v)-I_2'(u+d,v)\right)$ $I_k'(u,v)=\sum_{m,n}I_k(m,n)<I_k(u,v)$
Census	$\sum_{u,v}HAMMING\left(I_1'(u,v),I_2'(u+d,v)\right)$ $I_k'(u,v)=BITSTRING_{m,n}\left(I_k(m,n)<I_k(u,v)\right)$

Normalized cross correlation (NCC) is the standard statistical method for determining similarity. Its normalization, both in the mean and the variance, makes it relatively insensitive to radiometric gain and bias. The sum of squared differences (SSD) metric is computationally simpler than cross correlation, and it can be normalized as well. In addition to NCC and SSD, many variations of each with different normalization schemes have been used. One popular example is the sum of absolute differences (SAD), which is often used for computational efficiency.

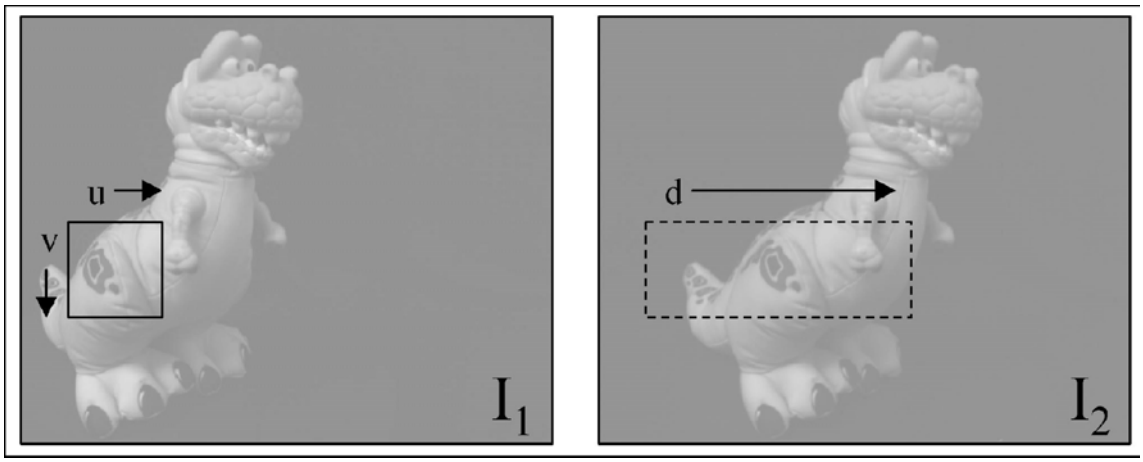

Figure 3: Block matching searches one image for the best corresponding region for a template region in the other image. Correspondence metrics are outlined in Table 2.

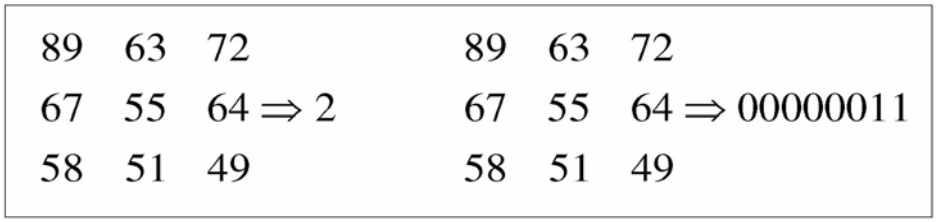

Figure 4: Example rank (left) and census (right) transforms.

Zabih and Woodfill [1] propose an alternative method for computing correspondence by applying local nonparametric transforms to the images before matching. In order to eliminate sensitivity to radiometric gain and bias, a rank transform is applied locally to regions in both images. The rank transform for a local region about a pixel is defined as the number of pixels in that region for which the intensity is less than that of the center pixel. The resulting values are based on the relative ordering of pixel intensities rather than the intensities themselves (Figure 4). Since the magnitudes of these values are much compressed, sensitivity to outliers (e.g., due to occlusion) is also reduced. After the rank transform is applied, block matching is performed using the L1 norm (i.e., sum of absolute differences). While the rank transform method reduces sensitivity to radiometric gain and bias, it also reduces the discriminatory power of the matching procedure since information is lost. The relative ordering of all of the pixels surrounding a given pixel to be transformed is encoded in a single value.

There is a variation of the rank transform, called the ***census transform***, that preserves the spatial distribution of ranks by encoding them in a bit string (Figure 4). Matching is then performed using the Hamming distance (i.e., the number of bits that differ) between bit strings. This transform increases the dimensionality of the image data by a factor of the local region size, making it computationally expensive. This algorithm requires a massively parallel machine for real-time implementation.

The naı¨ve implementation of any block matching method is very inefficient due to redundant computations. For an image with N pixels, a template size of n pixels, and a disparity search range of D pixels, the complexity of naı¨ve block matching is O(NDn) operations. By keeping running block sums, redundant computations may be avoided and block matching complexity may be reduced to O(ND) operations, making it independent of the template size.

Gradient Methods: Gradient-based methods or optical flow, seek to determine small local disparities between two images by formulating a differential equation relating motion and image brightness. In order to do this, the assumption is made that the image brightness of a point in the scene is constant between the two views. Then, the horizontal translation of a point from one image to the other is computed by a simple differential equation,

$$(\nabla_x E)v + E_t = 0$$

where $(\nabla_x E)$ denotes the horizontal component of the image gradient, E_t denotes the temporal (here referring to the intensity differences between left and right stereo images) derivative, and v denotes the translation between the two images. The complexity of matching along epipolar lines using optical flow is simply O(N).

In theory, gradient-based methods can only estimate disparities up to half a pixel since the local derivatives are only valid over that range. Since adjacent image pixels are typically highly correlated, a pixel or more can often be estimated in practice.

Feature Matching: Block matching and gradient methods are well-known to be sensitive to depth discontinuities, since the region of support near a discontinuity contains points from more than one depth. These methods are also sensitive to regions of uniform texture in images. Feature-based methods seek to overcome these problems by limiting the regions of support to specific reliable features in the images. Of course, this also

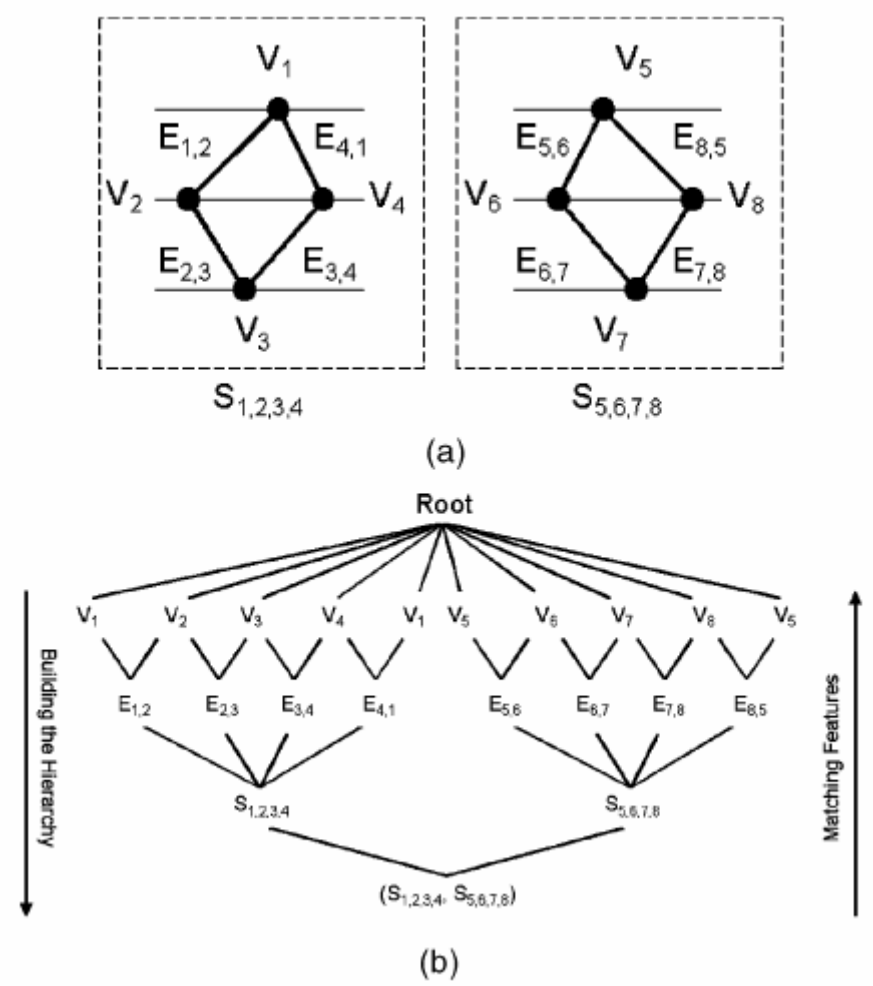

Figure 5: (a) Two hypothesized surfaces and (b) their component features are represented in a relational graph. The hierarchy is built from the lowest level to the highest (vertices, edges, and surfaces, in this example) and matching is performed from the highest to the lowest.

limits the density of points for which depth may be estimated. Due to the need for dense depth maps for a variety of applications and also due to improvements in efficient and robust block matching methods, interest in feature-based methods has declined in the last decade.

Venkateswar and Chellappa [2] have proposed a hierarchical feature-matching algorithm exploiting four types of features: lines, vertices, edges, and edge-rings (i.e., surfaces). Matching begins at the highest level of the hierarchy (surfaces) and proceeds to the lowest (lines). It allows coarse, reliable features to provide support for matching finer, less reliable features, and it reduces the computational complexity of matching by reducing the search space for finer levels of features. First, edges are extracted and the feature hierarchy is built from the bottom up based on structural (i.e., connectivity) and perceptual (i.e., parallel, collinear, and proximate) relationships. Incompatibility relations (e.g., intersects, overlaps, and touches) are also used to enforce consistent feature groupings. All potential features in the hierarchy are stored as hypotheses in a relational graph (see Figure 5). Inconsistent groupings are pruned from the graph by a truth maintenance system (TMS). Feature matching is then performed between the relational graphs of the stereo images, beginning with surfaces and proceeding to lines. For two surfaces to match, their component edges must match, etc. Once a higher-level feature match has been confirmed, the component features are no longer included in the search for other lower-level matches since a feature cannot belong to more than one group. This reduces the search space significantly at each level of the hierarchy. The authors report the matching complexity of each level of the hierarchy to be $O(N^4)$, where N is the number of features examined at that level, and the typical improvement provided by hierarchical matching to be a factor of 100. Of course, the improvement will vary with the percentage of higher-level features successfully matched.

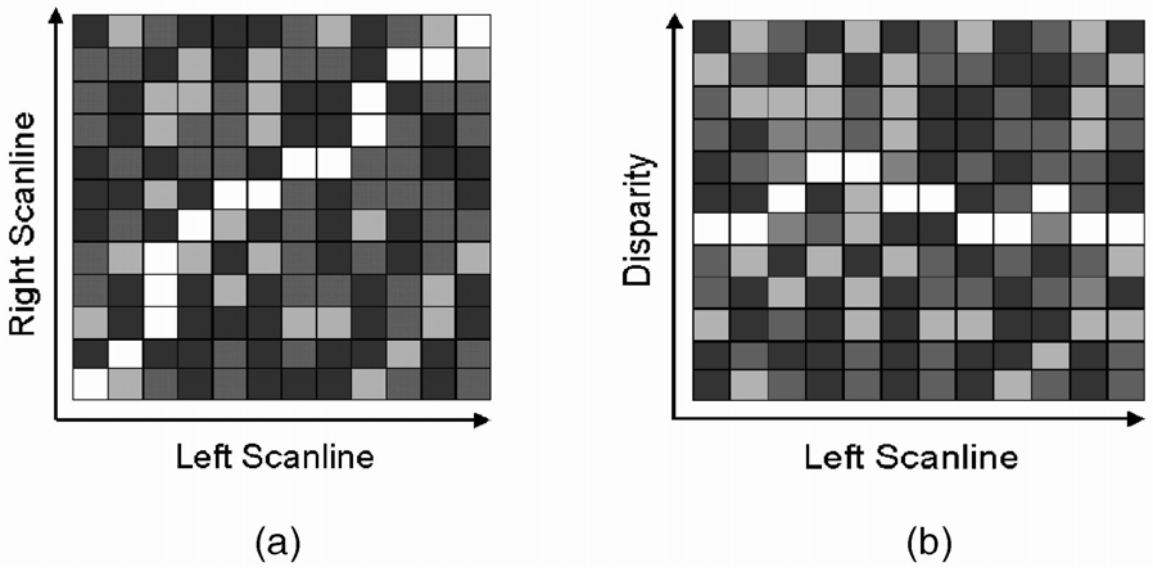

Figure 6: (a) An example disparity-space image using left-right axes and (b) another using left-disparity axes. Intensities shown represent the respective costs of potential matches along the scan-lines, with lighter intensities having lower cost.

3.2 Global Correspondence Methods:

So far we have talked about the local correspondence methods. No we shall look at some of the techniques that are regarded as global correspondence methods. Global correspondence methods exploit non-local constraints in order to reduce sensitivity to local regions in the image that fail to match, due to occlusion, uniform texture, etc. The use of these constraints makes the computational complexity of global matching significantly greater than that of local matching. We shall discuss two global correspondence approaches. They are:

1. Dynamic Programming and
2. Intrinsic Curves

Dynamic Programming Approach: Dynamic programming is a mathematical method that reduces the computational complexity of optimization problems by decomposing them into smaller and simpler sub-problems. For stereo matching, the epipolar monotonic ordering constraint allows the global cost function to be determined as the minimum cost path through a disparity space image (DSI). There are two ways to construct a DSI, shown in Figure 6.

1. The axes may be defined as the left and right scan lines
2. Define the axes as the left scan line and the disparity range

One of the principal advantages of dynamic programming, or any global search method, is that it provides global support for local regions that lack texture and would otherwise be matched incorrectly.

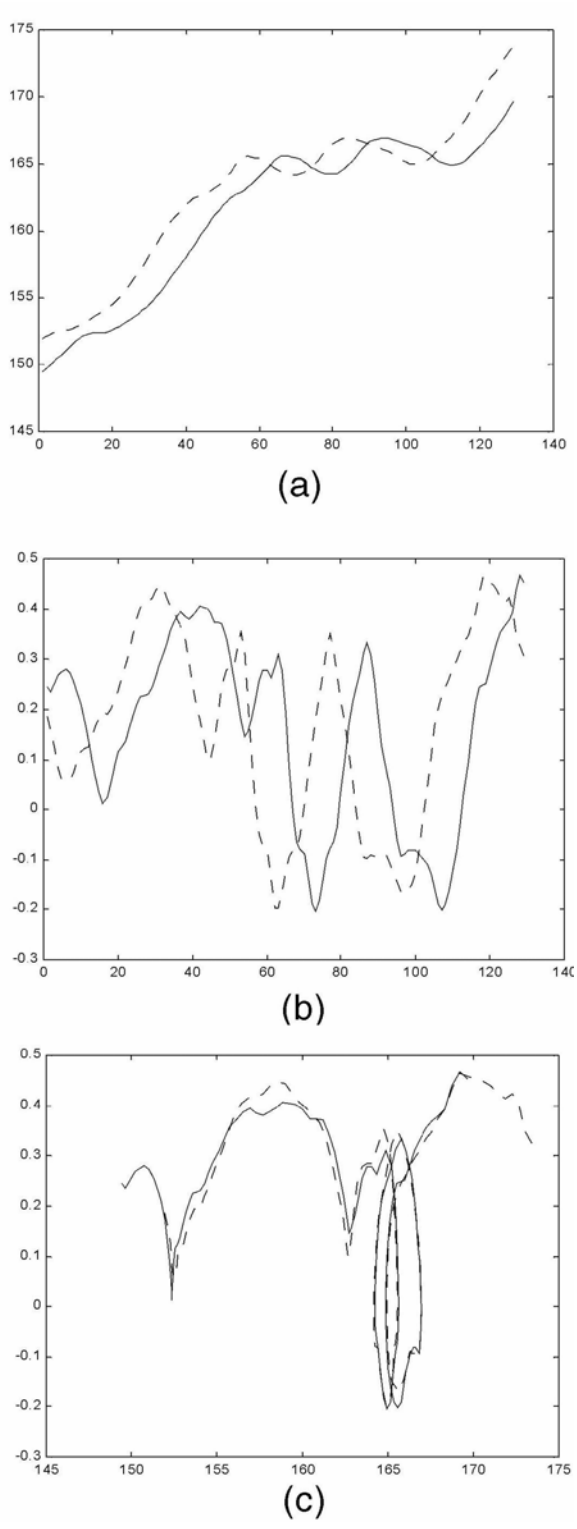

Figure 7: (a) Left and right image scanline intensities, (b) their derivatives, and (c) the intrinsic curves formed by plotting one against the other.

The principal disadvantage of dynamic programming is the possibility that local errors may be propagated along a scan-line, corrupting other potentially good matches.

Intrinsic Curves Approach: Tomasi and Manduchi [3] propose an alternative to conventional search for global matching, using a different representation of image scanlines, called intrinsic curves. An intrinsic curve is a vector representation of image descriptors defined by applying operators (e.g., edge and/or corner operators) to the pixels in a scanline. A simple example is shown in Figure 7. The intrinsic curves here are defined by plotting the intensities of scanline pixels against their respective derivatives. Thismappingis invariant to translation (i.e., disparity), so, in the ideal case, matching pixels map to the same points along a common curve. In the general case, however, due to noise and perspective differences, matching pixels do not always map to exactly the same points, as can be seen in Figure 7. Thus, the disparity search problem is cast in intrinsic curve space as a nearest neighbor problem.

The principal benefit of the intrinsic curve representation is its invariance to disparity. The nearest-neighbors distances between points on two curves representing left and right scanlines are not directly affected by the amount of disparity between them in image space. Therefore, multi-resolution methods commonly used to reduce computational complexity are unnecessary with this approach. Of course, this representation is still affected by occlusion and uniform or repetitive texture in the imaged scene.

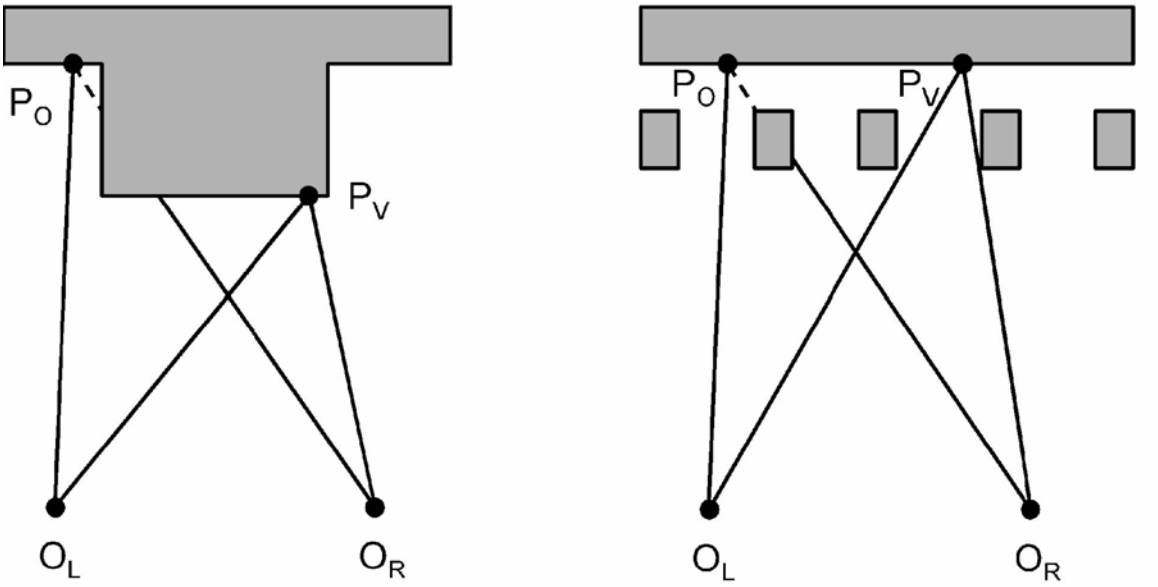

Figure 8: Examples of typical occlusion (left) and less common narrow occlusion (right) that may be observed when viewing a fence, for example. The points PV in the examples are visible to both cameras, so their depths may be estimated by stereopsis. The half-occlusion points PO in the examples are only visible in one image, so their depths may not be estimated.

4. The Problem of Occlusion:

The occlusion problem in stereovision refers to the fact that some points in the scene are visible to one camera but not the other, due to the scene and camera geometries. Figure 8 depicts two scenes, each with two points: one, P_V, being visible to both cameras and the other, P_O, visible to only one camera. We call the point P_O half occluded because it is occluded in one of the views and not the other. While the depth at point P_V may be computed by stereopsis, the depth at P_O is inestimable, unless additional views areaddedonwhich the point is not occluded or assumptions are made about the scene geometry. The half-occlusion in the left example of Figure 8 is commonly observed in most scenes. The half-occlusion observed in the right example is less common since narrow structures (e.g., fences) that might obstruct one's view are simply not observed in most scenes.

The simplest approaches to handling occlusion regions merely attempt to detect them either before or after matching. These regions are then either interpolated based on neighboring disparities to produce a dense depth map or simply not used for applications requiring only a sparse depth map. The most common approach of this type is to detect discontinuities in the depth map itself after matching. Median filters are

Table 3: List of Real Time Stereo Implementations:

REAL-TIME SYSTEM	IMAGE SIZE	FRAME RATE	RANGE BINS	METHOD	PROCESSOR	CAMERAS
INRIA 1993	256x256	3.6 fps	32	Normalized Correlation	PeRLe-1	3
CMU iWarp 1993	256x240	15 fps	16	SSAD	64 Processor iWarp Computer	3
Teleos 1995	320x240	0.5 fps	32	Sign Correlation	Pentium 166 MHz	2
JPL 1995	256x240	1.7 fps	32	SSD	Datacube & 68040	2
CMU Stereo Machine 1995	256x240	30 fps	30	SSAD	Custom HW & C40 DSP Array	6
Point Grey Triclops 1997	320x240	6 fps	32	SAD	Pentium II 450 MHz	3
SRI SVS 1997	320x240	12 fps	32	SAD	Pentium II 233 MHz	2
SRI SVM II 1997	320x240	30+ fps	32	SAD	TMS320C60x 200MHz DSP	2
Interval PARTS Engine 1997	320x240	42 fps	24	Census Matching	Custom FPGA	2
CSIRO 1997	256x256	30 fps	32	Census Matching	Custom FPGA	2
SAZAN 1999	320x240	20 fps	25	SSAD	FPGA & Convolvers	9
Point Grey Triclops 2001	320x240	20 fps 13 fps	32	SAD	Pentium IV 1.4 GHz	2 3
SRI SVS 2001	320x240	30 fps	32	SAD	Pentium III 700 MHZ	2

commonly used to eliminate outliers in depth maps, which are often caused by occlusion regions.

5. Real-Time Stereo Implementation:

In the past decade, real-time dense disparity map stereo (30 frames per second or faster) has become a reality, making the use of stereo processing feasible for a variety of applications, some of which are discussed in the next section. Until very recently, all truly real-time implementations made use of special purpose hardware, like digital signal processors (DSP) or field programmable gate arrays (FPGA). However, with ever increasing clock speeds and the integration of single instruction multiple data (SIMD) coprocessors (e.g., Intel MMX) into general-purpose computers, real-time stereo processing is finally a reality for common desktop computers. This section reviews the progression of real-time stereo implementations over the past decade. A summary of real-time stereo systems and their comparative performances is provided in Table 3.

6. Conclusion:

Computational stereo is an important research area for the computer vision community. Perhaps the most practically significant advance in the last decade has been the appearance of real-time stereo systems, first on special-purpose hardware and, more recently, on general-purpose computers. Due in large part to these real-time implementations, research on real-time stereo applications has blossomed in the latter part of this decade. Real-time algorithms, however, are still relatively simplistic, and most of the global matching and occlusion handling methods discussed do not currently run in real-time. More demanding potential applications (e.g., virtual reality) require both real-time algorithms and very precise, reliable, and dense depth estimates. Hence a lot of research works are still to be done in this area.

References:

[1] R. Zabih and J. Woodfill, "Non-Parametric Local Transforms for Computing Visual Correspondence," Proc. Third European Conf. Computer Vision, pp. 150-158, 1994.

[2] V. Venkateswar and R. Chellappa, "Hierarchical Stereo and Motion Correspondence Using Feature Groupings," Int'l J. Computer Vision, vol. 15, pp. 245-269, 1995.

[3] C. Tomasi and R. Manduchi, "Stereo Matching as a Nearest- Neighbor Problem," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, pp. 333-340, 1998.