Stereoscopic analysis of multiple images

Sébastien Roy

Jean Meunier

Département d'informatique et de recherche opérationnelle

Université de Montréal

Abstract

A new algorithm to recover depth from a sequence of two or more stereoscopic images is presented. The algorithm, which uses a dynamic programming approach, builds a dense depth map and allows the camera displacement between each image to be any combination of rotation and translation. Since no «smoothing constraint» on depth is used, occlusions and depth discontinuities along the border of objects are preserved and easy to identify. For a given cost function, the algorithm finds the optimal correspondence along epipolar lines. In our case, this function is the difference of intensities between corresponding points, adjusted with a factor accounting for occlusions. Tested on non-trivial synthetic image sequences with true depth map available, we obtain a mean disparity error of less than one pixel.

submitted august 7th 1993 to the *International Journal of Computer Vision*

1 Introduction

Depth estimation of objects in a scene is very useful in many applications of image processing. Stereoscopic fusion, a very important depth estimation mechanism of the human visual system, provides a passive way of extracting depth. By using at least two different views of a scene, the relative displacement (disparity) of each object in a scene provides a simple measure of how deep an object is. Assuming the position of the camera is known for each view, disparity can be obtained for a point in an image by finding the corresponding point in another image. Solving this «correspondence problem» for each point of an image gives a disparity map that can easily be converted into a depth map (Brown 1988, Horn 1986, Jähne 1991, Shirai 1987, Weng 1992b).

In order to be as useful as possible, the depth map must be dense and accurately reflect the depth discontinuities on the borders of objects. Moreover, an occlusion map should also be obtained to provide accurate detection of the parts of an image hidden in other images. This map provides a way to discard occluded regions from the depth map since depth cannot be recovered from those regions.

Although many algorithms can create high quality dense depth map (Fleck 1991, Roy 1992), their usefulness has been somewhat limited by the added work needed to calculate depth for all points of an image. The dynamic programming approach has proven to be a good tool to greatly improve the efficiency while providing accurate depth maps (Cox 1992). Usually, stereoscopic analysis is performed with two images. However, more images can be used to provide a better depth estimation. In particular, trinocular algorithms use three images to obtain more accurate depth map (Ayache 1989, Lee 1990).

The stereoscopic algorithm presented in this paper can efficiently provide a depth map and an occlusion map constructed from the analysis of multiple images obtained with arbitrary camera displacements. The depth map is dense and is not blurred across depth discontinuities. It features an original presentation of general epipolar geometry, and presents an image rectification process that does not depend on absolute camera positions and does not distort disparity measurements.

Unlike many stereoscopic algorithms (Shirai 1987, Ayache 1989, Weng 1992a), the matching process is based on image intensity levels and does not involve any a priori image segmentation or token extraction.

To simplify the stereoscopic analysis, a few important hypotheses have been made. Objects in the scene are assumed opaque and dull, but can overlap and be partially hidden from view. Lighting is assumed constant in position and intensity. The intensity variations induced by lighting on objects when the camera moves are not taken into account and are considered minimal. The camera displacement is an affine transformation, composed of arbitrary rotations and translations and is assumed known before the stereoscopic analysis.

After a preliminary description of the stereoscopic model and the equations needed for stereoscopic analysis, the algorithm is described in details. It is then applied to synthetic image sequences. Performance and accuracy are estimated by comparing the computed depth map with the true depth map.

2 Stereoscopic model

The cameras can be placed anywhere around the scene. Since depth can only be measured in the field of view common to all the images, the camera positions and orientations should maximize the size of this field.

As shown in figure 1, a point P on an object is projected on the projection planes of cameras a and b. The relative distance between the projected points P'_a and P'_b is related to the depth of P.

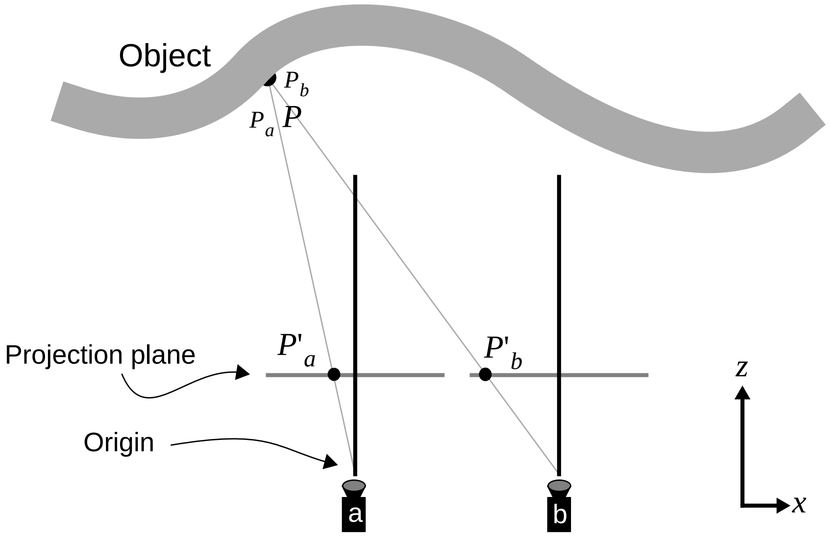


FIGURE 1. Simple stereoscopic model. An object point P has coordinates P_a and P_b relative to camera a and b respectively. They are projected to P'_a and P'_b for each camera.

For a given camera coordinate system, the projection plane is positioned at $z = f$, the focal distance. Homogenous coordinates are used to represent three-dimensional points.

The perspective projection P' of a point P is achieved by the equation

$$P' = H(\boldsymbol{Per} \cdot P)$$

where $H\left(\begin{vmatrix} x \\ y \\ z \\ h \end{vmatrix}\right) = \begin{vmatrix} x/h \\ y/h \\ z/h \\ 1 \end{vmatrix}$ is a homogenization function

and $\boldsymbol{Per} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$ is the projection matrix for focal distance f.

It follows that a point P_a (i.e. P relative to camera a) and its perspective projection P'_a have the form

$$P_a = \begin{vmatrix} x_a \\ y_a \\ z_a \\ 1 \end{vmatrix} \text{ and } P'_a = \begin{vmatrix} x'_a \\ y'_a \\ f \\ 1 \end{vmatrix} \text{ where } x'_a = \frac{fx_a}{z_a} \text{ and } y'_a = \frac{fy_a}{z_a}.$$

The displacement between cameras a and b is an affine transformation described by a homogenous matrix of the form $\boldsymbol{M}_{ab} = (m_{ij})_{ab} = \boldsymbol{R} \cdot \boldsymbol{T}$ where $\boldsymbol{R}$ is a rotation matrix and $\boldsymbol{T}$ is a translation matrix. To simplify notation, m_{ij} is used freely to represent an element of the matrix $\boldsymbol{M}_{ab}$.

2.1 Epipolar vectors

With perspective projection, one projected point $P'_a = (x'_a, y'_a, f, 1)^t$ can be the projection of a line made of points $P_a(z_a)$, of the form

$$P_a(z_a) = \begin{vmatrix} x'_a z_a / f \\ y'_a z_a / f \\ z_a \\ 1 \end{vmatrix}.$$

Projecting those points to the other camera's projection planes gives a set of collinear points $P'_b(z_a) = (x'_b, y'_b, f, 1)^t$ that will form the *epipolar line* (figure 2). We have

$$P'_b(z_a) = H(\boldsymbol{Per} \cdot \boldsymbol{M}_{ab} \cdot P_a(z_a)) = \left| \begin{matrix} f(z_a A_{ab} + m_{14}) / (z_a C_{ab} + m_{34}) \\ f(z_a B_{ab} + m_{14}) / (z_a C_{ab} + m_{34}) \\ f \\ 1 \end{matrix} \right|$$

where

$$A_{ab}(P'_a) = m_{11}\frac{x'_a}{f} + m_{12}\frac{y'_a}{f} + m_{13} \quad , \qquad B_{ab}(P'_a) = m_{21}\frac{x'_a}{f} + m_{22}\frac{y'_a}{f} + m_{23}$$

$$\text{and } C_{ab}(P'_a) = m_{31}\frac{x'_a}{f} + m_{32}\frac{y'_a}{f} + m_{33} \ . \qquad (1)$$

To simplify notation, when A_{ab}, B_{ab} and C_{ab} are used without arguments, P'_a is implied.

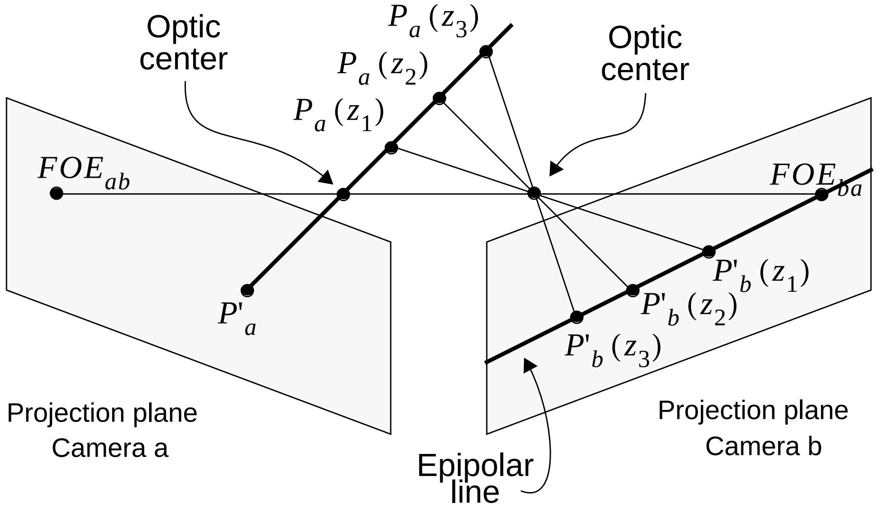


FIGURE 2. Epipolar line. All point $P_a(z_1), P_a(z_2), P_a(z_3)$ that projects on P'_a also projects on the epipolar line as $P'_b(z_1), P'_b(z_2), P'_b(z_3)$. All epipolar lines of points from image *a* intersect at the Focus Of Expansion FOE_{ba} and vice versa (see section 2.2).

Assuming that the depth z_a of any point $P_a(z_a)$ is limited to a known interval, we can reduce the epipolar line to an *epipolar segment*, even if the interval is very large (i.e. from f to ∞). The epipolar segment represents all possible matching points in image b of a point P_a in image a. Let $zmin_a$ and $zmax_a$ represent the bounds of the depth interval allowed for camera a. Let's also assume that they are selected so that any point in this interval projects into the visible parts of the all the other camera's projection planes.

Since the projection of a three-dimensional line is also a line on the projection plane, the epipolar segment can be defined by projecting the endpoints $P_a(zmin_a)$ and $P_a(zmax_a)$. Usually we have $zmin_a > 0$ and $zmax_a = \infty$ but often the camera geometry or the experimental conditions reduce this interval. Figure 3 shows the relation between the depth interval and the epipolar segment.

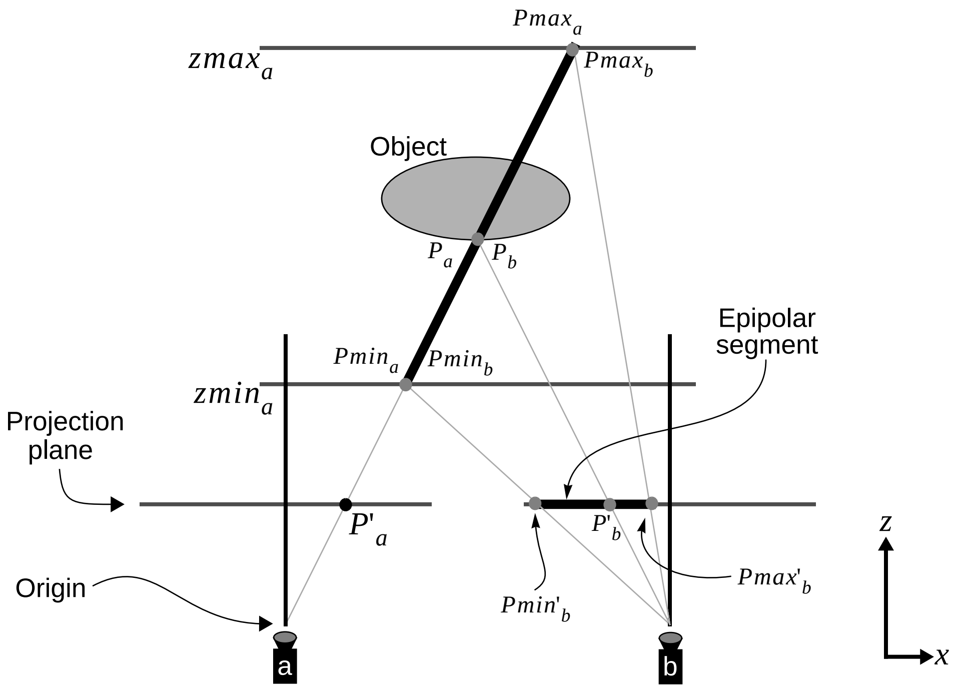


FIGURE 3. Epipolar segment. When z_a varies in the interval $[zmin_a, zmax_a]$, the projection of point $P_a(z_a)$ for camera b varies between $Pmin'_b$ et $Pmax'_b$.

For a given point P'_a, the depth interval $[zmin_a, zmax_a]$ gives a corresponding interval $[zmin_b, zmax_b]$ in the coordinate system of camera b. It is defined as

$$zmin_b = zmin_a C_{ab} + m_{34} \text{ and } zmax_b = zmax_a C_{ab} + m_{34}$$

where C_{ab} is defined as in equation 1.

The endpoints of the depth interval are

$$Pmin_a = P_a(zmin_a) = \begin{vmatrix} x'_a \cdot zmin_a / f \\ y'_a \cdot zmin_a / f \\ zmin_a \\ 1 \end{vmatrix} \text{ and } Pmax_a = P_a(zmax_a) = \begin{vmatrix} x'_a \cdot zmax_a / f \\ y'_a \cdot zmax_a / f \\ zmax_a \\ 1 \end{vmatrix}.$$

After projection to camera b, we obtain the endpoints of the epipolar segment

$$Pmin'_b = H(\boldsymbol{Per} \cdot \boldsymbol{M}_{ab} \cdot Pmin_a) \text{ and } Pmax'_b = H(\boldsymbol{Per} \cdot \boldsymbol{M}_{ab} \cdot Pmax_a)$$

expanding to

$$Pmin'_b = \begin{vmatrix} f(zmin_a A_{ab} + m_{14}) / zmin_b \\ f(zmin_a B_{ab} + m_{24}) / zmin_b \\ f \\ 1 \end{vmatrix}$$

and

$$Pmax'_b = \begin{vmatrix} f(zmax_a A_{ab} + m_{14}) / zmax_b \\ f(zmax_a B_{ab} + m_{24}) / zmax_b \\ f \\ 1 \end{vmatrix}$$

where A_{ab} and B_{ab} are defined by equation 1.

We define $\vec{E}_{ab}(P'_a)$ as an *epipolar vector* representing the direction of all possible points P'_b in image b that can match a given point P'_a in image a. The origin of this vector is given by an initial displacement vector $\vec{M}_{ab}(P'_a)$ defined as the displacement induced by the most distant point $Pmax'_b$ (figure 4). Algebraically, this gives

$$\vec{E}_{ab}(P'_a) = Pmin'_b - Pmax'_b \text{ and } \vec{M}_{ab}(P'_a) = Pmax'_b - P'_a$$

expanding to

$$\vec{E}_{ab}(P'_a) = \left| \begin{matrix} f(zmax_a - zmin_a)\ (m_{14}C_{ab} - m_{34}A_{ab}) / (zmin_b zmax_b) \\ f(zmax_a - zmin_a)\ (m_{24}C_{ab} - m_{34}B_{ab}) / (zmin_b zmax_b) \\ 0 \\ 1 \end{matrix} \right|$$

and

$$\vec{M}_{ab}(P'_a) = \left| \begin{matrix} f(zmax_a A_{ab} + m_{14}) / zmax_b - x'_a \\ f(zmax_a B_{ab} + m_{24}) / zmax_b - y'_a \\ 0 \\ 1 \end{matrix} \right| .$$

For brevity, the parameter P'_a is implied when $\vec{E}_{ab}$ and $\vec{M}_{ab}$ are used without arguments.

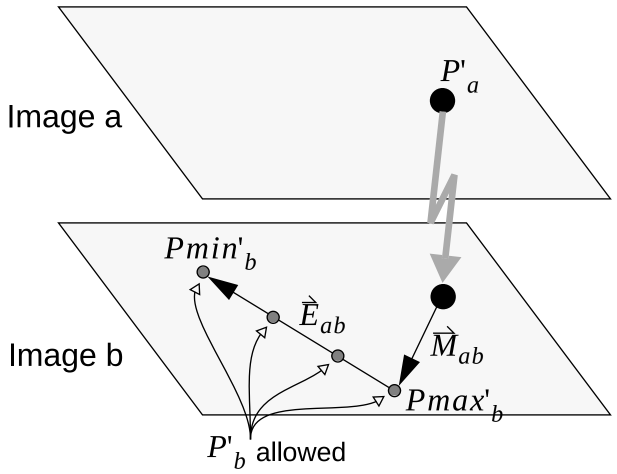


FIGURE 4. Initial displacement vector and epipolar vector. For a given point P'_a of image a, the matching point P'_b must lie on the line segment $[Pmin'_b, Pmax'_b]$ related to the two vectors $\vec{M}_{ab}$ and $\vec{E}_{ab}$.

2.2 Focus of expansion

The focus of expansion (FOE) is an image point representing the apparent motion of the camera. Defined as the intersection of all epipolar lines, the FOE is a point where the length of the epipolar vector $\vec{E}_{ab}$ is zero. Therefore, depth cannot be recovered at this point (see equations 3 and 4). Alternately, it can also be defined as the projection of the optical center of a camera on the projection plane of the other camera (figure 2). Only the relative displacement between cameras is needed to calculate this point. Algebraically, the FOE in image a for cameras a and b is

$$FOE_{ab} = (\tilde{x}_a, \tilde{y}_a, f, 1)$$

where

$$\tilde{x}_a = f\frac{(m_{22}m_{33} - m_{23}m_{32})\, m_{14} + (m_{13}m_{32} - m_{12}m_{33})\, m_{24} + (m_{12}m_{23} - m_{13}m_{22})\, m_{34}}{(m_{21}m_{32} - m_{22}m_{31})\, m_{14} + (m_{12}m_{31} - m_{11}m_{32})\, m_{24} + (m_{11}m_{22} - m_{12}m_{21})\, m_{34}}$$

and

$$\tilde{y}_a = f\frac{(m_{23}m_{31} - m_{21}m_{33})\, m_{14} + (m_{11}m_{33} - m_{13}m_{31})\, m_{24} + (m_{13}m_{21} - m_{11}m_{23})\, m_{34}}{(m_{21}m_{32} - m_{22}m_{31})\, m_{14} + (m_{12}m_{31} - m_{11}m_{32})\, m_{24} + (m_{11}m_{22} - m_{12}m_{21})\, m_{34}}.$$

2.3 Depth calculation from disparity

We can express the disparity between a point P'_a and its corresponding point P'_b with a single parameter e_{ab} that represents the displacement (or disparity) along the epipolar vector. This parameter is defined by the relation (figure 4)

$$P'_b = P'_a + \vec{M}_{ab} + e_{ab} \cdot \vec{N}_{ab} \ , \quad 0 \le e_{ab} \le \left\|\vec{E}_{ab}\right\| \qquad (2)$$

where the normalized epipolar vector $\vec{N}_{ab}$ is defined as $\vec{N}_{ab} = \dfrac{\vec{E}_{ab}}{\left\|\vec{E}_{ab}\right\|}$.

The stereoscopic fusion can now be accomplished by searching a value of e_{ab} along the epipolar vector $\vec{E}_{ab}(P'_a)$ for all points P'_a of image a. The disparity e_{ab} and the depth z_a for camera a are related by the equation

$$e_{ab} = DepthToDisparity_{ab}(z_a) = \frac{(zmax_a - z_a)\, zmin_b \left\|\vec{E}_{ab}\right\|}{(zmax_a - zmin_a)\, (z_a C_{ab} + m_{34})} \qquad (3)$$

where C_{ab} is defined in equation 1. The inverse relation gives z_a as a function of e_{ab}

$$z_a = DisparityToDepth_{ab}(e_{ab})$$

$$= \frac{zmax_a zmin_b \left\|\vec{E}_{ab}\right\| - e_{ab}\,(zmax_a - zmin_a)\, m_{34}}{e_{ab}\,(zmax_a - zmin_a)\, C_{ab} + zmin_b \left\|\vec{E}_{ab}\right\|} . \qquad (4)$$

3 The algorithm

An algorithm that performs stereoscopic analysis of two images taken from arbitrary positioned cameras is first presented.

A rectification process selects appropriate pairs of epipolar lines in two images. Each pair of lines creates a «solution space» that is used by an efficient dynamic programming algorithm to find an optimal matching for those lines (figure 5).

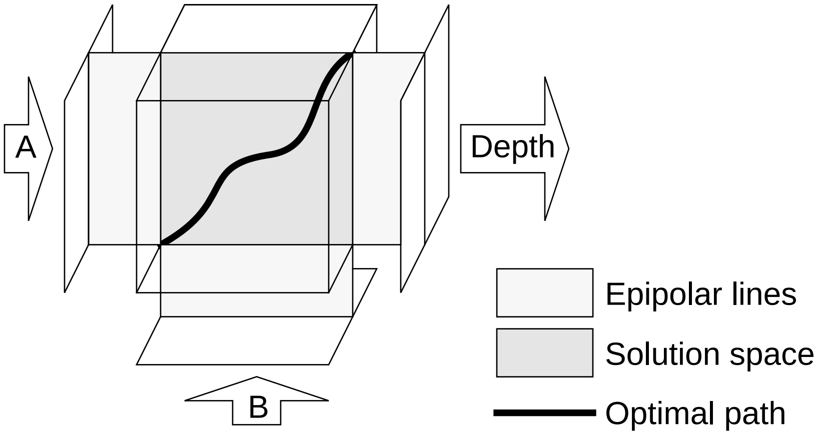


FIGURE 5. Matching process using dynamic programming. A pair of epipolar lines from rectified images a and b creates a solution space where a dynamic programming method finds an optimal path yielding depth for image a.

This two-image algorithm is then expanded to use more images simultaneously. To achieve this goal, the basic two-image algorithm is successively used over selected pairs of images. The cost function used for matching is also changed to take advantage of the added information provided by the extra images.

3.1 Image rectification

Since the camera movement can be any affine transformation, a rectification process must be applied to each image in order to adapt the matching process to each particular epipolar geometry. Some solutions for image rectification were proposed (Ayache 1989).

Our solution uses only the relative transformation between cameras without requiring any absolute positions.

A number of lines going through the FOE in image a are selected and extracted from the image. Those lines are epipolar lines since they all go through the FOE.

- If FOE_{ab} is in the visible part of the projection plane (i.e. inside of image a), the selected lines have angles in the interval $[0°, 180°)$ (figure 6a).
- If FOE_{ab} is not in the visible part of the projection plane (i.e. outside the image a) but not at infinity, the selected lines have angles in the interval $[\theta_1, \theta_2]$ where θ_1 and θ_2 are the angles of the lines joining two of the image's corners and FOE_{ab}. The two corners are selected to provide the widest angle interval (i.e. $|\theta_2 - \theta_1|$ is maximal) to allow the lines to sweep the whole image (figure 6b).
- If FOE_{ab} is at infinity, all selected epipolar lines are parallel and oriented toward the FOE (figure 6c). Obviously, each line must intersect the image.

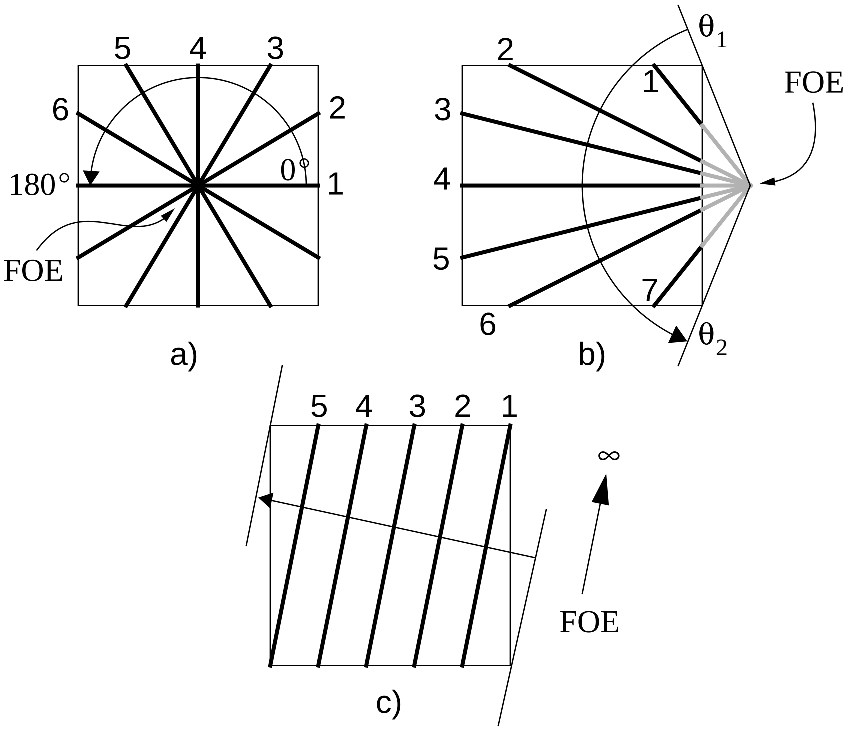


FIGURE 6. Image Rectification. Numbered lines are the selected epipolar lines. a) the FOE is inside the image, b) the FOE is outside the image but not at infinity, c) the FOE is at infinity.

Let’s assume that a selected epipolar line in image a is defined by a starting point S_a and an ending point T_a. The corresponding epipolar line $[S_b, T_b]$ in image b is defined as the space including all possible matching points of points in line $[S_a, T_a]$ in image a (figure 7). From equation 2, we have

$$S_b = S_a + \vec{M}_{ab}(S_a) \text{ or } S_b = S_a + \vec{M}_{ab}(S_a) + \vec{E}_{ab}(S_a),$$

$$T_b = T_a + \vec{M}_{ab}(T_a) \text{ or } T_b = T_a + \vec{M}_{ab}(T_a) + \vec{E}_{ab}(T_a)$$

The choice of either definition of S_b and T_b is made so as to get the longest line segment, i.e. $\|T_b - S_b\|$ is maximum. This ensures that the line will contain all the points that can be matched to points in $[S_a, T_a]$.

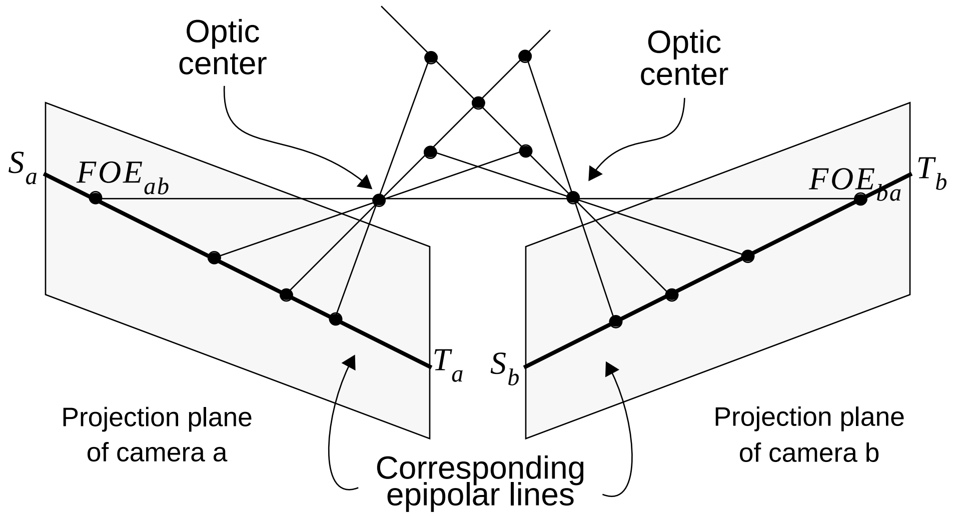

Optic center
Optic center
S_a
FOE_{ab}
FOE_{ba}
T_b
T_a
S_b
Projection plane of camera a
Projection plane of camera b
Corresponding epipolar lines

FIGURE 7. Corresponding epipolar lines. For an epipolar line $[S_a, T_a]$ in image a, the corresponding epipolar line is $[S_b, T_b]$ in image b. All epipolar lines intersect in their respective FOE.

The number of epipolar lines selected is controlled by a parameter α and the density of points along each line is controlled by a parameter β. Those parameters correspond to a scaling factor of the images. Rectified pixel values along the lines are calculated with bilinear interpolation. By increasing the pixel density β along epipolar lines, we achieve sub-pixel accuracy for disparity measurements. For example, a density of $\beta = 2$ allows a disparity precision of around half a pixel. Due to the limitations of the interpolation technique used, the density cannot be increased above a certain limit.

3.2 Matching along epipolar lines

For two selected epipolar lines $[S_a, T_a]$ and $[S_b, T_b]$, we can define a cartesian space with those lines as coordinate axes. As shown in figure 8, each point in this space is a possible match between two points. For each point with integer coordinates (i, j) in this cartesian space, the corresponding points on the epipolar lines are obtained by equations

$$P'_a = S_a + \frac{i}{\beta} \cdot \vec{L}_a, \; 0 \le i \le i_{max}, \; i \in \{0, 1, 2, \ldots\} \tag{5}$$

$$P'_b = S_b + \frac{j}{\beta} \cdot \vec{L}_b, \; 0 \le j \le j_{max}, \; j \in \{0, 1, 2, \ldots\} \tag{6}$$

$$\text{where } i_{max} = \lfloor \|T_a - S_a\| \cdot \beta \rfloor \text{ and } j_{max} = \lfloor \|T_b - S_b\| \cdot \beta \rfloor, \tag{7}$$

$$\vec{L}_a = \frac{T_a - S_a}{\|T_a - S_a\|} \text{ and } \vec{L}_b = \frac{T_b - S_b}{\|T_b - S_b\|}.$$

One can show that $\vec{L}_b$ equals $\pm\vec{N}_{ab}(P')$ where P' is on the epipolar line $[S_a, T_a]$.

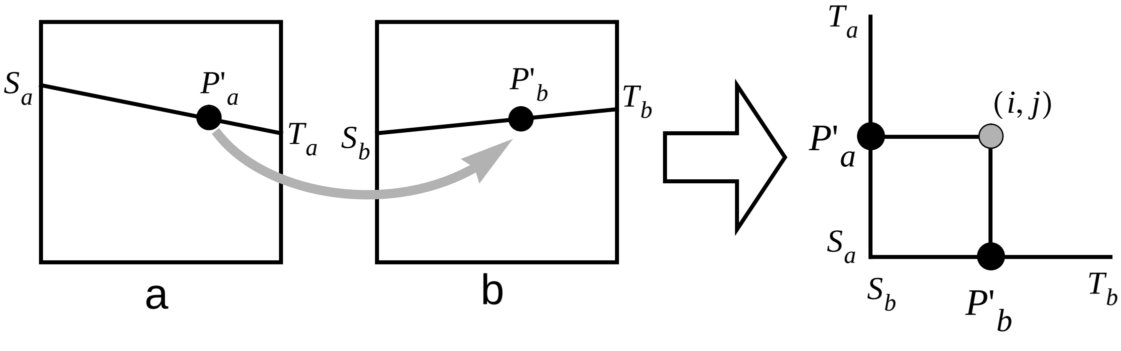


FIGURE 8. Solution space. A match between points P'_a and P'_b is represented in the solution space as a single point (i, j).

From the definition of disparity given by equation 2, and using equations 5 and 6, we can deduce the relation between a point (i, j) in the matching space and the disparity e_{ab}

$$\begin{aligned} e_{ab} \cdot \vec{N}_{ab}(S_a) &= SpaceToDisparity_{ab}(i, j) \cdot \vec{N}_{ab}(S_a) \\ &= S_b - S_a + \frac{j}{\beta} \cdot \vec{L}_b - \frac{i}{\beta} \cdot \vec{L}_a - \vec{M}_{ab}\left(S_a + \frac{i}{\beta} \cdot \vec{L}_a \right) \ . \end{aligned} \qquad \mathit{(8)}$$

One can show that the disparities are not distorted by the rectification process into the solution space since e_{ab} is linearly related to j for a given i in equation 8.

As an example, in the simple case of a left to right (horizontal) camera displacements with a depth interval reaching infinity, we have

$$\vec{L}_a = \vec{L}_b = (1, 0) \ , \vec{N}_{ab}\left(S_a + i \cdot \vec{L}_a \right) = (-1, 0) \ ,$$

$$S_a = S_b \ , \ \vec{M}_{ab}\left(S_a + i \cdot \vec{L}_a \right) = (0, 0)$$

which gives a relation

$$e_{ab} = \frac{i - j}{\beta} \ .$$

For a given point i on line $[S_a, T_a]$, we can find the interval restricting the matching points j along the line $[S_b, T_b]$. From equation 8, with e_{ab} taking its limit values 0 and $\left\| \vec{E}_{ab}(P'_a) \right\|$, we obtain

$$j_1(i) \cdot \vec{L}_b = \left(P'_a + \vec{M}_{ab}(P'_a) - S_b \right) \cdot \beta \text{ and } j_2(i) \cdot \vec{L}_b = j_1(i) \cdot \vec{L}_b + \vec{E}_{ab}(P'_a) \cdot \beta$$

where P'_a is a function of i (equation 5). For the point i, the search for the matching point j takes place between $j_1(i)$ and $j_2(i)$.

Finding a match for all points along epipolar lines $[S_a, T_a]$ and $[S_b, T_b]$ is equivalent to finding a path going through the solution space joining points $(0, 0)$ and (i_{max}, j_{max}) as defined in equation 7. This path is illustrated by figure 9.

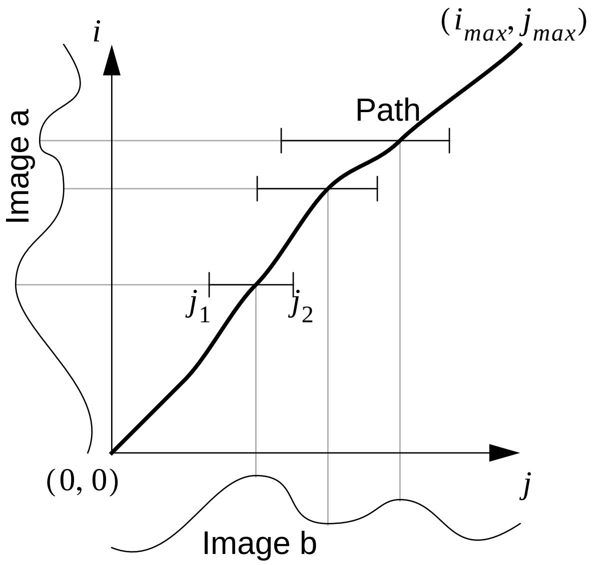


FIGURE 9. Path in the solution space. Finding a match for all possible points between a pair of epipolar lines gives a path in the solution space. For a given point i, the search interval is bounded by j_1 and j_2. Images intensities are pictured along the axis.

Notice that the matching processes performed on different pairs of epipolar lines are independent from one another. This allows a very straightforward parallel implementation.

3.3 Optimal path finding

An efficient dynamic programming approach is used to extract an optimal path from all possible paths in the solution space. The accuracy of this path will largely depend on the cost function used to evaluate each possible path.

In order to be able to use dynamic programming, some assumptions about the scene must be made in order to restrain the choices of path that can be found.

- Real objects: The scene is composed of objects that can exist in the physical world. Because of that, we can assume that the solution path is a single connected path where depth discontinuities are represented by horizontal and vertical path segments (figure 10a).

- Opaque objects: A point considered visible can only match one point. When it matches more than one point, it is considered occluded. The problem of transparency detection is thus avoided.

It directly follows that the solution path can not reverse direction. For a path reaching the solution point (i, j), the next point can only be $(i-1, j)$, $(i, j-1)$ or $(i-1, j-1)$ (figure 10b).

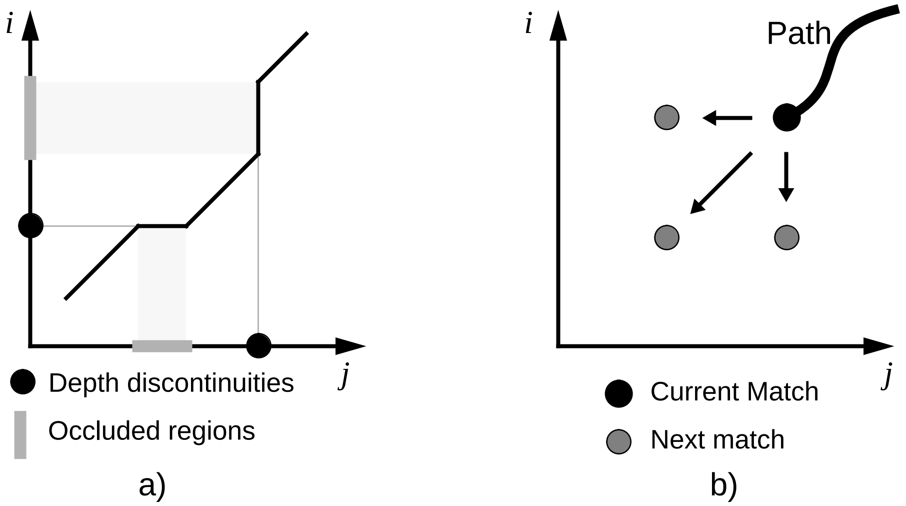


FIGURE 10. Characteristics of a path in solution space. a) Occluded regions are detected by vertical or horizontal path segments. b) A point on the path can have only one of three successor points.

By adding a «cost function» giving the pertinence of each matching pair of points, the correspondence problem is transformed into a minimum cost path finding problem. The path characteristics are such that dynamic programming can be used.

3.4 Cost function for two images

The cost function is based on the assumption that corresponding points should have similar image intensity levels. This forces objects in the scene to be dull so the displacement of the camera will not create specular reflections inducing intensity variations not directly related to stereoscopy.

The basic cost $B_{ab}(i, j)$ for a matching point (i, j) is the difference of intensity between the corresponding points. We have

$$B_{ab}(i,j) = \left|I_a(P'_a) - I_b(P'_b)\right|$$

where P'_a and P'_b are defined from (i,j) by equations 5 and 6, and $I_a(P')$ is the intensity of image a at point P'.

Following (Hillier 1986) and (Jain 1989), we define x_k as being the position (i,j) of a point on the path after k stages. We also define $C_n(x_1, \ldots, x_n)$ as the total cost of a path $x_1, \ldots, x_n$ at stage n. This total cost is defined as

$$C_n(x_1, \ldots, x_n) = C_{n-1}(x_1, \ldots, x_{n-1}) + B_{ab}(x_n) + OccPenalty(x_n, x_{n-1})$$

where $OccPenalty(x_n, x_{n-1}) \begin{array}{ll} = 0 & \text{if } x_n - x_{n-1} = (-1,-1) \\ = 1 & \text{otherwise} \end{array}$.

The occlusion penalty $OccPenalty(x_n, x_{n-1})$ is an extra cost added only when the preceding match along the path creates an occlusion (i.e. horizontal or vertical path segment). This factor is usually very low and can be increased when the images are expected to be highly corrupted by noise.

The optimal path $x_1^*, \ldots, x_n^*$ is defined as

$$C_n\left(x_1^*, \ldots, x_n^*\right) = \min_{x_1, \ldots, x_n} \{C_n(x_1, \ldots, x_n)\} \qquad .$$

Since we have the recursive relationship for stage k

$$C_k\left(x_1^*, \ldots, x_k^*\right) = \min_{x_k} \ \{C_k\left(x_1^*, \ldots, x_{k-1}^*, x_k\right)\} \ ,$$

and in our case $x_1^* = (i_{max}, j_{max})$, $x_n^* = (0,0)$, $C_1\left(x_1^*\right) = B_{ab}(i_{max}, j_{max})$, we can see that the optimal path is obtained after n stages of one-variable searches.

All disparities are found from the optimal path $x_1^*, \ldots, x_k^*$. For a given solution point $x_k = (i,j)$, we get an point P'_a in image a from equation 5. The disparity e_{ab} for this point is obtained from equation 8 :

$$e_{ab} = SpaceToDisparity_{ab}(x_k) \ .$$

The occlusion map for image a is easy to construct by following the optimal path while looking for vertical path segments that correspond to occlusions. In the same way, the occlusion map for image b would be obtained from the horizontal path segments (figure 10a).

4 Multiple image correspondence

The algorithm already presented uses two images to evaluates the depth of one of them, called the *reference image*. The extended stereoscopic algorithm takes two *basic images* while all the other images in the sequence are used as *extra images* (figure 11). All pairs of basic images contain the reference image and are processed by stereoscopic analysis while all the other images serve as extra images. The final depth map is obtained from averaging all depth maps. The final occlusion map is obtained from the union (logical or) of all occlusion maps.

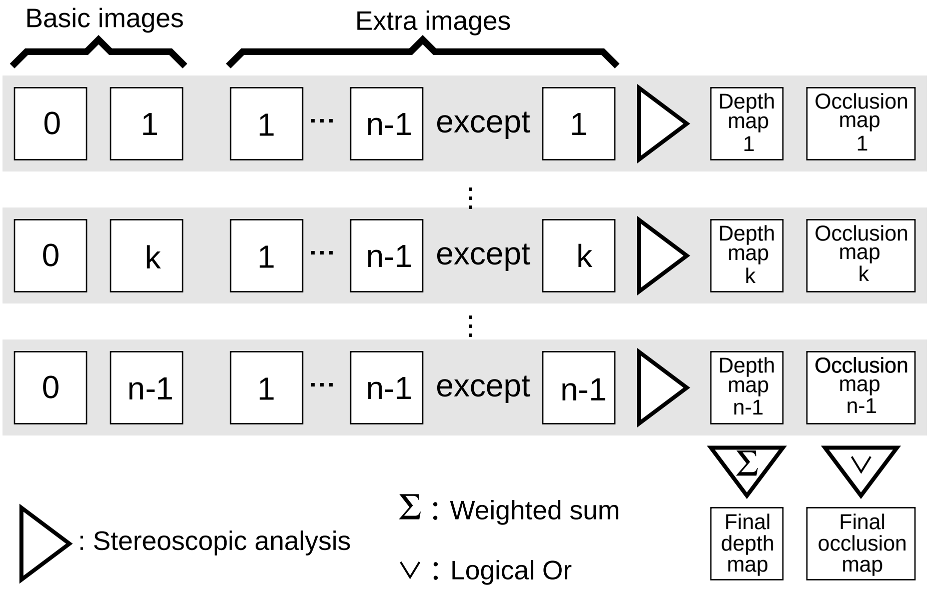


FIGURE 11. Extended algorithm for n images. After $n-1$ stereoscopic analyses, the final depth and occlusion maps for the reference image 0 are obtained from the $n-1$ depth and occlusion maps.

The dynamic programming correspondence process can only be applied to two images at a time. Extra images can add accuracy to the matching by providing additional information to the cost function.

As shown in figure 12, a match between points P'_0 and P'_1 of the two basic images 0 and 1 provides a disparity e_{01} that is converted to a depth measurement z_0 with (equation 4)

$$z_0 = DisparityToDepth_{01}(e_{01}) \ .$$

For each extra image k, this depth can be converted into a disparity e_{0k} between image 0 and k with (equation 3)

$$e_{0k} = DepthToDisparity_{0k}(z_0) \ .$$

The corresponding point P'_k in image k for this depth can be used in the new cost function and is obtained with

$$P'_k = P'_0 + \vec{M}_{0k} + e_{0k}\vec{N}_{0k},\ 0 \le e_{0k} \le \|\vec{E}_{0k}\| \ .$$

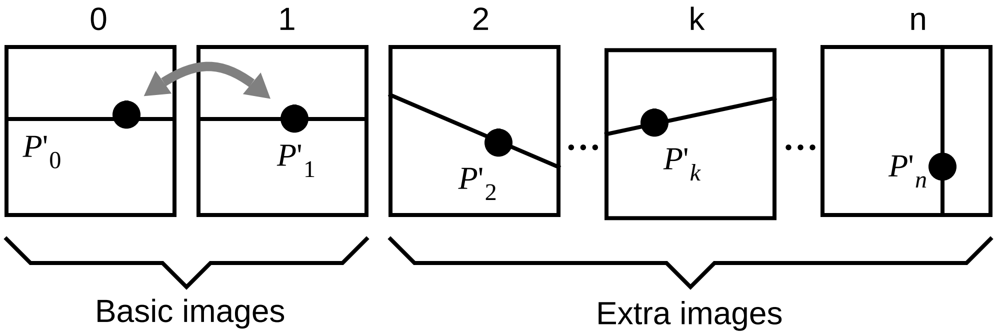


FIGURE 12. Extra image point selection. For a pair of matching points P'_0 and P'_1, we can compute one corresponding point in each extra image $(P'_2, \ldots, P'_k, \ldots, P'_n)$.

4.1 Cost function for multiple images

The two-image cost function already defined in section 3.4 can be easily modified to take into account all extra images. For a given pair of matching points P'_0 and P'_1, all corresponding points P'_k should have the same intensity level.

For the k^{th} stereoscopic analysis, the basic cost function $B_{0k}(i, j)$ is changed for

$$B_{0k}(i,j) = \left|I_0(P'_0) - I_k(P'_k)\right| + \sum_{l = 1,\, l \neq k}^{n-1} \left|I_0(P'_0) - I_l(P'_l)\right| .$$

If a point P'_l lies outside the visible part of the projection plane of camera l, it obviously cannot contribute to the summation term.

5 Results and discussion

The stereoscopic algorithm has been applied to sequences of 2 and more images. Two examples of those analyses are presented here with an evaluation of the quality and accuracy of the computed depth maps.

In these examples, the images are 256×256 pixels in size. The rectification parameters α and β are set to $\alpha = 2$ and $\beta = 2$. Those settings allow sub-pixel accuracy of about 0.5 pixel for disparity measurements.

The first example is obtained with a camera displacement in depth along the z axis (figure 13) generating two views of the scene.

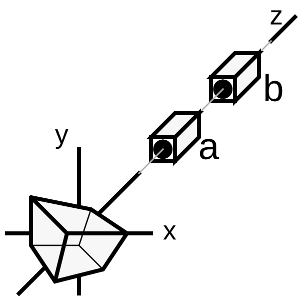


FIGURE 13. Camera setup with displacement in depth. In this two-camera example, the camera b is moved along the z axis from the reference camera a.

The scene is composed of a a cube whose edges are cylinders in front of a textured background (at infinity). This synthetic image is not trivial to analyze since we did not add texture to the cube, and we also selected a shape that creates occlusions.

In this example, the FOE is located in the center of the images, as in figure 6a. The rectification process applied to image a is shown in figure 14.

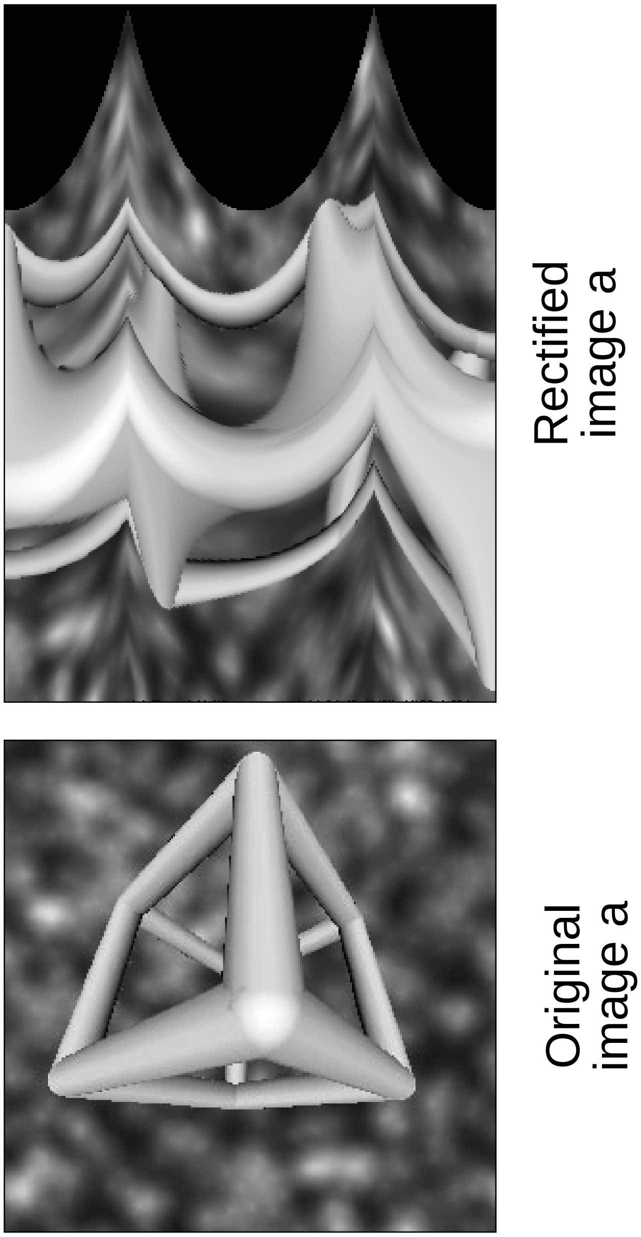


FIGURE 14. Image rectification. A number of epipolar lines intersecting in the FOE (at the center of image a) are selected and piled up vertically to form the rectified version of image a.

The rectified images a and b are then matched with the dynamic programming method. A sample cost function is shown in figure 15 with the optimal path found.

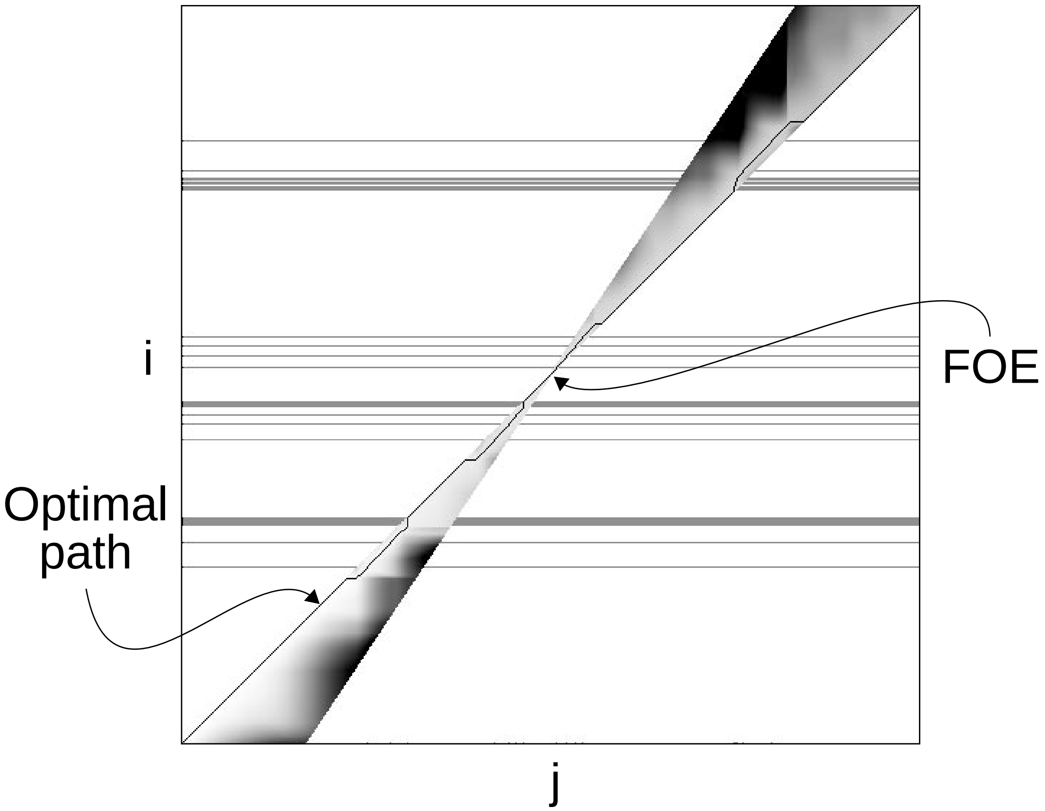


FIGURE 15. Solution space and optimal path. The search region is a strip crossing the solution space, with the characteristic bottleneck at the FOE. The horizontal bands show regions along *i* where occlusions are detected. Within the search region, darker shades of gray indicate higher costs. The optimal path is shown as a black line.

The optimal path gives the disparity and occlusion maps of image *a* as shown in figure 16. Note that disparities are shown as gray levels corresponding to displacement in pixels along the epipolar lines. They are easily converted into depth values using equation 4. Observe that the disparity changes toward zero for points near the FOE (at the center of the image). The mean error in the disparity map is 0.5 pixel and is mostly caused by the lack of texture on the cube and the particular positioning of the cameras.

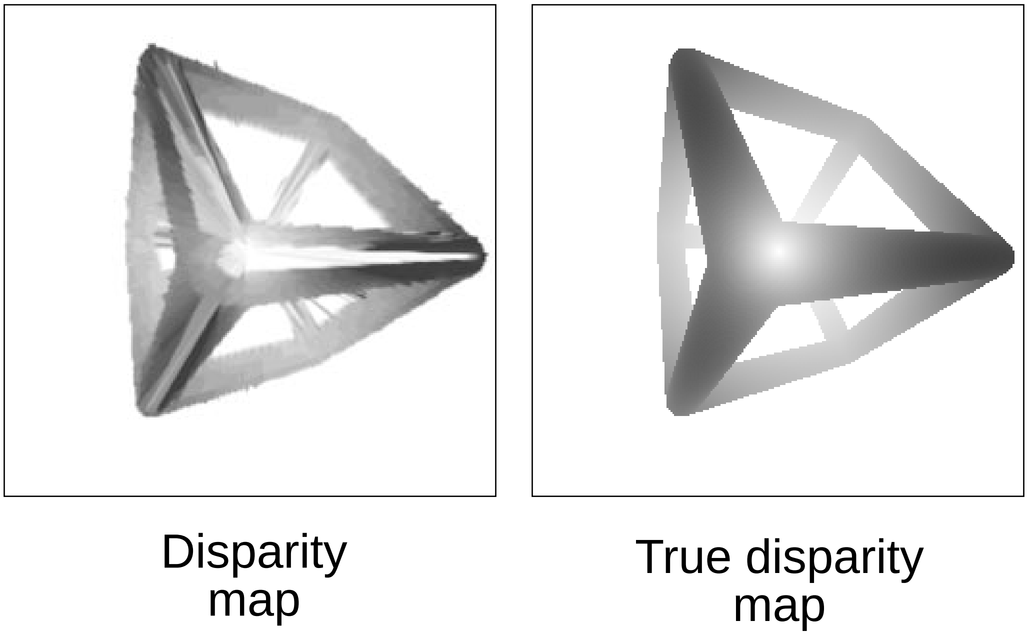


FIGURE 16. Disparity maps. The disparity map is given in pixels ($\text{white} = 0$, $\text{black} = 23$). The true disparity map is given as a reference.

For the second example, three cameras are set up as in figure 17. Taking camera a as the reference of this trinocular system, the camera b is moved horizontally while the camera c is moved vertically. Since those movements are perpendicular, the lack of texture doesn’t have as big an impact as in the binocular case. In this example, the rectification process is trivial since the FOEs are both at infinity (figure 6c) and aligned with the image axis.

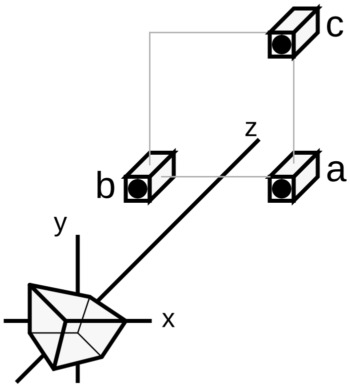


FIGURE 17. Trinocular camera setup. Taking camera *a* as the reference, camera *b* has been moved horizontally (along the *x* axis) while camera *c* has been move vertically (along the *y* axis).

The three images are shown in figure 18 and feature several occlusions as well as a lack of texture that complicates the task of the stereoscopic algorithm.

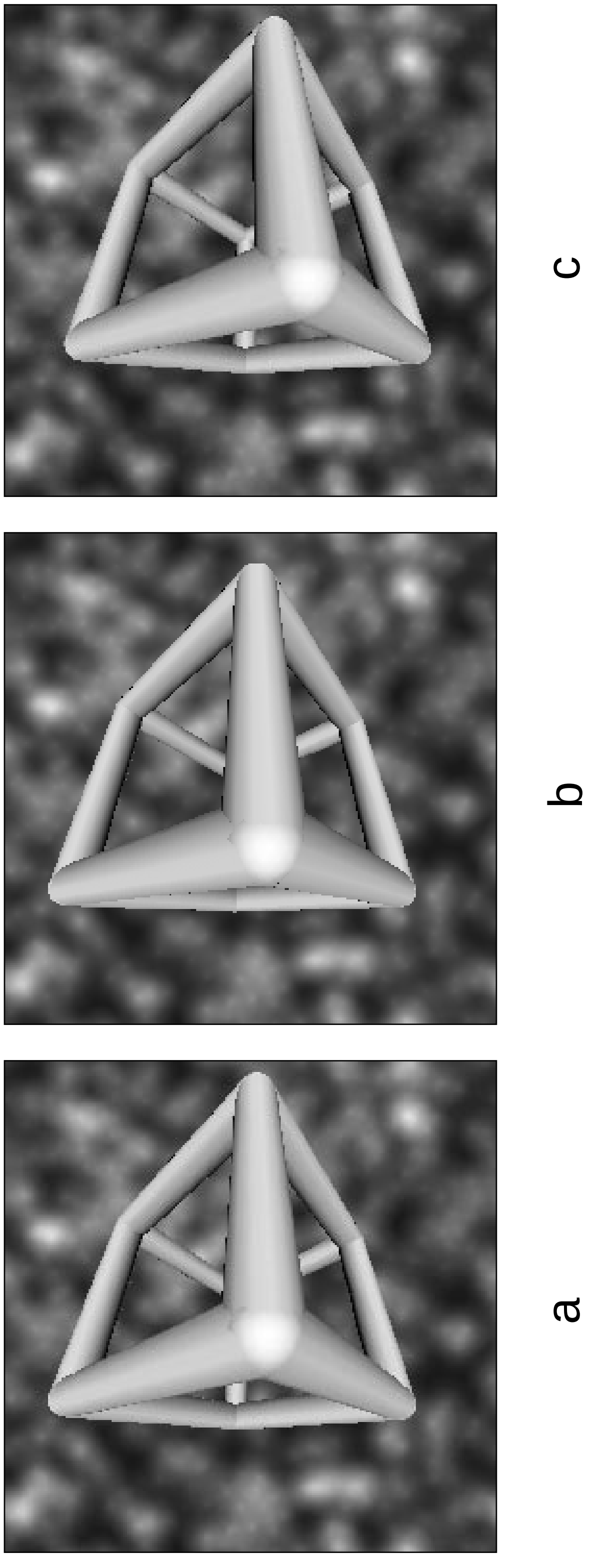


FIGURE 18. Three synthetic images. a) Reference image. b) horizontal displacement. c) vertical displacement. The background has been placed at infinity.

The cost function for a pair of epipolar lines is shown in figure 19 to illustrate the path finding process.

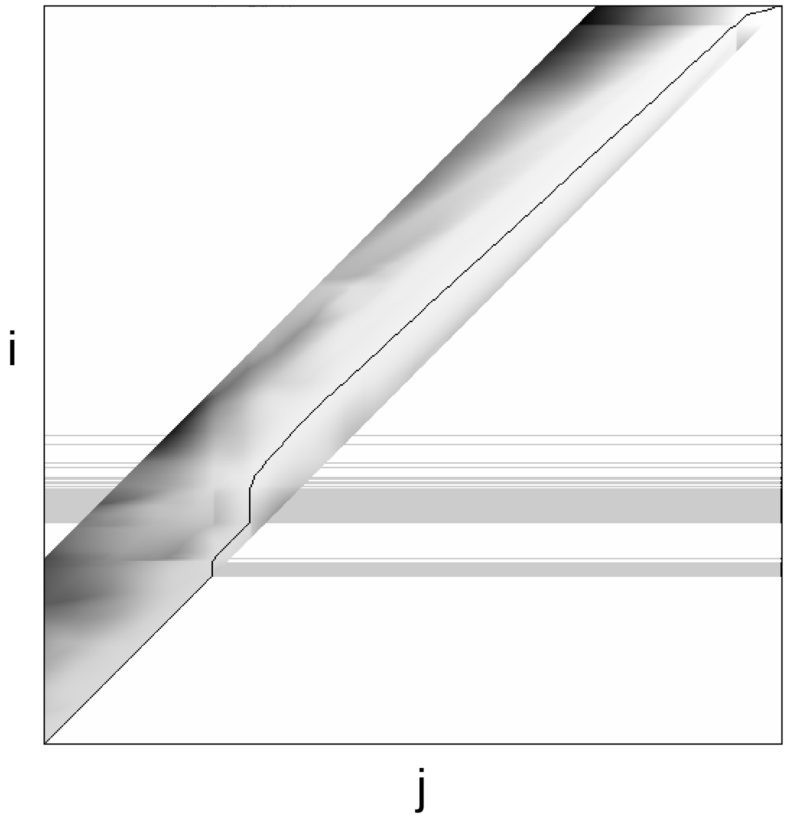
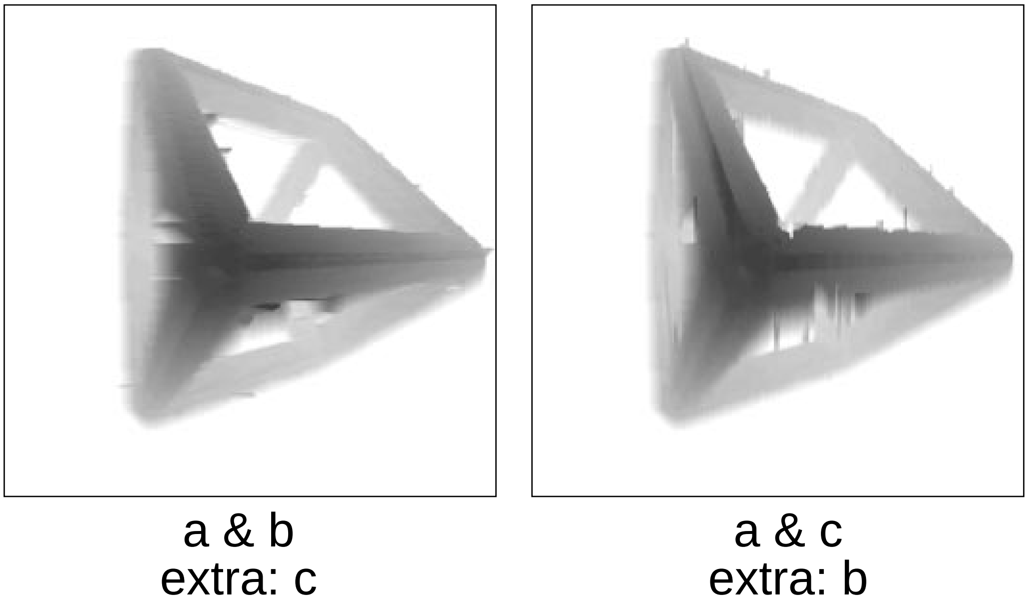


FIGURE 19. Solution space and optimal path. The search region is a narrow strip crossing the solution space. The horizontal bands show regions along *i* where occlusion are detected. Minimum cost is shown in white while maximum cost is shown in black. The optimal path is shown as a black line.

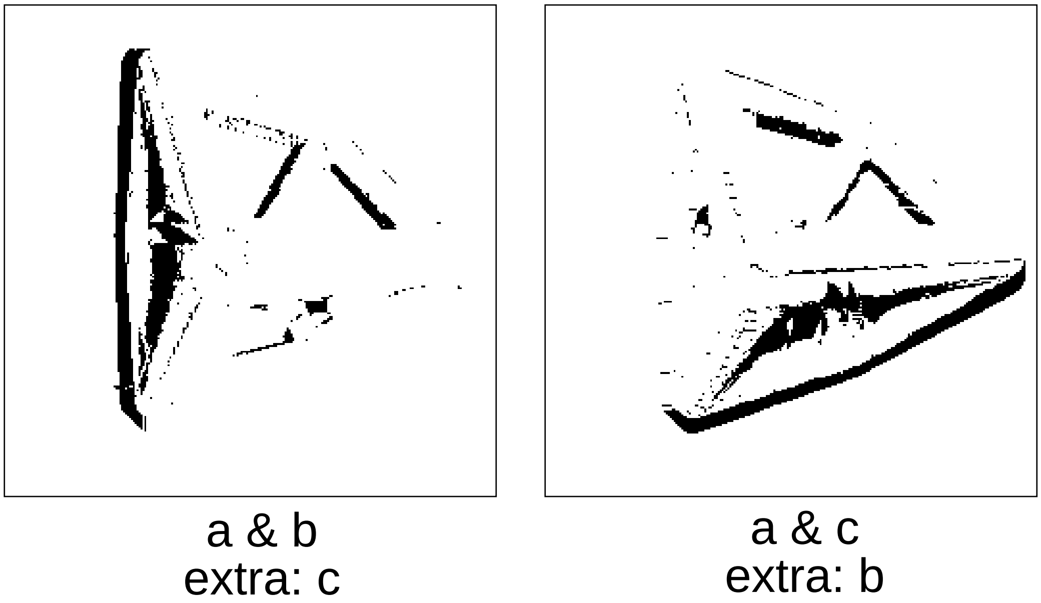
Two applications of the stereo algorithm are needed to get intermediate disparity maps for images *a* and *b*, and then image *a* and *c*. Those maps are shown in figure 20 while the corresponding occlusion maps are shown in figure 21.



a & b
extra: c

a & c
extra: b

FIGURE 20. Disparity maps. These disparity maps result from the stereoscopic analysis of successive pairs of basic images a, b and a, c. They are used to build the final depth map.



a & b
extra: c

a & c
extra: b

FIGURE 21. Occlusion maps. These occlusion maps result from the stereoscopic analysis of successive pairs of basic images a, b and a, c. They are used to build the final occlusion map. Black points are considered occluded.

As expected, we notice that occlusions dependent largely on the camera displacement. Since the final occlusion map (figure 22) is the union of the intermediate occlusion maps, a point is considered occluded as soon as it is occluded in any one of those maps.

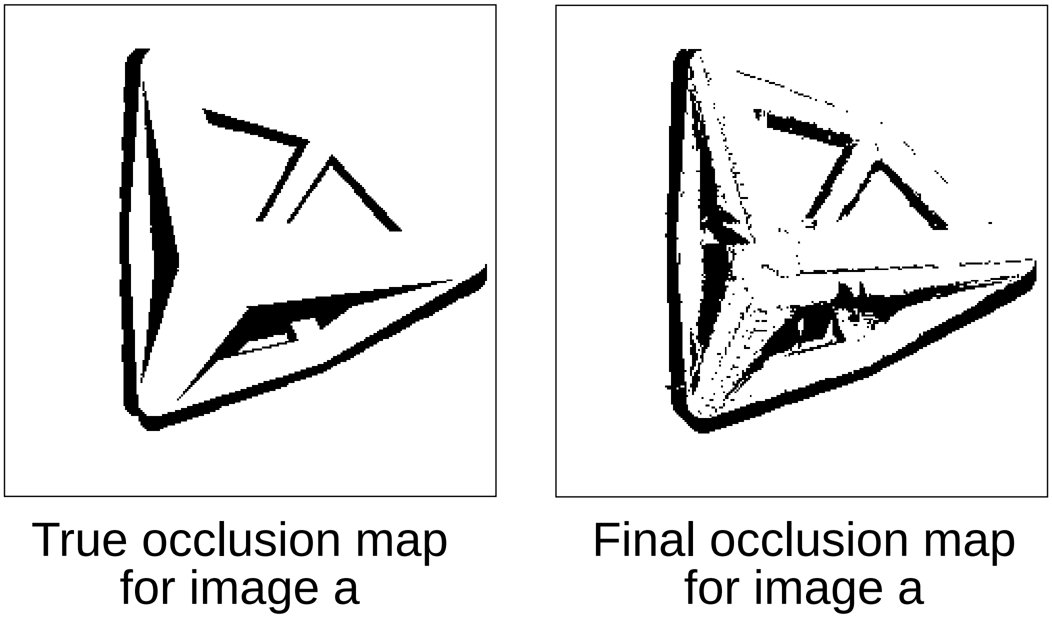


FIGURE 22. True and final occlusion maps. The final occlusion map for image a is obtained by the union (logical or) of the intermediate occlusion maps of figure 21. The true occlusion map is shown for comparison purposes. Black points are considered occluded.

The final disparity map is shown in figure 23 along with a computed reference map that gives the true disparity values. The disparity map obtained from a simple binocular stereoscopic analysis is also given to show the improvement brought by the trinocular example. In all those disparity maps, the occluded points have no depth information assigned to them. Depth values are only assigned to points visible in all images (a, b and c).

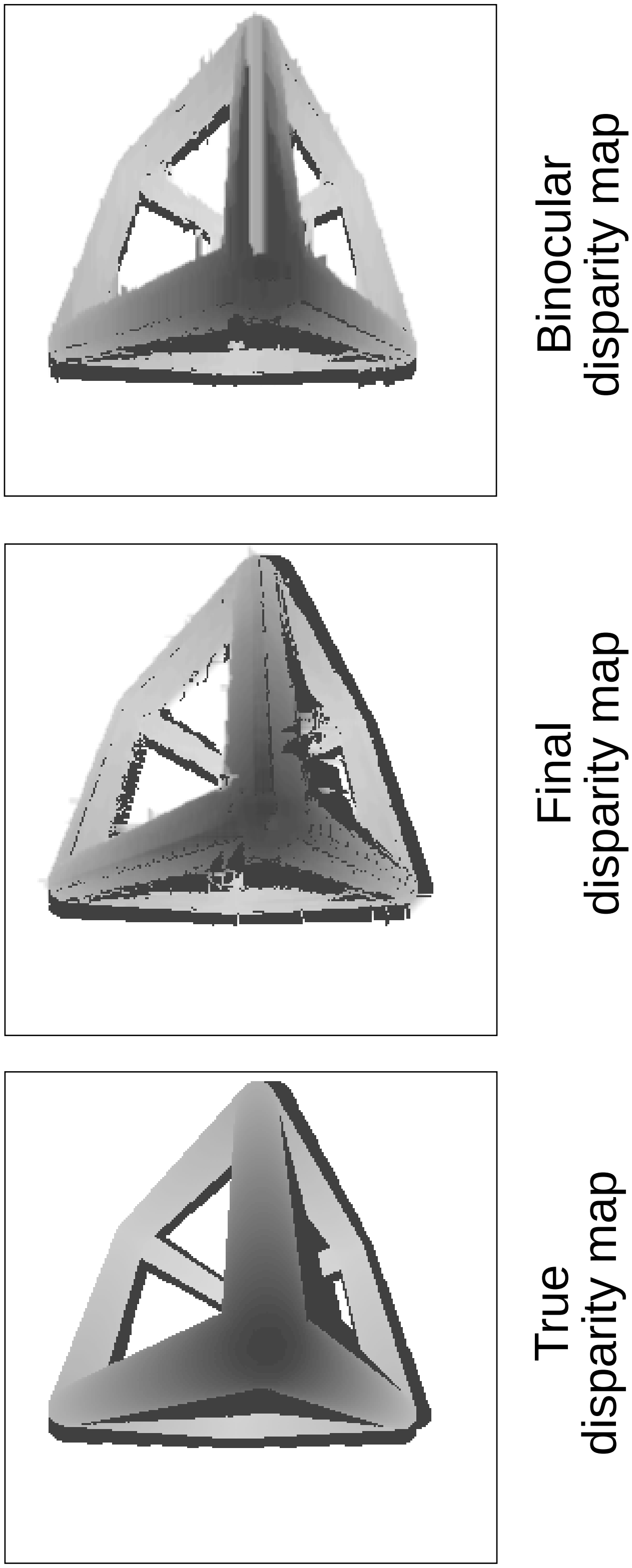


FIGURE 23. True and final disparity maps. The reference disparity map, containing true disparity values, is shown with the final disparity map obtained for image a, with occlusions shown in black. The disparity map

of a simple binocular analysis is also shown for comparison.

The mean difference between the final disparity map and the reference depth map of figure 23 is 0.26 pixel, when occluded pixels are not taken into account. The same measure taken in the binocular case yields an error of 0.40 pixel, also when occluded pixels are not taken into account.

Those results can be considered encouraging since the disparity accuracy was set to 0.5 pixel (i.e. $\beta = 2$). However, it must be noted that the textured background helps to lower somewhat the error measurements.

6 Conclusion

A new algorithm for analysis of multiple stereoscopic images was presented. Stereoscopic matching is done on two ‘basic’ images with a dynamic programming method. During the matching process, the new cost function to be minimized takes into account the other images. Supporting general camera displacements, the algorithm does not require any preprocessing of the images. The depth map and occlusion map are dense, and parallel computation is easy to achieve. Due to the absence of any «depth smoothing constraints», discontinuities along the contour of objects are well preserved in the depth map. In the future, allowing transparent objects and taking into account specular reflections of light will certainly prove to be challenging.

Acknowledgments

We thank Martin Trudeau for his helpful suggestions and comments.

References

Ayache, N. 1989, *Vision stéréoscopique et perception multisensorielle,* InterEditions: Paris

Brown C. 1988, *Advances in computer vision,* Volume 1, LEA publishers: New Jersey

Cox, I.J., Hingorani, S., Maggs, B.M. and Rao, S.B. 1992. Stereo without disparity gradient smoothing: a bayesian sensor fusion solution, In Proc. of British Machine Vision Conference, p. 337-346, Leeds, England

Fleck, M.M. 1991, A topological stereo matcher, *International Journal of Computer Vision*, 6:3, p. 197-226

Hillier, F.S. and Lieberman G.J. 1986, *Introduction to Operations Research*, Holden-Day, Oakland, California

Horn, B.K.P. 1986. *Robot vision*. The MIT Press: Cambridge, Massachussetts.

Jähne, B. 1991, *Digital Image Processing*, Springer-Verlag

Jain, A.K. 1989. *Fundamentals of digital image processing*, Prentice Hall, p. 359-361

Jones, D.J. 1992, Stereoscopic Sensing of 3D Shape and Distance, *Computer Vision : Introduction and perspectives*, McGill University

Griswold, N.C. and Yeh, C.P. 1988. A New Stereo Vision Model Based upon the Binocular Fusion Concept, *Computer Vision, Graphics, and Image Processing* 41, p. 153-171

Lee, H.-J. and Deng, H.-C. 1990, Three-Frame Corner Matching and Moving Object Extraction in a Sequence of Images, *Computer Vision, Graphics, and Image Processing* 52, p. 210-238

Roy, S. 1992. *Analyse d’images stéréoscopiques basée sur la détermination du flux optique*. M.Sc. thesis, Université de Montréal

Scheuing, A. and Niemann, H. 1986. Computing depth from stereo images by using optical flow, *Pattern Recognition letters* 4, p. 205-212

Shirai, Y. 1987, *Three-Dimensional Computer Vision*, Springer-Verlag, p. 122-140

Weng, J., Ahuja, N. and Huang, T.S. 1992b, Matching two perspective Views, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 14, no. 8

Weng, J., Huang, T.S. and Ahuja, N. 1989. Motion and Structure from Two Perspective Views: Algorithms, Error Analysis, and Error Estimation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, No. 5

Weng, J., Huang, T.S. and Ahuja, N. 1992a, Motion and Structure from Line Correspondences: Closed-Form Solution, Uniqueness, and Optimization, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 14, No. 3