3D RECONSTRUCTION AND TRACKING OF HUMAN FACES

FROM A STEREO IMAGE SEQUENCE

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Engineering

By

MOHAMMAD SHAFIKUL HUQ
B.Sc in CSE, BUET, Dhaka, Bangladesh, 1997

2001
Wright State University

WRIGHT STATE UNIVERSITY

SCHOOL OF GRADUATE STUDIES

May 03, 2001

I HEREBY RECOMMEND THAT THIS THESIS PREPARED UNDER MY SUPERVISION BY Mohammad Huq ENTITLED 3D Reconstruction and Tracking of Human Face from a Stereo Image Sequence BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science in Computer Engineering.

_____

A. Ardeshir Goshtasby, Ph.D.
Thesis Director

_____

Oscar N. Garcia, Ph.D.
Department Chair

Committee on Final Examination

_____

A. Ardeshir Goshtasby, Ph.D.

_____

Oscar N. Garcia, Ph.D.

_____

Ricardo Gutierez Osuna, Ph.D.

_____

Joseph F. Thomas, Jr., Ph.D.
Dean of the School of Graduate Studies

ABSTRACT

Huq, Mohammad. M.S., Department of Computer Science and Engineering, Wright State University, 2001. 3D Reconstruction and Tracking of Human Faces from a Stereo Image Sequence.

A computer vision algorithm is designed to track and reconstruct a talking face from two stereo image sequences captured from two cameras. At first, the cameras are calibrated and calibration parameters are used to reconstruct the face. Reconstruction is achieved through stereo matching. To do stereo matching, an energy minimizing algorithm is adopted. Seed points are collected from a laser sweep of the face. While the person stands still before the camera, a vertical laser line is swept over his face. A laser point appearing in the same scan line of both images is used to match points in the left and right images. The laser point appearing in both images works as a seed point. After sweeping the laser, the person starts talking and both cameras capture stereo image sequences of the person. Now, starting with the seed points and the energy minimizing algorithm, both image sequences are frame by frame processed to reconstruct the face for every image pair. A large window has been used to coarsely track the face. After coarse tracking, the image points are matched finely with the energy minimizing algorithm. The energy minimizing algorithm has two energy terms and two parameters. Definition of the energy terms and selection of the parameters automatically are discussed.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

Page

LIST OF FIGURES

LIST OF FIGURES (Continued)

Figure Page

LIST OF FIGURES (Continued)

Figure Page

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank all people who have been instrumental in preparation of this thesis. First of all, appreciation goes to Dr. Goshtasby for his in depth guidance and support during the course of this thesis. I am extremely grateful to him for providing assistance in literature survey and suggesting improvements to bring the thesis to its final form. I would also like to thank my coworkers, Marcel Jackwoski and Lijun Ding for participating in many helpful technical discussion. Finally, I would thank Dr. Oscar N. Garcia and Dr. Ricardo Gutierez Osuna for serving on my examining committee.

I also would like to thank the National Science Foundation for supporting this work during the past fifteen months.

And of course this work would not have been possible without invaluable support and encouragement of my family members and friends. My gratitude to them is beyond expression.

I. INTRODUCTION

We perceive depth using our two eyes. An object's images are formed on both our eye retinas. The images are not exactly the same, but are displaced with respect to each other. The amount of displacement depends on the distance of the object to the eyes. Objects in distant are displaced less than those closer. Displacement is usually proportional to inverse of depth. In stereo vision, two or more cameras are used to find depth. Stereo depth perception is the process by which a three-dimensional structure is recovered from two or more images taken from slightly different view points of a scene.

We obtain two images of a scene and from spatial processing of the pair reconstruct the scene. When an object moves, some temporal processing is needed to track it. While tracking, the object is reconstructed in 3D. In temporal processing, the left image and the right undergo the following stages: feature detection, feature matching, interpolation and reconstruction of the object. The key step here is the matching problem, known as stereo correspondence, to find the correspondence between 2D projected object points in the left and right images. The output of a stereo correspondence algorithm is a *disparity map*, specifying the relative displacement of matching points in the two images. The stereo correspondence problem is inherently underconstrained and is further complicated by the fact that the images typically contain noise. Two approaches have been used to solve this problem. *Feature-based* approaches only match points and lines with a certain amount of local information such as intensity, yielding only sparse disparity maps. *Area-based* approaches, on the other hand, yield a dense disparity map by matching small image patches as a whole and relying on the assumption that nearby

points usually have similar displacements. A typical area-based stereo matching algorithm proceeds as follows: for each point in one image, the displacement that aligns this point with that in the other image is found. The quality of match is measured by comparing windows centered at the two points, for example, using the sum of absolute difference or cross correlation. An area-based algorithm typically has the following steps:

1. For each disparity under consideration, compute a per-pixel matching cost.
2. Aggregate the costs
3. Across all disparities, find the best match based on the aggregated cost
4. Compute the missing disparities using interpolation.

Although area based matching yields a dense disparity map, it has the problem of not matching accurately if image gradient is low. For example, the forehead and cheeks on a human face look almost the same anywhere. Another problem with area matching is determination of the search area. It should be assured that the match exists somewhere within the search area. The first problem can be solved using an energy minimizing snake. This is a rather new idea. In an image area without enough gradient, an energy minimizing snake can distribute matching points uniformly. The second problem can be taken care of by introducing seed points. Seed points are those that are close to their correct matching positions. Laser has been used to generate seed points. This is also a new idea used in stereo matching. After matching, points with missing disparities are interpolated. Then projective geometry parameters are used to reconstruct the object. Snake is powerful for tracking, because if the object moves fast between two consecutive frames, snake will be able to track them well. In this case, the matching points in one frame will work as seed points for the next frame.

1.1 Problem description

We assume a talking person standing in front of a stereo camera setup. The two cameras will capture a pair of stereo image sequences – left camera image sequence and right camera image sequence. From these two sequences, ultimately we want to generate a 3D animation of the person.

1.2 Approach

There are many approaches that enable reconstruction of a realistic face in a virtual world such as using a plaster model [1], or interactive deformation [2]. Such models may give nice results, but require a lot of time for their preparation. Methods that are more efficient are such as laser scanning, structured light method not using laser [3] and stereo triangulation [4][5][6].

In our approach we have combined laser scanning and stereoscopy. At first, the stereo cameras are calibrated. Calibration involves finding the relationship between image coordinates and world coordinates through some parameters. These parameters are used later to reconstruct the object. To do stereo matching, an energy minimizing algorithm using seed points in the first image pair is adopted. Seed points are collected by a laser sweep. A person stands still before the camera. Then, a vertical laser line is swept over his face. A laser point appearing on both images tells where the point in the left image appears in the right image. The laser point works as a seed point. Sweeping is done in a slightly dark environment to let the laser beam become brighter. Brighter points are easy to detect. The laser is swept on the whole face from one side to the other to

collect seed points from all over the face. After this, the person starts talking in the presence of sufficient light. Using the seed points, the energy minimizing algorithm starts matching both images frame by frame and reconstructs the face for every image pair. A large window is used to track the face. This is coarse tracking. After coarse tracking, the image points are matched finely by applying the energy minimizing algorithm.

1.3 Organizaton of the Thesis

The rest of this thesis is organized as follows. Chapter 2 presents a brief review of algorithms and techniques used in stereo matching and tracking. The motivation for energy minimizing snake is also mentioned. Chapter 3 discusses some preliminaries needed to read the rest of the thesis. Chapter 4 shows how to calibrate cameras and get projective geometry parameters that relate scene coordinates to image coordinates. The method to collect seed points from laser sweep has been discussed in chapter 5. Chapter 6 describes the energy minimizing algorithm and finding the disparity map between the left and right images. Chapter 7 discusses the tracking of a face. The remaining two chapters show some results and offer some conclusions.

II. RIVIEW OF PAST WORKS

Stereo vision has been the subject of study for more than two decades. Stereo correspondence, 3D reconstruction and stereo tracking are the main problems to be solved. To solve the stereo correspondence problem, dynamic programming has been used efficiently to minimize (or maximize) functions of a large number of correspondences. Successful attempts at dynamic programming for solving the stereo correspondence problem are reported by Baker and Binford [1] and Ohta and Kanade [2]. In both cases they used edges as the basic primitives. However, these techniques are computationally expensive. A fast and automatic stereo correspondence algorithm based on dynamic programming has been proposed by Benshair amd Debrie [3]. All these methods treat stereo correspondence as the problem of finding an optimal path on a two-dimensional search plane.

Marr and Poggio [4] have enforced uniqueness and continuity constraints to limit the search. They used a confidence measure for a point in the left image to a set of possible corresponding points in the right image. This algorithm does not perform good on real images, mostly because the tokens and the features that are used are not sufficient to deal with most real images. Pollard, Mayhew and Frisby [5] enforce the disparity gradient constraint. Their algorithm first extracts, from both images, a number of tokens, each token characterizing a number of features. For example, edge points are detected and characterized by their strength and orientation. Tokens are then matched using an iterative winner-take-all procedure that enforces uniqueness.

In our work we have used the pixel intensity as the matching token. The energy minimizing snake enforces uniqueness and ordering constraint by energy minimization.

The snake, which has a grid shape, converges in several iterations always satisfying those two constraints. Besides distributing the matches uniformly and unambiguously over a large image region where sufficient features are not available, the snake shows good performance in tracking when the object changes shape.

The work by Waxman and Duncan [6] is one of the first attempts to relate stereo matching with motion. Their work of course was restricted to rigid motion, restricting to motion of a rigid body. Their work is motivated by the fact that when rigid motion is involved, there exists a correlation between relative image flow and stereo disparity. By employing information from both spatial and temporal image sequences, the structure computation can be made more accurate because there are more matching information.

We have worked here with a non-rigid object – a talking human face. The explicit relationship between spatial and temporal domain displacement may not exist when the object in motion is not rigid. However, the relationship between these two domains should still exist by some extent when the inter-frame motion is too small so that the motion is locally rigid. Work by Wen-Hung and Aggarwal [7] has some similarity with our work. They have worked with stereo and motion on non-rigid objects with a cooperative matching paradigm. They have incorporated temporal matching to spatial matching. We have used temporal matching information. One example of tracking in spatio-velocity space using energy minimizing snake is the work done by Natan Peterfreund [14]. His snake tracks moving boundary of a non-rigid object. He has used canny edge detector [15] to find whether or not an image region falls on the boundary. P. Nesi and R. Magnolfi [16] have used snake for tracking and synthesizing facial motion. They have divided the motion into two - global and local. Global motion has been

estimated from a 3D face model and then local motion has been tracked using snake. We have tracked global movement rather in a simpler way, then have used snake to estimate local movement. Our energy minimizing snake is powerful enough to work with a little prior information about an object's geometry.

III. PRELIMINARIES

3.1 Pinhole Camera Model

A camera can be approximated by a pinhole camera. The focal length is the distance of the hole to the image plane.

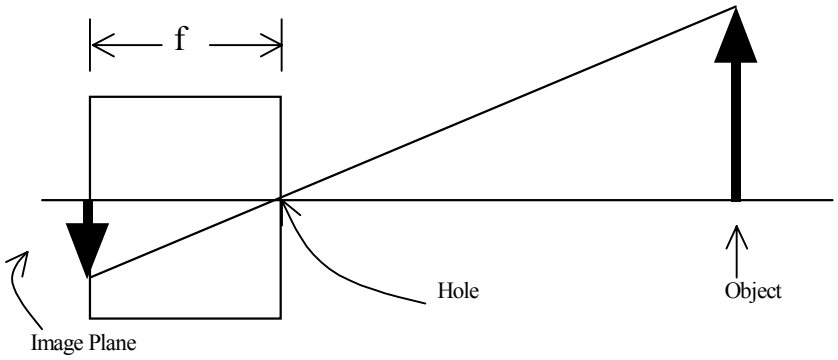


Fig. 3.1: A pinhole camera model.

3.2 Correspondence and Reconstruction Problem

Two pinhole cameras of the types discussed above form images m_1 and m_2 of a

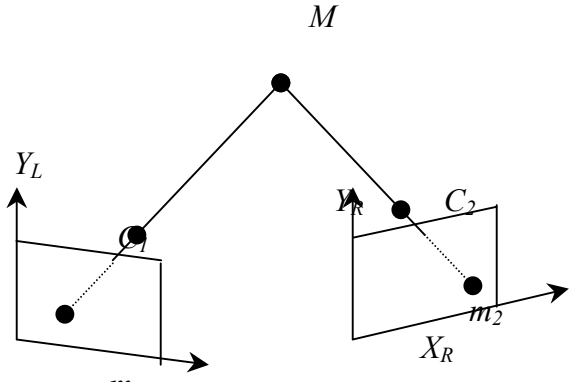


Fig. 3.2: The stereo setup.

physical point M. Fig. 3.2 shows three coordinate systems, one in each of the image planes (x_L,y_L) and (x_R,y_R) and one in 3D space (X,Y,Z), which is sometimes called the world coordinate system.

Given the two images, two problems have to be solved,

1. For a point m_1 in left plane, determine point m_2 in the right plane that it corresponds to. "Correspond" means they are the images of the same physical point M. This is the *correspondence problem*.
2. Given m_1 and m_2, compute the 3D coordinates of M in the world coordinate system. This is the *reconstruction problem*.

Given a point m_1 in the left image, it may initially be put into correspondence with any point m_2 in the right image. To find the correct correspondence, we must use constraints to reduce the number of potential matches for any given point m_1. These constraints are of three kinds. The first kind is from the imaging system. Probably, the most important such constraint is the epipolar constraint, which transforms a two-dimensional search into a one-dimensional search. The second kind of geometric constraint arises from the objects being looked at. We can assume, for example, that distances of object points to the imaging system vary slowly almost everywhere except at the depth discontinuities. The third kind of constraint is physical and arises from the way objects interact with light. For simplicity we assume that an object's surface is diffused and the brightness of a scene point appears the same in both images.

The second problem can be solved using the position C_1 and C_2. The result essentially depends on how accurately the positions of corresponding points are determined.

3.3 Epipolar, Uniqueness, Continuity and Ordering Constraints

Epipolar Constraint

Given m_1 in the image plane I_L, all possible physical points M that may have produced m_1 are on the infinite half-line (m_2,C_2). As a direct consequence, all possible matches m_2 of m_1 in I_R stay along this infinite half-line. This image point is an infinite half-line ep_2 going through the epipole E_2, which is the intersection of the line (C_1,C_2)

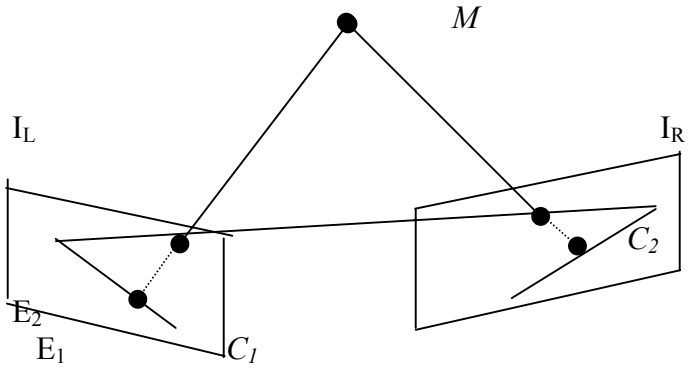


Fig. 3.3: The epipolar geometry.

with the plane I_R, E_2 is called the epipole of the second camera, and line ep_2 is called the epipolar line for point m_1. The correspondence constraint is that, for a given point m_1 in the plane I_L, its possible matches in plane I_R lie on this epipolar line. Therefore we have reduced the dimension of our search space from two dimensions to one. The epipolar constraint is of course symmetric and, for a point m_2 in plane I_R, we find epipolar line ep_1 that passes through the epipole E_1, which is the intersection of line (C_1,C_2) with plane I_L.

When plane I_L or plane I_R, or both, are parallel to line (C_1,C_2), the epipoles are at infinity and the epipolar lines in one plane (or both) become parallel.

Uniqueness

If the objects are opaque, one point in the left image should have at most one match in the right image. This is not true for transparent objects. This is also not always true when we are using line segments as tokens. Nonetheless, this constraint can often be used to reduce the number of possible correspondences.

Continuity

The basic idea of this constraint is that the world is mostly made up of objects with smooth surfaces. This means that the reconstruction function, which assigns a pair of matched pixels to a 3-D point *M*, is smooth almost everywhere. The reconstruction function is summarized as a function $z = f(d)$, where z is the distance of M to the cameras and *d* is the disparity.

Definition of disparity : given a point m_1 with coordinates (x_l,y) in the left image and its corresponding point m_2 with coordinates (x_r,y) in the right image plane, disparity is (x_r-x_l). A disparity of 0 implies that the 3-D point *M* is at infinity. If we bring point *M* toward the optical center C_1 along the infinite half-line (m_1,C_1), the disparity will increase from 0 to $+\infty$. This is assuming the epipoles are at infinity.

Ordering Constraint

Let us consider a 3-D point *M* and its projections m_1 and m_2 in I_L and I_R, respectively. Fig. 3.4 shows the configuration of the corresponding epipolar plane. Now let us choose a point *N* in the cone defined by M,C_1, C_2 containing the base line (C_1,C_2). *N* has images n_1 and n_2 in image planes I_L and I_R, respectively. Now we have (m_1,n_1) for I_L and (n_2,m_2) for I_R. The ordering is reversed. This ordering will hold for any scene point *N* in the hatched region. In practice it is difficult to eliminate the whole cross-hatched cone of Fig 3.4. If *M* and *N* are from different objects, it is highly probable that

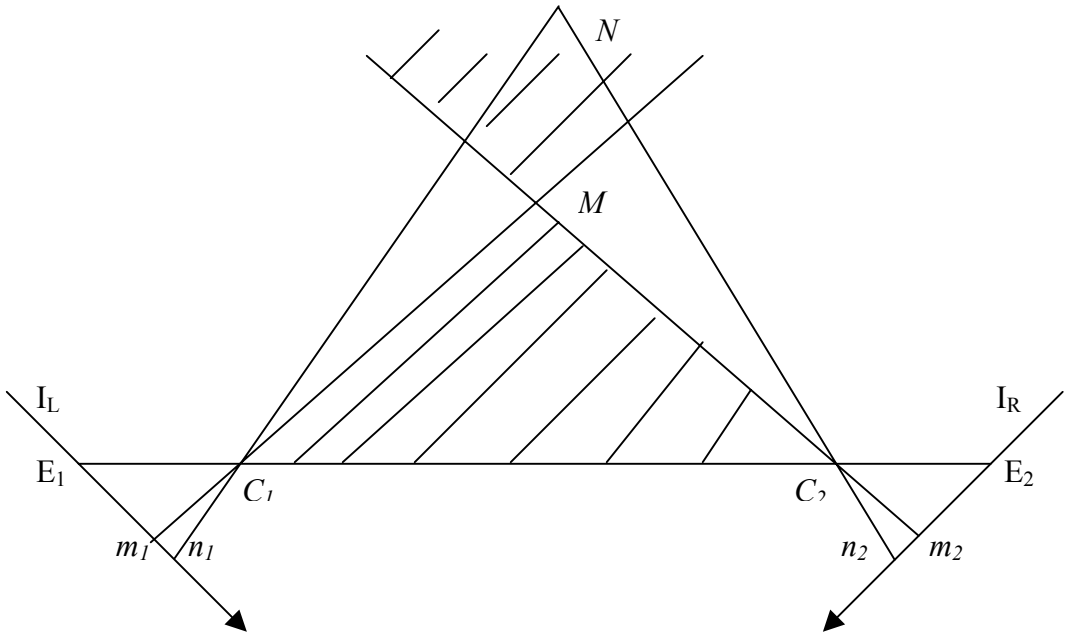
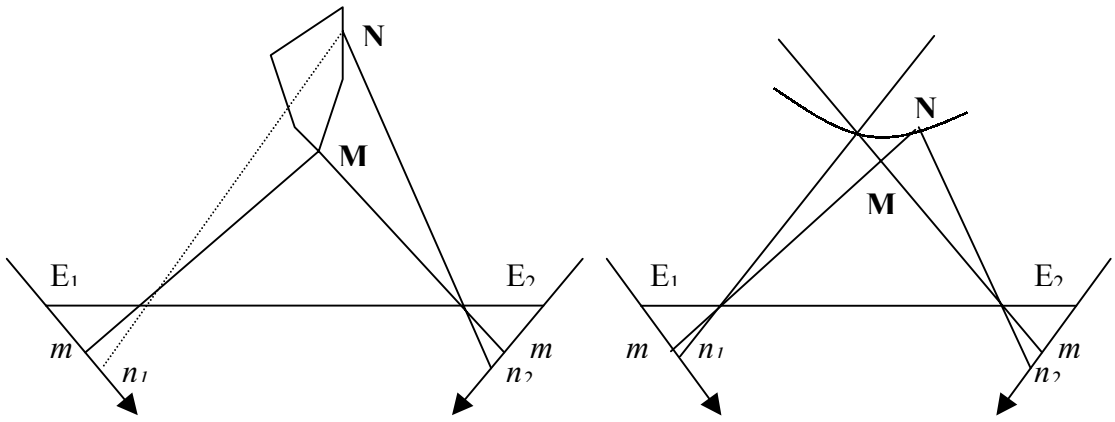


Fig. 3.4: Forbidden zone attached to *M*.

they will appear on the both image planes. However they will be in reverse order. If both points are from the same opaque object, one of the points will not appear in one plane. This absence will create occlusion. Occluded points will remain unmatched. To satisfy the ordering constraint, *N* should be outside the hatched zone as shown in Fig. 3.5.



a): Occlusion introduced b): Ordering constraint holding

Fig. 3.5: Object outside the forbidden zone.

It is clear from this figure that *N* will stay out of this region when change of depth is not much. Human face is a single object with continuous and small change of depth except the region surrounding the nose.

3.4 Camera Calibration

Calibration is defined as finding a camera system's intrinsic and extrinsic parameters from the relation between scene coordinates and image coordinates. These parameters can transform image points to scene points and vice-versa. To find the parameter values, we need to know image coordinates of some scene points whose 3D scene coordinates are known. Once found, these parameters can be applied to any stereo point pair to determine the 3D scene coordinates.

3.5 Our Stereo Setup

Ideally, focusing two cameras on the same object completes the stereo setup. But for the stereo process to work, several restrictions are imposed on the setup. They are,

1. The cameras should be identical. They should have the same focal length and the same zoom level.
2. Their optical axes should meet at some point near the center of the object.
3. X-axis of both cameras should be horizontal.
4. The two cameras should not be much apart from each other. This restriction will reduce the effect of perspective foreshortening.

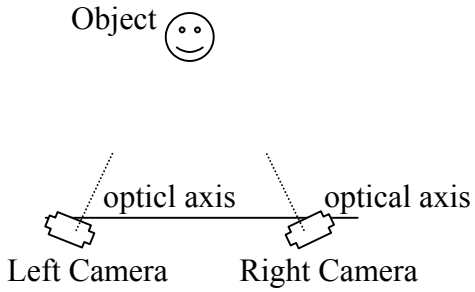


Fig. 3.6: Capturing images in a stereo setup.

With this setup, the two images of an object captured by the two cameras produce a stereo pair. For moving objects, it is assumed that both cameras capture the images at the same time.

3.6 Depth Perception from Stereo Setup

The setup in the Fig. 3.7 has two identical cameras that are tilted towards each other. It shows how they perceive depth of 3D points. P_f and P_n are two points in space.

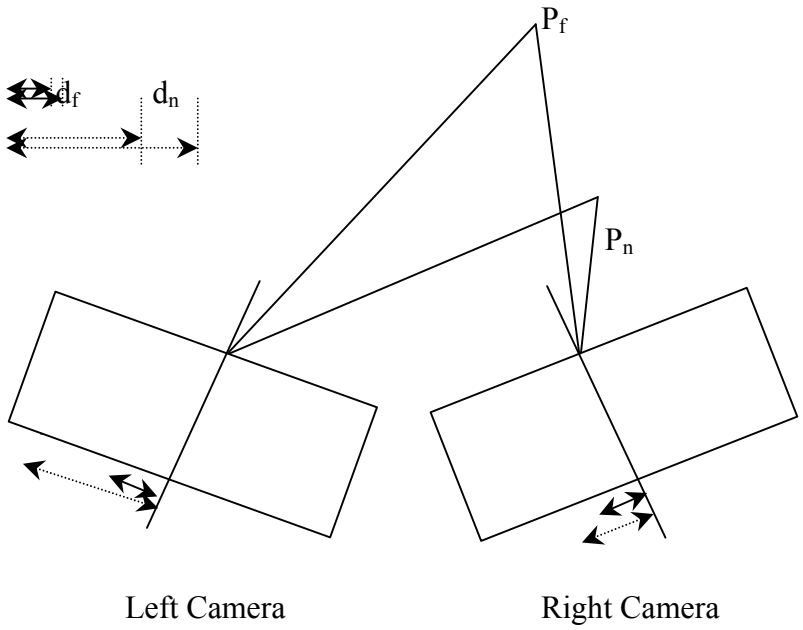


Fig. 3.7: Depth perception from stereo setup.

One is farther from the cameras than the other. Due to perspective foreshortening, they appear at different distances from the center of the image planes. As shown in the figure, the difference of horizontal image points in left and right images, which is called disparity, for point P_f is d_f and for P_n is d_n. Clearly $d_f < d_n$. This means disparity decreases

as the depth increases. From this relation, it is possible to find depth of any 3D scene point visible in both images.

IV STEREO GEOMETRY AND CAMERA CALIBRATION

4.1 Stereo Geometry

Two cameras slightly apart from each other by some distance *b* are focused at the same point. The distance between their focal points OL and OR is *b*, and *b* is called baseline length. With respect to the distance between focus point and the cameras *b* should not be big to introduce occlusion. A very small *b*, however, will introduce inaccuracy in the depth calculation. Both cameras are tilted towards each other. Y-axes in both cameras are vertical and parallel to each other.

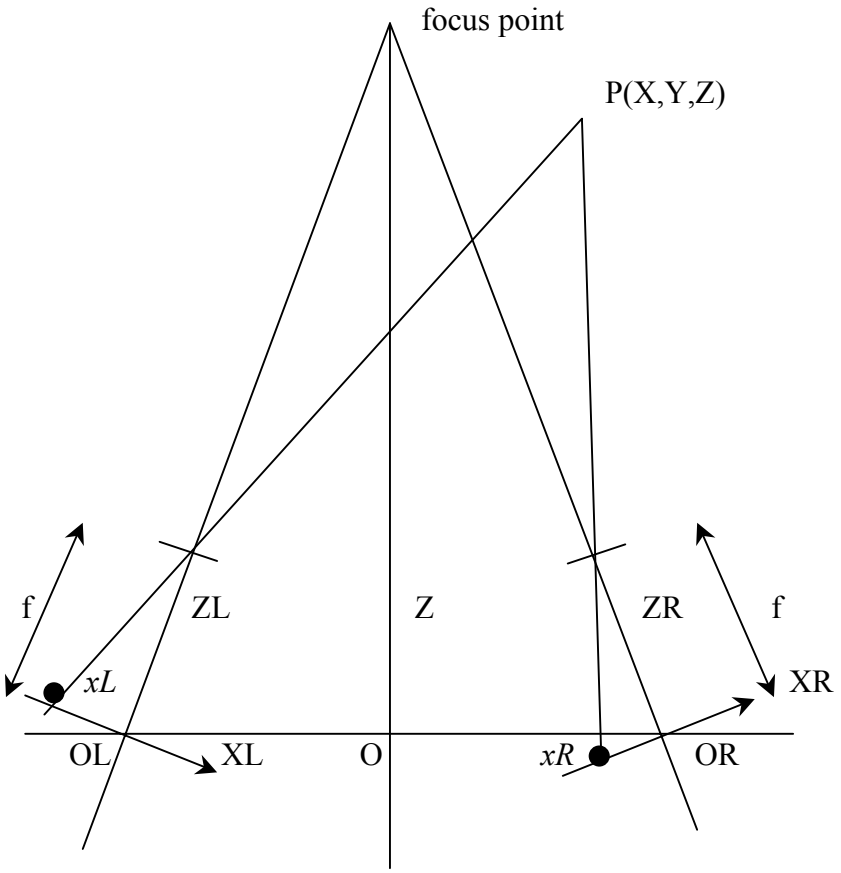


Fig. 4.1: Stereo Geometry

The camera coordinate systems *XLYLZL* and *XRYRZR* for the left and the right camera are shown in figure 4.1.

4.2 Projective Equations for Camera Calibration

Relation between (x_l,y_l) and (x_r,y_r) can be described by the following four projective equations:

$$x_l = \frac{m_1X+m_2Y+m_3Z+m_4}{m_5X+m_6Y+m_7Z+1} \qquad x_r = \frac{n_1X+n_2Y+n_3Z+n_4}{n_5X+n_6Y+n_7Z+1}$$

$$y_l = \frac{m_8X+m_9Y+m_{10}Z+m_{11}}{m_5X+m_6Y+m_7Z+1} \qquad y_r = \frac{n_8X+n_9Y+n_{10}Z+n_{11}}{n_5X+n_6Y+n_7Z+1}$$

… … … (4.1)

There are twenty two unknown parameters – m_1 to m_{11} and n_1 to n_{11}. We need five and a half points from the left image and corresponding five and a half points from

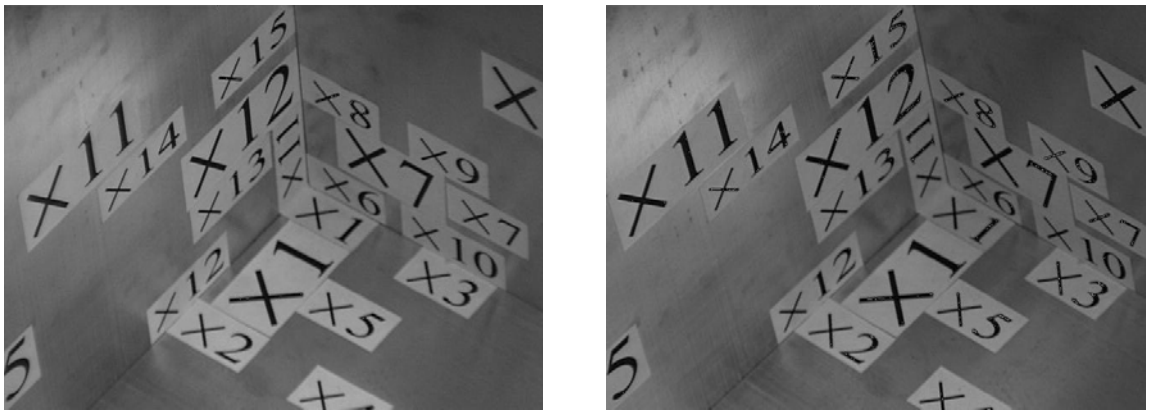


Fig 4.2 : Images Used in Calibration

the right image to know those parameters. We can find these points' coordinates with the mouse click. Mouse clicks may not be accurate. In that case. those parameter values will have some error. To minimize the error we use least-squares method. We click more than six points and use them in the least-squares formula. It has been observed that

twelve points are enough in the least-squares method. Taking more points does not change the accuracy. Fig 4.2 shows a stereo image pair of a box marked with some known point coordinates.

The equations of 4.1 can be simplified as :

$$m_1X+m_2Y+m_3Z+m_4-m_5x_lX-m_6x_lY-m_7x_lZ-x_l=0$$

$$m_8X+m_9Y+m_{10}Z+m_{11}-m_5y_lX-m_6y_lY-m_7y_lZ-y_l=0$$

$$n_1X+n_2Y+n_3Z+n_4-n_5x_rX-n_6x_rY-n_7x_rZ-x_r=0$$

$$n_8X+n_9Y+n_{10}Z+n_{11}-n_5y_rX-n_6y_rY-n_7y_rZ-y_r=0$$

… … … (4.2)

4.3 Finding the Camera Parameters

Taking the first two equations of (4.2) for any scene point (*Xi,Yi,Zi*) we may obtain

$$\Delta_{1i}=m_1X_i+m_2Y_i+m_3Z_i+m_4-m_5x_{li}X_i-m_6x_{li}Y_i-m_7x_{li}Z_i-x_{li}$$

$$\Delta_{2i}=m_8X_i+m_9Y_i+m_{10}Z_i+m_4-m_5y_{li}X_i-m6y_{li}Y_i-m_7y_{li}Z_i-y_{li}$$

Where $\Delta 1i$ and $\Delta 2i$ are errors. We have to find a set of *m* values for which these errors will be minimum for *N* number of scene points. *N* is preferably large. As mentioned before *N*

= 12 in our experiments. The error term for all *N* points is defined by

$$E = \sum_{i=1}^{n} \left[\Delta_{1i}{}^{2} + \Delta_{2i}{}^{2} \right]$$

Our target is to minimize as much *E* as possible. In that case, partial differentiation of *E* with respect to *m1*,*m2*, …, *m11* are set to zero.

$$\frac{\partial E}{\partial m_1} = 0, \quad \frac{\partial E}{\partial m_2} = 0, \quad \frac{\partial E}{\partial m_3} = 0, \ldots, \quad \frac{\partial E}{\partial m_{11}} = 0$$

From the first equality,

$$\frac{\partial E}{\partial m_1} = \sum 2X_i (m_1 X_i + m_2 Y_i + m_3 Z_i + m_4 - m_5 x_{li} X_i - m_6 x_{li} Y_i - m_7 x_{li} Z_i - x_{li}) = 0$$

or

$$m_1 \sum X_i{}^{2} + m_2 \sum X_i Y_i + m_3 \sum X_i Z_i + m_4 \sum X_i + m_5 \sum (-x_{li} X_i{}^{2}) + m_6 \sum (-x_{li} X_i Y_i) + m_7 \sum (-x_{li} X_i Z_i) - \sum X_i x_{li} = 0$$

or

$$m_1 \sum X_i Y_i + m_2 \sum Y_i{}^{2} + m_3 \sum Y_i Z_i + m_4 \sum Y_i + m_5 \sum (-x_{li} X_i Y_i) + m_6 \sum (-x_{li} Y_i{}^{2}) + m_7 \sum (-x_{li} Y_i Z_i) - \sum Y_i x_{li} = 0$$

or

$$m_1 \sum X_i Z_i + m_2 \sum Y_i Z_i + m_3 \sum Z_i{}^{2} + m_4 \sum Z_i + m_5 \sum (-x_{li} X_i Z_i) + m_6 \sum (-x_{li} Y_i Z_i) + m_7 \sum (-x_{li} Z_i{}^{2}) - \sum Z_i x_{li} = 0$$

or

$$m_1 \sum X_i + m_2 \sum Y_i + m_3 \sum Z_i + m_4 \sum - m_5 \sum x_{li} X_i - m_6 \sum x_{li} Y_i - m_7 \sum x_{li} Z_i - \sum x_{li} = 0$$

… … … (4.3)

There are eleven equations, each for a partial differentiation. From these eleven equations we can find all eleven *m* parameters, from the twelve points. *n* parameters can be found similarly.

After *m* and *n* parameters are found, (*X,Y,Z*) of a scene point is calculated by substituting its corresponding left image point (*xl*,*yl*) and right image point (*xr*,*yr*) in the equation set 4.3. There are four equations, but three are needed. We can use the least-squares method and use all four one to get more accurate (*X,Y,Z*). In this case the error terms are defined by

$$\Delta_{1i} = X_i(m_5 x_{li} - m_1) + Y_i(m_6 x_{li} - m_2) + Z_i(m_7 x_{li} - m_3) - (m_4 - x_{li})$$

$$\Delta_{2i} = X_i(m_5 y_{li} - m_8) + Y_i(m_6 y_{li} - m_9) + Z_i(m_7 y_{li} - m_{10}) - (m_{11} - y_{li})$$

$$\Delta_{3i} = X_i(n_5 x_{ri} - n_1) + Y_i(n_6 x_{ri} - n_2) + Z_i(n_7 x_{ri} - n_3) - (n_4 - x_{ri})$$

$$\Delta_{4i} = X_i(n_5 y_{ri} - n_8) + Y_i(n_6 y_{ri} - n_9) + Z_i(n_7 y_{ri} - n_{10}) - (n_{11} - y_{ri})$$

$$E = \sum_{i=1}^{n} (\Delta_{1i}{}^2 + \Delta_{2i}{}^2 + \Delta_{3i}{}^2 + \Delta_{4i}{}^2)$$

E has to be minimum to get the most accurate (X,Y,Z).

Taking partial differentiation of E with respect to X,Y and Z, we find

$$\frac{\partial E}{\partial X} = (m_5 x_{li} - m_1)[X_i(m_5 x_{li} - m_1) + Y_i(m_6 x_{li} - m_2) + Z_i(m_7 x_{li} - m_3) - (m_4 - x_{li})] +$$

$$(m_5 y_{li} - m_1)[X_i(m_5 y_{li} - m_8) + Y_i(m_6 y_{li} - m_9) + Z_i(m_7 y_{li} - m_{10}) - (m_{11} - y_{li})] +$$

$$(n_5 x_{ri} - n_1)[X_i(n_5 x_{ri} - n_1) + Y_i(n_6 x_{ri} - n_2) + Z_i(n_7 x_{ri} - n_3) - (n_4 - x_{ri})] +$$

$$(n_5 y_{ri} - n_8)[X_i(n_5 y_{ri} - n_8) + Y_i(n_6 y_{ri} - n_9) + Z_i(n_7 y_{ri} - n_{10}) - (n_{11} - y_{ri})] = 0$$

$$\frac{\partial E}{\partial Y} = (m_6 x_{li} - m_2)[X_i(m_5 x_{li} - m_1) + Y_i(m_6 x_{li} - m_2) + Z_i(m_7 x_{li} - m_3) - (m_4 - x_{li})] +$$

$$(m_6 y_{li} - m_9)[X_i(m_5 y_{li} - m_8) + Y_i(m_6 y_{li} - m_9) + Z_i(m_7 y_{li} - m_{10}) - (m_{11} - y_{li})] +$$

$$(n_6 x_{ri} - n_2)[X_i(n_5 x_{ri} - n_1) + Y_i(n_6 x_{ri} - n_2) + Z_i(n_7 x_{ri} - n_3) - (n_4 - x_{ri})] +$$

$$(n_6 y_{ri} - n_9)[X_i(n_5 y_{ri} - n_8) + Y_i(n_6 y_{ri} - n_9) + Z_i(n_7 y_{ri} - n_{10}) - (n_{11} - y_{ri})] = 0$$

$$\frac{\partial E}{\partial Z} = (m_7 x_{li} - m_3)[X_i(m_5 x_{li} - m_1) + Y_i(m_6 x_{li} - m_2) + Z_i(m_7 x_{li} - m_3) - (m_4 - x_{li})] +$$

$$(m_7 y_{li} - m_{10})[X_i(m_5 y_{li} - m_8) + Y_i(m_6 y_{li} - m_9) + Z_i(m_7 y_{li} - m_{10}) - (m_{11} - y_{li})] +$$

$$(n_7 x_{ri} - n_3)[X_i(n_5 x_{ri} - n_1) + Y_i(n_6 x_{ri} - n_2) + Z_i(n_7 x_{ri} - n_3) - (n_4 - x_{ri})] +$$

$$(n_7 y_{ri} - n_{10})[X_i(n_5 y_{ri} - n_8) + Y_i(n_6 y_{ri} - n_9) + Z_i(n_7 y_{ri} - n_{10}) - (n_{11} - y_{ri})] = 0$$

… … … (4.4)

From these three equations we find (*X,Y,Z*).

Sample 3D scene coordinates with their corresponding left image and right image coordinates are shown below.

X	*Y*	*Z*	x_l	y_l	x_r	y_r
66.5	26.5	0	310	131	344	130
0	37.5	43.5	429	297	462	297
172.5	0	151	53	245	44	247
68.5	0	79	242	289	263	291
12	14	0	371	231	413	231
101	14	0	235	92	266	92
14	82	0	503	155	537	154
55.5	56	0	387	118	418	117
0	17	10.5	393	264	434	265
0	101.5	41	558	226	584	226
0	16	71	382	368	412	369
0	78.5	71	508	300	530	301
0	71	10	498	207	534	206
11.5	0	15.5	342	273	383	274
108.5	0	11.5	193	114	222	114
71.5	0	48	244	233	271	234
129	0	106.5	135	259	140	260
47.5	0	118	269	391	285	392

Fig 4.3 : (x_l,y_l) and (x_r,y_r) from calibration image pair and their corresponding (*X,Y,Z*) values

These (X,Y,Z), (x_l,y_l) and (x_r,y_r) values are put into twenty two equations of (4.3) to find m and n parameters. Obtained m and n values have been shown next.

m parameter values		n parameter values	
m_1	-1.614941	n_1	-1.757358
m_2	1.765514	n_2	1.612260
m_3	-0.245561	n_3	-0.439582
m_4	361.634247	n_4	407.197815
m_5	-0.000379	n_5	-0.000372
m_6	-0.000382	n_6	-0.000433
m_7	-0.000257	n_7	-0.000234
m_8	-1.590302	n_8	-1.594210
m_9	-1.126352	n_9	-1.152009
m_{10}	1.565309	n_{10}	1.585569
m_{11}	264.140472	n_{11}	264.599884

Fig 4.4 : m and n parameters using (X,Y,Z), (x_l,y_l) and (x_r,y_r) of fig 4.3 into equations (4.3).

These m and n values are used in equations (4.4) to find scene coordinate (X,Y,Z) for any corresponding points (x_l,y_l) and (x_r,y_r) from stereo image pair.

V. FINDING STEREO DISPARITY BY MATCHING LASER POINTS

5.1 Introduction

If a physical point on the face can be detected in both left and right images, then, it is possible to find the disparity of correspondence. If that physical point has some good feature, then it will be easy to detect it. Laser is red and bright enough and so it is easily detected when falling on a face. A vertical laser beam is swept over the face from one side to the other. Five to ten seconds sweeping is enough to cover the whole face. To

Fig. 5.1 a): Left camera image from a laser sweep.

make the laser beam relatively bright, the sweeping is accomplished in low lighting. While sweeping, both cameras capture frames synchronously. Each of these frames

contains a vertical laser line. In each pair of frames the laser spine is detected, giving disparity between points in the spines. Fig. 5.1 shows a pair of images containing a laser line.

Fig. 5.1 b): Right camera image from a laser sweep.

5.2 Laser Spine Detection

After getting the images from laser sweep, the laser spines are detected. Every color image has red, green and blue frames. Laser is red. So, in the red frame the laser has more contrast than it has in the other two frames. A program that was already built in the Intelligent Systems Lab was used to detect the laser spines in red frames. This program uses the following steps:

1. Smoothing the image to removes noise.
2. Enhancing the image to give the laser more contrast.
3. Finding the mid point of an image region with a high intensity. Here, the center of a laser spine is detected resulting in edges that go through the spine.
4. Getting the strongest edge. One row might contain more than one edge. Here the strongest edge is taken.
5. Removing short segments.

Fig. 5.2 shows images after detecting laser spines in the images shown in Fig. 5.1.

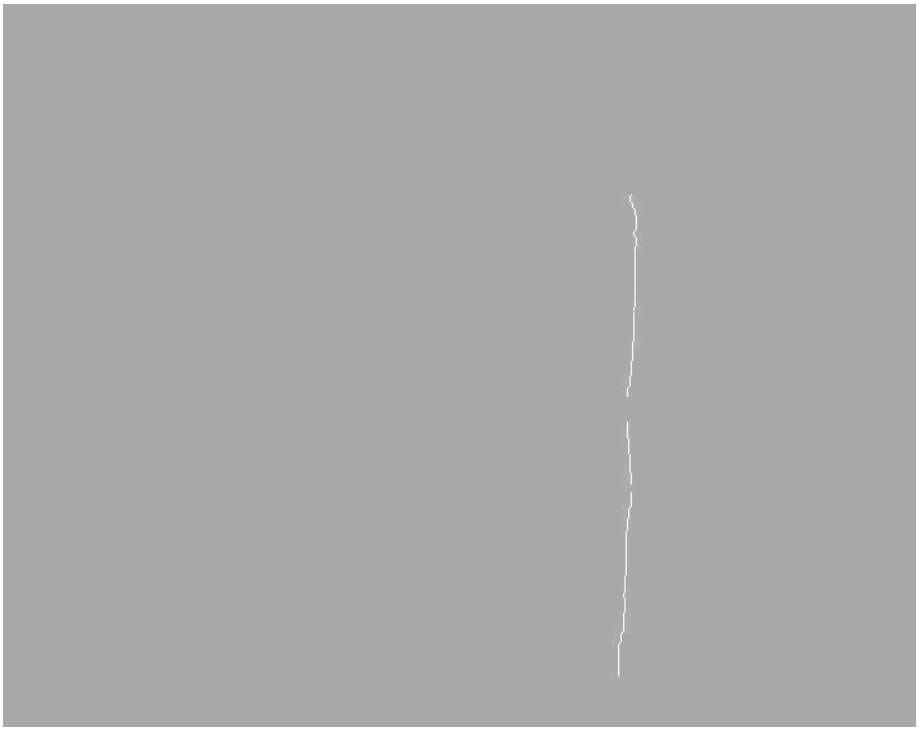


Fig. 5.2 a): Laser spine in left image.

Figure 5.2 shows that most of the laser points are detected. The right image has a false laser segment. Removal of this segment will be discussed in section 5.3.

Fig. 5.2 b): Laser spines in left and right images.

5.3 Finding Disparity

After detecting the laser spines in both images, the images are scanned horizontally to detect a left image laser point and it's corresponding right image laser point. These two points refer to the same physical point. Their coordinates are substituted in equation 4.4 to find the 3D coordinates of their corresponding physical point. These coordinates are written into a file, which we call *depth file*, for later use. The whole image is scanned horizontally to find coordinates of corresponding points and their physical 3D points.

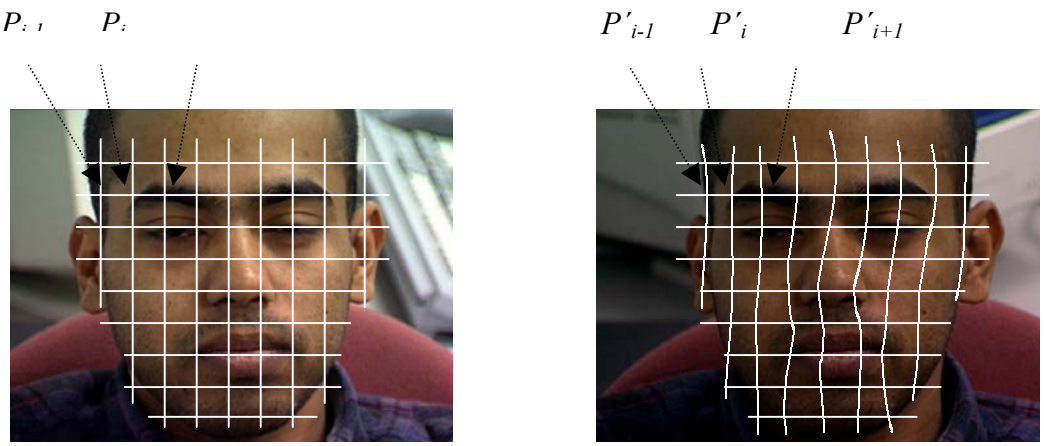
5.4 Removing False Edges

All the image pairs are processed according to Sections 5.2 and 5.3. This processing will give us depths of most of the face points on the face and also their coordinates in the left and right images. These coordinates are saved in the *depth file*. A program already developed in the Intelligent Systems Lab can load this file and show the reconstructed face. False edge points generate 3D points with their positions in front of or behind the face surface. A program has been developed to remove such points. This program takes two depth values as input. One depth value is approximately the nose tip. The other one is the depth near the ears. These two depths are determined interactively from the 3D view. The program keeps only the points inside these two depths and discards others. Refined data are saved in a file, which we call *refined depth file*.

VI. STEREO MATCHING USING AN ENERGY MINIMIZING ALGORITHM

6.1 The Grid Setup

It would be very time consuming if we matched all points in the left and right images. Rather, we will match some of the points and interpolate the rest. The left image is divided into rows and columns. Cross sectional points of the rows and columns will be matched. We are interested in points on the face. To separate the face region that we are interested in from the background, a half elliptical region is defined by three mouse clicks. Two are at the outer ends of the two eyes and the other is at the end of the chin. Fig. 6.1 shows a grid defined in this manner on the face.



Left camera image Right camera image

Fig. 6.1: Grid setup on the face.

Now if a point P_i in the left image is to match in the right image, in the right image we search for a match in both sides of corresponding grid point P'_i . The search area is

within left point P'_{i-1} and P'_{i+1}. The search area, of course, needs to be short to avoid incorrect matches and it should be ensured that P_i will match within P'_{i-1} and P'_{i+1}. The disparity information found from laser sweep is utilized here. To ensure that a point in the left image will find its match within the search area the grid in the right image is initialized using the laser disparity. When initialized, laser disparity brings the right image grid points very close to the exact match points. Some points' disparities remain unknown. They are interpolated from disparities of surrounding points.

6.2 Initializing the Grid from Laser Disparity

After processing the laser swept images, we produced a refined depth file that has information about correspondence between points in the left image and the right image.

Fig. 6.2 a): Initializing left image grid with laser disparity.

This file is used to initialize the grid points in both images. Points only within the half ellipse are initialized. Fig. 6.2 shows the initialized grid.

Fig. 6.2 b): Initializing right image grid with laser disparity.

Fig. 6.3 a): Left image grid after filling the holes.

Fig. 6.3 b): Right image grid after filling the holes.

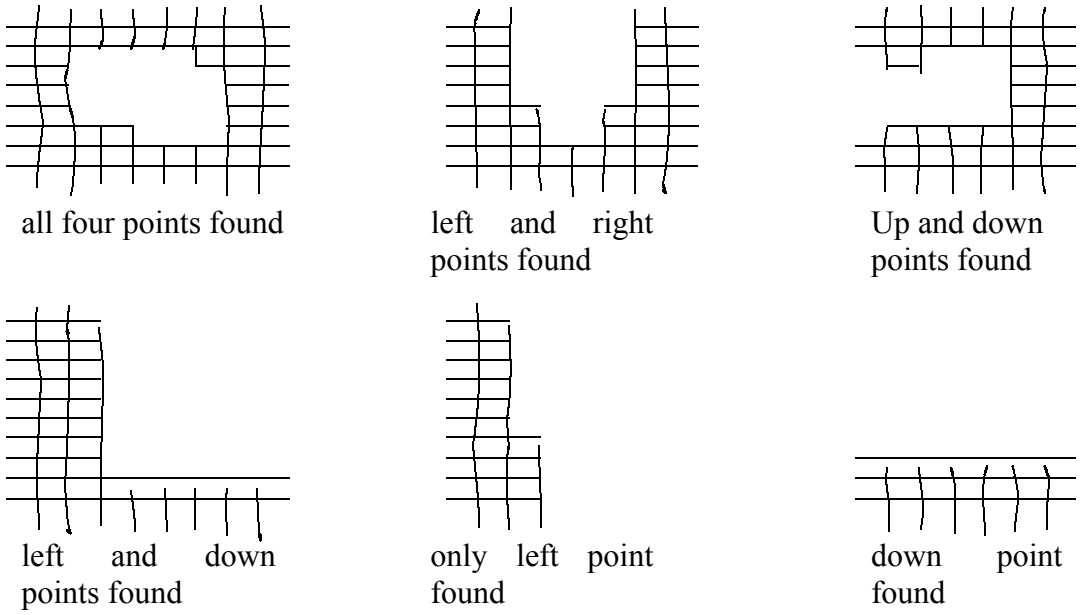


Fig. 6.4: Holes in the grid.

The refined depth file does not have information about all the points. This lack of information creates hole in the right grid. x coordinates of grid points in the hole are evaluated from surrounding points using linear interpolation. To interpolate within a hole, we look for already initialized grid point to left, right, up and down. There might be six or more cases depending on whether an initialized point exists or not. For the case shown in Fig. 6.4, we see that x_5 can be interpolated using either vertical points or horizontal points. Considering horizontal points (x_1,y_1) and (x_2,y_2) the interpolated value comes as $\left(x_1+\frac{3}{5}\times(x_2-x_1)\right)$ where 5 is the number of cells between x_1 and x_2 and 3 is the number of cells between x_1 and x_5. Again considering vertical points (x_3,y_3) and (x_4,y_4) the interpolated value comes as $\left(x_3+\frac{3}{5}\times(x_4-x_3)\right)$. For better approximation we have taken average of these two values. Thus we have

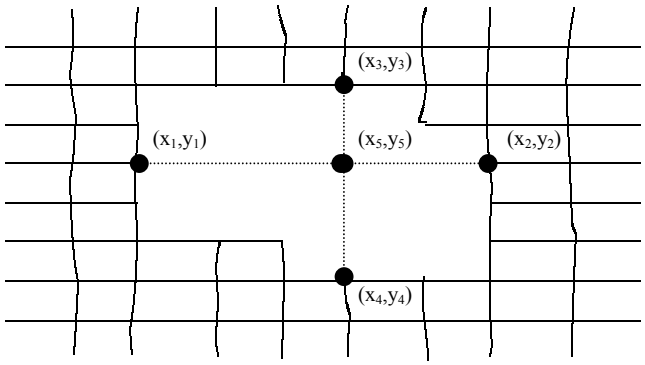


Fig. 6.5: Interpolating hole points.

$x_5=\frac{1}{2}\left(x_1+\frac{3}{5}\times(x_2-x_1)+x_3+\frac{2}{5}\times(x_4-x_3)\right)$. For the other cases, hole points' x-coordinates are evaluated from one or two surrounding points.

6.3 The Energy Minimizing Algorithm

We have a half elliptic grid with regular shape in the left image and another grid with irregular shape in the right image. Due to perspective foreshortening vertical lines in the left grid distort to the left or right from the position it has in the left grid. The horizontal lines remain the same. Grid points in the right image are situated very close to their matching positions. This happens due to the use of disparity from refined depth file and applying interpolation later. Now, to correct their matching the Energy Minimizing Algorithm is applied. We will take the grid points one by one from the left and see where they match the best in the right image. This approach raises two questions,

i) What image feature should we use for a search.

ii) Where and how much area is to be searched to find a match.

These two questions introduce two energy terms in the algorithm. The first one introduces the external energy and the second one the internal energy. These two energy terms are denoted respectively by E_{ext} and E_{int}. The Energy Minimizing Algorithm has the following steps:

1. Develop an energy equation E_i combining both E_{ext} and E_{int} that evaluates energy for a match point in the right image in such a way that less energy indicates a better match.
2. Define the grid points P_{Li} in the left image. Only these points will be searched for in the right image.

3. For a grid point P_{Li} in the left image calculate all energies E_i for all points p_{Ri} in a predetermined right image area A where P_{Li} is expected to exist. A is defined such that ordering constraint is not violated.
4. The point p_{Ri} that has the minimum E_i value is the best match for P_{Li}. Define new A surrounding the point p_{Ri}.
5. For all grid points in the left image follow steps 2 and 3 to find the match and calculate locally minimum E_i for each.
6. Sum all minimum energy values for all P_{Li} points. Assuming this summation is E_n, where n means nth iteration, if $E_{n-1} - E_n > \varepsilon$, where ε is a small threshold value, go to Step 3. Otherwise quit.

Energy minimizing algorithm is iterative. It stops only when the global minimum energy E_n does not decrease at all or decreases very little. The details of this algorithm will be discussed further in the next sections.

6.3.1 External and Internal Energies

External Energy

Assuming $P_{i,j}$ is a grid point where i and j mean the i *th* row and the j *th* column of the grid, a small image area surrounding $P_{i,j}$ will remain almost unchanged in the right image. This property allows us to use a well-known method called template matching. In template matching a template surrounding point $P_{i,j}$ is taken. Then the intensity values within the template are compared with intensity values of a same size template in the right image. There are two methods to compare two templates - Sum of Absolute

Difference (SAD) and Cross Correlation (CC). Each method measures how well both templates match. If a template is compared with *n* templates we get *n* values. Among these *n* values, the lowest value (for SAD) or the highest value (for CC) indicates that the match corresponding to this value is the best.

Sum of Absolute Difference (SAD):

Consider a template of *k*l* pixels with *k* rows and *l* columns. $I_{LC}(i,j)$ is the intensity value of pixel *(i,j)* in the template in the left image. $I_{RC}(i,j)$ is the intensity value for the template in the right image. *C* stands for color and R, G and B refer to templates in red, green and blue frames of a color image. Then SAD of these templates is calculated in the following way:

$$SAD = \sum_{C=R,G,B} \sum_{i=1..k,j=1..l} |(I_{LC}(i,j) - A_{LC}) - (I_{RC}(i,j) - A_{LC})|$$

$A_{LC} = \dfrac{\sum I_{LC}(i,j)}{k*l}$ is average intensity of left template and

$A_{RC} = \dfrac{\sum I_{RC}(i,j)}{k*l}$ is average intensity of right template.

A_{LC} and A_{RC} are used to normalize the templates. Normalization reduces the effect of shadows resulting from different positions of the two cameras or from uneven lighting in the field of view.

SAD = 0 if both templates are the same. The value increases as the matching becomes worse. The highest possible value is 3*(k*l*255) where intensity ranges from 0 to 255. So, SAD value ranges from 0 to 3*(k*l*255). This range can be transformed into a range from 0 to 1 by dividing it by 3*(k*l*255).

Cross Correlation (CC):

$$CCC = \frac{\sum_{C=R,G,B} \left(\sum_{i=1..k, j=1..l} ((I_{LC}(i,j) - A_{LC}) \times (I_{RC}(i,j) - A_{RC})) \right)}{\sum_{C=R,G,B} \left(\sqrt{\sum_{i=1..k, j=1..l} ((I_{LC}(i,j) - A_{LC})^2} \times \sqrt{\sum_{i=1..k, j=1..l} ((I_{RC}(i,j) - A_{RC})^2} \right)}$$

A_{LC} and A_{RC} are used to normalize the templates. As mentioned before, normalization reduces shadowing effect. It is essentially required for *CCC* calculation too. Fig 6.6 a) shows two 4*4 dissimilar templates. Fig. 6.6 b) shows them after applying normalization. In this example only one among three - red, green and blue templates has been shown.

81	67	34	250
253	10	187	78
3	65	190	21
56	112	32	1

250	34	1	56
76	12	5	73
43	195	234	45
2	54	2	231

Fig. 6.6 a): Two dissimilar templates.

-9	-23	-56	160
163	-80	97	-12
-87	-25	100	-69
-34	22	-58	-89

168	-48	-81	-26
-6	-70	-77	-9
-39	113	152	-37
-80	-28	-80	149

Fig. 6.6 b): Templates after normalization.

To calculate the CCC, corresponding cell values are multiplied. In case of dissimilar templates some of these values obtained from multiplications are found positive and

some are found negative. Summing these positive and negative values cancel each other resulting in a small value. In the example above, CCC is calculated to be 0.078382.

81	67	34	250
253	10	187	78
3	65	190	21
56	112	32	1

81	67	34	250
253	10	187	78
3	65	190	21
56	112	32	1

Fig. 6.7 a): Two similar templates.

-9	-23	-56	160
163	-80	97	-12
-87	-25	100	-69
-34	22	-58	-89

-9	-23	-56	160
163	-80	97	-12
-87	-25	100	-69
-34	22	-58	-89

Fig. 6.7 b): Similar templates after normalization.

From templates shown in Fig. 6.7, CCC comes 1.0, which is greater than the previous value. CCC has the highest value 1.0. When templates are similar, the values obtained from multiplications are found positive, and their summation gives a large value. A good match will have a value 1.0 or close to it. The value decreases down to –1 as similarity between two templates decreases. Fig. 6.8 shows external energy curve when a left grid point template is matched 20 pixels to the left and 20 pixels to the right of the corresponding grid point in right image. The curve will have a maxima near or on 1.0 if a match is found. Shape of the other parts of the curve is abrupt and depends on intensities in images. For a sharp change of intensity, the curve will be abrupt. For small change of intensity the curve will be smooth.

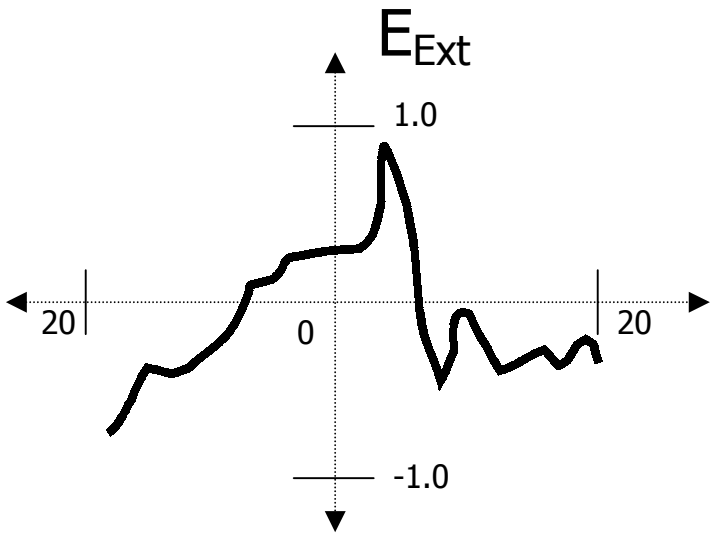


Fig. 6.8: External energy curve.

Internal Energy

External energy is not enough to find a correct match. In fact an image may have no significant gradient throughout a large region, for example the cheeks and the forehead of a face.

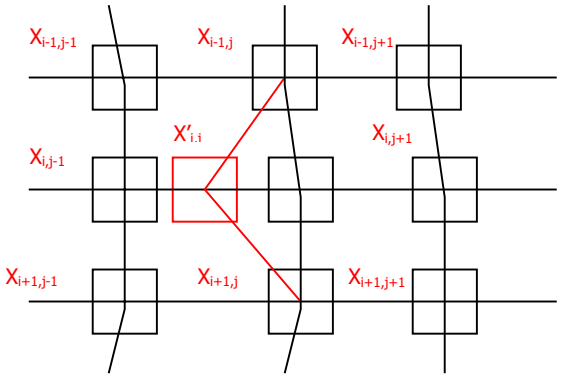


Fig. 6.9: False template matching.

Absence of gradient causes a left image template to match in several places in the right image. This will happen when gradient is low across the search area. In this case, the external energy assists to correct a false match. A false match generally causes a grid

point to move away from its correct matching position. Fig. 6.9 illustrates this situation. In the first iteration, the algorithm points at $X_{i+k,\ j+l}$ ($k,\ l$ = -1,0,1) are found from matching laser points or from interpolation.

In the figure, the template at $X_{i,j}$ commits a false match and moves far away at $X'_{i,j}$. Internal energy has been introduced to avoid this false match. This energy pulls the template towards its correct match position that is assumed to be situated somewhere close to the center of the surrounding eight grid points. Our objective is to find a match somewhere near $X''_{i,j}$ which is simply the average of x-coordinate values of surrounding grid points.

Thus $X''_{i,j} = \frac{1}{8}\left(\sum_{\substack{k=-1,0,1 \\ l=-1,0,1}} X_{i+k,j+l}\right)$.

To find a match for the template at grid point (i,j) in the left image, we search in the right image from grid point (i,j-1) all the way up to grid point (i,j+1), pixel by pixel. So $X'_{i,j}$ has a possible value from $X_{i,j-1}$+1 to $X_{i,j+1}$ –1. Internal Energy of the searching point (i,j) is defined by

$$E_{\text{int}} = \frac{2\,|\,X''_{i,j} - X'_{i,j}\,|}{(X_{i,j+1} - X_{i,j-1})}.$$

The expected match is the point with x-coordinate value $X''_{i,j}$. At this point, the internal energy is 0. When moving away from this point in both sides, internal energy will increase up to 1.0. $|\,X''_{i,j} - X'_{i,j}\,|$ has the maximum value very close to the half of $(X_{i,j+1} - X_{i,j-1})$. That is why multiplication by 2 is used to keep E_{int}'s maximum value near 1.0. The best match is expected to have a small internal energy.

Figure 6.10 shows the internal energy curve. In the middle of the search area internal energy is zero. The energy value increases linearly from center in both directions and reaches 1.0 at both ends.

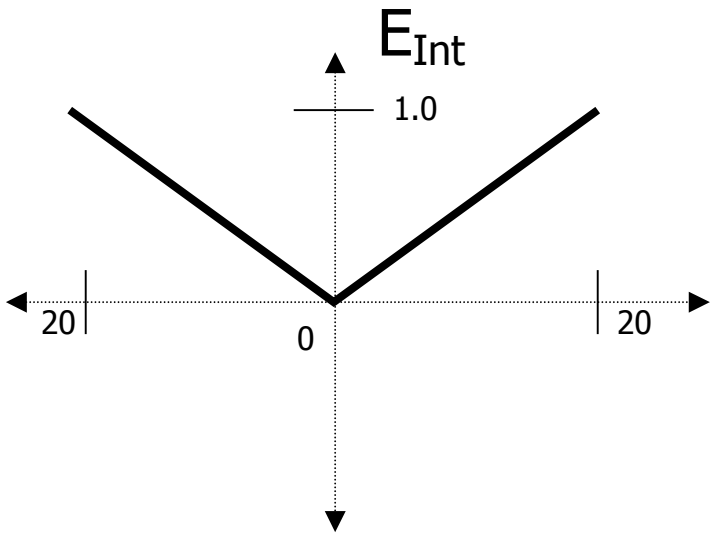


Fig. 6.10: Internal energy curves.

Developing The Energy Minimizing Equation

A good match has a high external energy and a low internal energy. Both may not be possible at the same time. So we set a compromise between these two, defining the following energy equation

$E = \alpha \times E_{\text{int}} - \beta \times E_{ext}$, where α,β are constants. … … … … 6.1

The template at left grid point (*i,j*) is matched with templates between right grid points (*i,j-1*) and (*i,j+1*). *E* is calculated for every match position and the point with minimum *E* is selected as the best match. Minimum *E* selects a match that is near $X''_{i,j}$ because its internal energy, acting as a pulling power towards the center.

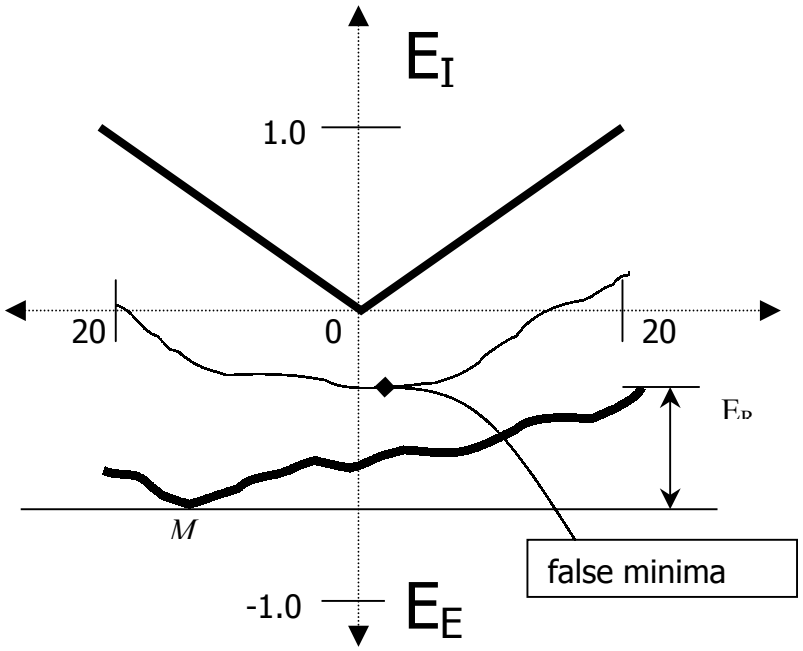


Fig. 6.11 a): False minima before energy normalization.

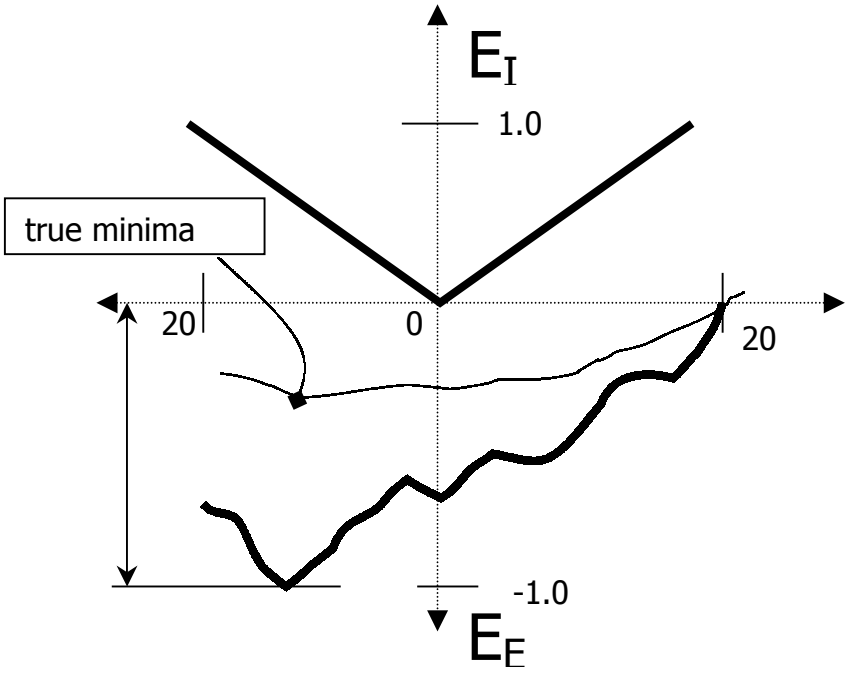


Fig. 6.11 b): True minima after energy normalization.

In equation 6.1 the external energy E_{ext} is calculated using cross correlation. Cross correlation gives a high value for a good match. That is why there is a negative sign before it. α and β are constant parameters that control the importance of one energy term against the other. A large value of α with respect to β means that the grid has a large internal energy and so it is stiff. On the other hand, large value of β emphasizes external energy and it is as if the grid is more elastic.

Slopes of the external and internal energy terms have a great impact on the best match position. In Fig. 6.9 (a), external energy has a slope smaller than the slope of internal energy. The upper thick curve is for internal energy and the lower thick curve is for the external energy. The thin curve between them is the resultant energy minimizing equation curve from equation 6.1. This curve shows that the minimum for external energy is near the center but that of the internal energy is at M. This will happen when external energy curve falls within a small range, E_R as shown in the figure. The image region with a slow change of intensity faces this problem. To bring the minima at *M*, external energy values are transformed from range E_R to range 0-1.0 as has been shown in Fig. 6.9 (b).

Caculating E

E_{ext} is calculated from the cross correlation between the left image template and the right image template. The template at grid point (*i,j*) in the left image is matched with the templates at points between grid points (*i,j-1*) and (*i,j+1*) in the right image. As the match is tested from left to right, the internal energy E_i decreases and becomes zero at $X''_{i,j.}$ This change influences the internal energy of the two points (*i,j-1*) at left and (*i,j+1*)

at right of point at $X'_{i,j}$. Their external energy remains the same. So, calculation of energy of point at $X'_{i,j}$ involves the calculation of energies at point $(i,j-1)$ and $(i,j+1)$. In fact to find a good match we take average of these three energies for each match and select the point with the minimum average energy. We call this average energy E_s, where s stands for snake.

$$
\begin{aligned}
E_s(i,j) &= \frac{1}{3}(E(i,j-1)+E(i,j)+E(i,j+1)). \\
&= \frac{1}{3}(\alpha_{i,j-1}E_{\text{int}}(i,j-1)-\beta_{i,j-1}E_{ext}(i,j-1) \\
&+\alpha_{i,j}E_{\text{int}}(i,j)-\beta_{i,j}E_{ext}(i,j) \\
&+\alpha_{i,j+1}E_{\text{int}}(i,j+1)-\beta_{i,j+1}E_{ext}(i,j+1)).
\end{aligned}
$$

All three points have different α and β values. Fig. 6.10 shows a part of the grid setup. The dashed template is moved within points $(i,j-1)$ and $(i,j+1)$ to find a match that has even less energy than its current match at $X_{i,j}$.

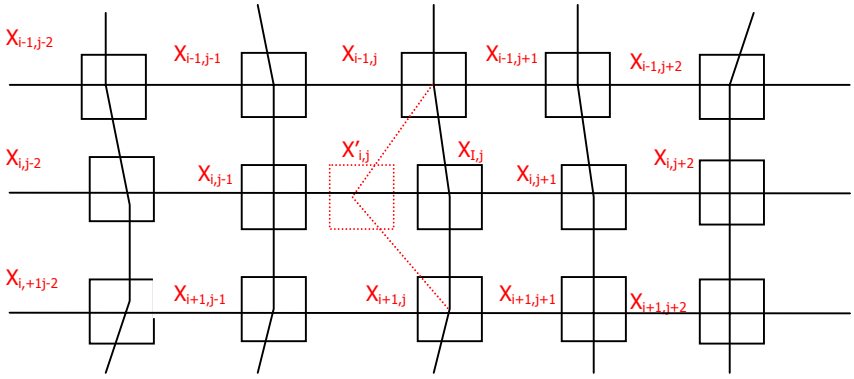


Fig 6.12: Calculating E_s (i,j).

$$
E_{\text{int}}(i,j-1)=\frac{2\times|X''_{i,j-1}-X_{i,j-1}|}{(X_{i,j}-X_{i,j-2})}, \text{ where } X''_{i,j-1}=\frac{1}{8}\Big(\sum_{\substack{k=-1,0,1\\ l=-2,-1,0}} X_{i+k,j+l}\Big).
$$

$$E_{\text{int}}(i,j) = \frac{2\times|X''_{i,j} - X'_{i,j}|}{(X_{i,j+1} - X_{i,j-1})}, \text{ where } X''_{i,j} = \frac{1}{8}\left(\sum_{\substack{k=-1,0,1 \\ l=-1,0,1}} X_{i+k,j+l}\right).$$

$$E_{\text{int}}(i,j+1) = \frac{2\times|X''_{i,j+1} - X'_{i,j+1}|}{(X_{i,j+2} - X_{i,j})}, \text{ where } X''_{i,j+1} = \frac{1}{8}\left(\sum_{\substack{k=-1,0,1 \\ l=0,1,2}} X_{i+k,j+l}\right).$$

$E_{ext}(i,j-1)$, $E_{ext}(i,j)$ and $E_{ext}(i,j+1)$ are calculated using the cross correlation method by matching the left image templates with corresponding templates in the right image at $X_{i,j-1}$, $X'_{i,j}$ and $X_{i,j+1}$.

Global Minimum Energy E_n is the summation of $E_s(i,j)$ for all grid points. As iteration continues, the grid points find good match and E_n becomes smaller. After several iterations E_n does not decrease further, indicating that the grid has become stable.

Iteration	En
1	-0.617
2	-0.791
3	-0.811
4	-0.821
5	-0.826
6	-0.829
7	-0.836
8	-0.838
9	-0.840
10	-0.841

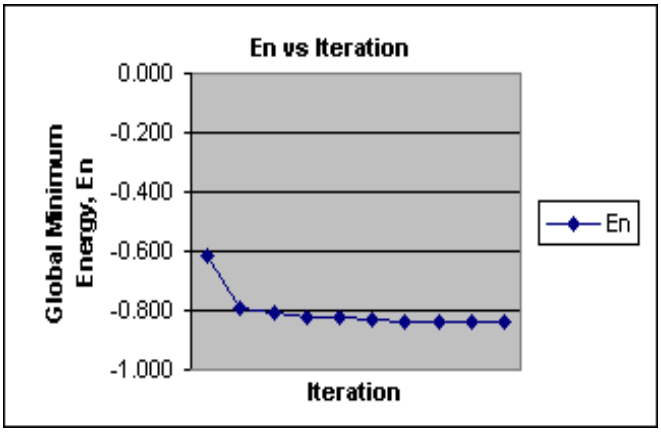


Fig. 6.13: Global energy convergence.

5.3.2 Energy Parameters Selection

In the equation $E = \alpha \times E_{\text{int}} - \beta \times E_{ext}$, α and β are constants. α is the weight of the internal energy and β is that of the external energy. How much control each energy

has on a match point, depends on their relative values. The energy that has the larger parameter value will have more control. Constant values for both may find good matches in some image regions, in some other image regions they might fail. In our case β is fixed as 1.0 and the values of α is changed with disparity gradient. Disparity gradient can be measured from bending of vertical grid line. If the gradient is lower than 3, α is assigned 1.0, otherwise α is the value of that disparity gradient which is $|(X_{i,j-1}+X_{i,j+1})/2-X_{i,j}|$.

5.3.2 Removing the Bad Matches

Bad matches are introduced mainly due to occlusion and illumination difference between images. When seen from two cameras, illumination will depend on bending of object surface and the direction of light source. On a human face, occlusion may appear near the nose. Bad matches also may appear on eyelids because of their hair. Hair on eyes

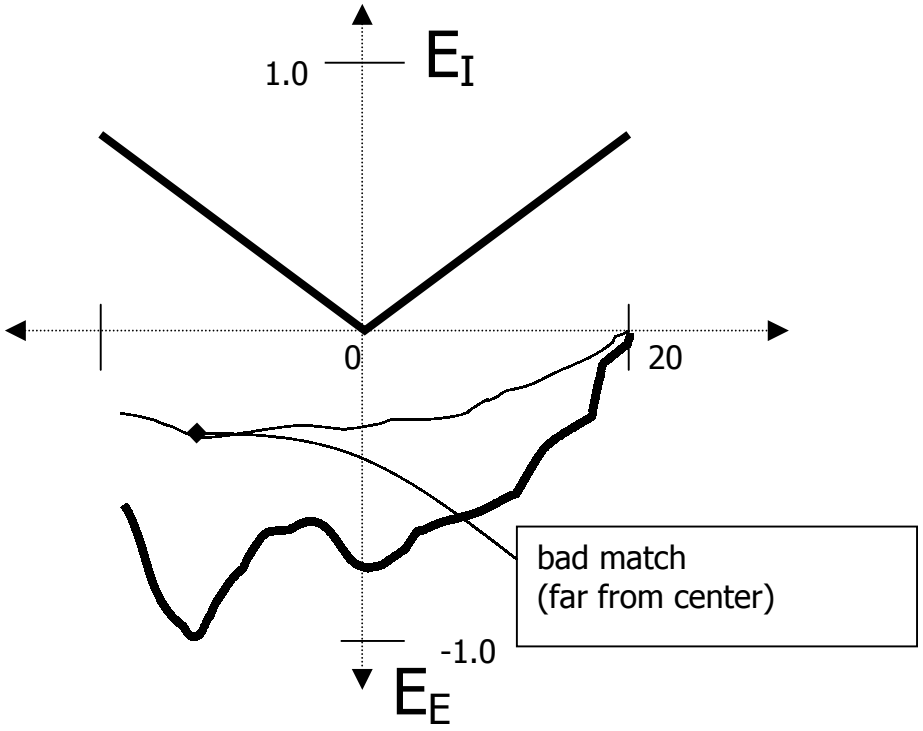


Fig. 6.14 a): Bad match.

has sharp change of depth, which violates the ordering constraint. In these cases we try to make an approximate match, forcing the grid points to be distributed evenly. Energy minimizing algorithm already has this force of even distribution in image region without enough feature points and gradient. To approximate bad matches we just increase this force. Bad matches are usually those points with sharp change of disparity. With respect to the surrounding good matches, which are very close to the center of their surrounding points after several iterations of energy minimizing algorithm, bad matches seem to appear far away from the center. Fig. 6.12 (a) shows a such bad match appearing far from the center.

To drag the bad matches close to the center of their surrounding points, we

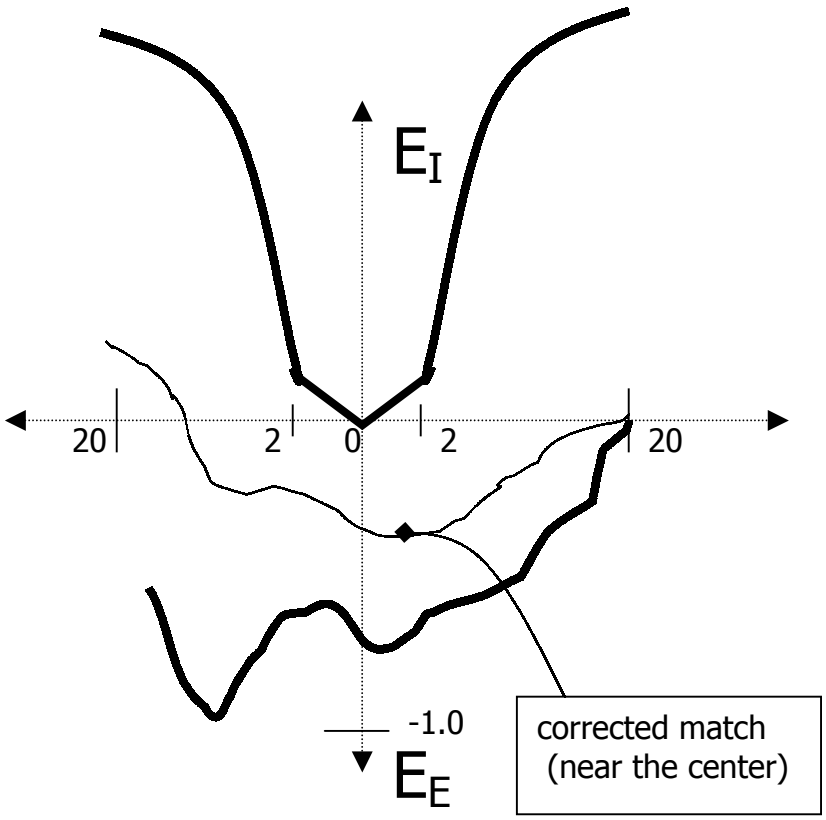


Fig.6.14 b) : Corrected match.

change the shape of the internal energy curve as has been shown in Fig. 6.12 (b). Near the center the curve remains linear like before. Then, it increases exponentially as distance from the center increases in both directions. These two different shapes of the curve serve two different purposes:

The *Exponent* part forces the bad matches to stay close to the center. In this part the energy minimizing equation changes to $E = \alpha \times E^{n}_{\text{int}} - \beta \times E_{ext}$ with *n* as an exponent for internal energy E_{int}. Value of *n* is greater than one.

Linearity near the middle is still maintained so that good matches that are not on the middle but near it are not forced to displace. In this part of the curve value of *n* is one.

5.3.3 Defining the Grid Boundary

The most sensitive part of this algorithm is defining the grid boundary. Fig. 6.10 shows that a grid point uses fourteen neighboring points to calculate its energy. But the boundary points do not have that many neighboring points. For these points, internal energy parameter α is set high with respect to external energy parameter value β.

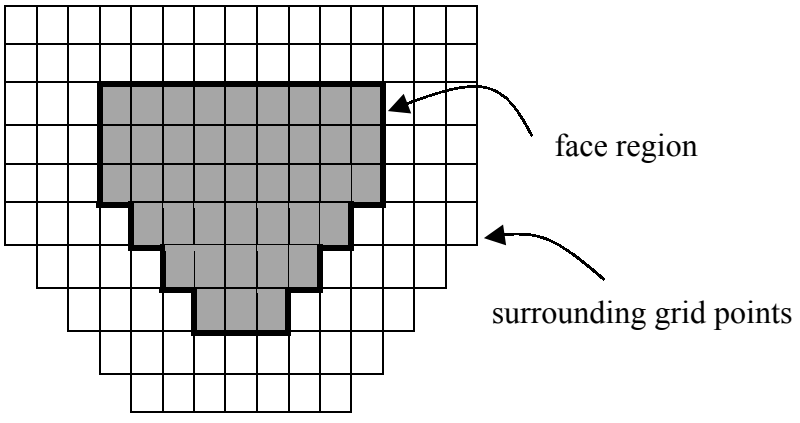


Fig. 6.15: Grid boundary set up.

Assigning high value will discourage left boundary points to go to far left and right boundary points to far right sacrificing good matches. This sacrifice affects matching of nearby interior points. To keep this effect away from the face region we have added some more grid cells surrounding the grid resulting a larger grid. In Fig. 6.15, the shaded grid cells are on the face region.

6.4 Face Reconstruction

6.4.1 Interpolation of 3D face points

Only cross sectional points of the left grid are matched. A grid cell has four such points. Other points within a left grid cell are matched to the points within a right grid cell by using linear interpolation. Fig. 6.16 shows two corresponding left and right grid cells. Left cell is rectangular. After matching, the right cell might transform into a trapezoid.

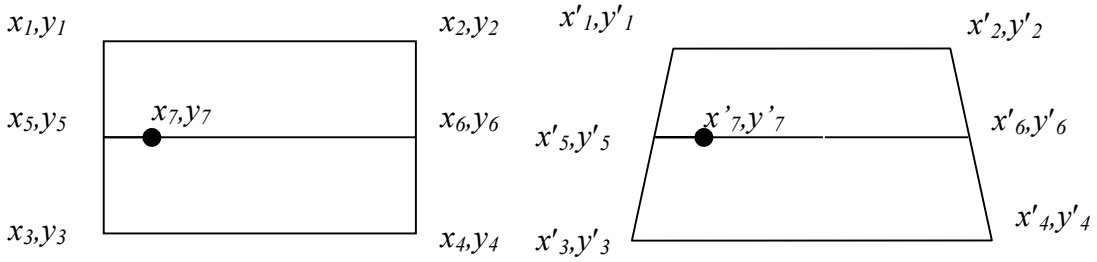


Fig. 6.16: Two corresponding grid cells.

Suppose, we want to interpolated points on line (x_5,y_5) - (x_6,y_6) with respect to line (x'_5,y'_5) - (x'_6,y'_6). First, we find how x_5 and x_6 change to x'_5 and x'_6. Because the scan lines are same in both images, values y_1, y_2, y_3 and y_4 remain the same.

$$x_5^{'} = x_1^{'} + \frac{y_5 - y_1}{y_3 - y_1} \times \left(x_3^{'} - x_1^{'}\right) \text{ and } x_6^{'} = x_2^{'} + \frac{y_5 - y_1}{y_3 - y_1} \times \left(x_4^{'} - x_2^{'}\right)$$

Now x_7 chages to x'_7 such a way that the ratio of (x_7-x_5) to (x_6-x_7) remains the same as the ratio of (x'_7-x'_5) to (x'_6-x'_7).

$$x_7^{'} = x_5^{'} + \frac{x_7 - x_5}{x_6 - x_5} \times \left(y_6^{'} - y_5^{'}\right) \text{ and } y_7^{'} = y_7$$

The point (x_7,y_7) is interpolated as (x'_7,y'_7) in the above equations.

Interpolation of all pixels in the left image may not pick all points in the right image. This will happen when x'_2- x'_1 is greater than x_2-x_1. Fig. 6.8 shows how missing points are created due to perspective foreshortening. Missing points will create vertical gaps in the right image. To avoid this, each half pixel on left image line is mapped to the right

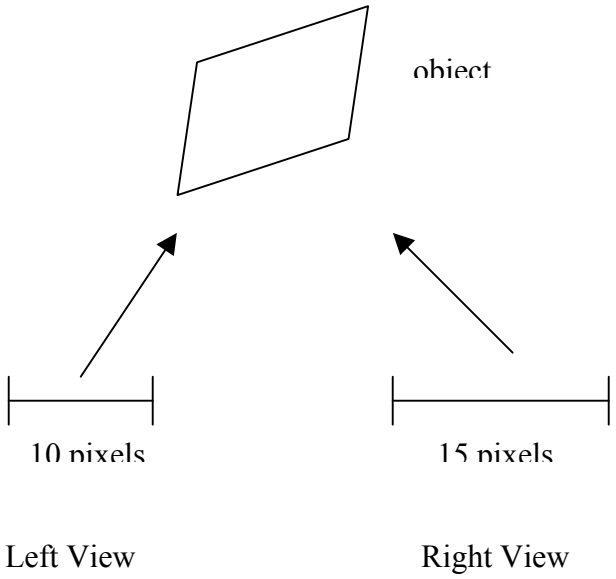


Fig. 6.17: Perspective foreshortening.

image. Exactly how much to advance along the left image line equals to the ratio (x'_2-x'_1)/(x_2-x_1). According to the object's position left image line also may be greater than the right image line.

6.4.1 **Acquiring Fractional Pixel Accuracy**

Even half pixel error in disparity makes an observable error in depth. Due to CCD limitation of the camera it is not possible to acquire accuracy beyond a pixel. However it is possible to estimate using curve fitting. Cross correlation shows some degree of matching on the two pixels on both sides of the pixel with the best match. Three energy values in these three pixels are interpolated to get the maxima. This maxima may come on a fractional pixel position. The three energy values are fitted into the curve $y = ax^2 + bx + c$.

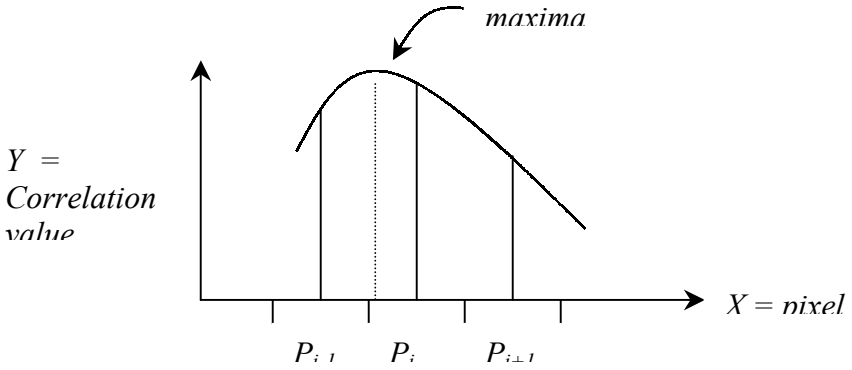


Fig. 6.18: Interpolation of the correlation values.

Substituting three values in the above equation, we can solve for the parameters a and b. Now at the maxima $y' = 2ax + b = 0$. or $x = -b/(2a)$. Here x is the estimated best match position in pixel.

6.4.3 3D Viewing of a Reconstructed Face

For 3D viewing, we need (X,Y,Z) coordinates of face points and color of each of them. We have already described how to find (X,Y,Z) when (x,y) values of two corresponding stereo points are given. To get the color, we simply read the color values at (x,y) in both images and then take the average.

A program was already developed in the Intelligent Systems Lab for viewing 3D objects when (X,Y,Z) values of its points and their color values are given.

VII. STEREO TRACKING

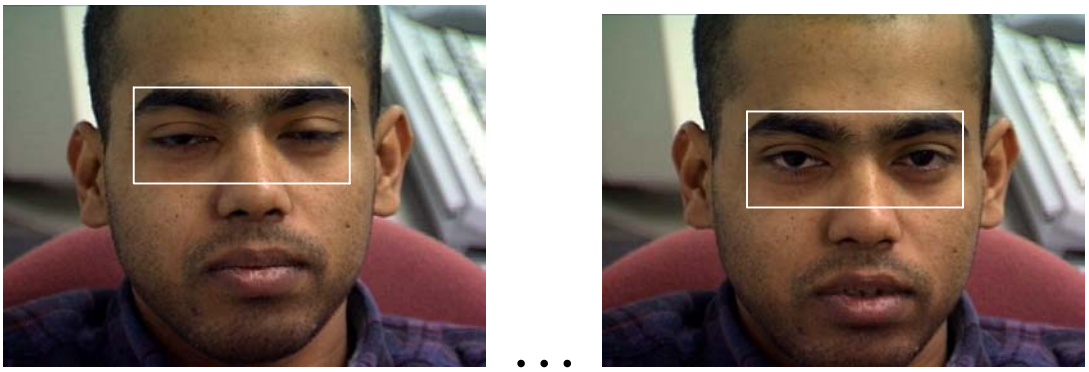
We have a sequence of image frames of a moving and talking face. So far it has been discussed how a 3D face can be reconstructed from one stereo image pair with the laser points as seed by applying the energy minimizing algorithm. Now we will focus on tracking the face and reconstructing it again so that ultimately we can animate a talking 3D face.

To track and then reconstruct the face sequentially in each pair of frames, the first question is what would be the seed points in the next pairs of frames. There are no laser points, hence, no disparity information. The solution follows the fact that between each sequential frame deformation is very small if the camera frame rate is high. The disparities of the face points do not change much. So, disparity information from one frame can be used in the next frame. After matching, all grid points in a stereo image pair are used as seed points in the next stereo image pair. Although the disparities do not change much, the point coordinates might change greatly due to a fast head movement. Accordingly disparity information in the grid will change. Our plan is to attach the grid always to the face. The grid will move as the head moves, hence moving the disparity information. The problem here is to find the approximate shift of the face in horizontal and vertical directions. We call this coarse tracking.

7.1 Coarse Tracking

To find the approximate head shift, a sliding window on the face is defined. It will find the shift amount by looking around for a match of minimum sum of absolute

difference. In one frame the window is defined somewhere on the face. In the next frame, the window is shifted horizontally and vertically up to a predetermined maximum amount. This amount depends on frame rate and speed of head movement. To avoid false matching, the window is defined in a region with enough gradient and texture information. We have chosen a window that contains the two eyes. The window position was defined while defining the grid. Face region around the eyes has enough gradient that a simple matching technique like taking sum of absolute differences is enough for this coarse tracking.



i *th* snap from Left Camera i+n *th* snap from Left Camera

Fig. 7.1: Sliding window for coarse tracking.

The head moves to either the left or right and either up or down. Sliding the window gives us the amount of these movements in pixels. We add the horizontal movement with x-coordinate and the vertical movement with y-coordinate of the grid points. Consequently the grid is initialized on the face in the next image pair. The movement of the face may be both rotation and translation. For translation depth of the

face points may change but their relative disparities do not change. If the face talks or rotates then there is clearly a depth

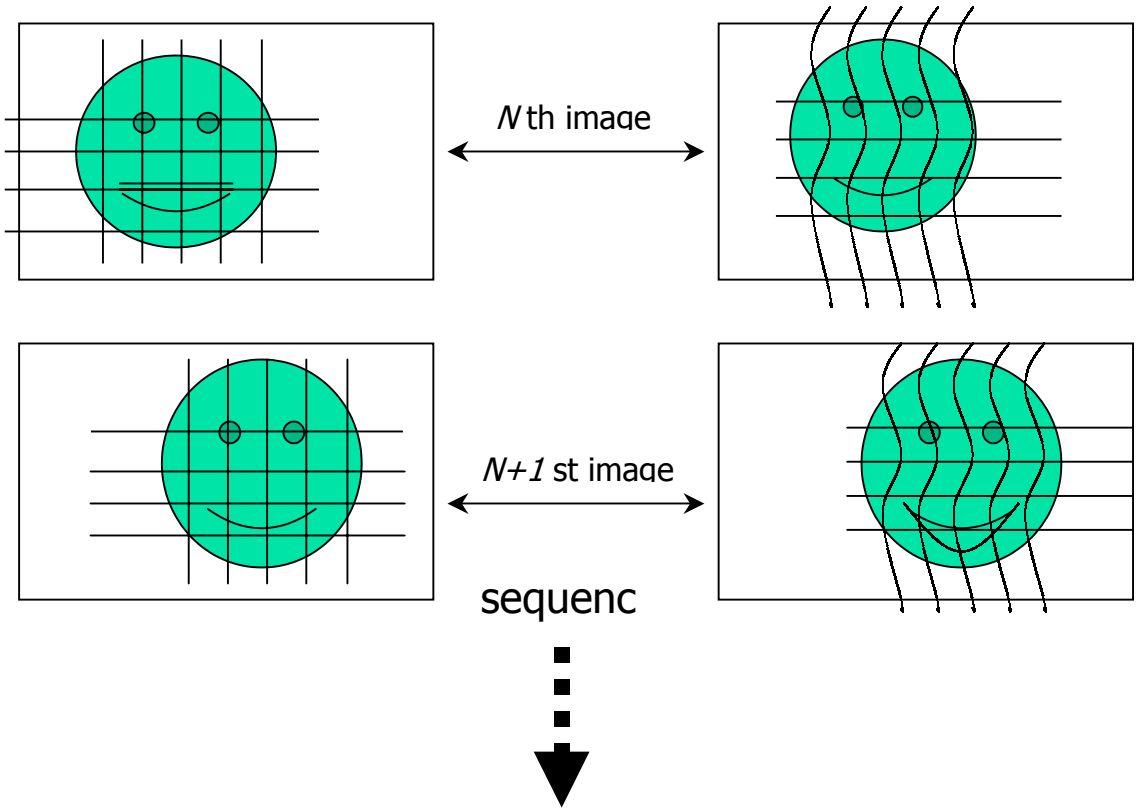
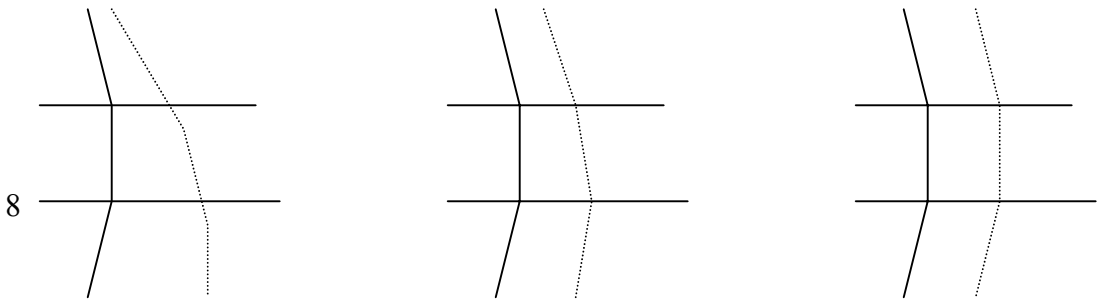


Fig. 7.2: Disparity changes very little as head moves.

change between consecutive frames. If frame to frame movement is not much, then depth i.e. disparity changes very little – by a few pixels only. So, most of the matches will still remain within the search area. The critical portion of the face is its mouth. While talking, the depth changes very sharp when both lips open suddenly after they were closed. So the disparities of points between the lips change a lot. In fact, the points go outside their search region. The energy minimizing algorithm can tackle this type of problem very well. This is discussed in the next section.

7.2 Fine Tracking

After the grid is positioned, we apply the energy minimizing algorithm. This algorithm corrects the approximate match. In a talking face, the points on the mouth and around it move fast and deform more than other region of the face. For this reason, the match point in the right image may fall outside the search region. If a seed point is outside the initial search region, the match will be always false and the snake will never converge. Away from the mouth region the points are almost stationary. They have less movement and deformation. Their matching points fall inside the search region. So they have a high probability to converge. The advantage of a grid snake is that this convergence will bring the matching points of the nearby points into their search region. Now they will find their matches and converge. This process continues extending all over



a) iteration *i* b) iteration *i+1* c) iteration *i+2*

Fig. 7.3: Snake convergence.

the grid until all points find their matches. Fig. 7.3 shows how grid points with matching points inside their search regions help bringing other points closer to their matching points. The solid vertical line is matched and dashed vertical line is mismatched.

Because of the pulling energy, the solid line will attract the dashed line closer to it. The dashed line also pulls the solid line towards it. Dashed line is mismatched, so it is

weaker than the solid line. Gradually the dashed line moves closer to solid line and stops when it finds its match. Because of this snake property, the energy minimizing algorithm needs iterations to come to a stable state. In the stable state, the global energy of the grid does not change much in next iterations.

VIII. RESULTS AND ANALYSIS

9 8.1 Reconstructed 3D Face

Fig. 8.1 (1)-(8) are reconstructed 3D faces shown from different views. Only the portion of the face that is visible from both the cameras has been shown. The broken

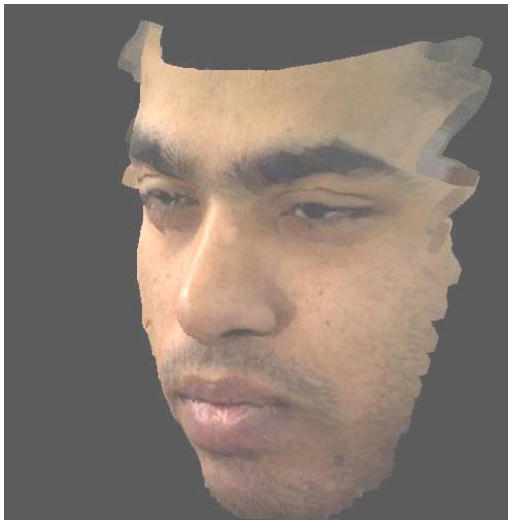
(1)

(2)

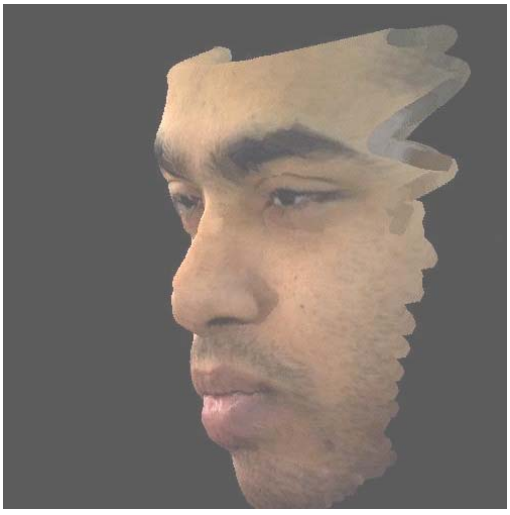
(3)

(4)

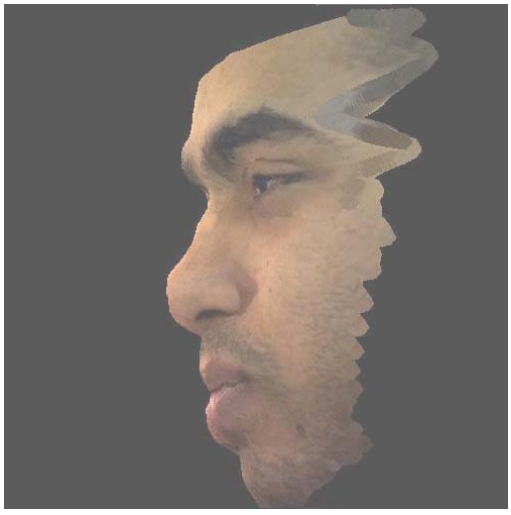
(5)



(6)



(7)



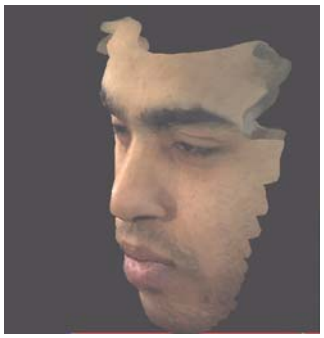
(8)

Fig. 8.1 (1)-(8) : Reconstructed 3D face from different views.

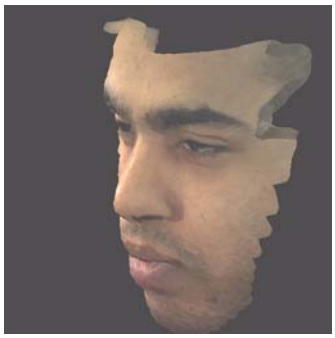
boundaries are due to the fact that behind the face is invisible or extremely sharp in depth. This causes occlusion or too much foreshortening, failing match.

8.2 A reconstructed 3D Image Sequence

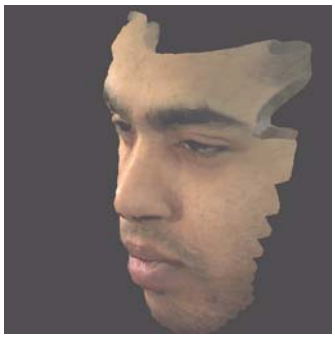
From two different viewpoints, Fig. 8.2 (1)-(26) and Fig. 8.3 (1)-(26) show a 3D image sequence of 26 frames reconstructed from the same stereo image sequence. In the stereo image sequence, the person says something and moves his lips and closes his eyes. All movements have been reconstructed.



(1)



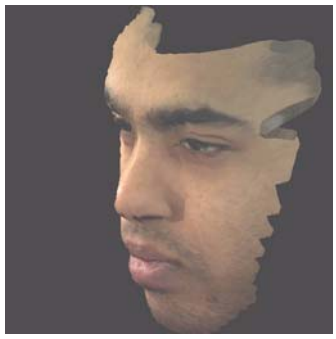
(2)



(3)

(4)

(5)



(6)

(7)

(8)

(9)

(10)

(11)

(12)

(13)

(14)

(15)

(16)

(17)

(18)

(19)

(20)

(21)


(22)


(23)


(24)


(25)

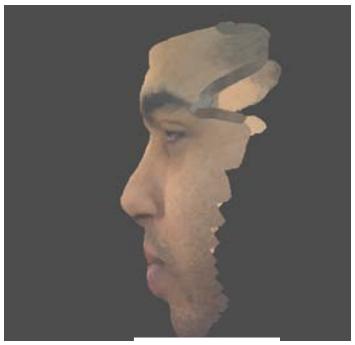

(26)

Fig. 8.2 (1) – (26): 3D reconstructed image sequence from one view.


(1)


(2)


(3)



(4)

(5)

(6)

(7)

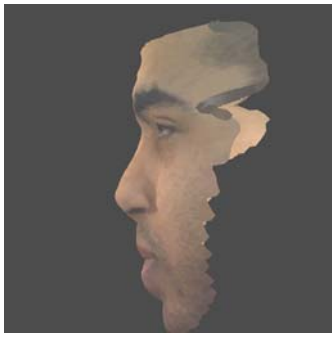
(8)

(9)

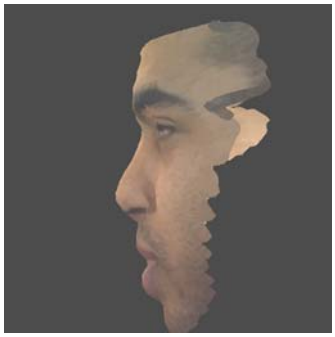
(10)

(11)

(12)



(13)



(14)

(15)

(16)

(17)

(18)

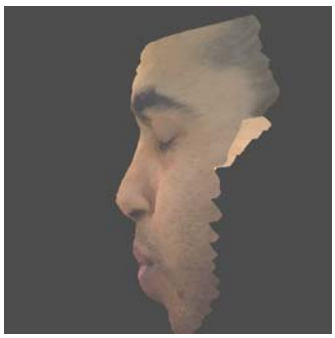
(19)

(20)

(21)

(22)

(23)



(24)

(25)

(26)

Fig. 8.3 (1) – (26): 3D reconstructed image sequence from another view.

8.3 Convergence and Speed

Cross Correlation matching is very sensitive to noise. Stereo images have noise due to occlusion and illumination differences. The occluded areas often observed are located near the nose. Matching was approximated by the pulling energy of the snake. The nose tip is shiny appearing differently in the two images. As the two cameras are positioned at different places, the shiny spot appears at different places on the nose. This makes illumination of the nose tip different and makes matching difficult. Smoothing and normalizing the templates have reduced this difficulty somewhat. Smoothing, however, weakens the feature points. Consequently a high-resolution grid and big templates are required in matching.

It has been observed that approximately for every fifteen millimeters of depth the disparity changes by 3 to 4 pixels when the face appears in a 480×640 image. So, even a one-pixel error in matching deforms the 3D shape by several millimeters. To obtain a smooth surface, curve fitting has been used to evaluate fractional disparities.

An energy minimizing algorithm needs some iterations for it to converge to a stable matching state. A 480×640 image pair with a 80×80 grid converges after twelve iterations using a 12×12 template. This needs almost 20 minutes on a SGI machine. It is possible to make it faster using a smaller template size. But if the template is too small, a wrong match might be found due to ambiguities and noise. Too large a template again may no longer match due to foreshortening and occlusion, resulting in loss of detail and blurring (or dislocating) object boundaries in the resulting disparity map.

8.4 Further Work

The described algorithm will run faster if image rectification is applied. Because of perspective, matching epipolar lines in the images may not represent the same scan line. That is why vertical search by some pixels has been done. Rectification makes the scan lines in the right image fall on the same line of the left image. This consequently eliminates the vertical search.

IX. CONCLUSION

In our work stereo matching has been accomplished using an energy minimizing snake. We have used an energy minimizing algorithm with grid snake. A grid contributes a lot in matching. In human face there are regions that will be matched even though they violate ordering constraint. The rear portion of hair on eyelids falls in a forbidden region, making the matching impossible. Besides, the area near the nose is sometimes invisible to one of the cameras. This creates occlusion. Forehead and cheeks have lack of gradient. An energy minimizing model has a pulling energy to approximately match in these regions. The energy equation has two parameters that have been selected automatically. The external energy parameter is fixed to 1.0 and the internal energy parameter is made dependent on the disparity gradient at the matching point. The pulling energy has been increased beyond a certain limit to avoid bad matches. Laser sweep has been successfully used to collect disparity information in the beginning. Besides faces, the algorithm should work well for other objects if some seed points can be introduced in the first iteration of processing of the algorithm.

3D reconstruction and tracking are steps used in many image analysis applications. One example is speech driven animation that needs 3D movements data from the feature points. These feature points are those face points that move when a person talks. After collecting the 3D movements from reconstructed 3D face and recognizing patterns of movements, the patterns could be associated with the feature points to generate 3D talking face animation that is driven by any speech.

BIBLIOGRAPHY

[1] Meet Geri. The new face of animation, *Computer Graphics World*, Volume 21, Number 2;1998.

[2] A. LeBlanc; P. Kalra; N. Magnenat-Thalmann; D. Thalmann. Sculpting with the Ball & Mouse Metaphor, Proceedings, *Graphics Interface '91*, Calgary, Canada, pp. 152-159;1991.

[3] M. Proesmans; L.V. Gool; Reading between the lines – a method for extracting dynamic 3D with texture, *Proceedings, VRST '97*, pp. 95-102;1997.

[4] P. Fua. From multiple stereo views to multiple 3-D surfaces, *Internationl Journal of Computer Vision 24(1)*, pp.19-35;1997.

[5] P. Fua. Face models from uncalibrated video sequences, *Proceedings, CAPTECH '98*, pp. 215-228;1992.

[6] Valente, S.; Dugelay, J. A visual analysis/synthesis feedback loop for accurate face tracking. *Signal Processing: Image Communication 16*, pp. 585-608;2001.

[7] Baker, H.H.; T.O. Binford. Depth from edge and intensity based stereo. *Proceedings, 7th International Joint Conference On Artificial Intelligence*, Vancouver, Canada 631 – 636;1981.

[8] Ohta, Y.; T. Kanade. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Transactions, Pattern Analysis and Machine Intelligence.* 7 (2), 139-154;1985.

[9] Benshair, A., Miche, P.; Debrie, R. Fast and automatic stereo vision matching algorithm based on dynamic programming method. *Pattern Recognition Letters*, 17: 457-466;1996.

[10] Marr, D.; Poggio T. Cooperative computation of stereo disparity. *Science*, 194:283-287;1976.

[11] Pollard S. B.; Mayhew, J.E.W.; Frisby, J.P. A stereo correspondence algorithm using a disparity gradient constraint. *Perception*, 14:449-470;1985.

[12] Waxman, A. M.; Duncan, J.H. Binocular image flow: Steps toward stereo-motion fusion. *IEEE Transactions. Pattern Analysis and Machine Intelligence.* PAMI-8:715-729; 1986.

[13] Wen-Hung Liao; Aggarwal, J.K. Cooperative matching paradigm for the analysis of stereo image sequences. *Internatinal Journal, Imaging System Technology*, *9*:192-200; 1998.

[14] Peterfreund, N. The velocity snake: deformable contour for tracking in spatio-velocity space. *Computer vision and Image Understanding, vol. 73, No. 3*, March, pp. 346-356;1998.

[15] Ding, A; Goshtasby, A. On the Canny edge detector. *Pattern Recognition, vol-34*, pp. 721-725;2001.

[16] Nesi, P.; Magnolfi, R. Tracking and synthesizing facial motions with dynamic contours. *Real – Time Imaging 2*, pp. 67-79;1996.