

M.TECH PROJECT REPORT

**SURFACE RECONSTRUCTION AND EVOLUTION
FROM MULTIPLE VIEWS**

By

M.SUDHAKAR
(EE10M04)

Adviser:

Dr. SOUMYA JANA
DEPARTMENT OF ELECTRICAL ENGINEERING

Co-Adviser:

Dr. AMIRTHAM RAJAGOPAL
DEPARTMENT OF CIVIL ENGINEERING



INDIAN INSTITUTE OF TECHNOLOGY HYDERABAD

Ordinance Factory Estate , Yeddumailaram, Medak-502205.

Declaration

I declare that this written submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.

M. Sudhakar

(Signature)

(M. Sudhakar)

EE10M04

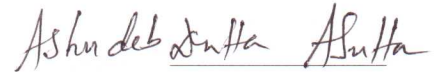
(Roll No.)

Approval Sheet

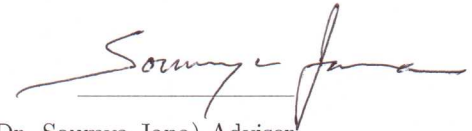
This Thesis entitled SURFACE RECONSTRUCTION AND EVOLUTION FROM MULTIPLE VIEWS by M. Sudhakar is approved for the degree of Master of Technology from IIT Hyderabad



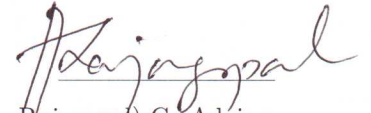
(Dr. Phanindra Varma Jampana) Examiner



(Dr. Ashdeb Dutta) Examiner



(Dr. Soumya Jana) Adviser
Dept. of Electrical Engineering
IITH



(Dr. Amirtham Rajagopal) Co-Adviser
Dept. of Civil Engineering
IITH

ACKNOWLEDGEMENTS

With profound sense of regard and gratitude I would like to first thank my advisor Dr. Sowmya Jana for having me as his student and providing me the opportunity to pursue my M.Tech research that led to this dissertation. It has been a privilege to have him as my advisor. His enthusiasm and love for research motivated me to pursue research as a career and have fun. He has been extremely helpful, provided me with great insight on various problems and immense encouragement through my stint at masters.

I would like to thank my project co-advisor Dr. Raja gopal for his immense guidance and support throughout the project. He helped me alot to understand the Finite Element Method concepts, and given access to his labs for some hands-on with Abaqus.

I would like to thank Dr. Asudeb datta, who advised me opt for IIT-Hyderabad at my admission. I also extend my thanks to Mr. Madhav Reddy, my senior who told me the importance of IIT at my admission. Also I would like thank IIT-Hyderabad for its support.

I would like to thank my team mates Mr. Kiran Kumar, Mr. Srikanth, Mr. Roopak to their support throughout my project. I also extend my thanks to my batchmates whom make time at IIT enjoyable.

Above all I would like to thank my parents and all my family members for constantly supporting me in building my career.

Abstract

Applications like 3D Telepresence necessitate faithful 3D surface reconstruction of the object and 3D data compression in both spatial and temporal domains. This makes us feel immersed in virtual environments there by making 3D Telepresence a powerful tool in many applications. Hence 3D surface reconstruction and 3D compression are two challenging problems which are addressed in this thesis.

3D surface reconstruction can be done by stereo based depth estimation and also by using silhouettes of the object from multiple views. The inherent problem in both these methods is to find the calibration parameters. We considered silhouette based 3D reconstruction with a novel auto calibration method. Traditional calibration methods are not suitable for real time applications since they either require external calibration object or requires a set of feature points which are visible to all the cameras. Our calibration method overcomes these difficulties there by making full body 3D reconstruction possible. Camera array with single axis motion around the object is considered and auto calibration is done using images taken from multiple views. Projection matrices found are used to project the silhouettes of the multiple 2D views into the space. Space carving technique is used to carve the shape of the 3D object required. The succeeding step after 3D reconstruction is to compress the 3D data in order make real time communication possible. 3D data will be huge and it has redundancy in both spatial and temporal domain. Compression in spatial domain is done by generating mesh of the 3D reconstructed data. Then 3D motion vector estimation should be done to estimate the deformed mesh points in temporal domain. But due to unavailability of proper data, motion estimation is done directly different set of images. This is done in two different ways. One is 3D motion vector estimation of 3D data using voxel matching. Other way is by using motion estimation of 2D images and motion vector estimation of their respective depth maps. Again this is done considering three different cases. First case is to find the motion vectors of 2D images alone and use them to compensate corresponding depth maps, second case is to find motion vectors of depth maps and use them to compensate image sequence and third case is to find motion vectors by using both image sequence and depth map sequence and compensate them by corresponding motion vectors. 3D reconstruction and motion estimation is implemented in Matlab and mesh generation is done using ABAQUS, COMSOL.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 5 |
| 1.1 | 3D Telepresence System | 5 |
| 1.1.1 | Acquisition of Multiple 2D views: | 6 |
| 1.1.2 | 3D Reconstruction: | 6 |
| 1.1.3 | 3D Compression: | 8 |
| 1.1.4 | 3D decompression and Rendering: | 8 |
| 2 | Problem Statement | 10 |
| 2.1 | 3D Reconstruction: | 10 |
| 2.2 | 3D Compression: | 10 |
| 3 | Literature Survey | 11 |
| 4 | Aquisition of Multiple 2D views | 15 |
| 4.1 | Basic Camera Model | 15 |
| 4.2 | Multi view Camera Model | 16 |
| 4.3 | Single Axis Motion | 17 |
| 4.4 | Auto Calibration | 18 |
| 4.4.1 | Case1: All points are visible to all cameras | 19 |
| 4.4.2 | Case 2: All Points are visible to more than one cameras | 20 |
| 5 | Silhouette based 3D Reconstruction | 24 |
| 5.1 | Object Reconstruction | 24 |
| 5.2 | Manifold Creation | 24 |
| 5.2.1 | Silhouette Generation | 25 |
| 5.2.2 | Multiple Cameras for Motion Capture | 26 |
| 5.2.3 | A Model of Object | 27 |
| 5.2.4 | Visual Hull Representation | 27 |
| 5.2.5 | Offline Reconstruction | 28 |
| 6 | 3D compression | 31 |
| 6.1 | Mesh generation | 31 |
| 6.2 | Motion Vectors | 32 |
| 6.2.1 | Compression in 3D Video | 33 |

| | | |
|----------|--|-----------|
| 6.2.2 | Motion Estimation in 3D | 35 |
| 6.2.3 | Motion Estimation for image and corresponding depth map | 36 |
| 7 | Results | 40 |
| 7.1 | Space Curving | 40 |
| 7.2 | Mesh Generation | 41 |
| 7.3 | Motion Estimation | 42 |
| 8 | Conclusion and Future Road Map | 45 |

Chapter 1

Introduction

The necessity for life like representation of the scene or person is significantly increasing in the field of communication and medical image processing. 3D Telepresence is one such application where we can feel the presence of person who is present at the other end(generally in a remote location). Creating life like representation is nothing but reconstructing the whole 3D model of a scene or person with the help 2D images. This problem is addressed by many people in many ways but it is still an open problem. Extracting back the 3D information from 2D images is not a straight forward problem. It is considered as an ill-posed problem. The following section gives the insight into various basic problems involved in a 3D telepresence system.

1.1 3D Telepresence System

Basic block diagram of a 3D telepresence system is shown in figure 1.1. A 3D Telepresence system should be able to reconstruct 3D object data captured from multiple images taken from different views and compress the data to transmit over communication channel. At the receiver end it should be able to decompress and represent the 3D data received maintaining color, luminous and texture consistency and render it to display.

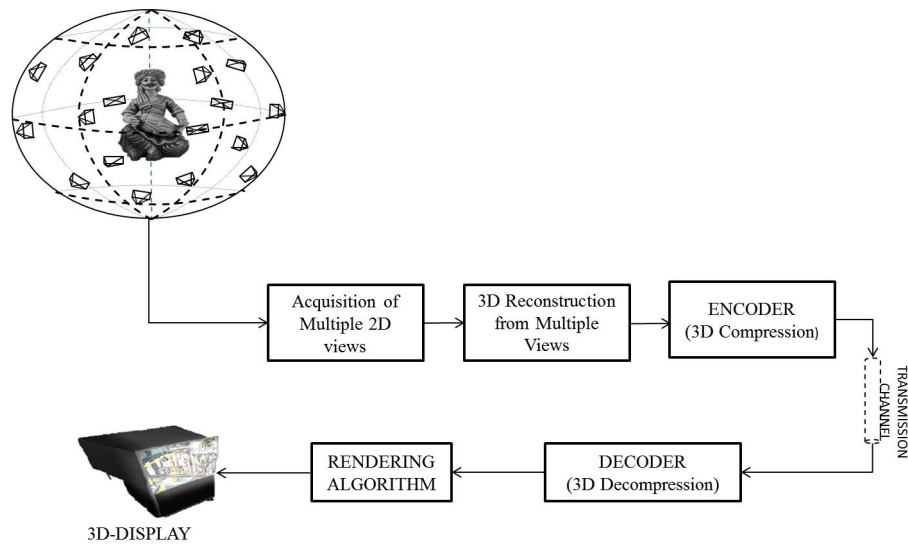


Figure 1.1: Block Diagram of the 3D Telepresence

The block by block description of 3D Telepresence system is as follows

1.1.1 Acquisition of Multiple 2D views:

A single camera cannot be able to represent the whole object, it can only capture single perspective(view) and it cannot give 3D information as well. Hence it is intuitive that a 3D Telepresence system need more than one cameras to represent whole object. As shown in the figure 1.1 the input to system is an array of 2D images captured in different views covering whole 360° space around the object(person). It is obvious that how many cameras are required? and where to place the cameras? are two basic questions that strikes to our mind. But there is no proper research done in this direction.

1.1.2 3D Reconstruction:

Multiple 2D images, captured by the network, play a central role in providing the depth related information that is difficult to perceive from individual 2D images. 3D reconstruction from single views is not a straight forward problem as we loose one dimension(depth) in the process of capturing image from a camera. This is considered as a ill-posed problem. The basic pinhole camera model can be seen in Figure 7.10

To lay down the mathematical framework for 3D object reconstruction, the working of a pin-hole camera needs to be completely understand and the transformation it affects on the 3D object when converting it into a 2D image. The set-up illustrated in Figure 1.3. shows how a 3D world coordinate is captured on to a 2D image plane.

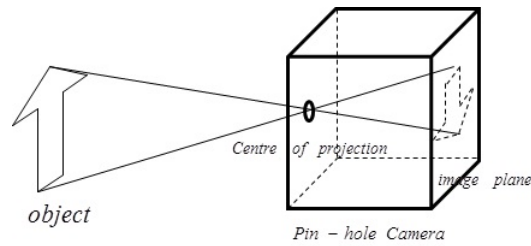


Figure 1.2: pinholecamera.

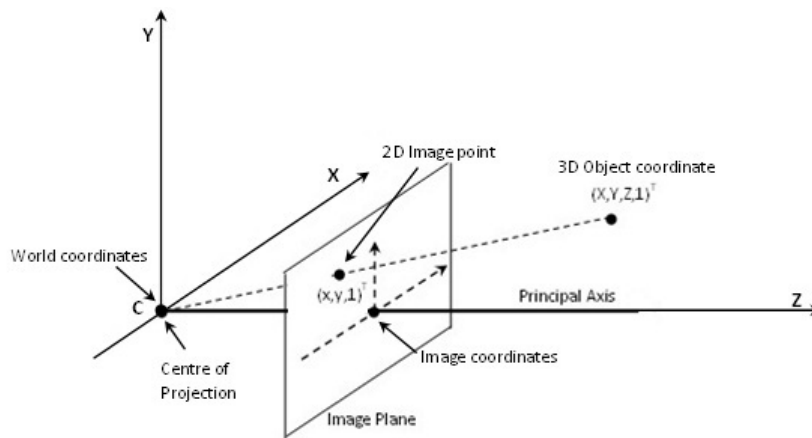


Figure 1.3: Camera and image plane placement in the world coordinate system.

The drop from three-dimensional space to a two-dimensional image is a projection in which one dimension is lost. So using a single camera cannot give this information back. Hence most existing data acquisition systems for 3D reconstruction use stereo cameras. The idea behind stereo vision is to mimic human biology by trying to recreate the behavior of human eyes. The eyes behave like two pin-hole cameras displaced by a certain distance. Each eye generates a slightly different perspective of the 3D scene and the brain then extracts position and depth information from these two 2D images. Stereo cameras with parallel axes model the human eye and are the ideal choice for 3D data acquisition applications. But stereo camera gives only depth information from only one perspective i.e., complete representation of the object is not possible. Multiple camera array is a possible solution to obtain complete information. This method of estimating the 3D depth information back from multiple cameras is known as 'Image Based 3D Reconstruction'. Image based 3D reconstruction constitutes both structure from stereo and structure from motion.

3D shape reconstruction using visual hulls generated based on Silhou-

ettes of images captured from multiple views is an interesting method which is suitable for multiple camera systems. This method is referred to as 'Silhouette based 3D reconstruction'. Methods like space carving and shadow carving are used in silhouette based 3D reconstruction.

1.1.3 3D Compression:

Generally for the best representation of the signal the samples should be continuous, but with continuous signals we face problem while encoding i.e we require infinite number of bits to encode which is not practical. So we digitize the signal so that we can encode and transmit. But when we digitize the signal we lose some information, therefore we cannot have best possible representation of the signal at the decoder or receiver. So, we should look for some better way of representing the signal so that no or less information is lost.

When an image is captured with the help of a camera data obtained will be as a set of intensity values representing a particular frame. 3D reconstructed data from multiple 2D views will generally be very large but there are limitations on bandwidth allocated to transmit, also the processing delay should be very low for real time transmission. We can never have a camera or a sensor that will capture points of interest i.e., the number of points that a camera can capture is nothing but fixed. So, there is high likelihood that we can have some redundant information in the views captured (Overlapping cameras is a trivial example). So how to get rid of this redundancy. One can pose a very interesting problem here i.e., Can we put a constraint on the number of views to be taken to have a minimal representation of the data?. what type sampling one should prefer to balance both quality and bandwidth requirements.

There many methods available to compress 2D images and 2D video, but 3D compression is not well explored. Methods like mesh generation, 3D motion vectors estimation can be used for 3D video.

In the transmitter (Encoder) 3D reconstruction of the object (person or scene) is done from 2D images captured in multiple views. The reconstructed 3D image is compressed and transmitted over the communication channel.

1.1.4 3D decompression and Rendering:

The decompression algorithm is run at the receiver based upon the compression algorithm used in the transmitter. 3D rendering is one of very important aspects of the 3D-Telepresence system. The 3D display should be able to provide viewer dependent view (motion parallax) and also it should be able to maintain color consistency, luminous consistency.

The most common solutions are stereoscopic displays with tracking. Stereoscopic displays [39] can emit only two distinguishable light beams from each pixel, this is the reason for the compromises: the viewer dependent view (that is, the 3D scene is correct only from a single point of view), thus the necessity of tracking [40] to create motion parallax, but still, this will provide a correct view only for the driver (who leads the session and wears the object that is tracked). Perspective for all other participants who are not looking at the same direction will be incorrect. Tracking systems also introduce a small amount of latency, which can be reduced, but still disturbing. All these limitations are responsible for the seasickness and headache after using these systems for longer sessions. There are many more 3D display technologies but failed to give elegant solution.

Chapter 2

Problem Statement

2.1 3D Reconstruction:

3D surface reconstruction of the object is one of the vital steps in achieving 3D Telepresence. So perfect 3D surface reconstruction of the object is considered as a major problem.

Solution: Multiple view camera array arranged on a circular frame around the object is considered (single axis motion). Silhouettes of the images captured from different views are generated. All the silhouettes are projected back to space and common intersection is obtained by using space carving approach. This common intersected volume gives the required 3D object representation.

Novelty: A new auto calibration method is proposed for finding the projection matrices of all the cameras which is suitable for real time applications. Silhouettes are projected into the space using these projection matrices.

2.2 3D Compression:

3D compression is also another vital process in achieving 3D Telepresence. Hence compressing the 3D reconstructed data for real time transmission is another major problem.

Solution: 3D Compression is considered in both spatial and temporal domain. In spatial domain mesh generation approach is used and in temporal domain 3D motion vectors are estimated and compensated.

Novelty: Estimation of 3D motion vectors.

Chapter 3

Literature Survey

The visual reconstruction problem addressed in this thesis can be broken down into two broad steps: modeling from images and rendering from models. To achieve maximum flexibility, both steps will rely on physically models. Then computer graphics rendering techniques will synthesize new views using these physically based dynamic scene models, even allowing arbitrary changes in any of the modeling parameters.

During the past decades many methods have been proposed to reconstruct the 3D shape of objects from images. Correspondence-based methods use pixel-wise correspondences among images as estimates of scene structure, which then allow new view synthesis by mapping this geometry into the new viewpoint [6]. The logical extension of correspondence-based methods is to extract full 3D structure from the scene. This approach reworks the correspondence problem in a global framework, allowing the simultaneous use of more information to resolve ambiguity and to increase precision. Another approach is apparent contours to compute the visual hull[2]. Space carving was proposed in[2][1] where photo-consistency is used to carve voxels. It has been shown, however, that these methods can only extract the line hull of the scene. For example, even perfect silhouettes of a coffee cup would lead to a model with a closed top, since no silhouette could distinguish the inside of the cup. So, space carving have the disadvantage that we cannot capture the concavity of the object.

The shadow curving is the refinement to the space curving[3].The shadow curving is the more time consuming problem because of the number of iterations needed for capture concavity.Even though we come across this process, there is no guarantee for capture the concavity in a better fashion. The problem of curve evolution driven by a PDE has been recently studied both from the theoretical standpoint and from the viewpoint of implementation with the development of level set methods that can efficiently and

robustly solve those PDE's. A nice recent exposition of the level set methods and of many of their applications can be found in [27]. By using the level set method we can obtain Euler-Lagrange equations of the geometry function, thereby obtaining a set of necessary conditions. In effect a set of partial differential equations, which we will solve as a time evolution problem by a level set method. And use the mesh mechanism [7] for representing the surface with minimum data. For compression we approach the motion estimation criteria.

The Baker and group in HP Labs designed an interactive telepresence system called HP[23] Coliseum see Fig. 3.1. In this framework they used five cameras fixed on a frame covering the field of view of the person of interest.



Figure 3.1: HP Coliseum in HP Labs

Coliseum builds 3D representation of object or person of interest Based on Image based Visual Hull (IBVH) technology of MIT [15]. For calibrating this set up they used a cube whose face have squares with different colors as a calibration object.

In Teleimmersion Lab, University of California, Berkeley Bajcsy and group have proposed interesting framework [22], [16], [17], [19], [18] see for 3D object reconstruction and 3D telepresence.

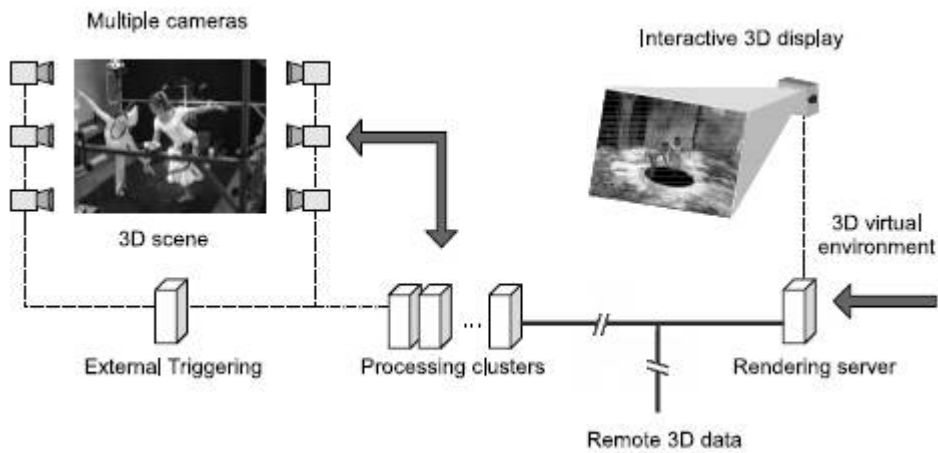


Figure 3.2: 3D Telepresence Framework in Teleimmersion Lab, UC Berkeley

They used 48 dragon fly cameras to capture the 3D object. These 48 cameras are divided into 12 clusters each containing 3 gray scale and one color camera which is shown Fig. 3.3. The Arrangement is shown in the Fig. 3.4. Three gray scale cameras in each cluster are used to find the depth maps and one color camera is used for rendering texture of the object.



Figure 3.3: Camera Cluster in Teleimmersion Lab, UC Berkeley

They proposed different methods for multiple camera array calibration one using virtual mesh (printed pattern on a blue infrared camera which is shown in Fig. ??) as a calibration object and other using two led bar as a virtual calibration object. Also, they used concept of vision graphs to localize all the cameras in the network with respect to a reference camera.

Camera calibration is a very important step in the process of 3D object reconstruction. Out of all different camera calibration methods Tsai's [21] algorithm is the most efficient algorithm for finding camera parameters but the complexity is high. Zhang [20] improved Tsai's algorithm in order to make implementation very easier. For calibrating single camera these methods are best suited. But calibration of a Multi camera array especially in

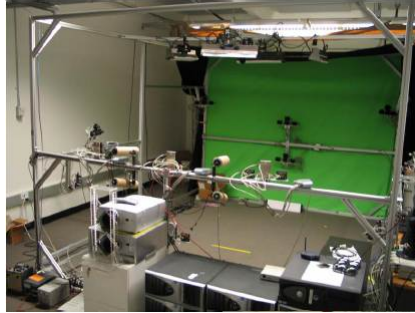


Figure 3.4: Camera Cluster arrangement in Teleimmersion Lab, UC Berkeley

real time or commercial applications these are suited because of using external calibration object and human intervention. However [21],[20] techniques are used in [23],[22],[16],[17],[19],[18] looked very cumbersome and are not practical as the calibration done in various steps and also there is human intervention. Hence extracting camera parameters for the scene captured is considered as very challenging problem which is known as “Auto” or “Self” calibration. Many researchers [31], [24], [28], [29],[30],[32],[33] tried to solve the auto calibration problem. Out of them Kanade’s[33] looks more generic. We observed that there is uniqueness problem in Kanade’s[33] algorithm, which we will address in chapter3.

For the representation of 3D reconstructed frame take 5-10MB of data for normal quality. So, the compression can be achieved in two ways i.e., spatial and temporal. The spatial compression can be achieved by representing the surface by using the interpolative basis, which is nothing but FEM [4]. The temporal redundancy can be achieved by using the block matching algorithm, the different types of block matching algorithms are discussed in [42]. But due to lack of 3D motion data we done the motion estimation for stereo sequence. The stereo reconstruction can be considered as 2.5D where we represent the one side of object by applying its depth map to its image. For the 3D depth map motion estimation and mechanism for data compression is discussed in [41].

Chapter 4

Aquisition of Multiple 2D views

4.1 Basic Camera Model

It is important to understand the mathematics behind how an image is formed by using a camera. Camera projects a 3D point on to the 2D image plane from the first principles of optics. This is known as projective transformation that defines how real-world objects are projected on the image plane. Figure 4.1 describes basic camera model that projects a 3D point in world coordinate system to a 2D point in camera coordinate system.

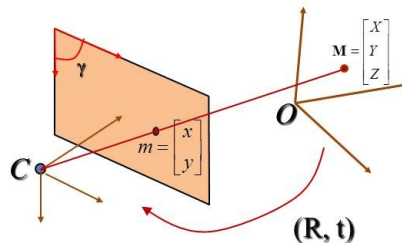


Figure 4.1: Basic camera model that defines projective transformation

Projective transformation is defined by camera intrinsic and extrinsic parameters. The following equation gives the basic structure projective geometry and the calibration parameters are useful in finding the 3D world coordinates.

$$s \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & | & t_1 \\ r_{21} & r_{22} & r_{23} & | & t_2 \\ r_{31} & r_{32} & r_{33} & | & t_3 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (4.1)$$

$$s \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathbf{K} [R \mid t] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (4.2)$$

$$s\tilde{m} = \mathbf{K} [R \mid t] \tilde{M} \quad (4.3)$$

where the 3×3 matrix K is dictated by the intrinsic parameters of the camera, and the 3×4 matrix $[R|t]$ by external parameters.

where (u_0, v_0) denotes the image coordinate of the point where the principal axis meets the image plane, f_x and f_y are focal lengths along image coordinate axes, and γ is a skewness index. If an image from camera is scaled by some factor, all of these parameters should be scaled (multiplied/divided, respectively) by the same factor.

Further, $[R \mid t]$, where the 3×3 matrix R is unitary, and indicates 3D rotation operation, where 3×1 vector t collects three translation parameters along the three world coordinate axes.

4.2 Multi view Camera Model

We have seen that single camera is not sufficient for extracting 3D information more over it can give information from single perspective. Hence we need multiple cameras to capture the object from different views so the full 3D reconstruction is achieved. Figure 4.2 shows the model of the multiple view camera array to capture the object from different views so full body 3D reconstruction is possible.

Mathematical modelling of multiple view camera array is not so straight as single camera. Han and Kanade[33](called Factorization Algorithm) proposed a simple model based on the assumption that all object points are visible from all the cameras which is practically not possible. Mathematical model for multicamera model based on factorization algorithm is given by

$$W_s = \begin{bmatrix} s_{11} \begin{bmatrix} x_{11} \\ y_{11} \end{bmatrix} & \dots & s_{1N} \begin{bmatrix} x_{1N} \\ y_{1N} \end{bmatrix} \\ \vdots & \ddots & \vdots \\ s_{M1} \begin{bmatrix} x_{M1} \\ y_{M1} \end{bmatrix} & \dots & s_{MN} \begin{bmatrix} x_{MN} \\ y_{MN} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_M \end{bmatrix} [X_1 \dots X_N]$$

$$\Rightarrow W_s = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_M \end{bmatrix} [X_1 \dots X_N] \quad (4.4)$$

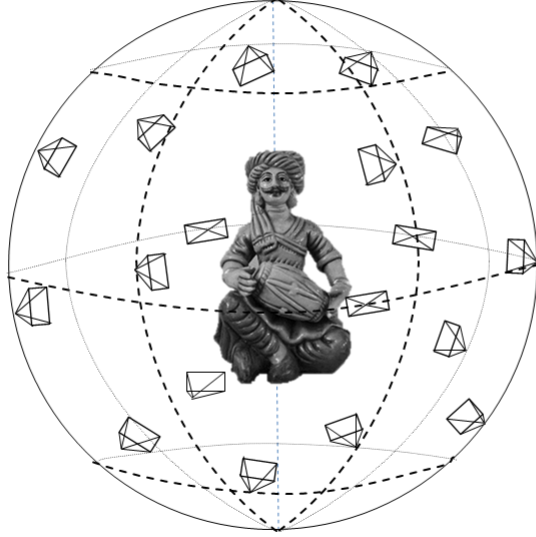


Figure 4.2: Multi view camera arrangement

This modeling is based on the assumption that M cameras sees N object points. P'_i s $\{i = 1 \dots M\}$ gives the projective transformation of each camera.

4.3 Single Axis Motion

Single axis motion is a special case of multiple view camera array arrangement. The cameras are placed on a circular frame around the 3D object such that principal axis of the cameras intersects the axis of the object coordinate system. This arrangement can be viewed as having a single camera on the circular frame and rotating the object about a single axis or vice versa. This arrangement can be seen in the figure 4.3

Before specializing to single axis rotation, consider first the general case of reconstruction from multiple pinhole cameras viewing a 3D scene. 3D points X in the scene are represented as homogeneous 4-vector $[X, Y, Z, 1]^T$, while their 2D projections x are represented as homogeneous 3-vectors $[x, y, 1]^T$. The action of each camera is represented by a 3 x 4 projection matrix P which is given by

$$s_{ij}x_{ij} = P_i X_j \quad (4.5)$$

The M camera matrices are indicated by P_i , $\{i = 1, \dots, M\}$ while the N

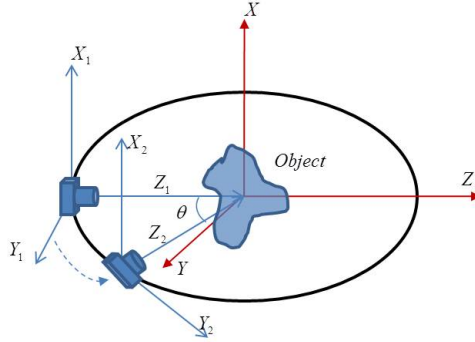


Figure 4.3: Single axis motion camera arrangement

3D points are $X_j, \{j = 1, \dots, N\}$. In this case M different cameras view a scene, there is no relationship between the P_i . Therefore $11M$ parameters are required to specify all the cameras. If we have identical internal parameters number of parameters required is reduced from $11M$ to $6M + 5$. If we have the camera in the single axis the number of parameters required is reduced from $6M + 5$ to $M + 8$.

If first camera centre is at position t on the X axis. Thus the first camera may be written

$$P_0 = H[I|t] \quad (4.6)$$

A rotation of the camera by θ yielding the camera $P_\theta = H[R_X(\theta)|t]$. In detail, with h_i the columns of H :

$$P_\theta = \begin{pmatrix} h_1 & h_2 & h_3 \end{pmatrix} \cdot \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & t \end{array} \right).$$

4.4 Auto Calibration

For 3D reconstruction using images from multiple views, simultaneous camera calibration and localization of multiple camera array is the most rudimentary and most significant step in order to extract the 3D attributes from the 2D images. As mentioned in the literature survey calibration using external calibration object is not preferred for multiple camera array, instead calibration of camera array from the scene itself is preferred. This is known as Auto calibration or self calibration. For single axis motion fixing the angle of rotation of principal axis with respect to objects axis provides more constraints in finding the camera parameters uniquely from the scene.

For single axis motion obtaining camera parameters with fixed angle of rotation can be made easy if we can get atleast the parameters of the reference camera which can be done by Zhang's[20] algorithm. But this has limitations for using this for real time and for unkown angle of rotation. Hence auto calibration is preferred. Here we use the factorization algorithm[33] as the reference and extend this by overcoming its limitations(all points should be visible to all cameras). The detailed explanation of the algorithm is given as follows

Let x be the measurement matrix of M cameras stacked that captures 3D points X which are visible from more than one cameras.

If we assume that the distance between the object center and the camera is large, then the scaling factor is independent of the position of the 3D object point and the camera projection can be modeled as:

$$s_i x_{ij} = P_i X_j$$

4.4.1 Case1: All points are visible to all cameras

Now, if all the N tracked feature points are visible from all the M cameras in the network, the global camera projection can be modeled as:

$$\begin{bmatrix} x_{11} & \dots & x_{1N} \\ y_{11} & \dots & y_{1N} \\ 1 & \dots & 1 \\ \vdots & & \vdots \\ x_{M1} & \dots & x_{MN} \\ y_{M1} & \dots & y_{MN} \\ 1 & \dots & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{s_1} K_1 [R_1 | t_1] \\ \frac{1}{s_2} K_2 [R_2 | t_2] \\ \vdots \\ \frac{1}{s_M} K_M [R_M | t_M] \end{bmatrix} \begin{bmatrix} X_1 & \dots & X_N \\ Y_1 & \dots & Y_N \\ Z_1 & \dots & Z_N \\ 1 & \dots & 1 \end{bmatrix} \quad (4.7)$$

$$x = PX \quad (4.8)$$

Problem statement: To choose $\{P, X\}$ such that $\|x - PX\|_F$ is minimized subject to the constraint that $\text{rank}(P) \leq 4$.

Solution: Choose P, X such that $\|x - \hat{P}\hat{X}\|_F$ is minimized subject to the constraint that $\text{rank}(P) \leq 4$.

where \hat{P}, \hat{X} are found by singular value decomposition of x and subsequently picks the best rank-4 estimate to obtain the solution. In this preliminary case, Han and Kanade's algorithm can be implemented and the global projection matrix can be obtained by rank-4 decomposition. The brief description of algorithm is as follows

$$\text{From SVD, } x_{3M \times N} = U_{3M \times 3M} \Sigma_{3M \times 3M} V_{3M \times N}^T$$

where the singular values in Σ are arranged in descending order. To obtain the rank-4 decomposition estimate, we write

$$\hat{\Sigma} = \begin{bmatrix} \Sigma_{11} & 0 & 0 & 0 \\ 0 & \Sigma_{22} & 0 & 0 \\ 0 & 0 & \Sigma_{33} & 0 \\ 0 & 0 & 0 & \Sigma_{44} \end{bmatrix}$$

where,

$$\Sigma = \begin{bmatrix} \Sigma_{11} & & & & \\ & \Sigma_{22} & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \Sigma_{3M,3M} \end{bmatrix}$$

If $U = [\underline{u}_1, \underline{u}_2, \dots, \underline{u}_{3M}]$, then $\hat{U} = [\underline{u}_1, \underline{u}_2, \underline{u}_3, \underline{u}_4]$. Similarly, if $V = [\underline{v}_1, \underline{v}_2, \dots, \underline{v}_{3M}]$, then $\hat{V} = [\underline{v}_1, \underline{v}_2, \underline{v}_3, \underline{v}_4]$. Now, solving for $x = U\Sigma V^T = PX$,

we write, $\hat{P} = \hat{U}\hat{\Sigma}H$ and $\hat{X} = H^{-1}\hat{V}^T$ for every invertible H

We now have $\hat{x} = \hat{P}\hat{X}$, thus for every choice of $(P, X) = (\hat{P}, \hat{X})$, we evaluate the frobenius norm to determine how much error the rank-4 decomposition would entail,

$$\begin{aligned} \|x - \hat{P}\hat{X}\|_F &= \|U\Sigma V^T - \hat{U}\hat{\Sigma}\hat{V}^T\|_F \\ &= \|\hat{U}^c\hat{\Sigma}^c\hat{V}^{cT}\|_F \\ &= \sum_{k=5}^{3M} \Sigma_{kk} \end{aligned}$$

where, $U = [\hat{U}|\hat{U}^c]$ and $V = [\hat{V}|\hat{V}^c]$. Thus, the task is now to choose P, X such that $\|x - \hat{P}\hat{X}\|_F$ is minimized subject to the constraint that $\text{rank}(P) \leq 4$.

4.4.2 Case 2: All Points are visible to more than one cameras

In this case all points may not be visible to all cameras. Let Θ be the visibility matrix which defines what features points are visible to what camera. Now the observation matrix cannot be modeled as equation (4.1), this is because x will be having holes if any particular camera cannot see any point in X

Problem Statement: To choose $\{P, X\}$ such that $\|\Theta \odot (x - PX)\|_F$ is minimized subject to the constraint that $\text{rank}(P) \leq 4$.

Solution: Choose P, X such that $\|\Theta \odot (x - \hat{P}\hat{X})\|_F$ is minimized subject to the constraint that $\text{rank}(P) \leq 4$.

where \hat{P}, \hat{X} are estimated by singular value decomposition (rank 4 decomposition) of x . The detailed algorithm to compute \hat{P}, \hat{X} is as follows

Initialization step: we break the M cameras into q clusters, such that

every camera in the j^{th} cluster can see N_j tracked feature points. We now apply Han and Kanade's method to cluster j to obtain the initial estimates of \hat{P}_j and \hat{X}_j .

$$\text{From SVD, } x_{3M_j \times N_j} = U_{3M_j \times 3M_j} \Sigma_{3M_j \times 3M_j} V^T_{3M_j \times N_j}$$

where the singular values in Σ are arranged in descending order. For want of simplicity, we shall drop the index j referring to the j^{th} cluster, keeping in mind that this same procedure is adapted for every cluster. To obtain the rank-4 decomposition estimate, we write

$$\hat{\Sigma} = \begin{bmatrix} \Sigma_{11} & 0 & 0 & 0 \\ 0 & \Sigma_{22} & 0 & 0 \\ 0 & 0 & \Sigma_{33} & 0 \\ 0 & 0 & 0 & \Sigma_{44} \end{bmatrix}$$

where,

$$\Sigma = \begin{bmatrix} \Sigma_{11} & & & & \\ & \Sigma_{22} & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \Sigma_{3M,3M} \end{bmatrix}$$

If $U = [\underline{u}_1, \underline{u}_2, \dots, \underline{u}_{3M}]$, then $\hat{U} = [\underline{u}_1, \underline{u}_2, \underline{u}_3, \underline{u}_4]$. Similarly, if $V = [\underline{v}_1, \underline{v}_2, \dots, \underline{v}_{3M}]$, then $\hat{V} = [\underline{v}_1, \underline{v}_2, \underline{v}_3, \underline{v}_4]$. Now, solving for $x = U\Sigma V^T = PX$,

$$\text{we write, } \hat{P} = \hat{U}\hat{\Sigma}H \quad \text{and} \quad \hat{X} = H^{-1}\hat{V}^T \quad \text{for every invertible } H$$

We now have $\hat{x} = \hat{P}\hat{X}$, thus for every choice of $(P, X) = (\hat{P}, \hat{X})$, we evaluate the frobenius norm to determine how much error the rank-4 decomposition would entail,

$$\begin{aligned} \|x - \hat{P}\hat{X}\|_F &= \|U\Sigma V^T - \hat{U}\hat{\Sigma}\hat{V}^T\|_F \\ &= \|\hat{U}^c \hat{\Sigma}^c \hat{V}^{cT}\|_F \\ &= \sum_{k=5}^{3M} \Sigma_{kk} \end{aligned}$$

where, $U = [\hat{U}|\hat{U}^c]$ and $V = [\hat{V}|\hat{V}^c]$. Thus, the task is now to choose P, X such that $\|x - \hat{P}\hat{X}\|_F$ is minimized subject to the constraint that $\text{rank}(P) \leq 4$. At the end of the first step, we have \hat{P}_j and \hat{X}_j , for every cluster $j = 1, 2, \dots, q$. Our goal is to populate the global image point matrix $x_{3M \times N}$ by using the directly observed image coordinates of the points that are visible and by estimating the coordinates of those points which are otherwise invisible to a given camera. In the second step, estimating invisible points is done by calculating point correspondences using fundamental matrices, trifocal tensors or multifocal tensors as per the situation. Since the first step gives

us \hat{P} from which the corresponding camera matrix P can be obtained and the knowledge of the camera matrices is used to generate the fundamental matrix or the trifocal or multifocal tensors required to generate the point correspondences.

Visibility Matrix Θ : While the point correspondences are being computed, a mask $\Theta_{3M \times N}$ is generated whose entries in the i^{th} column is 1 if the tracked feature point is visible to the i^{th} camera, else it is set to 0. This mask Θ is important as it helps us calculate the error metrics that will be used to verify whether or not the iterative procedure is indeed converging and also to help decide when to stop the iteration.

In the third step, the global image point matrix $x_{3M \times N}$ is created using both the directly observable points and the global camera projection matrix is created by stacking individual camera matrices in the order corresponding to that of the estimated invisible points. global image matrix. Once this global image point matrix is obtained, in step 4, step 1 is again computed using the new global image point matrix.

Error CriteriaThe error criterion is computed by applying a mask to the regular error minimization constraints that were used to determine \hat{P} and \hat{X} . Thus, we evaluate the Frobenius norm to determine how much error the invisible point estimation carries,

$$\begin{aligned} \|\Theta \cdot (x - \hat{P}\hat{X})\|_F &= \|\Theta \cdot (U\Sigma V^T - \hat{U}\hat{\Sigma}\hat{V}^T)\|_F \\ &= \|\Theta \cdot (\hat{U}^c\hat{\Sigma}^c\hat{V}^{cT})\|_F \end{aligned}$$

Once the Frobenius norm is calculated, the error metric for successive iterations is compared and if it converging, after suitable number of iterations, the process is halted. If the error metric is not within bounds, the next iteration goes back to step 2 and estimates the invisible points by using the camera matrices obtained from the fresh \hat{P} and \hat{X} evaluated in step 4 of the previous iteration. With these new estimates, the global image coordinate matrix and the global camera matrix are evaluated and step-1 is evaluated to obtain the fresh set of \hat{P} and \hat{X} that will be used in the next iteration. The error metric is calculated and is within bounds or converging, the process is halted, else the iteration goes back to step-2 and runs all over again.

At the end of the last iteration when the error metric is finally found to be converging and well within a certain threshold, the freshly updated \hat{P} and \hat{X} matrices represent the camera matrices and the world coordinates of the N tracked feature points. Thus, employing this iterative procedure will enable us to not only perform self-calibration and obtain the camera parameters of all the M cameras in the network but also obtain the shape information of the 3D object as the N tracked feature points will help determine the shape of the object.

Chapter 5

Silhouette based 3D Reconstruction

5.1 Object Reconstruction

There has been a rich research on the methods of 3D reconstruction from multiple images in the past few years. Research in 3D shape reconstruction from multiple view images has conventionally been applied in robot vision and machine vision systems, in which the reconstructed 3D shape is used for recognizing the real scene structure and object shape. For those kinds of applications, the 3D shape itself is the target of the reconstruction. The literature relates methods from 2D images to our approach most of 3D reconstruction. Our method begins with these same fundamentals, but the implementation is real time. Our survey has a particular focus on real-time and free-space based methods. In addition, we review works on the Shape from Points problem, where one would like to infer dense 3D geometry only from a sparse 3D point-wise reconstruction without occlusion information or color and texture from images. Much of the Shape from Points literature explicitly or implicitly makes use of the same discretization of space, and therefore some theoretical results from this literature are notably of interest. A potential application for real-time reconstruction that we have experimented with is improving visualization for remote-controlled or tele presence.

5.2 Manifold Creation

A different class of 3D model-building methods is shape from motion (SFM). In these methods, usually a single camera is moved through a static environment to create a model of the scene. Because the camera is moving relatively slowly, the inter-frame motion is usually quite small, so feature-tracking algorithms can be used to track features through long

camera motions. These features can then be used to estimate shape, for example using the Factorization method or non-linear optimization. These approaches usually work only with sparse features, and therefore face the challenge of finding surfaces to connect the feature points.

SFM approaches use feature tracking through video sequences captured by camera motion through a fixed scene. To apply these approaches to dynamic scenes, a collection of cameras could be treated as positions of the moving camera through the fixed scene of a single image from all cameras. The feature tracking algorithms usually require very small feature motion from frame to frame, however, so the real camera spacing would have to be small enough to simulate inter-frame camera motion which could easily amount to (tens of) thousands of images. This approach combines strengths both of explicit 3D modeling and of image-based rendering. Multi-camera video sequences of dynamic events are used to automatically estimate global scene structure corresponding to each frame of video, resulting in a 3D triangle mesh. The real images can be mapped back to this structure using texture mapping to add realistic visual appearance. The technique is repeated over time to capture complex dynamic events.

This shape and appearance digitization algorithm can be decomposed into two steps: first, recover 3D shape, and then estimate scene appearance. Shape digitization itself is estimation of visible surfaces in each video image and the merging of these estimates via volumetric integration into a single global 3D model. Decomposing shape and appearance helps avoid problems found in voxel coloring, in which mistakes early in the process hinder both shape and appearance modeling. Decomposing shape digitization allows local information to propagate to the global structure in a hierarchical framework, increasing the scope of the modeling process and decreasing noise as the hierarchy is traversed.

5.2.1 Silhouette Generation

In this section we discuss how we can get the silhouette of the subject from each of the images. This is also called foreground object extraction. Since we assume that the background images are available, a straight forward technique would be to subtract the background image from the background image and threshold to get the silhouette. This might work under ideal conditions. However, in the presence of noise, lighting changes and shadows, lacking of background information we need more sophisticated methods. The subtraction leaves only non-stationary or new objects, which include the objects entire silhouette region. The technique has been used for years in many vision systems as a preprocessing step for object detection

and tracking. The results of the simple background subtraction algorithms are fairly good; in addition, many of them run in real-time. However, many of these algorithms are susceptible to both global and local illumination changes such as shadows and highlights. These cause the consequent processes, e.g. tracking, recognition, etc., to fail. The accuracy and efficiency of the detection are clearly very crucial to those tasks. One of the fundamental abilities of human vision is color constancy. Humans tend to assign a constant color to an object even under changing illumination over time or space. The perceived color of a point in a scene depends on many factors including physical properties of the point on the surface of the object. Important physical properties of the surface in color vision are surface spectral reflectance properties, which are invariant to changes of illumination, scene composition or geometry. On Lambertian surfaces, the perceived color is the product of illumination and surface spectral reflectance. The object can be model in colour space as $E_i = R_i, G_i, B_i$. The foreground object will extract by using the difference in the R,G,B value of the back ground and the object surface. In other words we select the background which has different R,G,B values compared to the object.



Figure 5.1: Object image and its respective Silhouette image

5.2.2 Multiple Cameras for Motion Capture

Human motion capture is the activity of acquisition, processing the data and expressing motion in mathematical terms. The motion capture task can be classified into a number of systematically different groups, initialization, tracking, pose estimation and gesture recognition. Motion capture can be obtained by one of three technologies: optical, magnetic and electro-mechanical all of which involve markers or devices attached to the subject. A method for markerless motion capture that does not use such markers but uses multiple images which is obtained by cameras placed around object to estimate the pose of the subject. There exist a number of algorithms to estimate the pose from images captured from a one camera, a task that is extremely difficult and ambiguous. Segmentation of the image into different, possibly self-occluding, body parts and tracking them is an inherently difficult problem.

The basic steps involved in any markerless motion capture algorithm are:

1. Extract the model of object from multiple images
2. Initial pose estimation of object
3. Track the pose in subsequent frames

5.2.3 A Model of Object

Since we know object form, the use of object models can greatly simplify the object motion estimation problem. It also makes the estimation more accurate and robust. Representation of object can be done using different models based on the application requirements for which we intend to use our model. There can be voxel models or other articulated models. The most common representation of objects is using segments and joints, where we assume that the body parts are rigid. The rigid parts can be related in a tree structure, where the nodes are taken as joints. Estimating the each segment position with respect to its neighbor gives us the required posture. We can have various models based on the level of articulation and representation of segments. The segments can be represented using quadrics, line segments, ellipsoids, etc. The total number of joints and segments are decided by the actions we would like our model to perform.

5.2.4 Visual Hull Representation

For multiple view 3D reconstruction, we need to find the volume which corresponds to the silhouettes in all the images. This volume is called the visual hull of the object with respect to the views. Visual hull is not an exact reconstruction but the closest we can get to with the available information from the images. It is used extensively in virtual reality applications and object body pose estimation. For pose estimation, a 3D object model is usually fit to the visual hull to estimate pose parameters. One approach to visual hull construction is calculating a polygonal surface by intersecting silhouette cones, computed easily by back projection polygonal approximations of silhouette contours of the object. Volumetric reconstruction represents an alternative approach to visual hull construction. In this case, the 3D space is divided into elementary cubic elements (i.e., voxels) and tests are performed to label each voxel as being inside, outside or on the boundary of the visual hull.

Focus of our work is on multiple videos of a single subject taken in a controlled environment. The experiment set up is as follows: The placement of cameras is all around the subject and pointing towards the center of the capture space. This configuration is essential for estimation in the bounding

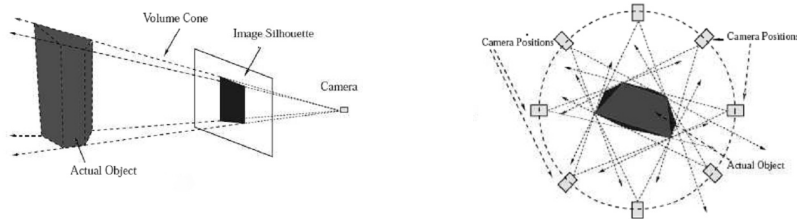


Figure 5.2: Silhouette Re-projection from views

volume space. We also assume that the background is static, which we can differentiate the subject and background easily. The cameras are assumed to be calibrated. The problem of estimating the voxel data for each time instant given the multiple view images. This is done using some existing algorithms. Correlation between successive frames can also be used here. Our reconstruction is aimed at applications of activity recognition and pose reconstruction for animation. So we are not interested in the features of our subject.

5.2.5 Offline Reconstruction

While Structure from Motion addresses the problem of reconstructing sparse 3D scene information from 2D images, two-view and multi-view stereo addresses how to obtain dense 3D from images. In stereo, the pose and calibration of each image camera is usually assumed known and given as input. Two-view stereo considers reconstruction from just two calibrated views, one of which is designated as the reference view. The 3D representation used is a discrete per pixel mapping from image space to camera relative scene depth w.r.t. the reference image. This representation is called a depth map. The two-view problem boils down to finding the optimal depth map via dense image matching and triangulation. In this way, it is similar to feature based Structure from Motion but with a match generated for every pixel rather than at sparsely detected feature points. Depth maps are often referred to as a 2.5D representation because they do not encode connectivity between depth estimates. For example, if simply assuming 8 connectivity between neighboring depth pixels to generate a back projected surface, foreground objects will incorrectly join with the background of the scene. There are several algorithms and formulations, varying primarily in terms of the texture-based matching cost, spatial regularization, and depth map optimization scheme. This section instead concentrates on multi-view reconstruction from > 2 images. The multi view stereo literature is vast.

While an exhaustive review would be excessively lengthy and peripheral to this thesis, it is important to situate our work in relation to this body of research. Both our work and stereo have a similar goal: to recon-

struct a 3D model from a set of input images. While a multitude of methods with differing properties exist, a common trend is that they compete to find a reconstruction that is as accurate and complete as possible. This emphasis can be seen in the very popular multi-view benchmark dataset and survey, which collects impressive results from the state of the art. As a result of this focus on quality, most multi-view methods are designed to operate on small sets of images (on the order of ten or a hundred), and they can take up to hours to process such datasets. Real-time reconstruction from video is rare. In contrast to the norm, this thesis work attempts to reconstruct more approximate models, but in real time. Commonly, stereo methods are cast as an optimization problem where a representation of the 3D scene or object to reconstruct is fit to the image data.

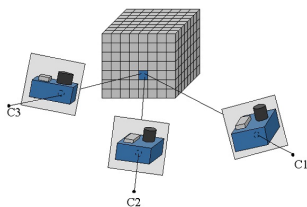


Figure 5.3: (a) Consistent voxel

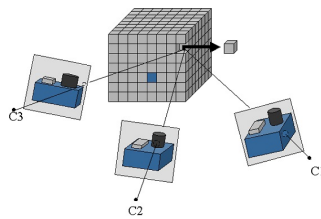


Figure 5.4: Non-consistent voxel

The approaches vary and are distinct from each other broadly in terms of the optimization framework, 3D parametrization, and cost function or functional to optimize. The objective function invariably contains some form of a texture-based photo-consistency matching cost. This measures how well the recovered 3D surface projections match between the input images based on scores derived from color or intensity differences, such as Normalized Cross Correlation (NCC) scores used in, or the Sum of Squared Differences (SSD). Photo-consistency is the primary reconstruction cue in stereo. Other common terms in the objective function relate to secondary reconstruction cues such as silhouette constraints (in the case of reconstruction of a single object segmented in image space), visibility (enforcing correctness when optimizing the photo-consistency with respect to occlusions in the images), and spatial regularization or smoothness priors. Optimization of the chosen objective can be performed using a number of standard techniques, which include iterative derivative-driven numerical optimization (e.g. gradient descent for surface deformation or level-set evolution or conjugate gradient for surface-patch refinement). The choice of optimization procedure influences properties of the method such as convergence, reconstruction quality, initialization, and speed. The applicability of a given optimization procedure also depends on both the form of the objective and the parametrization or representation of the reconstruction. to local minima. In the case of object reconstruction via surface evolution, Shape from Sil-

houettes(SFS) can provide an approximate initial surface by back-projecting the (segmented) objects silhouettes to form generalized cones in 3D space and then computing the volumetric intersection of all such cones. For scene reconstruction where silhouettes cannot directly apply, initialization is a limitation.

Chapter 6

3D compression

6.1 Mesh generation

This chapter introduces a complete framework for automatic adaptation of a 3D face model to a human face for visual communication applications like video conferencing or video telephony. First, facial features in a facial image are estimated. Then, the 3D face model is adapted using the estimated facial features. This framework is scalable with respect to complexity. Two complexity modes, a low complexity and a high complexity mode, are introduced. For the low complexity mode, only eye and mouth features are estimated and the low complexity face model Candide is adapted. For the high complexity mode, a more detailed face model is adapted, using eye and mouth features, eyebrow and nose features, and chin and cheek contours. Experimental results with natural videophone sequences show that with this framework automatic 3D face model adaptation with high accuracy is possible.

In the last few years, virtual humans and especially animated virtual faces (also called talking heads) have achieved more and more attention and are used in various applications. In modern computer games, virtual humans act as football players or Kung Fu fighters. In movies, highly realistic animated virtual humans are replacing real actors (e.g., in the science fiction movie *Final Fantasy*). On the Internet, animated virtual faces are acting as news announcers or sales agents. In visual communication applications, like video telephony or video conferencing, the real faces of the participants are represented by virtual face clones of themselves. If we take a closer look at the technology behind these animated faces, the underlying shape of a virtual face is often built from a 3D wireframe consisting of vertices and triangles. This wireframe is textured using textures from a real persons facial image. Synthetic facial expressions are generated by animating the 3D wireframe. Usually, the face is animated by movement of the wireframes vertices. In order to produce natural looking facial movements,

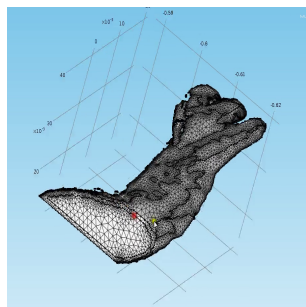


Figure 6.1: Meshing of Dinosaur Hand

an underlying animation structure (providing rules for animation) is needed, simulating the behavior of a real human face.

The creation of such an animated face requires generating a well-shaped and textured 3D wire-frame of a human face, as well as providing rules for animation of this specific 3D wireframe. There are different ways to create an animated face. One possibility is that an animated face is created manually by an experienced 3D modeler or animator. However, an automatic approach is less time consuming and is required for some applications. Dependent on the specific application and its requirements, different ways for the automatic creation of an animated face exist. For 3D modeling of the shape of the head or face, i.e., for generation of the 3D wire-frame, techniques that are common for the 3D modeling of objects in general could be used. With a 3D scanner, a laser beam is sent out and reflected by the objects surface. Range data from the object can be obtained and used for 3D modeling. Other approaches use range data from multi-view images (Niem, 1994) obtained by multiple cameras for 3D modeling. All these techniques allow a very accurate 3D modeling of an object, i.e., a human head or face.

6.2 Motion Vectors

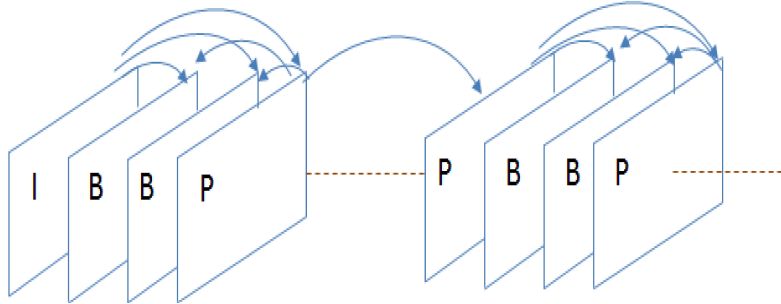
In video editing motion vectors are used to compress video by storing the changes to an image from one frame to the next. In a video sequences exhibit very close similarity, except for the fact that the objects or the parts of a frame in general may get somewhat displaced in position. This assumption is mostly valid except for the frames having significant change of contents.

The motion estimation block in a video codec computes the displacement between the current frame and a stored past frame that is used as the reference. We consider a pixel belonging to the current frame, in association with its neighborhood as the candidates block. The difference in position

between the candidates and its match in the reference frame is defined as the motion vector. The motion vector is stored in the place of candidate block with better prediction. The difference between predicted block and the candidate block called prediction error. While we are coding we will code the only prediction error instead of total candidate block. At reconstruction we will get back the candidate block from the reference frame (Intraframe) and the prediction error [8].

The video sequence consists of

- Intraframe coded pictures (I-pictures)
- Interframe predicted pictures (P-pictures)
- Bi-directionally predicted pictures (B-pictures)



P frame formed from 'I' . 'B' formed from 'I' , 'P'

6.2.1 Compression in 3D Video

The compression in 3D video can be done by incorporating motion estimation for 3D frames. The motion estimation of 3D video is gaining momentum because of the advancement in 3D technologies. Here we proposed an idea to find the motion vectors for 3D video. Here we have the 3D mesh model. Here we want to find the node displacement instead of block displacement. By incorporating this node displacement we make less computation, more compression. And we have to take care of the occluded points in the object.

The Encoder uses the 3D frames. The 3D frames are FEM modeled, which can be reconstructed from the nodes. We will find the motion estimation from the reference frame to candidate frame by estimating the motion of nodes. We have the Decoder running internally in Encoder. Because to calculate the motion vectors from the encoded 3D surface. The 3D surface which is formed by using the motion vector based on FEM models gives the better compression. For finding matching we will go for one of the existing matchings

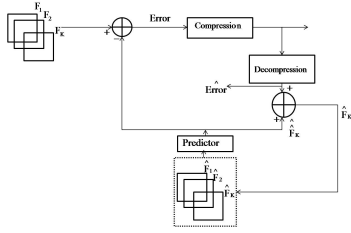


Figure 6.2: 3D encoder

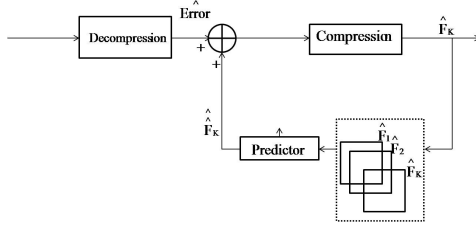


Figure 6.3: 3D Decoder

methods [8]. The reconstruction of 3D video in decoder is done by adding the motion vectors to the nodes in reference frame. The time varying mesh is a new multimedia representation, which is a sequence of 3D models that are composed of vertices, edges, and some attribute components such as colour. An extended block matching algorithm (EBMA) to reduce the temporal redundancy of the geometry information in the time-varying mesh by extending the idea of the 2-D block matching algorithm to 3-D space. In EBMA, a cubic block is used as a matching unit. Motion Compensation in the 3-D space is achieved efficiently by matching the mean normal vectors calculated from partial surfaces in cubic blocks. This 3D models are generated frame by frame. In the sequence of 3D models of time varying mesh, not only the geometry information but also the number of vertices changes; therefore, there is large amount of data. For example, each 3D models will take 5-10MB of memory to represent it in VRML file format[12].

However, there was significant difference between time varying mesh and 3D animation models, which are having same number of vertices and connectivity in each model. Therefore in 3D animation there is one-to-one mapping of vertices exist between successive models. Time-varying mesh is generated independently regardless of its neighbors. There are two categories of time-varying meshes on the analogy of 2D video coding techniques, intra and inter-frame coding. Intraframe coding is a compression method to reduce the spatial redundancy within a frame. From this point of view, 3D mesh compression technique so far have been intra-frame codings. On the other hand inter frame coding exploits the temporal correlation between consecutive frames. Most of the previous inter-frame compression methods have focused on 3D animation compression. The time varying mesh has more than 50000 vertices per frame, which is rather time consuming for vertex correspondences. So, it is better to go for 3D motion estimation by using surface normal of triangular mesh[12][13].

Due to lacking of 3D motion models now here we are doing the motion estimation for the data sets of Zhejiang University. The data set contains image and its respective depth map. We done the motion estimation in two ways. One is do the motion estimation in 3D itself by using the

block matching algorithm. And one is do the 2D motion estimation using image and depth map seperately.

6.2.2 Motion Estimation in 3D

We extend the Block matching algorithm of 2D to find the motion vectors in 3D video. A matching criterion quantifies the similarity between the current block and the candidate blocks. In 2Dvideo, a number of matching criteria have been used such as SSD,SAD, and MSD, etc [42]. These methods have an implicit constraint that the sizes of the two blocks are identical. 3D motion estimation uses a cubic block-splitting surface as its basic unit. Splitting is performed on the current frame that is to be encoded. The bounding box of the current frame F_m is calculated and then it is divided into cubic blocks along the and directions. The size of cubic block is $sXsXs$ (width X height X depth) .

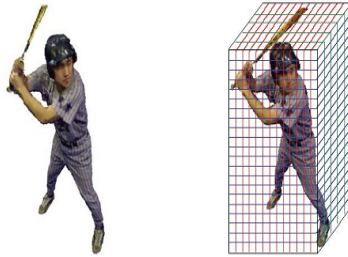


Figure 6.4: block splitting of 3D object [12]

The parameter is set 16 in our experiments. Thereby, a set of s^3 cubic blocks is obtained for a total of N_q cubic blocks. Here, we denote the n th block in the m th frame as B_m^n . Therefore, the frame can be represented as $F_m = B_m^c | 0 \leq c \leq N_{q-1}$. Each cubic block, , contains a partial surface of the current frame . Only blocks that contain surface of the 3-D model are used for MC. We determine the size of the cubic blocks by examining the energy of the residuals of the MC. Smaller cubic block sizes can produce better MC results, although smaller blocks increase the number of motion vectors, degrading the compression efficiency. After splitting process, B_m^c is compared with the candidate blocks B_{m-1}^l (for $0 \leq l \leq N_{CB}^n - 1$) in the decoded previous frame F_{m-1}^* , where N_{CB}^n denotes the number of the candidate blocks for B_m^c [12] . Note that the maximum value of N_{CB}^n depends on the search area because the search area become a center for B_{m-1}^l . Because the cubic blocks are defined based on the bounding box without considering the models shape, the partial surface in a cubic block is not necessarily at its center. It is desirable to have the partial surface at the center of the

cubic block to make the distribution of residuals biased toward zero [41].

$$\varepsilon(i, j, k; \underline{v}) = B(i, j, k) - \hat{B}((i, j, k) + \underline{v}) \quad (6.1)$$

$$v^*(i, j, k) = \arg \min_{\underline{v}} \|\varepsilon(i, j, k; \underline{v})\| \quad (6.2)$$

$$= \arg \min_{\underline{v}} \|B(i, j, k) - \hat{B}((i, j, k) + \underline{v})\| \quad (6.3)$$

Motion vector of 3D can be found by extending the 2D Block matching algorithm to the z-direction. In order to solve this problem, we adopt the 3D direction motion to find Z-adjustment motion estimation of the 3D equivalent of 2D with assigned depth maps. We adopt 3D motion to search for the depth map in Z-direction motion estimation provides information for 3D data compression. According to the Z-direction motion vectors, the system performs the motion compensation. Next, the 3D video sequence can be compressed easily according to the motion information.

6.2.3 Motion Estimation for image and corresponding depth map

The underlying supposition behind motion estimation is that the patterns corresponding to objects and background in a frame of video sequence move within the frame to form corresponding objects on the subsequent frame. The idea behind block matching is to divide the current frame into a matrix of macro blocks that are then compared with corresponding block and its adjacent neighbors in the previous frame to create a vector that stipulates the movement of a macro block from one location to another in the previous frame. This movement calculated for all the macro blocks comprising a frame, constitutes the motion estimated in the current frame. The search area for a good macro block match is constrained up to 'p' pixels on all four sides of the corresponding macro block in previous frame. This p is called as the search parameter. Larger motions require a larger p, and the larger the search parameter the more computationally expensive the process of motion estimation becomes.

Here we have used block matching algorithm for finding the motion vectors. We used the absolute difference as matching criteria for finding motion vectors of current frame. We initially divide the frames as square blocks of size w . Then you search for the match of each block of current frame in reference frame. After selecting a block in current frame you search for matching in reference frame search window. Optimum size of search window is " $2w*2w$ ".

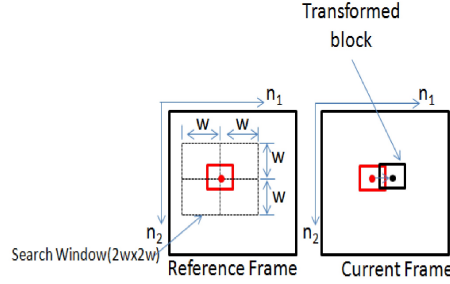


Figure 6.5: Motion estimation

For each block 'B' in current frame we will find the corresponding matching block ' \hat{B} ' in reference frame with Minimum Absolute Error criteria

$$\varepsilon(i, j; \underline{v}) = B(i, j) - \hat{B}((i, j) + \underline{v}) \quad (6.4)$$

$$v^*(i, j) = \arg \min_{\underline{v}} \|\varepsilon(i, j; \underline{v})\| \quad (6.5)$$

$$= \arg \min_{\underline{v}} \|B(i, j) - \hat{B}((i, j) + \underline{v})\| \quad (6.6)$$

'*' indicate the optimum. After finding the optimum motion vector we do the motion compensation by using reference frame and motion vector . And we obtain the error frame by doing the difference between motion compensated frame and the current frame.

$$\hat{EB}(i, j; \underline{v}) = B(i, j; v^*(i, j)) \quad (6.7)$$

$$\varepsilon^*(i, j) = \varepsilon(i, j, v^*(i, j)) \quad (6.8)$$

'EB' is the motion compensated block. At the receiver side we will reconstruct the current frame from the current frame, motion vectors and error frame.

Now we are doing the motion vectors differently for image and depth map sequence. Our main idea is to send a single motion vector either image or depth map sequence. At receiver we have to reconstruct the current frames of image and depth map by using this single motion vector. For this we done the different cases of motion estimation

- Find the motion vector by using image sequence

- Finding the motion vector using depth map sequence
- Find the motion vector using both image and depth map

Motion Vectors using image sequence

We obtain the motion vectors by using the images itself. Here we do not bother about the depth map images. We apply the same motion vectors for the image and depth map images. So, we obtain optimum motion compensation for image sequence only. In this case we require less data to encode image sequence error frames and more data for depth map error frames.

$B_1(i, j) \Leftrightarrow v_1^*(i, j)$ [For (i, j) , pick $\underline{v} = v_1^*(i, j)$]

Image Sequence

$$\begin{aligned} \hat{E}B_1 &= EB_1((i, j) + v_1^*(i, j)) \\ \varepsilon_1(i, j) &= \varepsilon_1^*(i, j; v_1^*(i, j)) \end{aligned}$$

Depth Map Sequence

$$\begin{aligned} \hat{E}B_2 &= B_2((i, j) + v_1^*(i, j)) \\ \varepsilon_2(i, j) &= \varepsilon_2(i, j; v_1^*(i, j)) \end{aligned}$$

Motion Vectors using depth map sequence

This is counter part to the above case. Here we obtain the motion vectors by using the depth map sequence. And we estimate the current frames using this motion vectors. In this case we require more data to encode error frames of image sequence and less data to encode the error frames of depth map sequence.

$B_2(i, j) \Leftrightarrow v_2^*(i, j)$ [For (i, j) , pick $\underline{v} = v_2^*(i, j)$]

Image Sequence

$$\begin{aligned} \hat{E}B_1 &= EB_1((i, j) + v_2^*(i, j)) \\ \varepsilon_1(i, j) &= \varepsilon_1(i, j; v_2^*(i, j)) \end{aligned}$$

Depth Map Sequence

$$\begin{aligned} \hat{E}B_2 &= B_2((i, j) + v_2^*(i, j)) \\ \varepsilon_2(i, j) &= \varepsilon_2^*(i, j; v_2^*(i, j)) \end{aligned}$$

Motion Vectors using both image and depth map sequence

This is optimum way to find motion vectors by using the both image and depth map sequence. And we estimate the current frames using this motion vectors. In this way we can achieve moderate requirement to encode error frames of both image and depth map sequences.

$$v^*(i, j) = \arg \min_{\underline{v}} (\lambda \|\varepsilon_1(i, j; \underline{v})\| + (1 - \lambda) \|\varepsilon_2(i, j; \underline{v})\|) \quad (6.9)$$

Image Sequence

$$\begin{aligned} \hat{E}B_1 &= EB_1((i, j) + v^*(i, j)) \\ \varepsilon_1(i, j) &= \varepsilon_1(i, j; v^*(i, j)) \end{aligned}$$

Depth Map Sequence

$$\begin{aligned} \hat{E}B_2 &= B_2((i, j) + v^*(i, j)) \\ \varepsilon_2(i, j) &= \varepsilon_2(i, j; v^*(i, j)) \end{aligned}$$

Here we have to find the λ , such that we can achieve the maximum compression,

$$\lambda = \frac{\sigma_2}{\sigma_1} = \frac{\textit{DynamicrangeofDepthmap}}{\textit{Dynamicrangeofimage}} \quad (6.10)$$

Chapter 7

Results

7.1 Space Curving

The surface reconstruction for different object by using space curving technique is shown below. We taken different 2D views of image and converted them to its silhouette by representing the the object with '1' and background with '0'. By back projecting the silhouette from the respective position, it gave the 3D object surface. Below figures shows the image of doll, its silhouette and the 3D surface formed by using space curving technique .

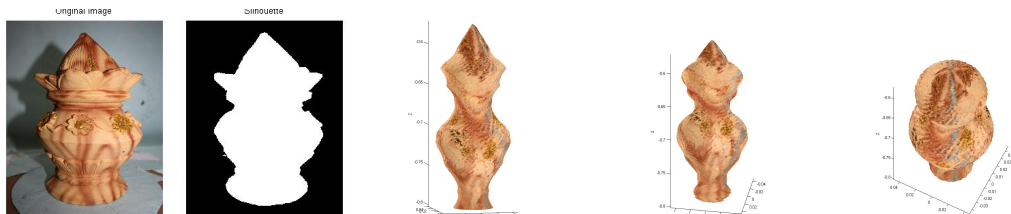


Figure 7.1: Silhouette of image and its 3D reconstructed views

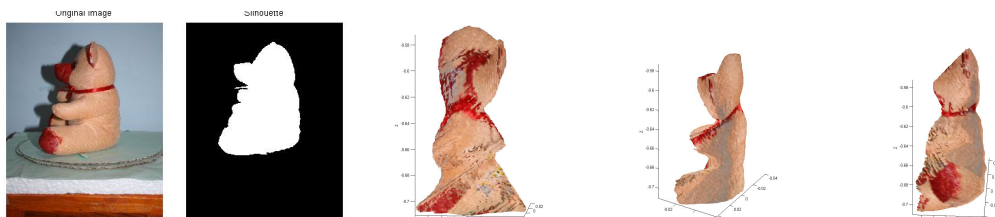


Figure 7.2: silhouette of teddy and its 3D views

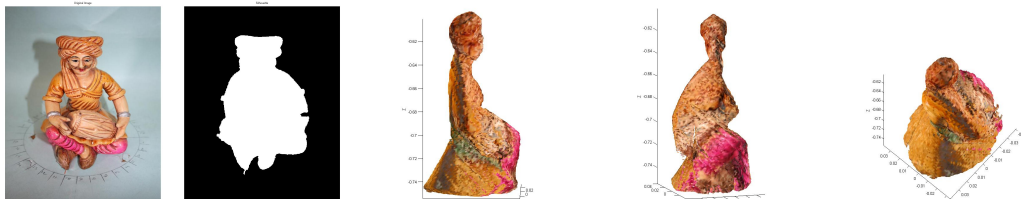


Figure 7.3: silhouette of doll and its 3D views

7.2 Mesh Generation

The reconstructed surface from space curving technique take nearly 10MB space for single frame with a minimum quality. So, to reduce spacial redundancy we used meshing, where we represent the data in terms of interpolative basis. For Mesh generation we used the *Abaqus, Comsol Multiphysics* packages.

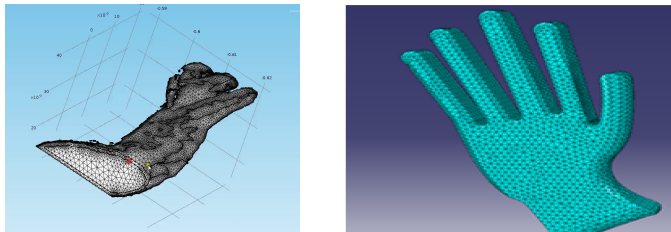


Figure 7.4: Meshing of Dinosaur Hand VRML using COMSOL and Hand model created using Abaqus

7.3 Motion Estimation

For reducing the temporal redundancy for creating 3D video we are extended the concept of 2D block matching algorithm to 3D block matching algorithm. But due to the lack of 3D video sequence we are used the 3D block matching algorithm for image and its depth map sequence. For motion estimation purpose we divided the total 3D grid into cubic elements and search for matching block in reference frame in 3-dimensions. Below figure shows the 3D block matching algorithm.



Figure 7.5: 2D image and its 3D with Depth map assignment

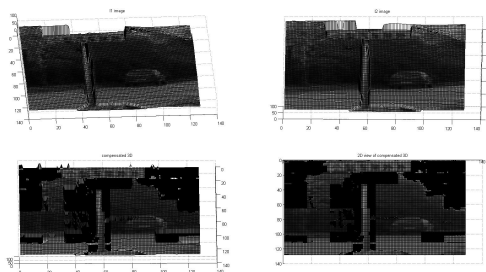


Figure 7.6: 3D motion estimation using Block Matching Algorithm

Apart from the 3D motion estimation, we worked on the data rate reduction for image and its respective depth map sequence. By exploiting relation between the image and depthmap using the weighting function. Below we can observe that if we estimate the motion estimation by using the only images the data to transmit error in depth map is increased in fig 7.7. If we estimate the motion vector based on depth map then data for transmit error in image increased in fig 7.8. By using the weighting function we obtained the minimum data rates with less distortion fig 7.9. The correlation functions for different cases shown in fig 7.10.



Figure 7.7: Image frames, error frames of image and depth map using motion vectors of image. Corresponding autocorrelation values are $R_{image}(0) = 54.56$ $R_{depth}(0) = 1892.85$

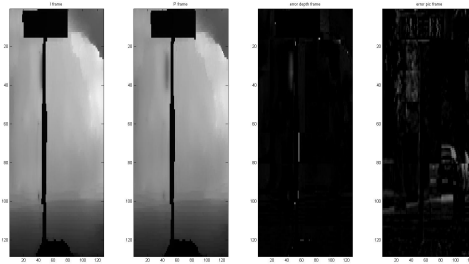


Figure 7.8: Normalized depth map images frames and error frame of depthmap and image using motion vectors of depth map images. Corresponding autocorrelation values are $R_{image}(0) = 101.48$ $R_{depth}(0) = 996.13$

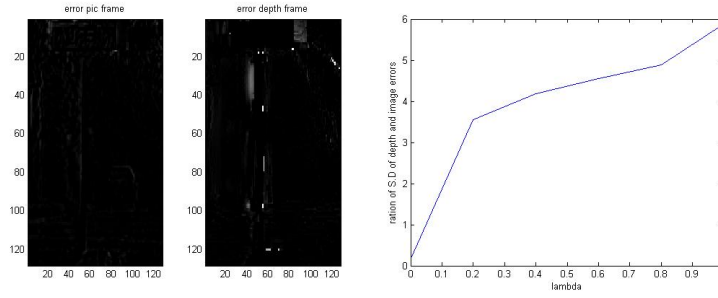


Figure 7.9: Error frames of image and depth map for combined motion vector calculation, Graph of λ vs Standard deviation ratio of depth map and image.

λ is optimum for

$$\text{Standarddeviationratio} = \frac{\text{DynamicRangeofdepthmap}}{D.R.of\text{image}} = \frac{\sigma_{\text{depthmap}}}{\sigma_{\text{image}}} = 5.47$$

corresponding $\lambda = 0.917234863996076$, autocorrelation values are $R_{\text{image}}(0) = 55.06$ $R_{\text{depth}}(0) = 1382.46$

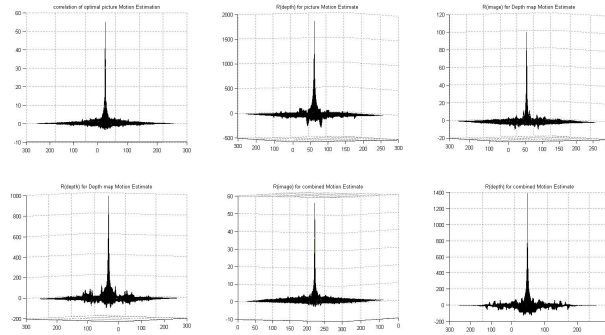


Figure 7.10: Correlation function of error for different motion estimate cases

Here by using λ we are exploiting the correlation between the image and depth map. The error becomes less in both image and depth map compare to the above two case.

Chapter 8

Conclusion and Future Road Map

We briefly explained the methods to reconstruct the 3D surface of an object from a sequence of calibrated images .i.e., Space curving, Shadow curving, variational approach. Among these we reconstructed the 3D surface by using the space curving with 36 2D images. We reconstructed the 3D surface by reprojecting the silhouette of image to the global 3D model. The precision is depending on the number of images we are using.

For transmission of the 3D data over the channel with less bandwidth we opted Finite Element Method for spatial redundancy reduction. We created the mesh of the models using *Abaqus*, *Comsol* packages. And for the temporal redundancy reduction purpose we done the motion compensation. But due to the lack of 3D motion data we done the motion estimation on 2D image and the depth map sequence.

Our future approach is to find the autocalibration for our own setup of cameras. By using our own setup we are going to create our own 3D motion models, which will be use for the spatial and temporal compression.

Bibliography

- [1] Kutulakos, K.N.; Seitz, S.M.; , "A theory of shape by space carving," *Computer Vision*, 1999. The Proceedings of the Seventh IEEE International Conference on , vol.1, no., pp.307-314 vol.1, 1999
- [2] A. W. Fitzgibbon, G. Cross, and A. Zisserman,"3D Structure from Multiple Images of Large-Scale Environments", Springer LNCS 1506, pages 155–170, 1998
- [3] Silvio Savarese , Marco Andreetto , Holly Rushmeier , Fausto Bernardini , Pietro Perona,3d reconstruction by shadow carving: Theory and practical evaluation ,*International Journal of Computer Vision* 71(3), 305336, 2007*International Journal of Computer Vision* 71(3), 305336, 2007
- [4] Tsap, L.V.; Goldgof, D.B.; Sarkar, S.; , "Nonrigid motion analysis based on dynamic refinement of finite element models," *Computer Vision and Pattern Recognition*, 1998. Proceedings. 1998 IEEE Computer Society Conference on , vol., no., pp.728-734, 23-25 Jun 1998
- [5] Zhengyou Zhang; , "Flexible camera calibration by viewing a plane from unknown orientations," *Computer Vision*, 1999. The Proceedings of the Seventh IEEE International Conference on , vol.1, no., pp.666-673 vol.1, 1999
- [6] E. Chen and L. Williams. View Interpolation for Image Synthesis. SIGGRAPH93,pp. 279-288, 1993.
- [7] Kurillo, G. Vasudevan, R. Lobaton, E. Bajcsy, R. , "A Framework for Collaborative Real-Time 3D Teleimmersion in a Geographically Distributed Environment," *Multimedia*, 2008. ISM 2008. Tenth IEEE International Symposium on , vol., no., pp.111-118, 15-17 Dec. 2008
- [8] Yu-Cheng Fan; Shu-Fen Wu; Bing-Lian Lin; , "Three-Dimensional Depth Map Motion Estimation and Compensation for 3D Video Compression," *Magnetics, IEEE Transactions on* , vol.47, no.3, pp.691-695, March 2011
- [9] R. I. Hartley. Euclidean reconstruction from uncalibrated views. In J. Mundy, A. Zisserman, and D. Forsyth, editors, "Applications of Invari-

- ance in Computer Vision”, LNCS 825, pages 237-256. Springer-Verlag, 1994.
- [10] J. Mundy and A. Zisserman. Repeated structures: Image correspondence constraints and ambiguity of 3D reconstruction. In J. Mundy, A. Zisserman, and D. Forsyth, editors, ”Applications of invariance in computer vision”, pages 89-106. Springer-Verlag, 1994.
- [11] Seung-Ryong Han; Yamasaki, T.; Aizawa, K.; , ”Geometry compression for time-varying meshes using coarse and fine levels of quantization and run-length encoding,” *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on* , vol., no., pp.1045-1048, 12-15 Oct. 2008
- [12] Seung-Ryong Han; Toshihiko Yamasaki; Kiyoharu Aizawa; , ”3D Video Compression Based on Extended Block Matching Algorithm,” *Image Processing, 2006 IEEE International Conference on* , vol., no., pp.525-528, 8-11 Oct. 2006
- [13] Seung-Ryong Han; Yamasaki, T.; Aizawa, K.; , ”Time-Varying Mesh Compression Using an Extended Block Matching Algorithm,” *Circuits and Systems for Video Technology, IEEE Transactions on* , vol.17, no.11, pp.1506-1518, Nov. 2007
- [14] Hartley, R.I.; , ”An algorithm for self calibration from several views,” *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR ’94., 1994 IEEE Computer Society Conference on* , vol., no., pp.908-912, 21-23 Jun 1994
- [15] W. Matusik, C. Buehler, R. Raskar, S. Gortler, L. McMillan, “Image-based Visual Hulls, SIGGRAPH 2000, pp. 369-374.
- [16] S. Jung, R. Bajcsy, “A Framework for Constructing Real-time Immersive Environments for Training Physical Activities,” *Journal of Multimedia* vol. 1, no. 7, November/December 2006.
- [17] G. Kurillo, L. Zeyu, R. Bajcsy, “Wide-area external multi-camera calibration using vision graphs and virtual calibration object,” *Second ACM/IEEE International Conference on Distributed Smart Cameras, 2008* , pp.1-9, 7-11 Sept. 2008.
- [18] R. Vasudevan, G. Kurillo, E. Lobaton, T. Bernardin, O. Kreylos, R. Bajcsy, K. Nahrstedt, “High-Quality Visualization for Geographically Distributed 3-D Teleimmersive Applications,” *IEEE Transactions on Multimedia*, vol. 13, no. 3, pp.573-584, June 2011.
- [19] G. Kurillo, R. Vasudevan, E. Lobaton, R. Bajcsy, “A Framework for Collaborative Real-Time 3D Teleimmersion in a Geographically Dis-

- tributed Environment,” *Tenth IEEE International Symposium on Multimedia, 2008*. pp.111-118, 15-17 Dec. 2008.
- [20] Z. Zhang, “A Flexible New Technique for Camera Calibration,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 22, No. 11, November 2000.
- [21] R.Y. Tsai, “A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses,” *IEEE J. Robotics and Automation* vol. 3, no. 4, pp. 323-344, Aug.1987.
- [22] S. Jung, R. Bajcsy, “Learning Physical Activities in Immersive Virtual Environments,” *IEEE International Conference on Computer Vision Systems* pp. 5, 04-07 Jan. 2006.
- [23] H. Baker, D. Tanguay, I. Sobel, D. Gelb, M. Gross, W. Culbertson, T.Malzenbender, “The coliseum immersive teleconferencing system,” *In Proceedings of International Workshop on Immersive Telepresence* France 2002.
- [24] O. Faugeras, T. Luong, and S. Maybank, “Camera Self-Calibration: Theory and Experiments,” *Proc Second European Conf. Computer Vision*, pp. 321-334, May 1992.
- [25] Olivier Faugeras, Renaud Keriven, Complete Dense Stereovision using Level Set Methods, ECCV- Volume I ,1998
- [26] El-Melegy, M.T.; Al-Ashwal, N.H.; , ”A variational technique for 3D reconstruction from multiple views,” *Computer Engineering and Systems*, 2007. ICCES '07. International Conference on , vol., no., pp.38-43, 27-29 Nov. 2007
- [27] Faugeras, O.; Keriven, R.; , ”Variational principles, surface evolution, PDE’s, level set methods and the stereo problem,” *Biomedical Imaging*, 2002. 5th IEEE EMBS International Summer School on , vol., no., pp. 83 pp., 15-23 June 2002
- [28] O. Faugeras and G. Toscani, “The Calibration Problem for Stereo,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 15-20, June 1986.
- [29] R. Hartley, “Self-Calibration from Multiple Views with a Rotating Camera,” *Proc. Third European Conf. Computer Vision*, pp. 471-478, May 1994.
- [30] R.I. Hartley, “An Algorithm for Self-Calibration from Several Views,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 908-912, June 1994.

- [31] M. Pollefeys, "Self-calibration and Metric 3D Reconstruction from Uncalibrated Image Sequences," *Ph.D. Thesis*, ESAT-PSI, K.U. Leuven, 1999.
- [32] B. Triggs, "Autocalibration from Planar Scenes," *Proc. Fifth European Conf. Computer Vision*, pp. 89-105, June 1998.
- [33] Mei Han, Takeo Kanade, "Creating 3D Models with Uncalibrated Cameras," *Applications of Computer Vision, 2000, Fifth IEEE Workshop*, vol. no., pp. 178-185, 2000.
- [34] Martin, W. N. and J. K. Aggarwal: 1983, 'Volumetric Descriptions of Objects from Multiple Views'. *IEEE Proc. Pattern Anal. Machine Intel* 1.5(2), 150-158.
- [35] Debevec, P. E., C. J. Taylor, and J. Malik: 1996, 'Modeling and Rendering Architecture from Photographs: A hybrid geometry and image-based approach'. In: *Proc. SIGGRAPH'96*. pp. 11-20.
- [36] Kakadiaris, I. A. and D. Metaxas: 1995, '3D Human Body Model Acquisition from Multiple Views'. In: *Proc. Int. Conf. on Computer Vision*. pp. 618-623.
- [37] Faugeras, O.: 1995, 'Stratification of three-dimensional vision: projective, affine, and metric representations'. *J. Opt. Soc. Am.* A12(3), 465-484.
- [38] Koenderink, J. J. and A. J. van Doorn: 1991, 'Affine structure from motion'. *J. Opt. Soc. Am.* A(2), 377-385.
- [39] D. Ezra, G. J. Woodgate, B. A. Omar, N. S. Holliman, J. Harrold and L. S. Shapiro "New autostereoscopic display system", pp. 10-14, *Sharp Technical Journal* Volume 62, August 1995.
- [40] G. J. Woodgate, D. Ezra, J. Harrold, Nicolas S. Holliman, G. R. Jones, R. R. Moseley, (1998), *Autostereoscopic 3D display systems with observer tracking*, *Image Communication Special Issue on 3D Video Technology (EURASIP - 1998)*, pp. 131
- [41] Yu-Cheng Fan; Shu-Fen Wu; Bing-Lian Lin; "Three-Dimensional Depth Map Motion Estimation and Compensation for 3D Video Compression," *Magnetics, IEEE Transactions on*, vol. 47, no. 3, pp. 691-695, March 2011.
- [42] Aroh Barjatya, Block matching algorithms for motion estimation, Final project paper, DIP, at Utah state university, spring 2004.