3D POINT CLOUD CLASSIFICATION USING DEEP LEARNING

A PROJECT REPORT

Submitted by

JENNISHA CHRISTINA MARTIN [REGISTERNO: 211417104097]
SIVASANKARI R [REGISTERNO: 211417104261]
SAKTHI R V [REGISTERNO: 211417104231]

in partial fulfillment for the award of the

degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2021

BONAFIDE CERTIFICATE

Certified that this project report **"3D POINT CLOUD CLASSIFICATION USING DEEP LEARNING"** is the bonafide work of **"JENNISHA CHRISTINA MARTIN(211417104097) SIVASANKARI R(211417104261), SAKTHI R V (211417104231)"** who carried out the project work under my supervision.

SIGNATURE

Dr.S.MURUGAVALLI,M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE

Mrs. SANGEETHA KALYANARAMAN,M.E

ASSOCIATE PROFESSOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidates were examined in the Anna University Project Viva-Voce Examination held on.....

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like express our heartfelt and sincere thanks to our Directors **Tmt. C. VIJAYARAJESWARI, Dr . C. SAKTHIKUMAR, M.E. Ph.D.,** and **Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.Murugavalli , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank **Project Guide Mrs.Sangeetha Kalyanaraman,M.E** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

We would like to thank **god** for showering his blessing on us and we also thank our **parents** much for their support both emotional and financial over the years.

JENNISHA CHRISTINA MARTIN
SIVASANKARI R
SAKTHI R V

ABSTRACT

In the last few years, the availability of 3D content is still less than 2D counterpart. Hence many 2D-to-3D image conversion methods have been proposed. Methods involving human operators have been most successful but also time- consuming and costly. Automatic methods, that make use of a deterministic 3D scene model, have not yet achieved the same level of quality for they rely on assumptions that are often violated in practice. Here two types of methods are developed. The first is based on learning a point mapping from local image/ attributes, such as color, spatial position. The second method is based on globally estimating the entire depth map of a query image directly from a repository of 3D images (image + depth pairs or stereo pairs) using a nearest-neighbour regression type idea. It demonstrates the ability and the computational efficiency of the methods on numerous 2D images and discusses their drawbacks and benefits. This learning has lately attracted increasing attention due to its wide applications in many areas, such as computer vision, autonomous driving, and robotics. As a dominating technique in AI, deep learning has been successfully used to solve various 2D vision problems. However, deep learning is still in its infancy due to the unique challenges faced by the processing with deep neural networks. Recently, deep learning has become even thriving, with numerous methods being proposed to address different problems in this area.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVATIONS

S.NO	ABBREVATION	EXPANSION	PAGE NO.
1.	MSE	Mean Squared Error	43
2.	IOU	Intersection Over Union	43
3.	CE	Cross Entropy	44
4.	EMD	Earth Mover Distance	44
5.	CD	Chamfer Distance	44
6.	DDD	Domain Driven Design	12

LIST OF SYMBOLS

S.NO	SYMBOL	DEFINITION	PAGE NO.
1.	X	Ground – Truth Shape	43
2.	X^	Reconstructed Shape	43
3.	n	Normal Distance	43
4.	L1 , L2	Distance	43
5.	I(-)	Indicator Function	43
6.	N	Total Number of Voxel	44

CHAPTER 1

CHAPTER 1
INTRODUCTION

1.1. OVERVIEW

The main aim of this Project is “Image Reconstruction” - to reconstruct a 2D image to a 3D image , The convenience of 3D-capable hardware today, such as TVs, Blu-Ray players, gaming consoles, and smart phones, is not yet matched by 3D content production. Today there exists an urgent need to convert the existing 2D content to 3D. The last two decades have seen an increase in the number of devices for the acquisition of 3D point clouds. In the same period the resolution, accuracy and acquisition rate of these devices have increased. This has led to a marked increase in the resolution, accuracy and size of point clouds that have to be processed .We focus on the works which use deep learning techniques to estimate the 3D shape of generic objects either from a single or multiple RGB images. This trend is set to continue with devices becoming cheaper and more accessible to the general public as Presently the biggest challenge for processing point clouds is their size .

1.2 PROBLEM DEFINITION

The proposed methods carry the philosophy of machine learning. They apply to arbitrary scenes and require no manual explanation. Two types of methods are proposed: The first one is based on learning a point mapping from local image/ attributes, such as color, spatial position, and motion at each pixel, to scene-depth at that pixel using a regression type idea . The second one is based on globally ,
Estimating the entire depth map of a query image directly from a repository of 3D images (image + depth pairs or stereopairs) using a nearest-neighbor regression type idea. It introduces local method and evaluates the qualitative performance and the computational efficiency of both the local and global methods. The improved quality of the depth maps produced by the global method relative to state-of-the- art methods together with up to 4 orders of magnitude reduction in computational effort and weakness of the methods are also demonstrated.

CHAPTER 2

CHAPTER 2
LITERATURE SURVEY

1. Deep learning for 3D Point clouds.

Year: 2020

Author: Yulan Guo Hanyun Wang, Qingyong Hu, Hai Liu ,Li Liu and Mohammed Bonnamoun.

Description:

This paper presents a comprehensive review of recent progress in deep learning . Methods for point clouds, it covers three major tasks, including

I. **3D shape classification** –a 3D shape has three dimensions. The D in '**3D**' stands for dimensional. In a world with three dimensions, you can travel forwards, backwards, right, left, and even up and down. The ability to travel up into space and back down differentiates **3D** from 2D.

II. **3D object detection and tracking** -3D object tracking simplifies to greedy closest-point matching, CenterPoint, first detects centers of objects using a key point detector and regresses to other attributes, including 3D size, 3D orientation, and velocity. In a second stage, it refines these estimates using additional point features on the object.

III. **3D point cloud segmentation**- is the process of classifying point clouds into multiple homogeneous regions. Moreover, this paper also presents comparative results on several publicly available datasets, together with insightful observations and inspiring future research directions.

Merits: A promising solution to address the raw point clouds with the Conv Nets. Since Conv Nets has the advantage of overlapping during convolutional operation may benefit the future.

Demerits: It is quite expensive and it has included some challenges - the irregular network connection will reduce the performance**.**

2. 3D Point Cloud Semantic Segmentation.

Year: 2019

Author: Jiaojiao Tian ,Yuxing Xie and Xiao Xiang Zhu.

Description:

This article summarizes available data sets and relevant studies on recent developments in 3D point cloud semantic segmentation (PCSS) and point cloud segmentation (PCS). Firstly, This article outlines the acquisition and evolution of the 3D point cloud from the perspective of remote sensing and computer vision, as well as the published benchmarks for PCSS studies . This literature review keep up with latest deep learning techniques, This article refers to point cloud semantic segmentation/classification/labeling, i.e. the task of associating each point of a point cloud with a semantic label, as PCSS , and summarizes existing studies on this topic. Also traditional and advanced techniques used for Point Cloud Segmentation (PCS) and PCSS are reviewed.

Merits: High accuracy suitable for large area and not affected by weather . Also Global data is available compared to ALS complete building façade, information is available as 4D information middle accuracy and not affected by weather in any case.

Demerits: Expensive affected by mirror reflection long scanning time. Close range limited accuracy .

3.Multiscale Hierarchical Network for 3D Point Cloud Semantic Segmentation

Year: 2019

Author: Xiaoli liang and Zhonglian Fu

Description: In this paper we proposed a multiscale hierarchical network (MHNet) for 3D point cloud semantic segmentation. First, a hierarchical point Cloud feature extraction structure is constructed to learn multiscale local region features. Then, these local features are subjected to feature propagation to obtain the features of the entire point set for pointwise label prediction. To take full advantage of the correlations of propagated information between the different scale coarse layers and the original points, the local features of each scale are characterized by feature propagation to obtain the features of the original point clouds at the corresponding scale. The global features propagated from different scales are integrated to constitute the final features of the input point clouds. The concatenated multiscale hierarchical features, including both local features and global features, can better predict the segmentation probability of each point cloud. Finally, the predicted segmentation results are optimized using the conditional random filed (CRF) with a spatial consistency constraint. The efficiency of MHNet is evaluated on two 3D datasets (S3DIS and ScanNet), and the results show performance comparable or superior to the state-of-the-art on both datasets.

Merits: To take fully advantage of the correlations of Propagated information between the different scale coarse layers and the original points, the local features of each scale are characterized by feature propagation to obtain the features of the original point clouds at the corresponding scale.

Demerits: Although there is considerable noise points in the segmentation result of MH Net, the number of noise point's only accounts for a small part of the total point clouds. Even all noise is corrected, the accuracy will not be greatly improved.

4. **3D Point Cloud Segmentation.**

Year: 2018

Author: Anh Nguyen and Bac Le.

Description: 3D point cloud segmentation is the process of classifying point clouds into multiple homogeneous regions, the points in the same region will have the same properties. The segmentation is challenging because of high redundancy, uneven sampling density, and lack explicit structure of point cloud data. This problem has many applications in robotics such as intelligent vehicles, autonomous mapping and navigation. Many authors have introduced different approaches and algorithms. In this survey, we examine methods that have been proposed to segment 3D point clouds. The advantages, disadvantages, and design mechanisms of these methods are analyzed and discussed. Finally, we outline the promising future research directions.

Merits: A method that takes advantage of contextual information combine with geometric reasoning or learning techniques would improve the segmentation result.

Demerits: Moreover, due to the limitations of the 3D sensors, the foreground is often highly entangled with the background. These problems present a difficult challenge when designing a segmentation algorithm.

5. Deep Learning on Point Sets for 3D Classification and Segmentation.

Year: 2017

Author: Charles R. Qi ,Hao Su ,Kaichun Mo ,Leonidas J. and Guibas

Description: Point cloud is an important type of geometric data structure. Due to its irregular format, most researchers transform such data to regular 3D voxel grids or collections of images. This, however, renders data unnecessarily voluminous and causes issues. In this paper, we design a novel type of neural network that directly consumes point clouds, which well respects the permutationinvariance of points in the input. Our network, named Point Net, provides a unified architecture for applications ranging from object classification, part segmentation, to scene semantic parsing. Though simple, Point Net is highly efficient and effective.Empirically, it shows strong performance on par or even better than state of the art. Theoretically, we provide analysis towards understanding of what the network has learnt and why the network is robust with respect to input perturbation and corruption.

Merits : Point clouds are simple and unified structures that avoid the combinatorial irregularities.

Demerits : Point cloud is Expensive and affected by mirror reflection long scanning time. Point cloud deals with only limited accuracy**.**

CHAPTER 3

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

3D reconstruction helps to recover correct shape and size of the objects. It helps to enhance the quality of images. The extensive literature survey shows many methods are existing for 3D reconstruction. The different methodologies, advantages and disadvantages have been severed and listed. This helps to implement 3D reconstruction technique efficiently. This paper gives detail survey of algorithms used for 3D reconstruction of images and their advantages and disadvantages. One of the merits is to take full advantage of the correlations of Propagated information between the different scale coarse layers and the original points, the local features of each scale are characterized by feature propagation to obtain the features of the original point clouds at the corresponding scale and the demerits are it has a Close range limited accuracy.

3.2 PROPOSED SYSTEM

A new class of methods is proposed to aim at 2D-to-3D image conversion that is based on the radically different approach of learning from examples. One method that is proposed is based on learning a point mapping from local image attributes to scene-depth. The other method is based on globally estimating the entire depth field of a query directly from a repository of image + depth pairs using nearest-neighbor-based regression. It objectively validates the algorithms' performance against state-of-the-art algorithms. While the local method was outperformed by other algorithms, it is extremely fast as it is, basically, based on table look-up. However, the global method performed better than the state-of-the-art algorithms in terms of cumulative performance across two datasets and two testing methods, and has done so at a fraction of CPU time. Anaglyph images produced by the algorithms result in a comfortable 3D experience but are not completely void of distortions.

Clearly, there is room for improvement in the future. With the continuously increasing amount of 3D data on-line and with the rapidly growing computing power in the cloud, the proposed framework seems a promising alternative to operator-assisted 2D-to-3D image and video conversion.

3.3 TECHNOLOGY STACK

- Input Requirements : PNG File .
- Output Requirements : Mat lab Plot Format .
- Hardware Environment : Laptop or Desktop – 1 GB ram & 500 GB hard disk& I3 Processor.
- Software Environment : Mat lab 2018 a & windows 10.

There are two types of 2D-to-3D image conversion methods: semi-automatic methods and automatic methods.

A. Semi-automatic methods:

Semi-automatic methods are more effective. This method has been effectively used commercially by such companies as Imax Corp., Digital Domain Productions Inc. etc. In order to shorten operator involvement in the process and lower the cost while speeding up the conversion, research has recently focused on the most labor-intensive steps of the manual involvement, namely spatial depth assignment. Further simplify operator involvement by first computing optical flow, then applying structure-from- motion estimation and finally extracting moving object boundaries. The role of an operator is to correct errors in the automatically computed depth of moving objects and assign depth in undefined areas.

B. Automatic Model:

The difficult of depth estimation from a single 2D image is the main step in 2D-to-3D image conversion. Methods called multiview stereo, attempt to improve depth by estimating scene geometry from multiple images not taken instantaneously. Such methods are similar in spirit to the methods proposed here, the main difference is that while these methods use images known to show the same scene as the query image, all images accessible in a large repository and automatically select suitable ones for depth retrieval. Real-time methods have been implemented in Blu-Ray 3D players by LG, Samsung, Sony and others.

DDD offers its TriDef 3D software for PCs, TVs and mobile devices. Recently, machine-learning-inspired techniques employing image parsing have been used to estimate the depth map of a single monocular image . Such methods have the potential to automatically generate depth maps, but work only on few types of images (mostly architectural scenes). Metric based on histogram of gradients was used for selecting most similar depth fields from a database. It has been observed that there is no significant quality degradation but a significant reduction of the computational complexity .

CHAPTER 4

CHAPTER 4

SYSTEM DESIGN

4.1 ER DIAGRAM

An Entity–relationship model (ER model) describes the structure of a database. The diagram given below describes the process involved in reconstructing a 2d image to a 3d image . Firstly, validation of the object and its background takes place by identifying the x, y, z coordinates of the image captured ,if the validations are true then the depth of the object is calculated for each of the coordinates , once the depth is calculated then another validation takes place to check whether the objects and its depth calculated for each of the three coordinates are organized and whether the calculated values for each of the coordinates are forming a curve ,if these conditions are true then 3d enhancement takes place , thus the reconstruction of a 2d to 3d image .

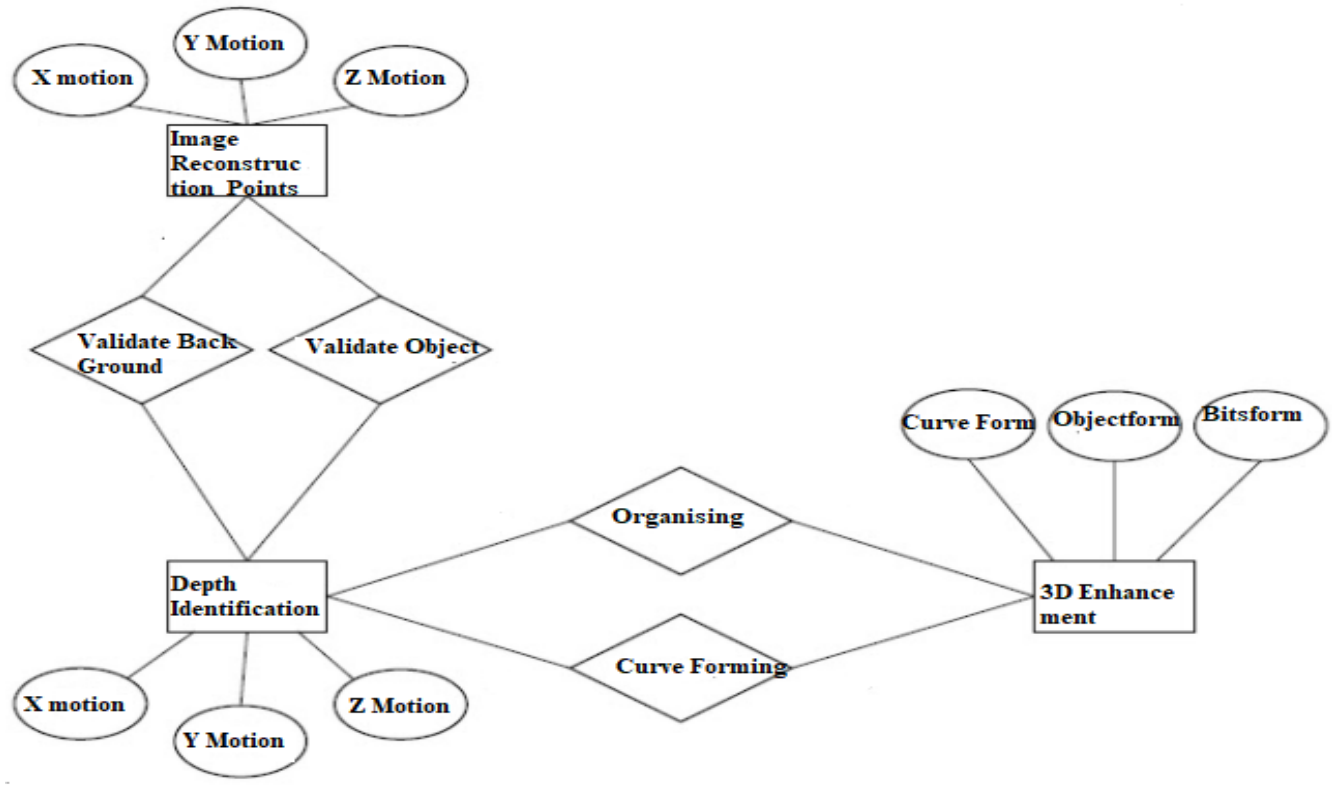


Figure: 4.1 ER DIAGRAM

4.2 DATA DICTIONARY

Data Dictionary is a centralized repository of metadata . Meta data is data about data. Here , The sample Description table consists of the fields shown below these fields describe the images assembled to be reconstructed and their type and also what type of filaments does each particles and 2d crystal does the image consist. The Sample Preparation table extracts the data from the sample description table to form the array of resolutions that the images consist of and also the em_solution_composition field calculates the composition of single particles the images consists of .The image Processing and Reconstruction Table consists of fields for selecting the particles from an image that is apt for reconstructing, the em_Particle_picking list field consist of these list of particles for reconstructing, and also this table consists of the em_filament_Selection and em_filament_reconstruction fields required for the proper reconstruction of the image. The Specimen Preparation table consists of fields for identifying the object from its background. The structure analysis table consists of fields for refining the images and identifying the curves from all perspective of the object and also categorizing the object identified .The Data collection table consists of fields for scanning the image and also identifying and calculating the diffraction phase and pattern of the image .

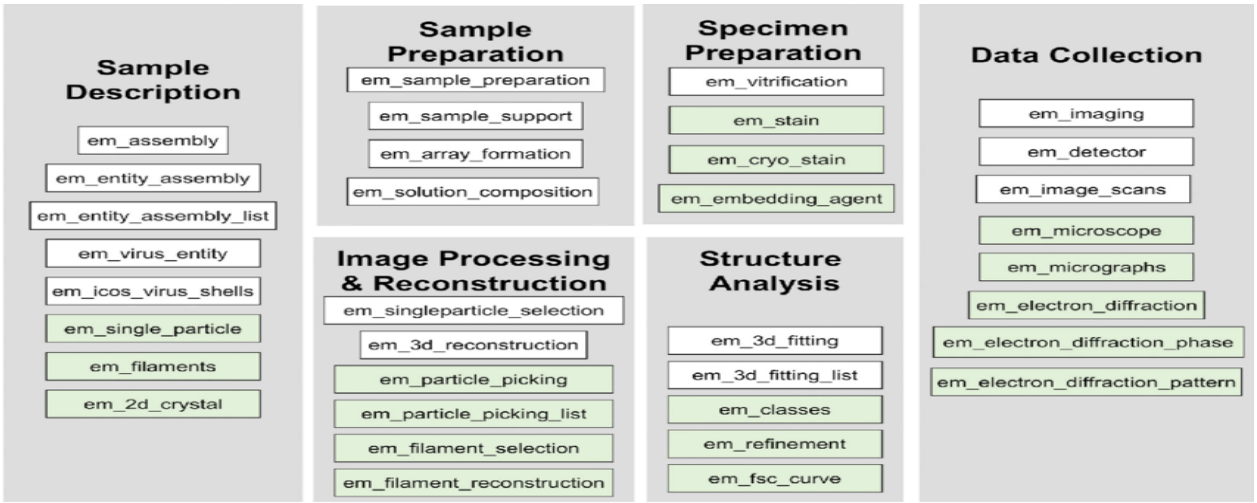


Figure: 4.2 DATA DICTIONARY

4.3 UML DIAGRAMS

4.3.1 USECASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system . Use case diagrams model the functionality of a system using actors and use cases. Here there are two actors - the input image is one actor and 3D covered image is an another actor. Between this two the following actions takes place, Firstly the 2D input image is given in X,Y coordinate and it will extract the depth to the max level (i.e) feature point extraction and then the input of two 2D images were stitched together(i.e) matching point pair selection which is to find the centralised of the image location and need to fix that location. Next we need to analyse the space after space analysis then we need to shape the image (i.e) curve fitting image. After shaping is done we need to render it so for that to fix an object in the space and then that object is rotated till 360 degree and it shown each and every angle and then it show in viewable 3D covered image .Thus the reconstruction of a 2d to 3d image .

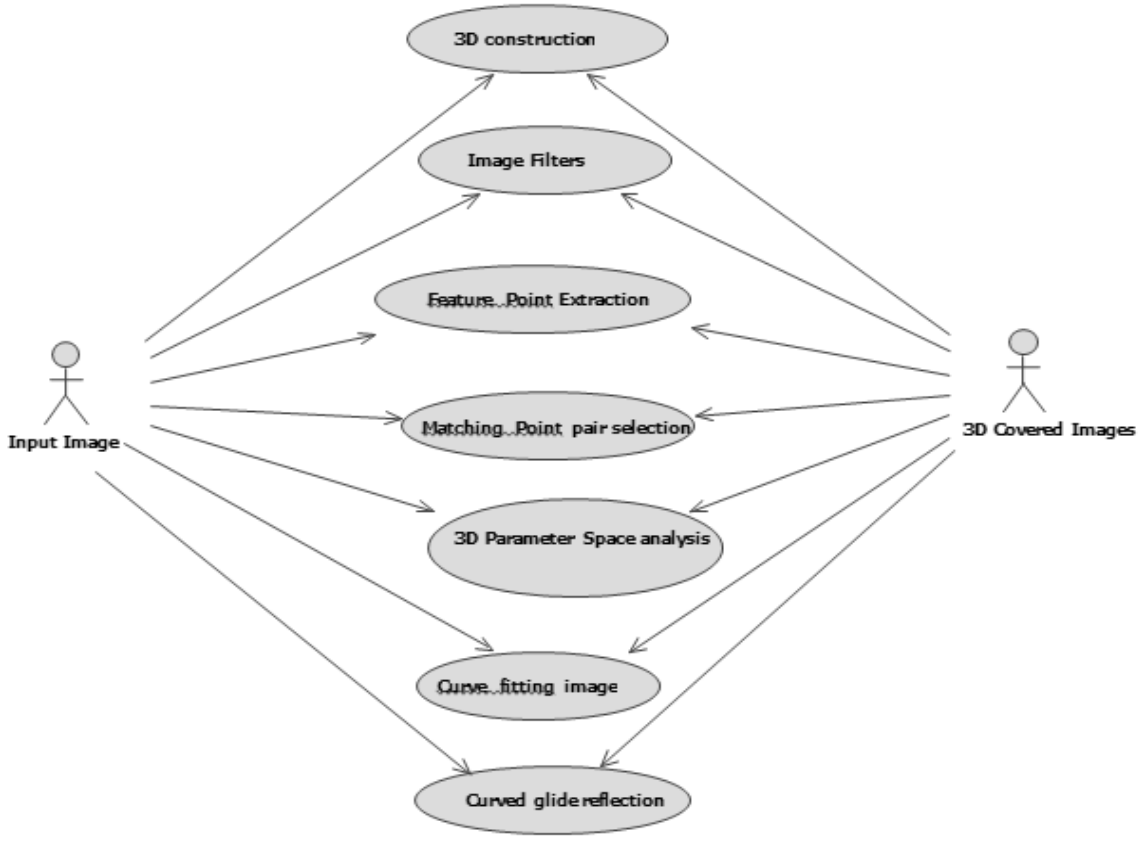


Figure: 4.3 USE CASE DIAGRAM

4.3.2 CLASS DIAGRAM

Class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints.The class diagram describes about the reconstruction of 2D to 3D images. First process is the image construction here we need to give the input image for which we need to construct the output for that the association between each class take place. After this the image selection happens and for that image which for which we selected depth is identified and the length will be calculated. Data acquisition is for object setting and pre-checking images. Image Construction is for source of the image and object arrival and reconstruct the image as per the input given After we re-constructed the image from 2d to 3d. Thus the process of this class diagram.

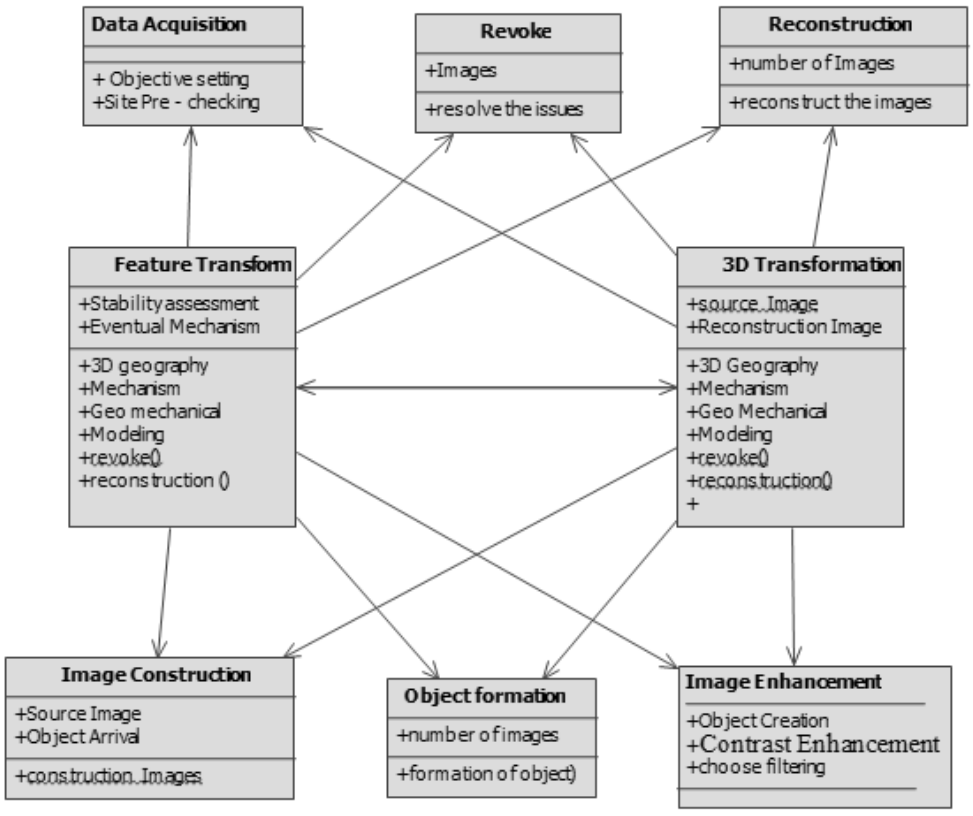


Figure: 4.4 CLASS DIAGRAM

4.3.3 SEQUENCE DIAGRAM

Sequence diagrams describe interactions among classes in terms of an exchange of messages over. UML sequence diagrams are used to show how objects interact in a given situation. A popular use for them is to document the dynamics in an object-oriented system. The above sequence diagram describes about the reconstruction of 2D to 3D images. First process is the image construction here we need to give the input image for which we need to construct the output. Next step is image formation it will form the image into two X, Y coordinate and output will be in three X, Y, Z coordinate. After this the image selection happens and for that image which for which we selected depth is identified and the length will be calculated .Next we need to validate the background for that the input of two 2D images were stitched together so that we could come to know the clear view of the background and finally fix that background and reconstruct the image as per the input given

After we re-constructed the image from 2d to 3d then the image is enhanced, thus the process of this sequence diagram.

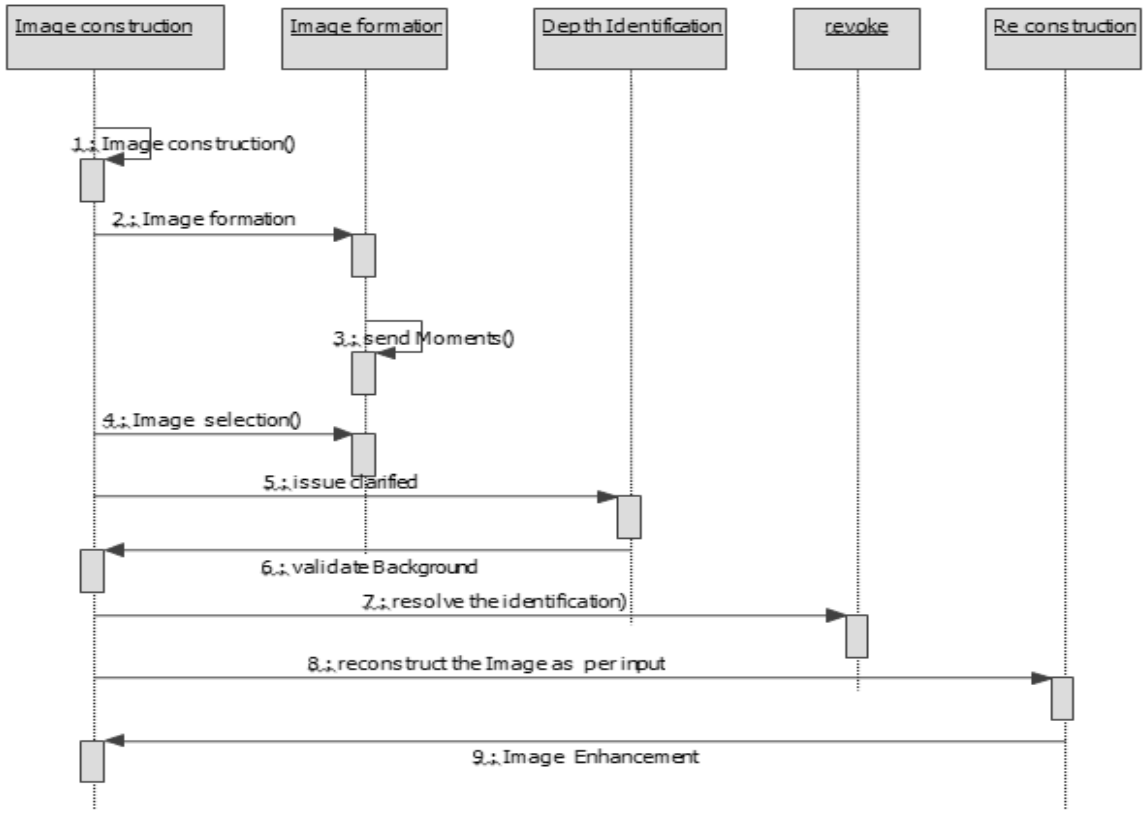


Figure: 4.5 SEQUENCE DIAGRAM

4.3.4 ACTIVITY DIAGRAM

Activity diagram are typically used for business process modeling for modeling the logic captured by a single use case or usage scenario, or for modeling the detailed logic of a business rule. Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Firstly, The Data Acquisition step takes place i.e. where the data is collected. The image Formation Step refers to the image being captured, the image selection step refers to the selection of a image from many to be reconstructed. The next step is image validation in this step the image is validated a multi perspective analysis of a 2d image takes place along with the image background validation ,if the above conditions are satisfied then the Reconstruction takes place ,finally a 3d image is formed .

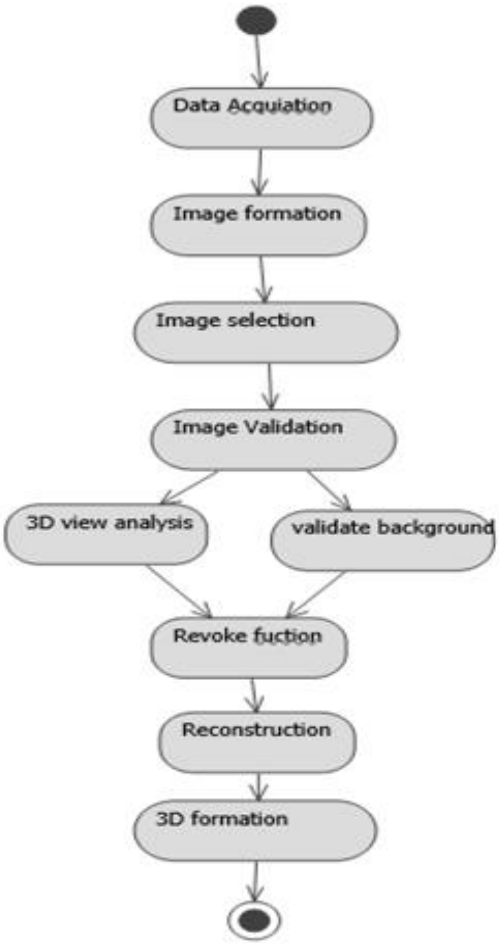


Figure: 4.6 ACTIVITY DIAGRAM

CHAPTER 5

CHAPTER 5

SYSTEM ARCHITECTURE

5.1 ARCHITECTURE OVERVIEW

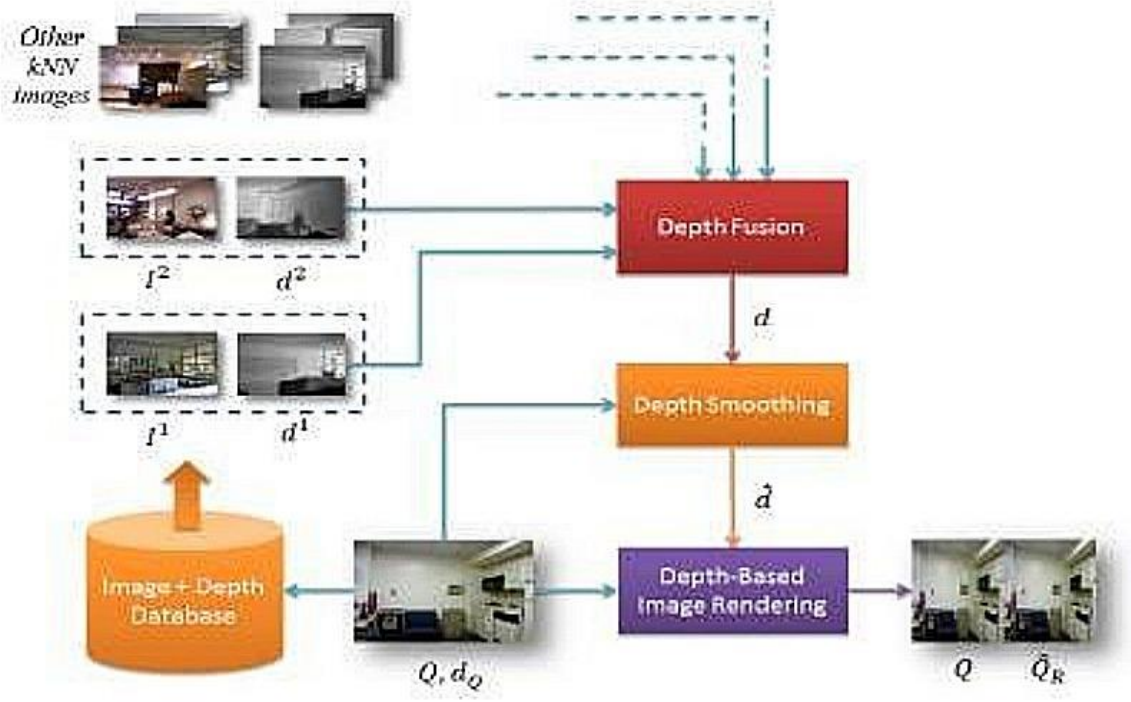


Figure: 5.1 a) SYSTEM ARCHITECTURE

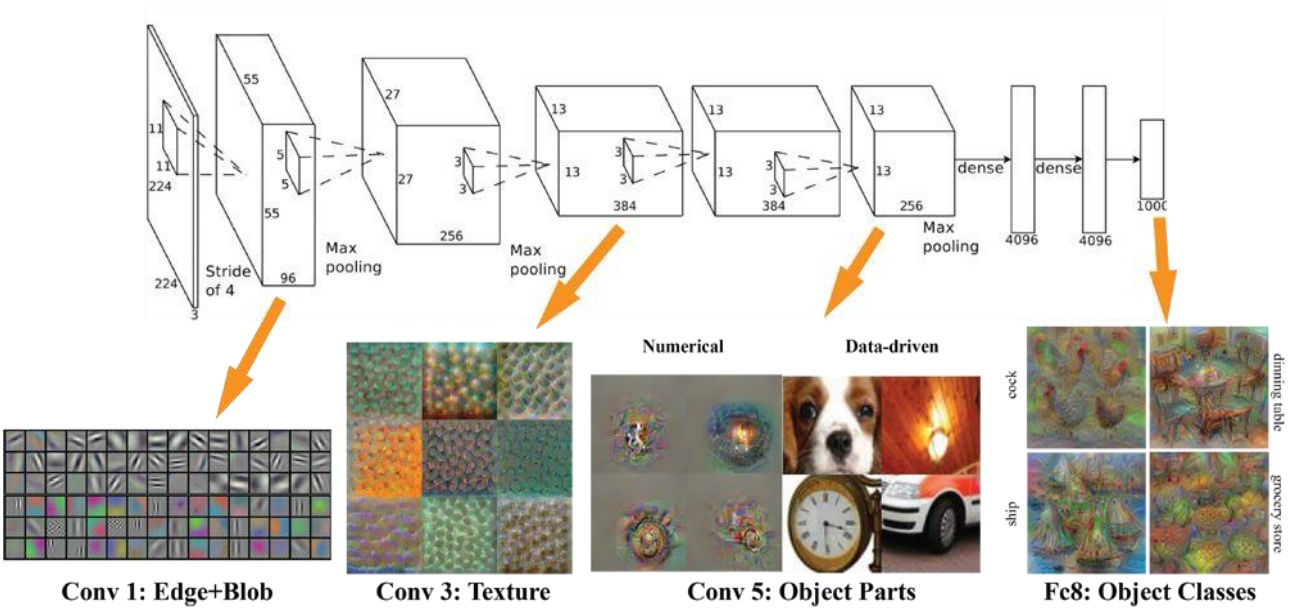


Figure: 5.1 b) SYSTEM ARCHITECTURE

The main goal of this project is to prototype a system which reconstructs rudimentary 3D images from a batch of 2D images . The point transformation is applied basically in real time, because it is based on purely local image/video attributes, such as color, spatial position, and motion at each pixel. A second method is developed that estimates the global depth map of a query image or video frame directly from a repository of 3D images (image + depth pairs or stereopairs) using a nearest- neighbor regression type idea. This approach is built upon a key observation and an assumption.

- The key observation is that among millions of 3D images available on-line, there likely exist many whose 3D content matches that of a 2D input .
- An assumption is made that two images that are photometrically similar also have similar 3D structure (depth).
- Given monocular query image Q, assumed to be the left image of a stereopair that is to be computed, relies on the above observation and assumption to "learn" the entire depth from a repository of 3D images.

- **Search for representative depth fields:**

  Find k 3D images in the repository I that have most similar depth to the query image, for example by performing a k nearest-neighbor (kNN) search using a metric based on photometric properties.

- **Depth fusion:**

  Combine the k representative depth fields, for example, by means of Median filtering across depth field

- **Depth smoothing:**

Process the fused depth field to remove spurious variations, while preserving depth discontinuities, for example, by means of cross- bilateral filtering

.

- **Stereo rendering:**

  Generate the right image of a fictitious stereopair using the monocular Query image and the smoothed depth field followed by suitable processing of occlusions and newly- exposed areas.

5.2 MODULE DESIGN SPECIFICATION

Our project is having the following modules:

The project will be split into 3 Modules

1. Data Collection
2. depth map generation/fusion
3. 3D visualization. Images will be obtained off-line.

ALGORITHM MODEL DESIGN EXPLANATION :

1. Experiment Setup: Data Collection

One of the main goals of the experimental setup is to keep the project within scope. The experiment will be designed such that one can recreate 3D images from 2D images without deviating much from the themes. Furthermore, the goal of the experiment is to create a scenario in which on can avoid some of the obstacles faced during the 2D to 3D conversion Process .

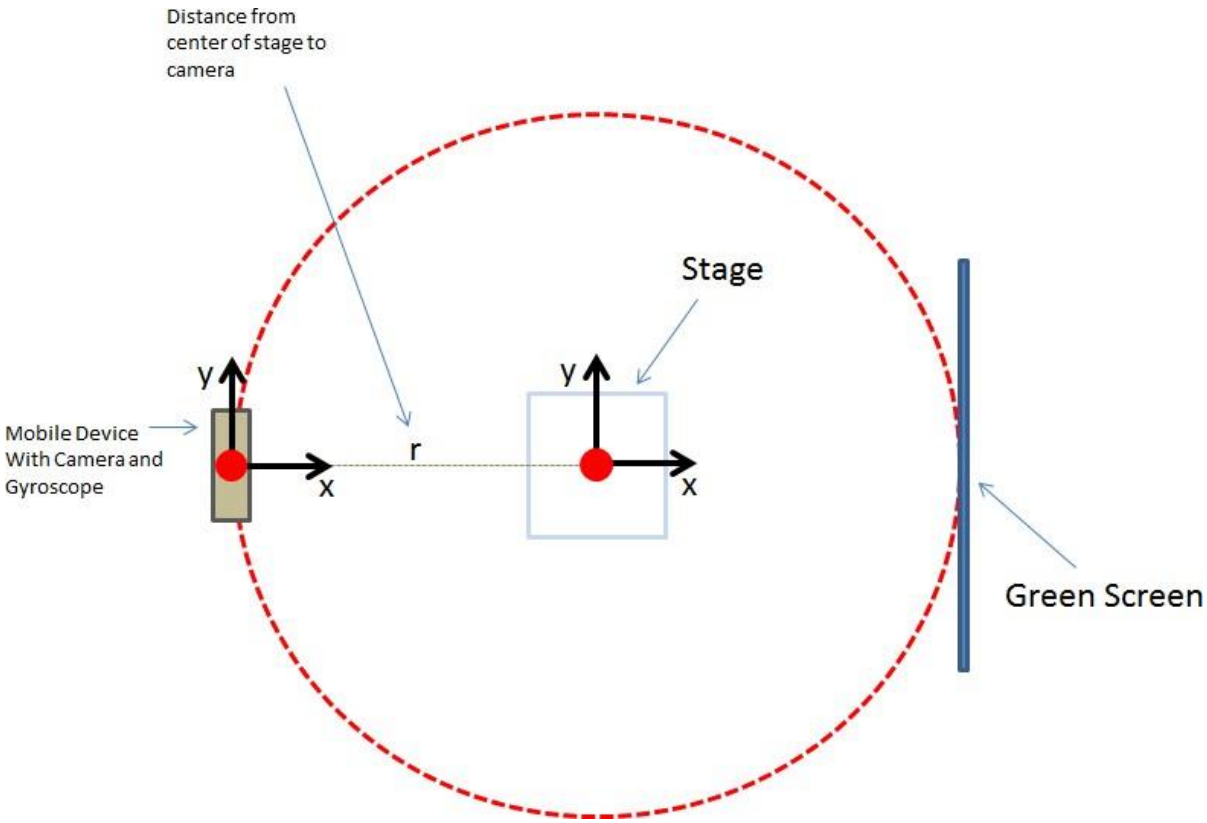


Figure 5.2: The experimental setup.

The setup will consist of: Camera device with camera and gyroscope capabilities, a stage to hold the object to be reconstructed in 3D, and a green screen. The distance between the Camera device and the center of the stage shall be held constant through the experiment. During the experiment, the camera will be rotated around the stage at constant intervals.

An experiment will be designed to allow for the collection of well-conditioned 2D images. The term 'well-conditioned' shall refer to a scene with the following characteristics: uniform illumination, easily segmented, and straightforward correspondences when referencing multiple images of the same scene.

The experiment will consist of: a Camera device for image collection , a stage for setting objects to be reconstructed in 3D, and a green screen to aid in segmentation. The distance between the camera and stage will be held constant.

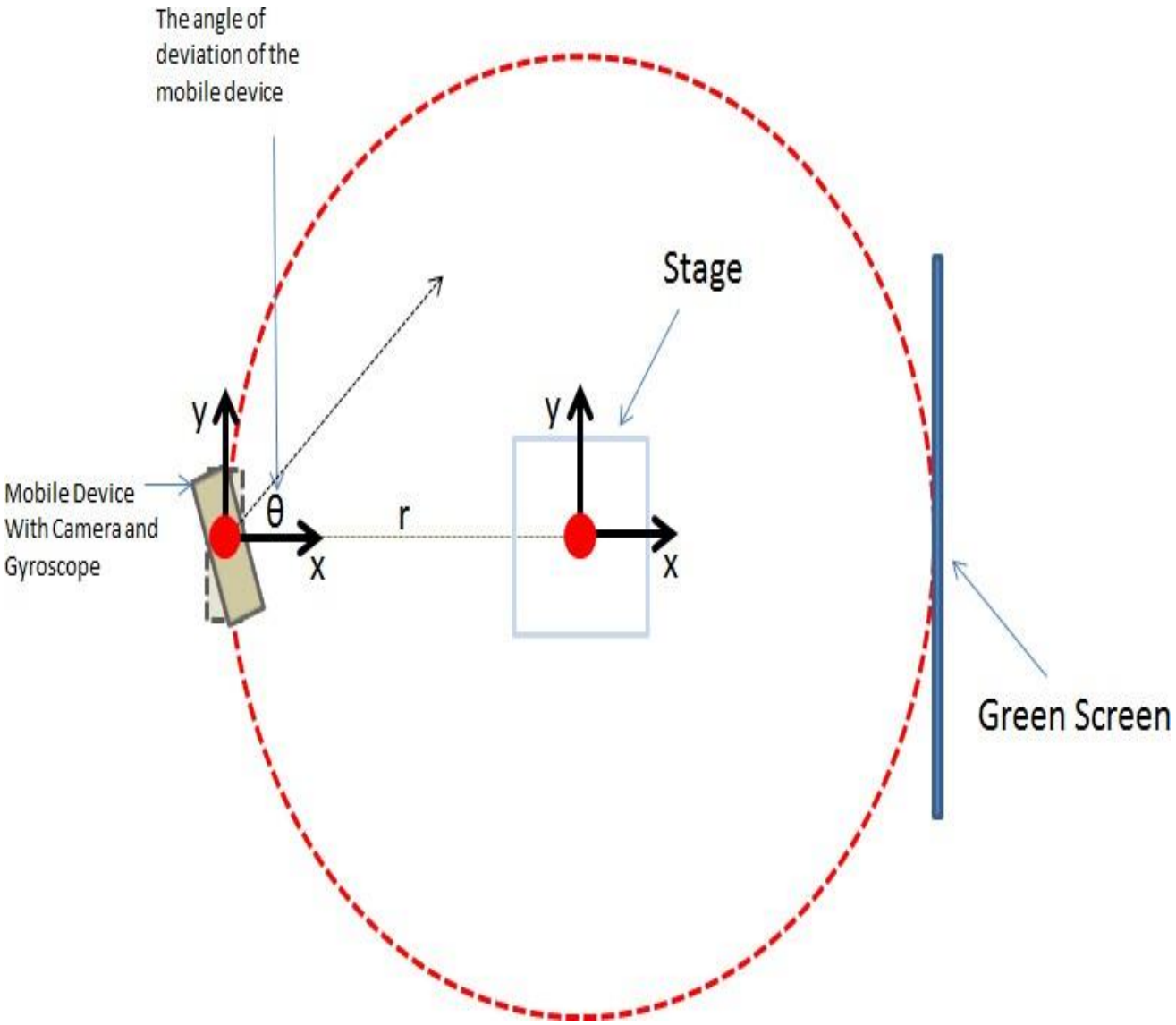


Figure 5.3 The Distance between camera and the Stage at angle 90

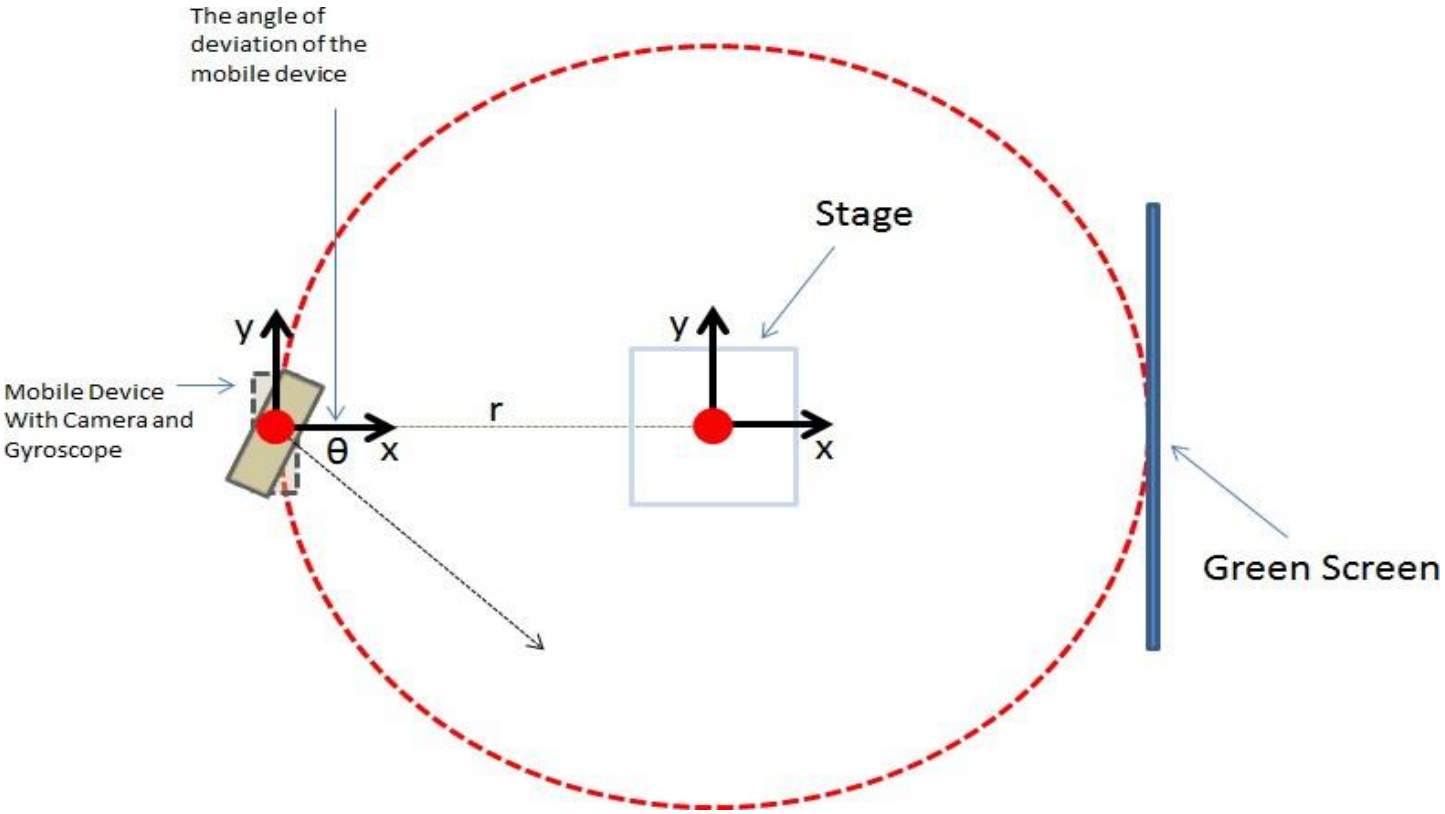


Figure 5.4 The Distance between camera and the Stage at angle 180

During the experiment, multiple images will be obtained by the camera device at various angles between the camera and stage. The first image obtained, at a deviation $\theta=0$ will be known as the template image. θ shall be varied at evenly spaced intervals and symmetric about $\theta = 0$ (i.e. symmetric about the template image)

The angle between the camera and scene (θ) will be varied in order to provide the 3D reconstruction algorithm with images of the scene at various perspectives. These images will be used to recreate the depth information of the scene. $\Delta\theta$ will be constant throughout the entire experiment. N images will be collected at $\Delta\theta$. N will be determined empirically such that the images obtained provide enough data to reconstruct a 3D scene with a reasonable amount of processing.

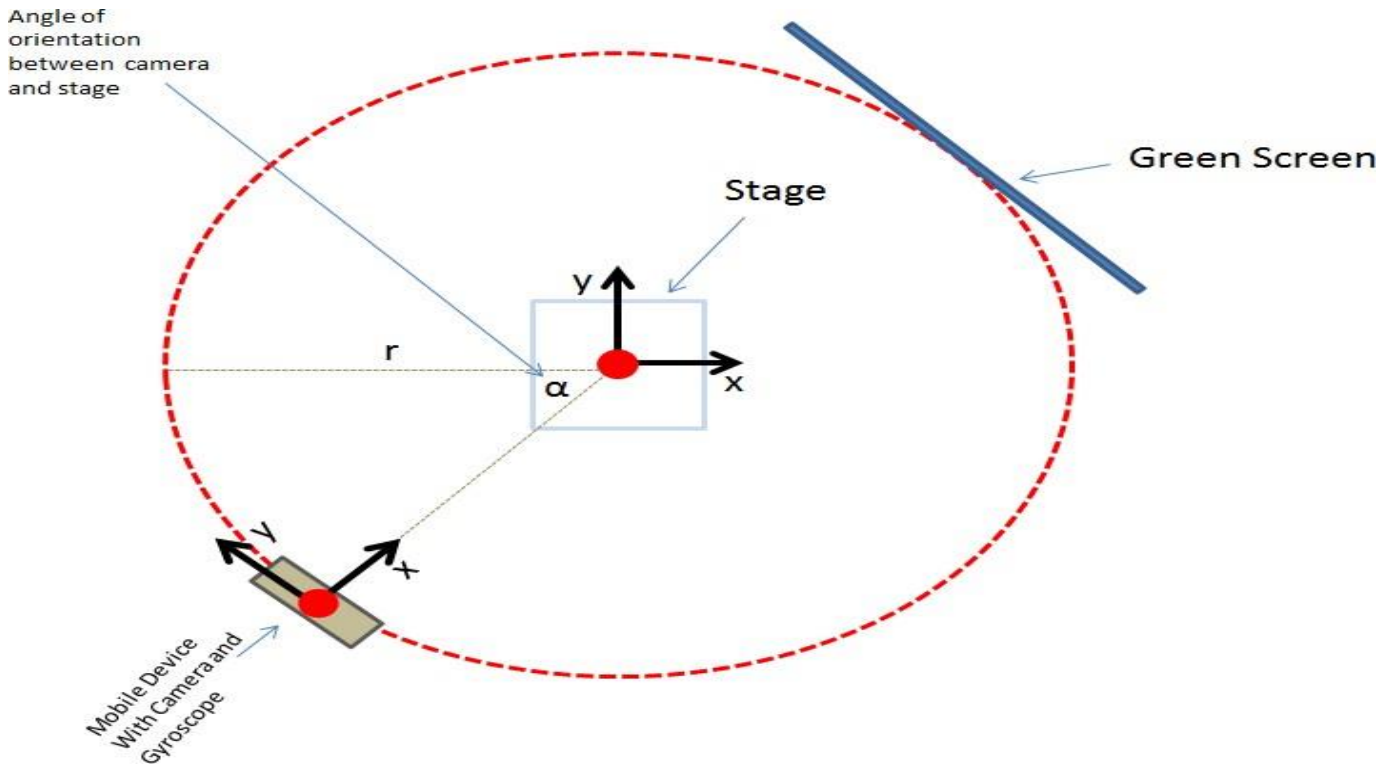


Figure 5.5 : During the experiment, multiple images will be obtained by the camera device at various points on a circle surrounding the stage. This angle shall be referred to as α.

The camera shall be place at equally spaced points on the circle surrounding the stage (see Figure 5.4). That is to say, $\Delta\alpha$ will be constant. The goal of the rotation about the state at intervals of $\Delta\alpha$ is to provide the 3D reconstruction algorithm with scene information at 360^{o}. The amount of data collected will be determined empirically such that the images obtained provide enough data to reconstruct a 3D scene with a reasonable amount of processing.

2. Generating 3D Information

The second portion of the project will involve generating a depth map using the 2D images collected at each α (i.e. at different points on the circle in Figure 3). It may be possible that only a small set of the images are used during 3D reconstruction . The depth generation algorithms are roughly classified into three categories which utilize different kinds of depth cues : the binocular,

monocular and pictorial depth cues. In this project, pictorial depth cues will be explored. Other conventional depth estimation methods include: the use of color channels, vanishing points, image warping , edge information , and object classification.

The approach to obtain the depth map will involve the following idea: items in the scene closest to the observer will translate much less than objects furthest away from the observer as θ varies. For key- point detection and image fusion, techniques will be employed. The goal is to represent the depth map in the form of a 256 bit image such that many of the techniques learned can be used to process the depth map images.

3. Visualization

The final portion of the project will involve rendering the 3D data. In this step, the depth information generated in Step 2 will be rendered using OpenGL. GLUT will be used to provide a simple method for a user to observe the rendered 3D object.

PSEUDO CODE :

Input: P_1,..,P_m – detected planes

F_1,..,F_n – detected features of images 1,..,n

C_1,..,C_n – camera poses of images 1,..,n

Output: M – reconstructed 3D model

G – 3d topology graph

```
for each P in {P_1,..,P_m}
//pairwise matching
    for i=1 to n-1
        backproject features F_i and F_(i+1) to plane P
        for each feature in F_i find most similar in F_(i+1)
        for each feature in F_(i+1) find most similar in F_i
        cross check
    end
    //generate feature hypotheses
    Build feature trails based on the pairwise matches
    for each trail
        if trail has at least 4 observations
            create 3D feature hypothesis
        end
    end
    //establishment and verification topology
    for each 3D feature pair
        if at least 3 associated 2D observations are connected
            connect 3D features in the topology graph
        end
    end
    remove unconnected 3D feature hypotheses
end
return 3D model M and topology graph G
```

CHAPTER 6

CHAPER 6

SYSTEM IMPLEMENTATION

6.1 CLIENT SIDE CODING (CV_Project.m) :

```
%% Calibration Parameters
% Left RGB Camera Intrinsic Parameters
fx_rgb_left = 517.57040; %546.13865;
fy_rgb_left = 518.73475; %535.59453;
cx_rgb_left = 320; %264.48866; %
cy_rgb_left = 240; %220.24823; %

% Right RGB Camera Intrinsic Parameters
fx_rgb_right = 520.08735; %516.06233;
fy_rgb_right = 509.26543; %509.59230;
cx_rgb_right = 320; %259.49525; %
cy_rgb_right = 240; %234.21632 %274.55578;

% Kinect Depth camera parameters
fx_d =  5.7616540758591043e+02;
fy_d = 5.7375619782082447e+02;
cx_d = 3.2442516903961865e+02;
cy_d = 2.3584766381177013e+02;

% Extrinsic parameters between RGB and Depth camera for
Kinect V1
% Rotation matrix
R =  inv([  9.9998579449446667e-01, 3.4203777687649762e-03,
-4.0880099301915437e-03;
    -3.4291385577729263e-03, 9.9999183503355726e-01, -
2.1379604698021303e-03;
    4.0806639192662465e-03, 2.1519484514690057e-03,
9.9998935859330040e-01]);

% Translation vector.
T = -[  2.2142187053089738e-02, -1.4391632009665779e-04, -
7.9356552371601212e-03 ]';

%% Load last pair of images for 3D reconstruction later
% Load Left RGB and Depth scene image
left_rgb_img = imread('LeftRgbobjects_scene21.png');
left_depth_img = imread('LeftDepthobjects_scene21.png');

% Load Right RGB and Depth scene image
right_rgb_img = imread('RightRgbobjects_scene21.png');
right_depth_img = imread('RightDedpthobjects_scene21.png');
```

```
% Lets have a look at the images
figure();
subplot(1,2,1)
imshow(left_rgb_img );
subplot(1,2,2)
imshow(left_depth_img,[0.8,3.0],'Colormap', jet(255));
figure();
subplot(1,2,1)
imshow(right_rgb_img);
subplot(1,2,2)
imshow(right_depth_img,[0.8,3.0],'Colormap', jet(255));
%% Step1: Back-project every 2D point from depth image into
3D space
% Left Camera:
[n_row, n_col] = size(left_depth_img);
left_depth_map = zeros(n_row, n_col, 3);
for x = 1:n_row
    for y = 1:n_col
        % Back-project every 2D point from depth image into
3D space
        left_depth_map(x, y, 1) = (y - cx_d) *
double(left_depth_img(x, y)) / fx_d;
        left_depth_map(x, y, 2) = (x - cy_d) *
double(left_depth_img(x, y)) / fy_d;
        left_depth_map(x, y, 3) = double(left_depth_img(x,
y));

    end
end

% Right Camera:
[n_row, n_col] = size(right_depth_img);
right_depth_map = zeros(n_row, n_col, 3);
for x = 1:n_row
    for y = 1:n_col
        % Back-project every 2D point from depth image into
3D space
        right_depth_map(x, y, 1) = (y - cx_d) *
double(right_depth_img(x, y)) / fx_d;
        right_depth_map(x, y, 2) = (x - cy_d) *
double(right_depth_img(x, y)) / fy_d;
        right_depth_map(x, y, 3) = double(right_depth_img(x,
y));
    end
end
```

```
%% Step 2: Project every transformed 3D point into the RGB
image
% Left Camera:
index = 0;
pc_rbg_left = zeros(n_row * n_col, 3);
pc_rbg_left_col = zeros(n_row * n_col, 3);
for x = 1:n_row
    for y = 1:n_col
        if left_depth_map(x, y, 3) > 0
            % Apply transformation between Depth and RGB
camera
            XYZ_depth = [left_depth_map(x, y, 1),
left_depth_map(x, y, 2), left_depth_map(x, y, 3)];
            XYZ_rgb = R * XYZ_depth' + T;
            % Project every transformed 3D point into the
RGB image
            x_rgb = round(fx_rgb_left * XYZ_rgb(1) /
XYZ_rgb(3) + cx_rgb_left);
            y_rgb = round(fy_rgb_left * XYZ_rgb(2) /
XYZ_rgb(3) + cy_rgb_left);

            if x_rgb > 0 && y_rgb > 0 && x_rgb < n_col  &&
y_rgb < n_row
                % look up color corrsponding to a given 3D
point
                color = left_rgb_img(y_rgb, x_rgb, :);
                index = index + 1;
                % create new point cloud with color
                pc_rbg_left(index,:) =  XYZ_rgb;
                pc_rbg_left_col(index, :) = color;
            end
        end
    end
end
pc_rbg_left = pc_rbg_left(1:index,:);
pc_rbg_left_col = pc_rbg_left_col (1:index,:);

figure()

pcshow(left_depth_map)
figure()
pcshow(pc_rbg_left, pc_rbg_left_col / 256.0)


% Right camera:
index = 0;
pc_rbg_right = zeros(n_row * n_col, 3);
pc_rbg_right_col = zeros(n_row * n_col, 3);
for x = 1:n_row
```

```
    for y = 1:n_col
         if right_depth_map(x, y, 3) > 0
            % Apply transformation between Depth and RGB
camera
            XYZ_depth = [right_depth_map(x, y, 1),
right_depth_map(x, y, 2), right_depth_map(x, y, 3)];
            XYZ_rgb = R * XYZ_depth' + T;
            % Project every transformed 3D point into the
RGB image
            x_rgb = round(fx_rgb_right * XYZ_rgb(1) /
XYZ_rgb(3) + cx_rgb_right);
            y_rgb = round(fy_rgb_right * XYZ_rgb(2) /
XYZ_rgb(3) + cy_rgb_right);

            if x_rgb > 0 && y_rgb > 0 && x_rgb < n_col  &&
y_rgb < n_row
                % look up color corrsponding to a given 3D
point
                color = right_rgb_img(y_rgb, x_rgb, :);
                index = index + 1;
                % create new point cloud with color
                pc_rbg_right(index,:) =  XYZ_rgb;
                pc_rbg_right_col(index, :) = color;
            end
        end
    end
end
pc_rbg_right = pc_rbg_right(1:index,:);
pc_rbg_right_col = pc_rbg_right_col (1:index,:);

figure()

pcshow(right_depth_map)
figure()
pcshow(pc_rbg_right, pc_rbg_right_col / 256.0)


%% Step3: Transformation from left 3D coordinate to right 3D
coordinate camera frame

% Transformation vectors between left and right RBG cameras
T_rbg = [ -954.57277, 99.75455, 462.56569] - [66.49332,
16.62784, 128.72005*0.5]; %[-1034.35109, 66.12586,
410.88503] * 0.9 ;
R_rbg = [ -0.03190 0.73381 -0.15668 ]; %[ -0.01157 0.68566 -
0.10888 ] ;
R_rbg = rotationVectorToMatrix(R_rbg);
R_rbg = inv(R_rbg);
```

```
% Calculate corresponding XYZ translated from left to right
len = length(pc_rbg_left);
pc_rbg_left_to_right = zeros(len, 3);
for i=1:len
    pc_rbg_left_to_right(i, :) = (R_rbg *  pc_rbg_left(i,
:)' + T_rbg')';
end

% Merge point cloud 3D from both left and right cameras:
figure()
pcshow([pc_rbg_left_to_right; pc_rbg_right],
[pc_rbg_left_col/256.0; pc_rbg_right_col/256])

ptime=[20 40 60 80 100];
tput= randi([15999 16100],1,5)
figure();
plot(ptime,tput,'b--o');
xlim([20, 100]);ylim([0,18000]);
xlabel('Neural Network Estimation)')
ylabel('throughput(MBPS)')
title('Throughput Vs Neural Network Estimation');


speed=[0 10 20 30 40 50 60]
tp=randi([45 60],1,7)
figure();
plot(speed,tp,'b--o');
xlim([0, 80]);ylim([30,90]);
xlabel('Number of Images(N)')
ylabel('Average Segmentation)')
title('(Number of Images(N) Vs Average Segmentation');


con=[0 10 20 30 40 50]
tp=randi([0 300],1,6)
figure();
plot(con,tp,'b--o');
xlim([0, 60]);ylim([0,500]);
xlabel('Number of Images(N)')
ylabel('throughput(MBPS)')
title('Number of Images Vs Throughput');
```

6.2 SERVER SIDE CODING :

```
close all;clc;clear;% adding pathsfsep = '/';Path1 =
sprintf(['..' fsep '..' fsep 'Wrappers/MATLAB/compiled'],
1i);Path2 = sprintf(['..' fsep '..' fsep
'Wrappers/MATLAB/supplem'], 1i);addpath(Path1);
addpath(Path2); % Define phantom dimensionsN = 256; % x-y-z
size (cubic image) % define parametersparamsObject.Ob =
'gaussian';paramsObject.C0 = 1; paramsObject.x0 = -
0.25;paramsObject.y0 = 0.1;paramsObject.z0 =
0.0;paramsObject.a = 0.2;paramsObject.b =
0.35;paramsObject.c = 0.7;paramsObject.phi1 =
30;paramsObject.phi2 = 60;paramsObject.phi3 = -25; %
generate 3D phantom [N x N x N]:tic; [G] =
TomoP3DObject(paramsObject.Ob,paramsObject.C0,
paramsObject.x0, paramsObject.y0, paramsObject.z0,
paramsObject.a, paramsObject.b, paramsObject.c,
paramsObject.phi1, paramsObject.phi2, paramsObject.phi3, N);
toc; % check 3 projectionsfigure; slice =
round(0.5*N);subplot(1,3,1); imagesc(G(:,:,slice), [0 1]);
daspect([1 1 1]); colormap hot; title('Axial
Slice');subplot(1,3,2); imagesc(squeeze(G(:,slice,:)), [0
1]); daspect([1 1 1]); colormap hot; title('Y-
Slice');subplot(1,3,3); imagesc(squeeze(G(slice,:,:)), [0
1]); daspect([1 1 1]); colormap hot; title('X-Slice');


clear
    close all
    fsep = '/';
    Path1 = sprintf(['..' fsep '..' fsep
'Wrappers/MATLAB/compiled'], 1i);
    Path2 = sprintf(['..' fsep '..' fsep
'Wrappers/MATLAB/supplem'], 1i);
    addpath(Path1); addpath(Path2);

    % generate 3D phantom (modify your PATH bellow):
    curDir   = pwd;
    mainDir  = fileparts(curDir);
    pathtoLibrary = sprintf(['..' fsep '..' fsep
'PhantomLibrary' fsep 'models' fsep 'Phantom3DLibrary.dat'],
1i);
    pathTP = strcat(mainDir, pathtoLibrary); % path to
TomoPhantom parameters file

    disp('Using TomoPhantom to generate 3D phantom');
    N = 256;
    ModelNo = 13;
```

```
    [G] = TomoP3DModel(ModelNo,N,pathTP);
    figure;
    slice = round(0.5*N);
    subplot(1,3,1); imagesc(G(:,:,slice), [0 1]); daspect([1
1 1]); colormap hot; title('Axial Slice');
    subplot(1,3,2); imagesc(squeeze(G(:,slice,:)), [0 1]);
daspect([1 1 1]); colormap hot; title('Y-Slice');
    subplot(1,3,3); imagesc(squeeze(G(slice,:,:)), [0 1]);
daspect([1 1 1]); colormap hot; title('X-Slice');

    angles_num = round(0.5*pi*N); % angles number
    angles = linspace(0,179.99,angles_num); % projection
angles
    Horiz_det = round(sqrt(2)*N); % detector column count
(horizontal)
    Vert_det = N; % detector row count (vertical) (no reason
for it to be > N, so fixed)
    %%
    disp('Using astra-toolbox (GPU) to generate 3D
projection data');
    proj3D_astra = sino3Dastra(G, angles, Vert_det,
Horiz_det);
    %%
    disp('Using TomoPhantom to generate 3D projection
data');
    proj3D_tomophant = TomoP3DModelSino(ModelNo, Vert_det,
Horiz_det, N, single(angles), pathTP);
    %%

    % comparing 2D analytical projections with ASTRA
numerical projections
    slice2 = 160;
    compar_im = squeeze(proj3D_astra(:,slice2,:));
    sel_im = proj3D_tomophant(:,:,slice2);
    disp(norm(sel_im(:) - compar_im(:))/norm(compar_im(:)))

    % figure;imshow(squeeze(sino_tomophan3D(:,150,:)), []);
    max_val = 100;
    figure;
    subplot(1,3,1); imagesc(sel_im, [0 max_val]);
title('Analytical projection');
    subplot(1,3,2); imagesc(compar_im, [0 max_val]);
title('Numerical projection');
    subplot(1,3,3); imagesc(abs(sel_im - compar_im), [0
max_val]); title('image error');

    figure;
```

```
    subplot(1,3,1);
imagesc(squeeze(proj3D_astra(:,slice2,:)), [0 max_val]);
title('Astra projection');
    subplot(1,3,2);
imagesc(squeeze(proj3D_astra(slice2,:,:)), [0 max_val]);
title('Tangentogram');
    subplot(1,3,3);
imagesc(squeeze(proj3D_astra(:,:,slice2)), [0 max_val]);
title('Sinogram');

    figure;
    subplot(1,3,1);
imagesc(squeeze(proj3D_tomophant(:,:,slice2)), [0 max_val]);
title('Analytical projection');
    subplot(1,3,2);
imagesc(squeeze(proj3D_tomophant(slice2,:,:))', [0
max_val]); title('Tangentogram');
    subplot(1,3,3);
imagesc(squeeze(proj3D_tomophant(:,slice2,:)), [0 max_val]);
title('Sinogram');
    %%
    disp('Reconstructing data using ASTRA toolbox (CGLS
method)');
    reconstructon = rec3Dastra(proj3D_tomophant, angles,
Vert_det, Horiz_det);
    figure;
    slice = round(0.5*N);
    subplot(1,3,1); imagesc(reconstructon(:,:,slice), [0
1]); daspect([1 1 1]); colormap hot; title('Axial Slice
(reconstruction)');
    subplot(1,3,2);
imagesc(squeeze(reconstructon(:,slice,:)), [0 1]);
daspect([1 1 1]); colormap hot; title('Y-Slice');
    subplot(1,3,3);
imagesc(squeeze(reconstructon(slice,:,:)), [0 1]);
daspect([1 1 1]); colormap hot; title('X-Slice');

close all;clc;clear;
    % adding paths
    fsep = '/';
    Path1 = sprintf(['..' fsep '..' fsep
'Wrappers/MATLAB/compiled'], 1i);
    Path2 = sprintf(['..' fsep '..' fsep
'Wrappers/MATLAB/supplem'], 1i);
    addpath(Path1); addpath(Path2);

    ModelNo = 4; % Select a model from Phantom2DLibrary.dat
    % Define phantom dimensions
    N = 512; % x-y size (squared image)
```

```
    % Generate 2D phantom:
    curDir   = pwd;
    mainDir  = fileparts(curDir);
    pathtoLibrary = sprintf(['..' fsep '..' fsep
'PhantomLibrary' fsep 'models' fsep 'Phantom2DLibrary.dat'],
1i);
    pathTP = strcat(mainDir, pathtoLibrary); % path to
TomoPhantom parameters file
    [G] = TomoP2DModel(ModelNo,N,pathTP);
    figure; imagesc(G, [0 1]); daspect([1 1 1]); colormap
hot;
    %%
    fprintf('%s \n', 'Generating sinogram analytically and
numerically using ASTRA-toolbox...');
    angles = linspace(0,180,N); % projection angles
    P = round(sqrt(2)*N); %detectors dimension
    % generate 2D analytical parallel beam sinogram (note
the 'astra' opton)
    [F_a] = TomoP2DModelSino(ModelNo, N, P, single(angles),
pathTP, 'astra');
    [F_num_astra] = sino2Dastra(G, (angles*pi/180), P, N,
'cpu');

    % calculate residiual norm (the error is expected since
projection models not the same)
    err_diff = norm(F_a(:) -
F_num_astra(:))./norm(F_num_astra(:));
    fprintf('%s %.4f\n', 'NMSE for sino residuals:',
err_diff);

    figure;
    subplot(1,2,1); imshow(F_a, []); title('Analytical
Sinogram');
    subplot(1,2,2); imshow(F_num_astra, []);
title('Numerical Sinogram (ASTRA)');
    %%
    fprintf('%s \n', 'Adding noise and artifacts to
analytical sinogram...');
    dose =  1e4; % photon flux (controls noise level)
    [sino_noise] = add_noise(F_a, dose, 'Poisson'); % adding
Poisson noise
    [sino_noise_zingers] = add_zingers(sino_noise, 0.5, 10);
% adding zingers
    [sino_noise_zingers_stripes] =
add_stripes(sino_noise_zingers, 1, 1); % adding stripes
```

```
    figure; imshow(sino_noise_zingers_stripes, []);
title('Analytical Sinogram degraded with noise and
artifacts');
    %%
    fprintf('%s \n', 'Reconstruction using ASTRA-toolbox
(FBP)...');

    rec_an = rec2Dastra(sino_noise_zingers_stripes,
(angles*pi/180), P, N, 'cpu');
    rec_num = rec2Dastra(F_num_astra, (angles*pi/180), P, N,
'cpu');

    figure;
    subplot(1,2,1); imagesc(rec_an, [0 1]); daspect([1 1
1]); colormap hot; title('Degraded analytical sinogram
reconstruction');
    subplot(1,2,2); imagesc(rec_num, [0 1]); daspect([1 1
1]); colormap hot; title('Numerical sinogram
reconstruction');
    %%


close all;clc;clear;
    % adding paths
    fsep = '/';
    Path1 = sprintf(['..' fsep '..' fsep
'Wrappers/MATLAB/compiled'], 1i);
    Path2 = sprintf(['..' fsep '..' fsep
'Wrappers/MATLAB/supplem'], 1i);
    addpath(Path1); addpath(Path2);


    % Define object dimensions
    N = 256; % x-y-z size (cubic image)

    % define parameters
    paramsObject.Ob = 'gaussian';
    paramsObject.C0 = 1;
    paramsObject.x0 = -0.25;
    paramsObject.y0 = 0.1;
    paramsObject.a = 0.2;
    paramsObject.b = 0.35;
    paramsObject.phi1 = 30;

    % generate 2D phantom [N x N ]:
    [G1] = TomoP2DObject(paramsObject.Ob,paramsObject.C0,
paramsObject.x0, paramsObject.y0, paramsObject.a,
paramsObject.b, paramsObject.phi1, N);
```

```
    figure; imagesc(G1, [0 1]); daspect([1 1 1]); colormap
hot;

    % generate corresponding 2D sinogram
    angles = single(linspace(0,180,N)); % projection angles

  P = round(sqrt(2)*N);

    [F1] =
TomoP2DObjectSino(paramsObject.Ob,paramsObject.C0,
paramsObject.x0, paramsObject.y0, paramsObject.a,
paramsObject.b, paramsObject.phi1, N, P, angles);
    figure; imagesc(F1, [0 50]); colormap hot;

    % generate another object
    paramsObject.Ob = 'rectangle';
    paramsObject.C0 = 0.75;
    paramsObject.x0 = 0.25;
    paramsObject.y0 = -0.1;
    paramsObject.a = 0.2;
    paramsObject.b = 0.4;
    paramsObject.phi1 = -45;

    [G2] = TomoP2DObject(paramsObject.Ob,paramsObject.C0,
paramsObject.x0, paramsObject.y0, paramsObject.a,
paramsObject.b, paramsObject.phi1, N);
    figure; imagesc(G2, [0 1]); daspect([1 1 1]); colormap
hot;
    % generate corresponding 2D sinogram
    [F2] =
TomoP2DObjectSino(paramsObject.Ob,paramsObject.C0,
paramsObject.x0, paramsObject.y0, paramsObject.a,
paramsObject.b, paramsObject.phi1, N, P, angles);
    figure; imagesc(F2, [0 50]); colormap hot;

    % build a new model
    G = G1 + G2;
    F = F1 + F2;

    figure;
    subplot(1,2,1); imagesc(G, [0 1]); daspect([1 1 1]);
colormap hot; title('New model');
    subplot(1,2,2); imagesc(F, [0 70]); daspect([1 1 1]);
colormap hot; title('Sinogram');

    % reconstruct the model
    REC = rec2Dastra(F, double(angles*pi/180), P, N, 'cpu');
    figure; imagesc(REC, [0 1]); daspect([1 1 1]); colormap
hot;
```

CHAPTER 7

CHAPTER 7
PERFORMANCE ANALYSIS

7.1. ACCURACY METRICS AND PERFORMANCE CRITERIA

Let X be the ground truth 3D shape and Xˆ the reconstructed one. Below, we discuss some of the accuracy metrics and performance criteria used to 3D reconstruction algorithms.

Accuracy metrics The most commonly used quantitative metrics for evaluating the accuracy of 3D reconstruction algorithms include:

(1) The Mean Squared Error (MSE) :

It is defined as the symmetric surface distance between **the reconstructed shape Xˆ** and the **ground-truth shape X**, i.e.,

$$d(X, X\hat{}\,) = 1\ nX\ X\ p\in X\ d(p, X\hat{}) + 1\ nX\hat{}\ X\ p\hat{}\in X\hat{}\ d(\hat{}p, X).$$

Here, **nX and nXˆ are, respectively, the number of densely sampled points on X and Xˆ**, **and d(p, X) is the distance**, e.g., the L1 or L2 distance, of p to X along the normal direction to X. The smaller this measure is, the better is the reconstruction.

(2) Intersection over Union (IoU):

The IoU measures the ratio of the intersection between the volume of the predicted shape and the volume of the ground-truth, to the union of the two volumes,

i.e**.,** $IoU = V\hat{}\cap V\ V\hat{}\cup V = P\ i\ \{I(V\hat{}\ i >) * I(Vi)\}\ P\ i\ \{I(I(V\hat{}\ i >) + I(Vi))\}$

where **I(·) is the indicator function**, **Vˆ i is the predicted value at the i−th voxel**, **Vi is the ground truth**, and **V is a threshold**. The higher the IoU value, the better is the reconstruction. This metric is suitable for volumetric reconstructions. Thus,

when dealing with surface-based representations, the reconstructed and ground-truth 3D models need to be vowelized.

(3) Mean of Cross Entropy (CE) loss [103]:

It is defined as follows;

CE = − 1 N XN i=1 {pi log ˆpi + (1 − pi) log(1 − pˆi}.

where N is the total number of voxels or points, depending whether using a volumetric or a point-based representation. **p and pˆ are, respectively, the ground-truth and the predicted DEEP LEARNING-BASED 3D OBJECT RECONSTRUCTION - A SURVEY 21 value at the i-voxel or point**. The lower the CE value is, the better is the reconstruction.

(4) Earth Mover Distance (EMD) and Chamfer Distance (CD).

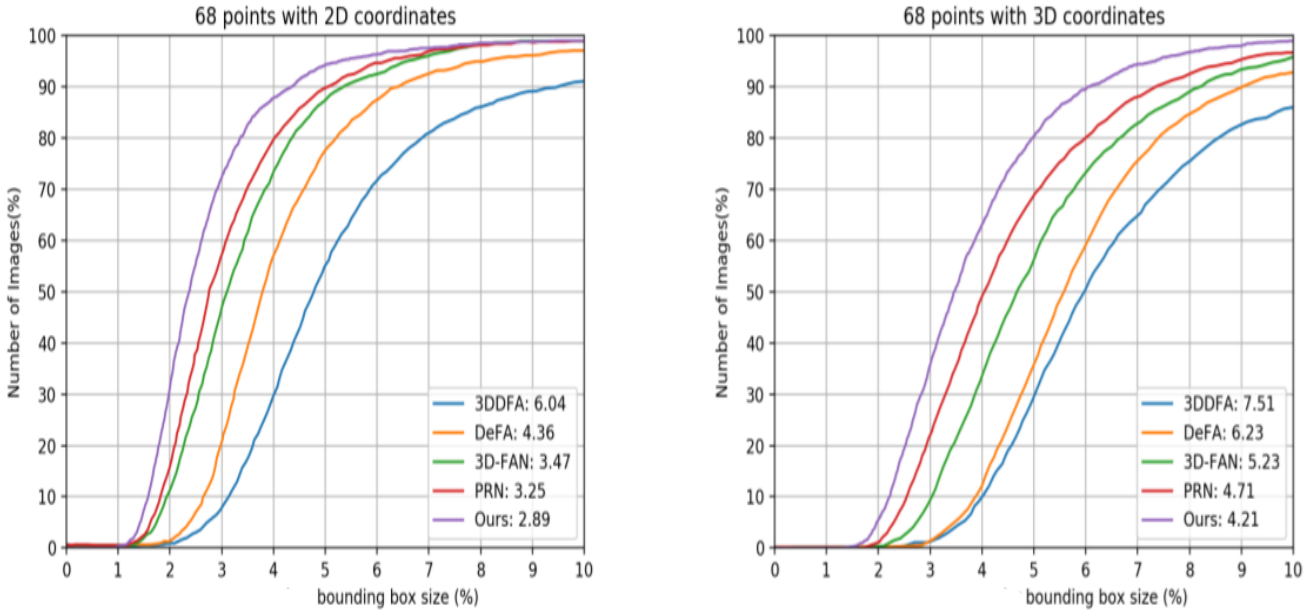


Figure 7.1 a) 68 Points with 2D VS 3D Coordinates

A better formula for calculating the size of a bounding box would be :

(length times width) plus (length plus width)

The bounding box coordinates are a value between 0 and 1 relative to the image size. For example, for an image with 0-degree orientation, a BoundingBox.left value of 0.9 puts the left coordinate close to the right side of the image . To display the box, translate the bounding box coordinate values to pixel points on the image and rotate them to 0 degrees, as shown in each of the following formulas.

ROTATE_0

left = image.width*BoundingBox.Left
top = image.height*BoundingBox.Top

ROTATE_90

left = image.height * (1 - (<face>.BoundingBox.Top <face>.BoundingBox.Height))

top = image.width * <face>.BoundingBox.Left

ROTATE_180

left =image.width(image.width*(<face>.BoundingBox.Left+<face>.BoundingBox.Width))

top = image.height * (1 - (<face>.BoundingBox.Top + <face>.BoundingBox.Height))

ROTATE_270

left = image.height * BoundingBox.top

top = image.width * (1 - BoundingBox.Left - BoundingBox.Width)

Using the following formulas, we calculate the bounding box's width and height as pixel ranges on the image in your code.

The width and height of a bounding box is returned in the BoundingBox.Width and BoundingBox.Height fields. The width and height values range between 0 and 1 relative to the image size. image.width and image.height are pixel values for the width and height of the source image.

box width = image.width * (<face>.BoundingBox.Width)

box height = image.height * (<face>.BoundingBox.Height)

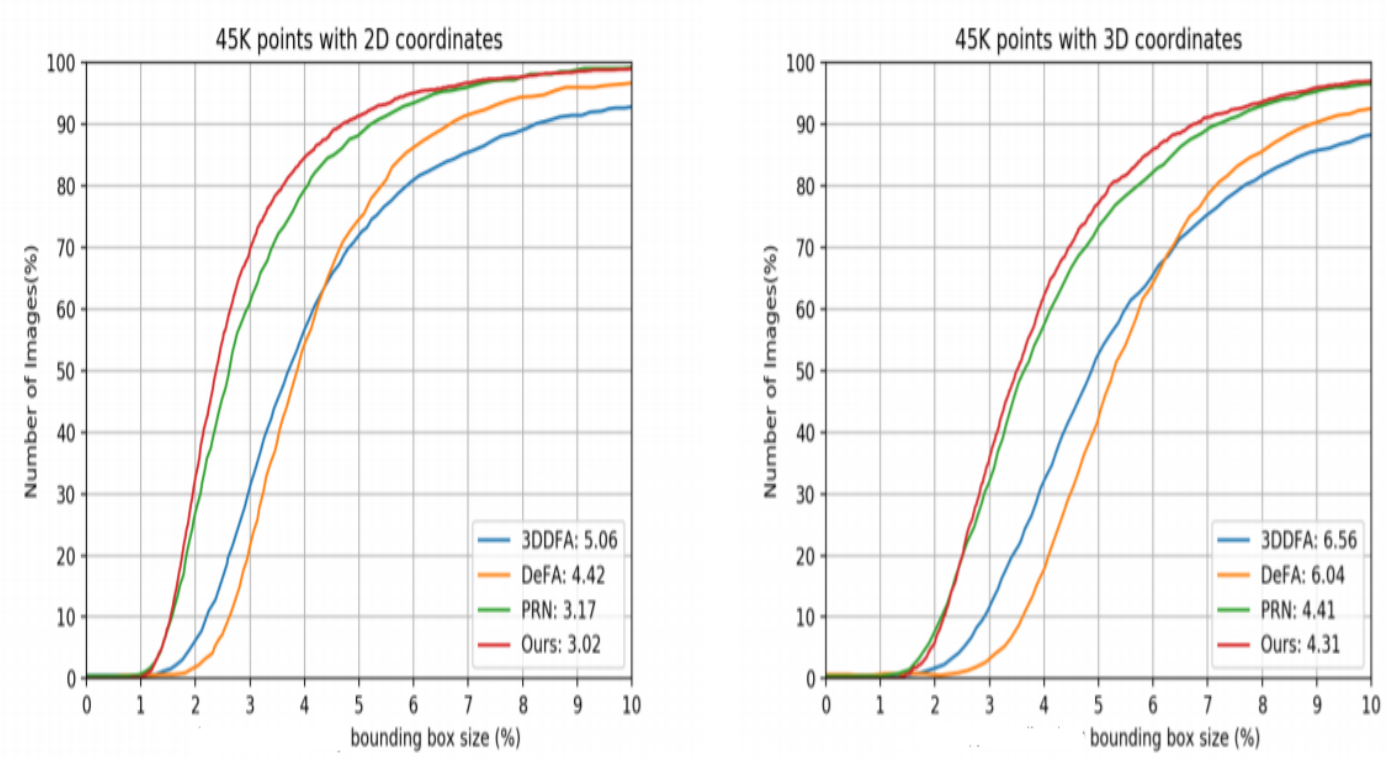


Figure 7.1 b) 45k Points with 2D VS 3D Coordinates

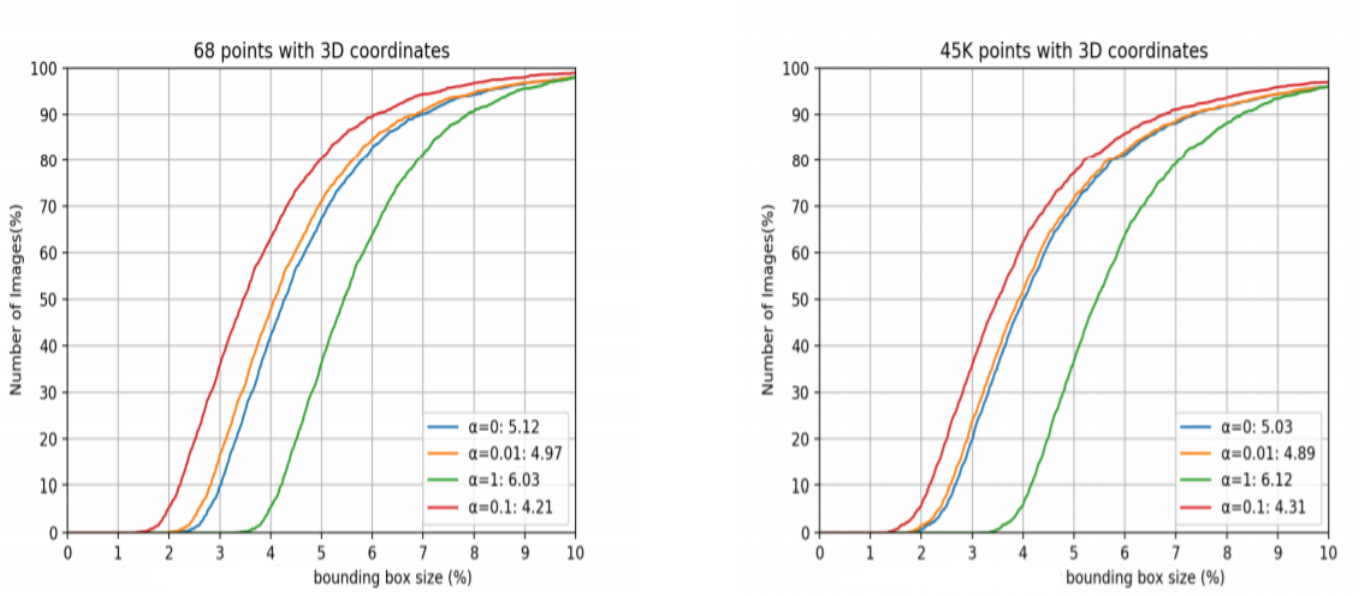


Figure 7.1 c) 68 Points & 45k points with 3D Coordinates

7.2. PERFORMANCE CRITERIA IN ADDITION TO QUANTATIVE METRICS

There are several qualitative aspects that are used to evaluate the efficiency of these methods. This includes:

(1) Degree of 3D supervision:

An important aspect of deep learning-based 3D reconstruction methods is the degree of 3D supervision they require at training. In fact, while obtaining RGB images is easy, obtaining their corresponding ground-truth 3D data is quite challenging. As such, techniques that require minimal or no 3D supervision are usually preferred over those that require ground-truth 3D information during training.

(2) Computation time :

While training can be slow, in general, its is desirable to achieve real time performance at runtime.

(3) Memory footprint :

Deep neural networks have a large number of parameters. Some of them operate on volumes using 3D convolutions. As such, they usually require a large memory storage, which can affect their performance at runtime and limit their usage, the prohibitive memory footprint of such large-scale point clouds is a heavy burden to the server side.

CHAPTER 8

CHAPTER 8
CONCLUSION

8.1 CONCLUSION

A new class of methods is proposed to aim at 2D-to-3D image conversion that is based on the radically different approach of learning from examples. One method that is proposed is based on learning a point mapping from local image attributes to scene-depth. The other method is based on globally estimating the entire depth field of a query directly from a repository of image + depth pairs using nearest-neighbor-based regression. It objectively validates the algorithms' performance against state-of-the-art algorithms. While the local method was outperformed by other algorithms, it is extremely fast as it is, basically, based on table look-up. However, the global method performed better than the state-of-the-art algorithms in terms of cumulative performance across two datasets and two testing methods, and has done so at a fraction of CPU time. Anaglyph images produced by the algorithms result in a comfortable 3D experience but are not completely void of distortions. Clearly, there is room for improvement in the future.

FUTURE ENCHANCEMENTS

With the continuously increasing amount of 3D data on-line and with the rapidly growing computing power in the cloud, the proposed framework seems a promising alternative to operator-assisted 2D-to-3D image and video conversion as well as in Robotics , to enable high level vision for robots to understands and perceive things as humans do.

APPENDICES

APPENDICES

A1. SAMPLE SCREENS

- **INPUT IMAGES :**

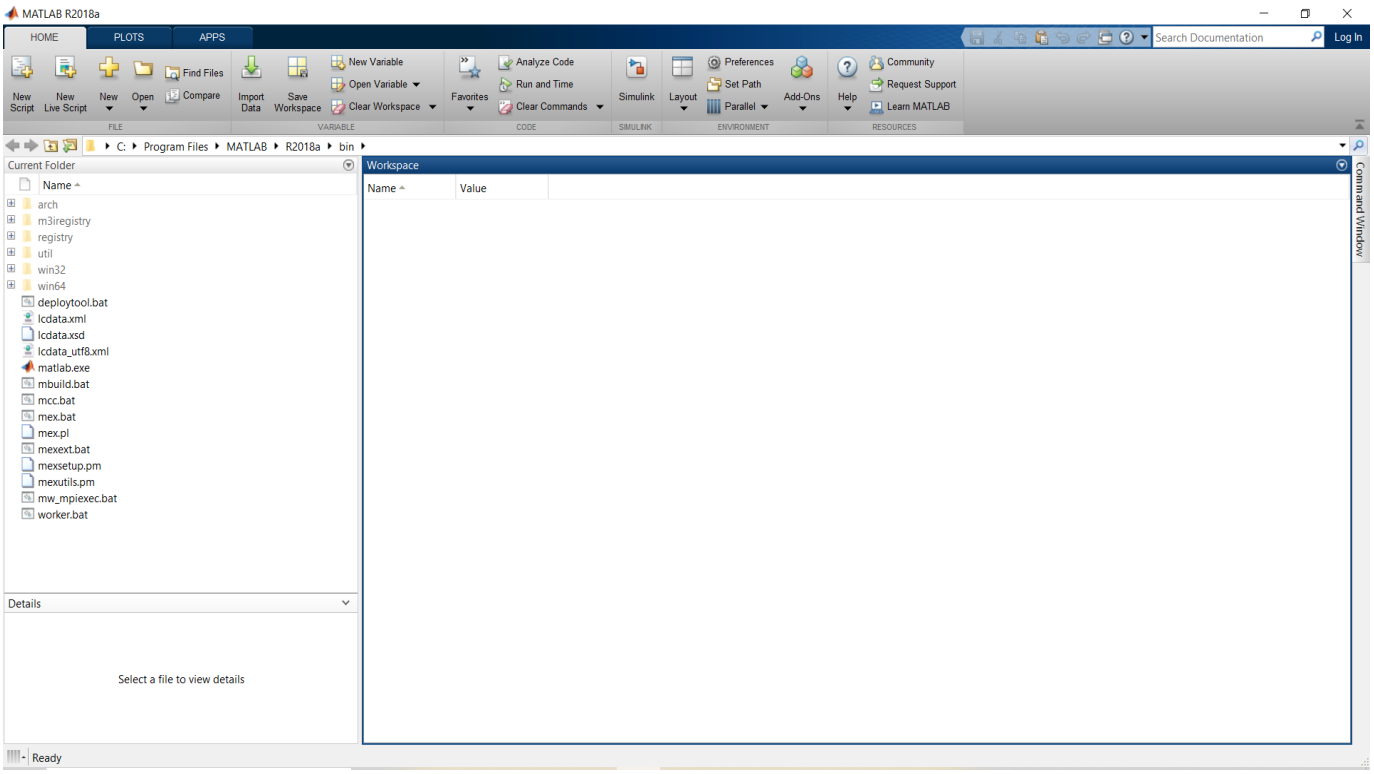


Figure: A1.1 PROJECT SETUP

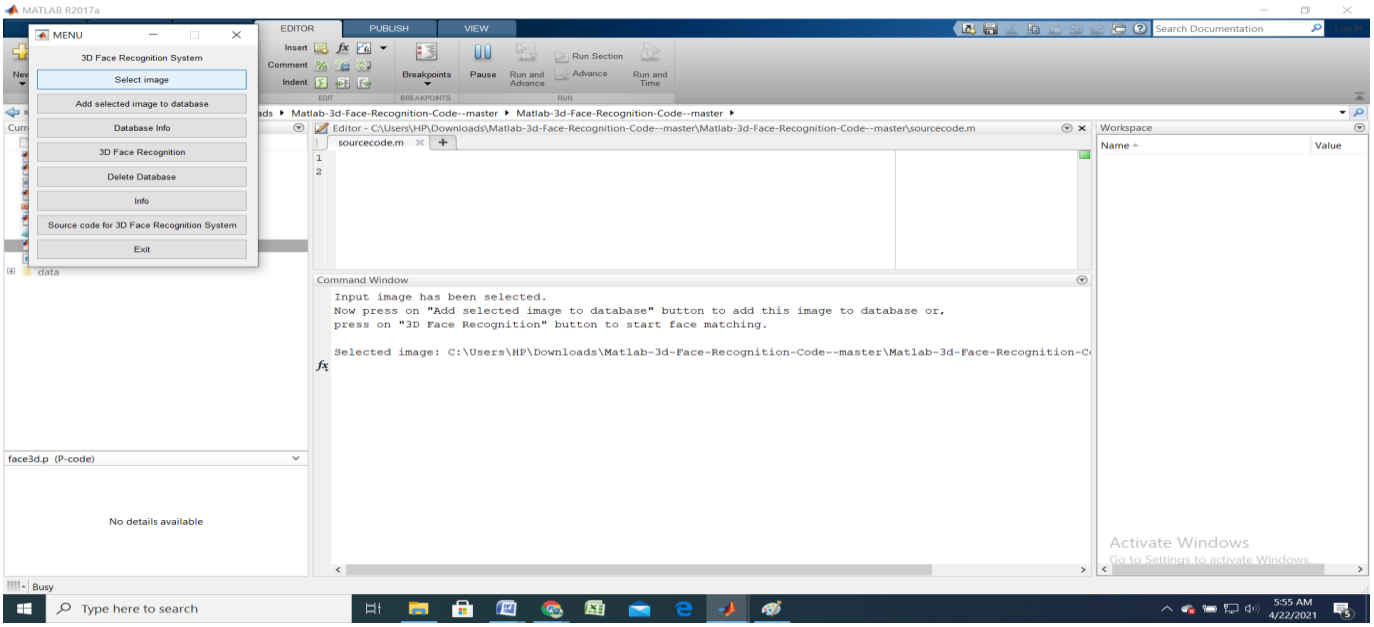


Figure: A1.2 IMAGE SELECTION

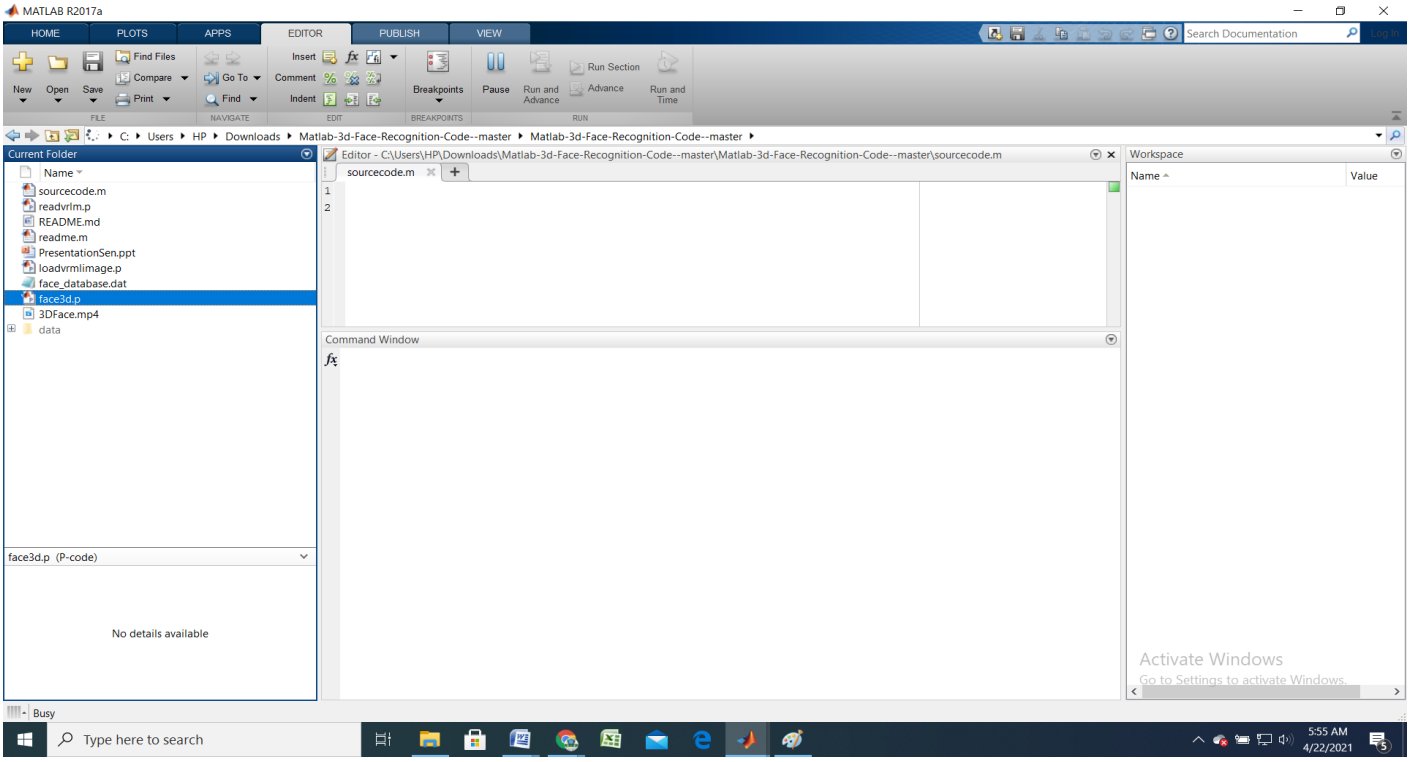


Figure: A1.3 Specifying the Source Code Folder

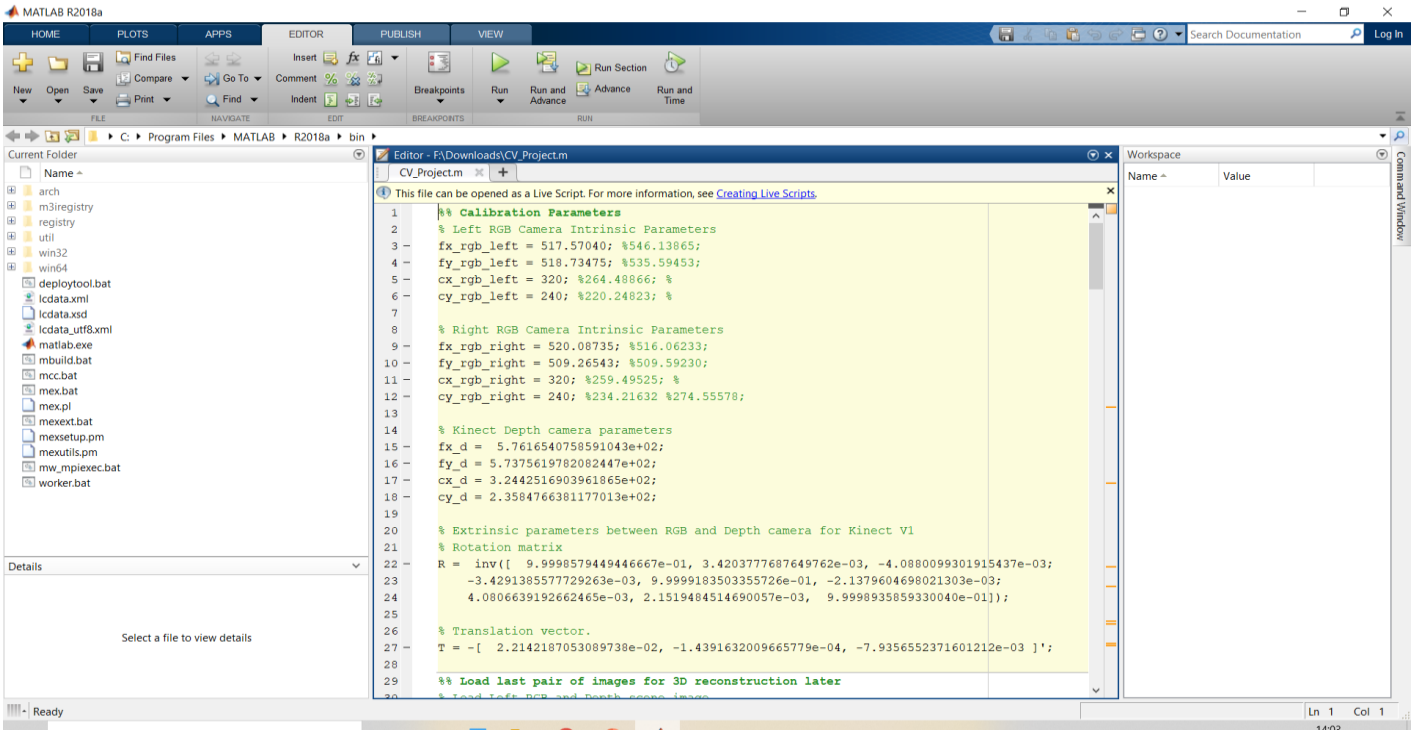
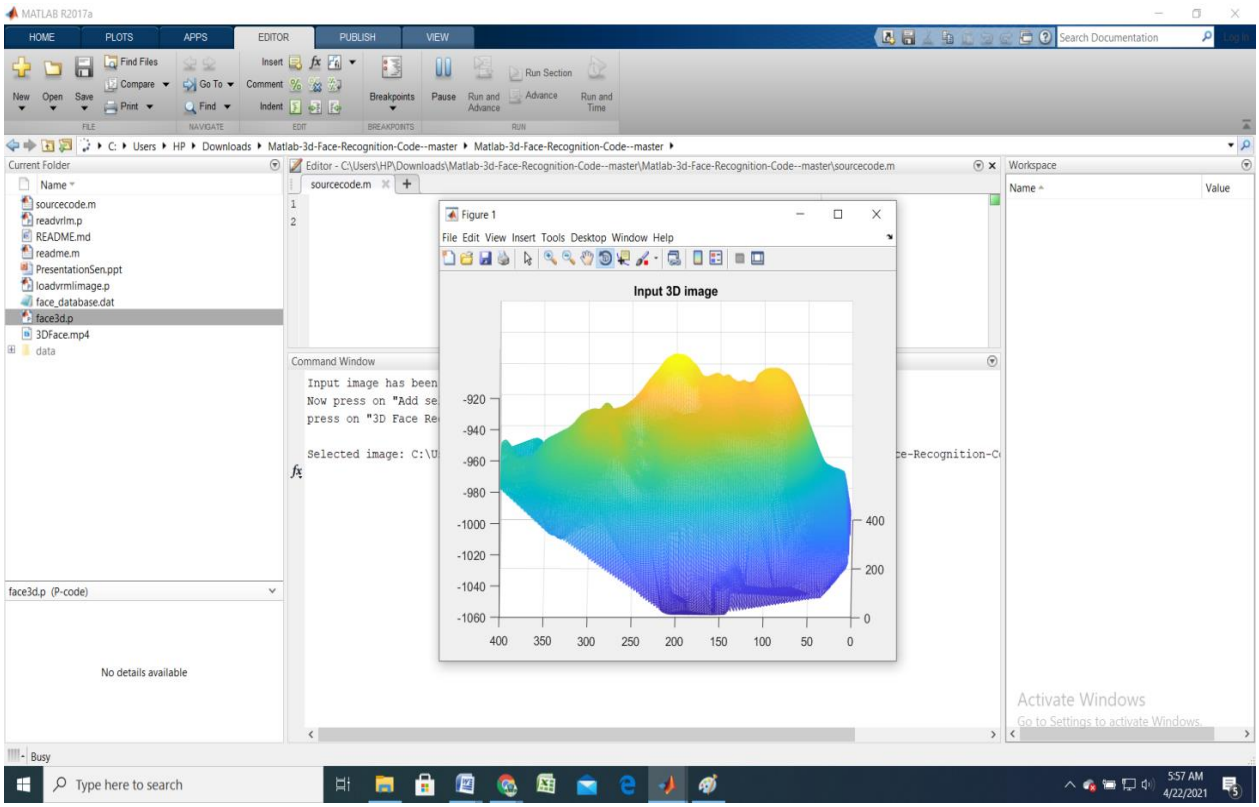
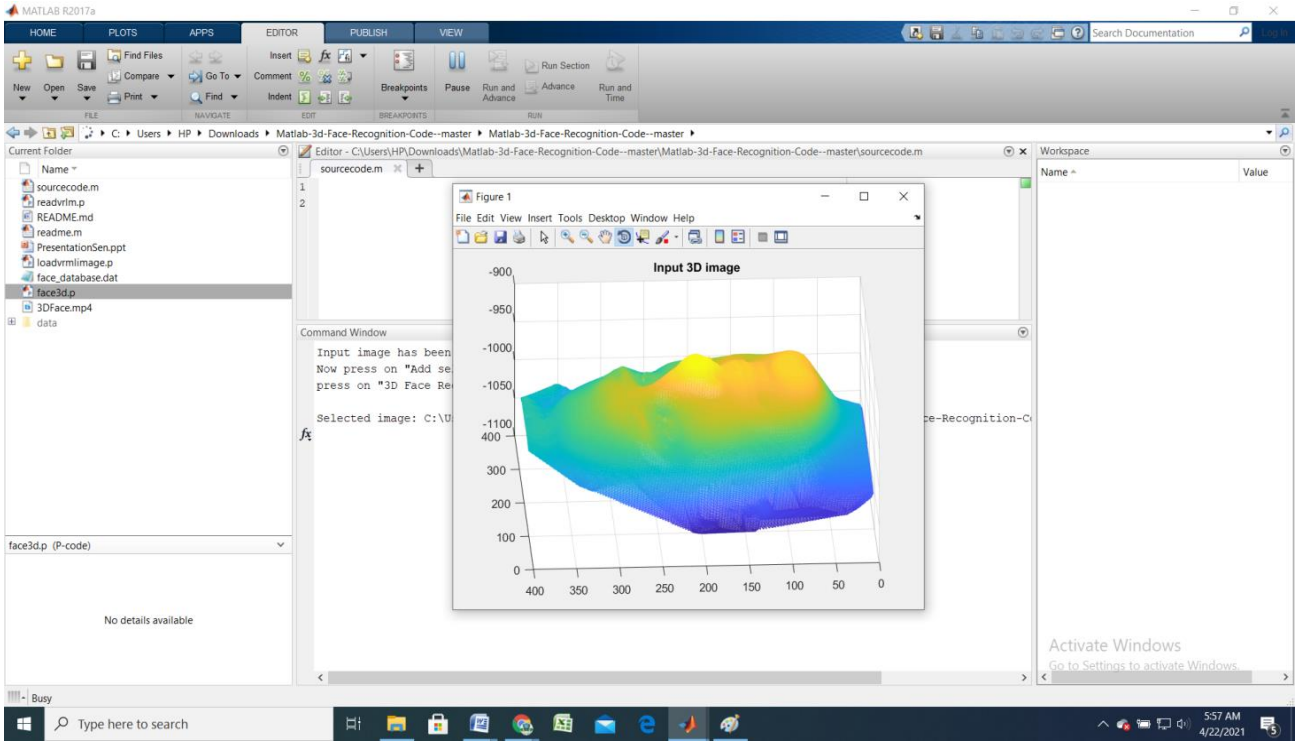


Figure: A1.4 Source Code

- **OUTPUT IMAGES : (For Image 1)**





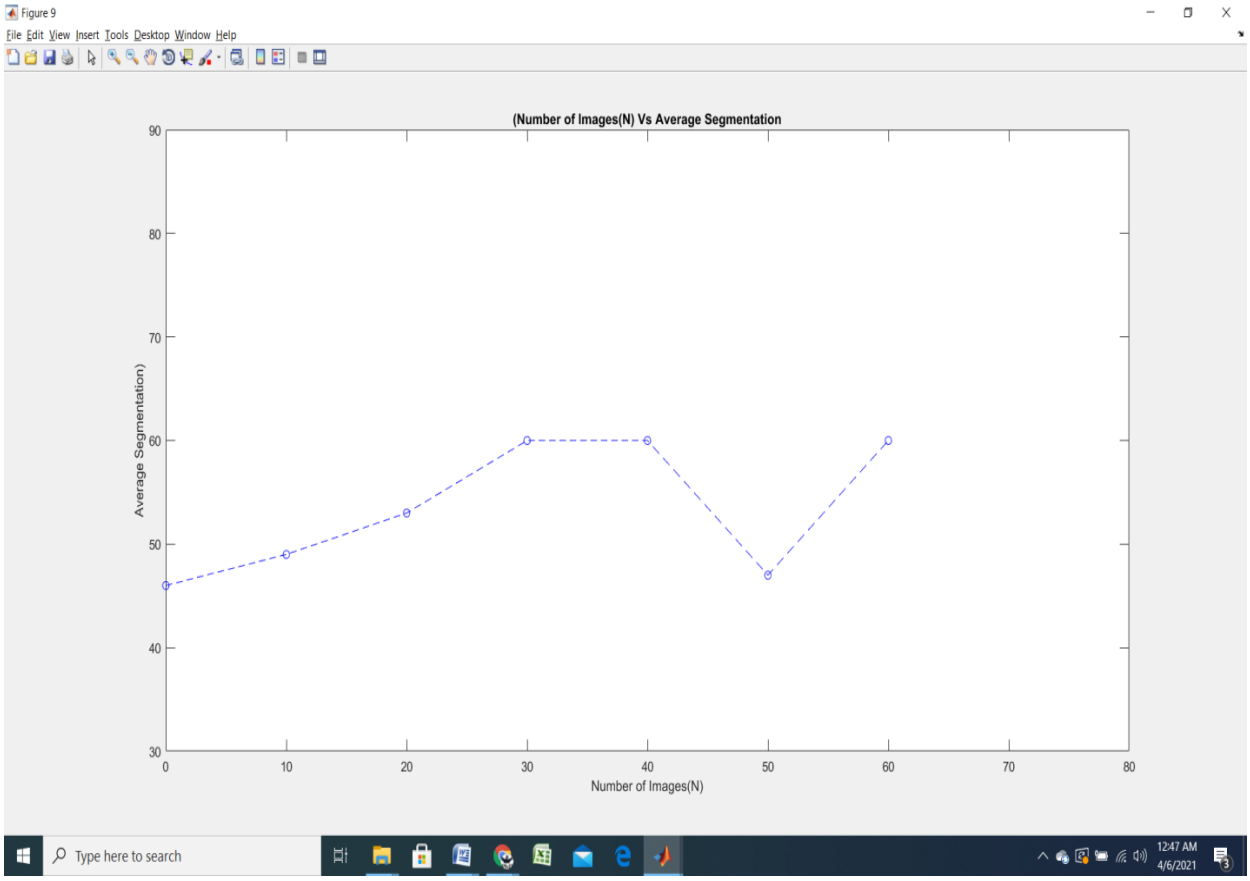


Figure 9
File Edit View Insert Tools Desktop Window Help
(Number of Images(N) Vs Average Segmentation
Average Segmentation)
Number of Images(N)
90
80
70
60
50
40
30
0
10
20
30
40
50
60
70
80
Type here to search
12:47 AM
4/6/2021

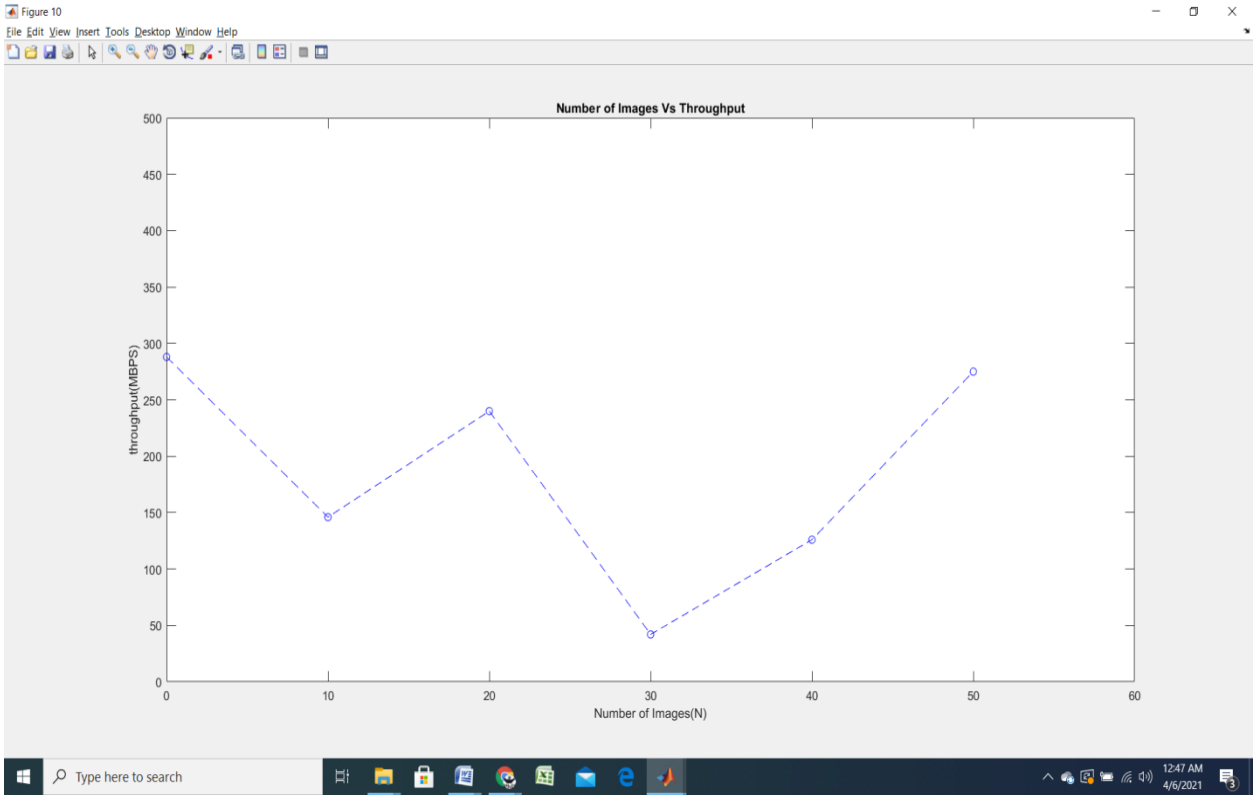


Figure 10
File Edit View Insert Tools Desktop Window Help
Number of Images Vs Throughput
throughput(MBPS)
Number of Images(N)
500
450
400
350
300
250
200
150
100
50
0
0
10
20
30
40
50
60
Type here to search
12:47 AM
4/6/2021

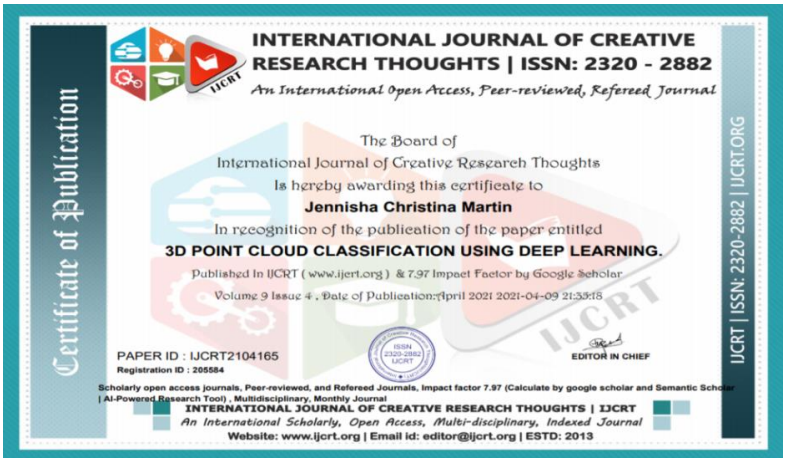
A2. PUBLICATIONS

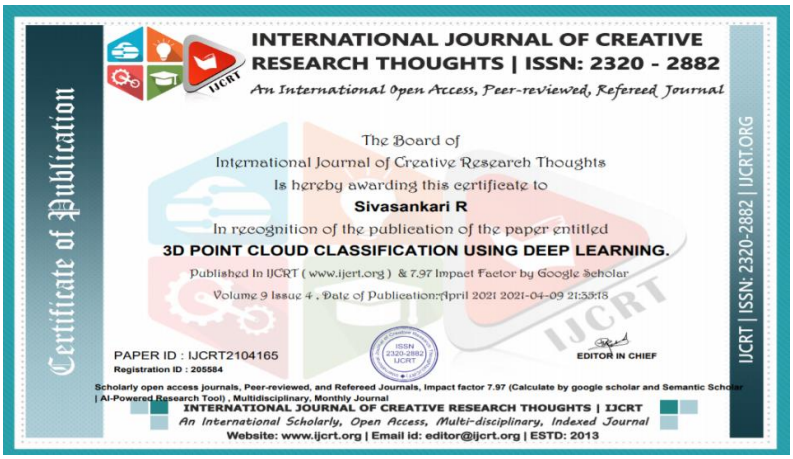
Journal name- International Journal of Creative Research Thoughts

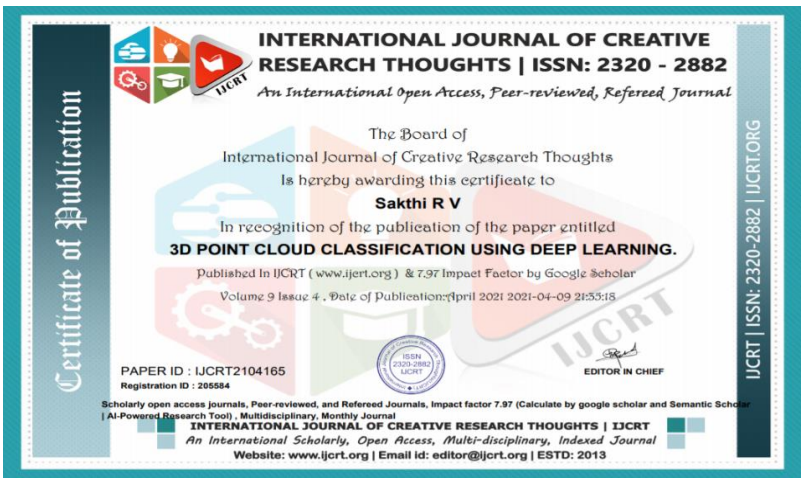
Paper title- 3D Point Cloud Classification Using Deep Learning

Publication issue – Volume 9 , Issue 4 , April 2021

Paper Status- Paper Accepted and it has passed through initial screening Phase







IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

3D POINT CLOUD CLASSIFICATION USING DEEP LEARNING

Sangeetha Kalyanaraman
Associate Prof. Computer Science
Panimalar Engineering College (Affiliated to Anna University)
Chennai, India

Jennisha Christina Martin
B.E Computer Science
Panimalar Engineering College (Affiliated to Anna University)
Chennai, India

Sivasankari R
B.E Computer Science
Panimalar Engineering College (Affiliated to Anna University)
Chennai, India

Sakthi R V
B.E Computer Science Panimalar Engineering College (Affiliated to Anna University)
Chennai , India

Abstract — In the previous couple of years, the provision of 3D content continues to be less than 2D counterpart. Hence many 2D-to-3D image conversion methods are proposed. Methods involving human operators are most successful but also tedious and expensive. Automatic methods that make use of a deterministic 3D scene model, have not yet achieved the identical level of quality for they depend on assumptions that are often violated in practice. Here two types of methods are developed. The primary is predicated on learning some extent mapping from local image/ attributes, like color, spatial position. The second method relies on globally estimating the complete depth map of a question image directly from a repository of 3D images employing a Nearest-neighbour regression type idea. It demonstrates the flexibility and therefore the computational efficiency of the methods on numerous 2D images and discusses their pitfall and benefits

I. INTRODUCTION

It demonstrates the flexibility and also the computational efficiency of the methods on numerous 2D images and discusses their drawbacks and benefits. Point cloud learning has lately attracted increasing attention thanks to its wide applications in many areas, like computer vision, autonomous driving, and robotics. As a dominating technique in AI, deep learning has been successfully wont to solve various 2D vision problems. However, deep learning on point clouds remains in its infancy thanks to the unique challenges faced by the processing of point clouds with deep neural networks. Recently, deep learning on point clouds has become even thriving, with numerous methods being proposed

to deal with different problems during this area. Video segmentation is a vital building block for prime level applications, like scene understanding and interaction analysis. While outstanding results are achieved during this field by the state-of-the-art learning and model-based methods ,they are restricted to certain styles of scenes or require an outsized amount of annotated training data to realize object segmentation in generic scenes. Various deep learning algorithms are proposed for the segmentation of point clouds. The point transformation applied basically in real time because it's supported purely local image attributes, like color, spatial position, and motion at each pixel. A second method is developed that estimates the world depth map of a question image or video frame directly from a repository of 3D images (image + depth pairs or stereopair) employing a nearest- neighbor regression type idea.

The key observation is that among innumerable 3D images available on-line, there likely exist many whose 3D content matches that of a 2Dinput . An assumption is created that two images that are photometrically similar even have similar 3D structure (depth).Given monocular query image Q, assumed to be the left image of a stereopair that's to be computed, relies on the above observation and assumption to "learn" the complete depth from a repository of 3D images. Search for representative depth fields: Find k 3D images within the repository I that have most similar depth to the query image, for instance by performing a k nearest-neighbor (kNN) search employing a metric supported photometric properties. Depth fusion: Combine the k representative depth fields, for instance, by means of median filtering across depth field. Depth smoothing: Process the fused depth field to get rid of spurious variations, while preserving depth

Discontinuities, for instance, by means of cross-bilateral filtering. Stereo rendering: Generate the correct image of a fictitious stereopair using the monocular query image and also the smoothed depth field followed by suitable processing of occlusions and newly-exposed areas. In the previous few years, the supply of 3D content continues to be but 2D counterpart. Hence many 2D-to-3D image conversion methods are proposed. Methods involving human operators are most successful but also time-consuming and costly. Automatic methods, that make use of a deterministic 3D scene model, haven't yet achieved the identical level of quality for they depend on assumptions that are often violated in practice. Here two varieties of methods are developed. The first is predicated on learning some extent mapping from local image/attributes, like color, spatial position. The second method is predicated on globally estimating the whole depth map of a question image directly from a repository of 3D images (image + depth pairs or stereo pairs) employing a nearest-neighbour regression type idea. Point cloud semantic segmentation plays a critical role in autonomous driving, robot navigation, augmented reality and 3D reconstruction. Currently, with the event of deep learning technology, semantic segmentation has made great progress, but it also faces many difficulties. The unordered and unstructured properties of 3D point clouds make it difficult to be presented as 2D images. the most contributions of our work are listed below: We design a brand new network MH Net, which may fully incorporate both local and global multiscale hierarchical features for highly accurate point cloud semantic segmentation.

II. RELATED WORKS

Some of the research works carried out by researches as related to 3D Point Cloud Classification is discussed in the succeeding Sub-sections.

2.1 Yuxing Xie; Jiaojiao Tian; Xiao Xiang Zhu
3D Point Cloud Semantic Segmentation(2019)

Here, the merits are:
1. in color (RGB, multi-spectral) information suitable for large area. High accuracy suitable for large area not affected by weather global data is available compared to ALS complete Building facade information is available 4D Information middle accuracy not affected by the Weather.

And, the Demerits are :
1. Affected by mirror reflection long scanning time Close-range limited accuracy.

2.2 Xiaoli Liang; Zhongliang Fu.
MH Net: Multiscale Hierarchical Network for 3D Point Cloud Semantic Segmentation (2019).

Here, the merits are :
To take fully advantage of the correlations of Propagated information between the different scale coarse layers and the original points, the Local features of each scale are characterized by feature propagation to obtain the features of the original point clouds at the corresponding scale.

And the Demerits are :
Although there is considerable noise points in the segmentation result of MH Net, the number of noise point's only accounts for a small part of the total point clouds. Even If all the noise points are corrected, the overall accuracy of the point clouds will not be greatly improved

2.3 Yulan Guo;Hanyun Wang;Qingyong Hu; Hai Liu ;Li Liu ; Mohammed Bennamoun. Deep Learning for 3D Point Clouds (2020)

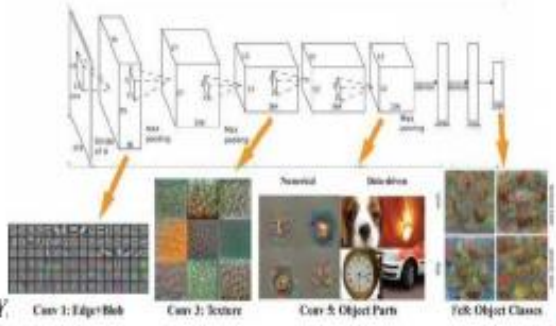
Here the merits are :
A promising solution is to address the raw point clouds with the ConvNets. Since ConvNets
Has the advantage of overlapping during Convolutional operation may benefit the future.

And the demerits are:
It is very expensive and it has included some challenges to over it the irregular network connection will reduce the performance level in the
System.

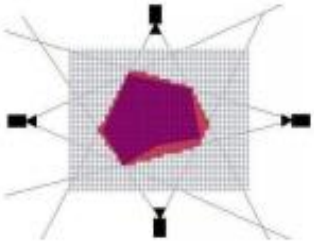
III. METHODOLOGY

In recent years, there has been an increasing demand for applications to monitor the targets related to land-use, using remote sensing images. Proposed the automatic approach to localize and detect building footprints, road networks and vegetation areas. Automatic interpretation of visual data is a comprehensive task in computer vision field. The Deep learning approaches improve the capability of classification in an intelligent way. Deep Learning algorithms gives high accuracy compared to the semi supervised machine learning algorithms.

C. Other methods: voxel coloring, shape carving But requires images to segmented, cameras Calibrated.



SY



IV. FUNCTIONALITIES MODULE I :

SEARCH FOR REPRESENTATIVE DEPTH FIELDS :

Find k 3D images in the repository I that have most similar depth to the query image, for example by performing a k nearest-neighbor(kNN) search using a metric based on photometric Properties.

MODULE 2: *DEPTH FUSION*

Combine the k representative depth fields, for example, by means of median filtering across depth field.

MODULE 3 : *DEPTH SMOOTHING*

Process the fused depth field to remove spurious variations, while preserving depth discontinuities, for example, by means of cross- bilateral filtering.

MODULE 4 : *STEREO RENDERING*

Generate the right image of a fictitious stereopair using the monocular query image and the smoothed depth field followed by suitable processing of occlusions and newly-exposed areas.

TESTING

A. Idea: Find a shape consistent with images
I. Shape: voxel grid
II. For each voxel -> compute occupied or free

B. Shape from Silhouette
I. Extract silhouette images from different views
II. Project voxel on each image
Lies within silhouette – occupied

Performance:

A. Enables estimating 3D shape with fewer viewpoints

i. Extreme case: a single view point

ii. Earliest work: shape from shading

iii. Assumes lambertian reflectance & constant Albedo.

B. Includes prior knowledge on

I. reflectance, shape, viewpoints, II. learnt from data

C. Less reliant on finding accurate feature correspondences

3D Recurrent Unit:

A. LSTM units arranged in 3D grid structure B. A unit receives
I. A feature vector from encoder
II. Hidden states of neighbour by convolution
C. Each unit reconstructs a part of final output.

V. IMPLEMENTATION

Training data:

I. 3D CAD for input images and ground truth occupancy grid.

II. ShapeNets, PASCAL 3D, Online Products
III. images augmented with random crops from Pascal
IV. Viewpoints sampled randomly.

Training:

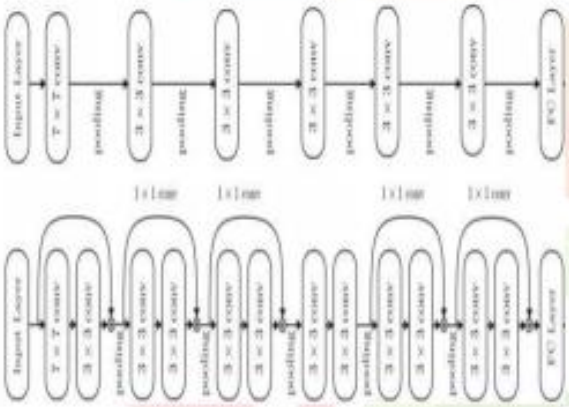
I. Variable length inputs across different mini Batches.
II. Combine single & multi view reconstruction.

Network:

I. Input 127x127, Output 32x32x32
II. 60k iterations, leaky relu (slope: 0.1)
III. Implemented in Theano, Adam for SGD update

Encoder:

A. Encode image into low dim features .
B. Involves standard convolution, pooling and fully connected layers.
C. Uses leaky Relu as activation function D. Also has a residual variant .



Decoder:

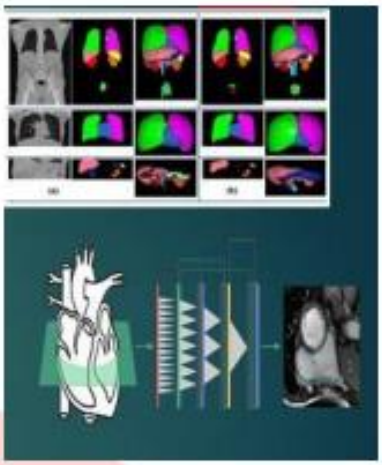
A. The hidden state is passed to the decoder.
B. Decoder up samples to target output resolution,
I. Applies nonlinearities, 3D deconvolution, Unpooling.
C. Output of last activation is converted to occupancy probability,
I. Uses voxel-wise softmax.
D.Finally we minimize the following cross entropy loss and back propagate.

$$L(X,y)=\sum_{i,j,k} y_{(i,j,k)}\log\left(p_{(i,j,k)}\right)+\left(1-y_{(i,j,k)}\right)\log\left(1-p_{(i,j,k)}\right)$$

GRU Variant:

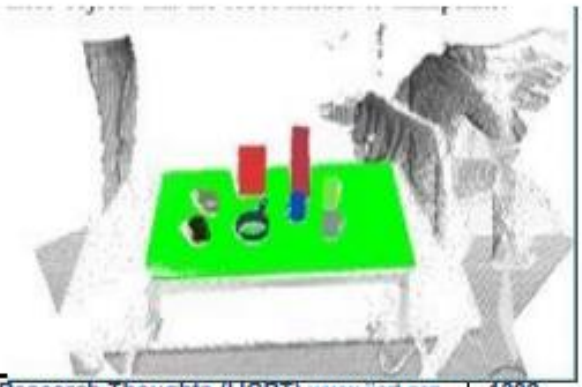
VI. FUTURE USE

In medical science for identifying cancer, tumors ,by classifying the 3d points rendering high dimensional internal views.



In Agriculture imaging, it might used for crop detection

In Robotics , to enable high level vision for robots to understands and perceive things as humans do.



$$u_t = \sigma\left(W_{fx}\mathcal{T}(x_t) + U_f * h_{t-1} + b_f\right)$$

$$r_t = \sigma\left(W_{ix}\mathcal{T}(x_t) + U_i * h_{t-1} + b_i\right)$$

$$h_t = \left(1 - u_t\right) \odot h_{t-1} + u_t \odot \tanh\left(W_h\mathcal{T}(x_t) + U_h * \left(r_t \odot h_{t-1}\right) + b_h\right)$$

VII. RESULT ANALYSIS

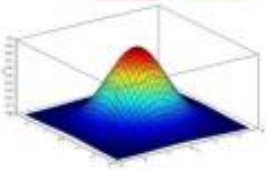
A typical 2D-to-3D conversion process consists of two steps: depth estimation for a given 2D image and depth based rendering of a new image in order to form a stereo pair. While the rendering step is well understood, the challenge is in estimating depth from a single image. Therefore, throughout the focus is on depth recovery.

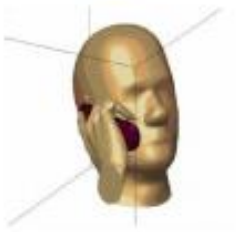
SINGLE VIEW CONSTRUCTION

MULTI VIEW 3D RECONSTRUCTION

ESTIMATE A 3D SHAPE GIVEN A SET OF IMAGES

Stereo reconstruction obtain point correspondences and camera calibration with estimate internal and external params.
Calculate projection rays with intersection gives 3D point.





VIII. CONCLUSION

A new class of methods is proposed to aim at 2D-to-3D image conversion that is based on the

radically different approach of learning from examples. One method that is proposed is based on learning a point mapping from local image attributes to scene-depth. The other method is based on globally estimating the entire depth field of a query directly from a repository of image + depth pairs using nearest-neighbor-based regression. It objectively validates the algorithms' performance against state-of-the-art algorithms. While the local method was outperformed by other algorithms, it is extremely fast as it is, basically, based on table look-up. However, the global method performed better than the state-of-the-art algorithms in terms of cumulative performance across two datasets and two testing methods, and has done so at a fraction of CPU time.

IX. REFERENCES

[1] A. Boulch, B. Le Saux, and N. Audebert, ''Unstructured point cloud semantic labeling using deep segmentation networks,'' in Proc. Eurographics Workshop 3D Object Retr., 2017, pp. 17–24.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, ''DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs,'' IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 4, pp. 834–848, Apr. 2018.

[3] E. Shelhamer, J. Long, and T. Darrell, ''fully convolutional networks for semantic segmentation,'' IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 4, pp. 640–651, Apr. 2017.

[4] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, ''A review on deep learning techniques applied to semantic segmentation,'' Apr. 2017, arXiv:1704.06857. [Online]. Available: https://arxiv.org/abs/1704.06857.

[5] L. P. Tchapmi, C. B. Choy, I. Armeni, J. Gwak, and S. Savarese, ''SEGCloud: Semantic segmentation of 3D point clouds,'' Oct. 2017, arXiv: 1710.07563. [Online]. Available: https://arxiv.org/abs/1710.07563.

[6] D. Maturana and S. Scherer, ''VoxNet: A 3D convolutional neural network for real-time object

recognition,'' in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Sep./Oct. 2015, pp. 922–928.

[7] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, ''3D ShapeNets: A deep representation for volumetric shapes,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1912–1920.

[8] C. R. Qi, H. Su, M. Kaichun, and L. J. Guibas, ''PointNet: Deep learning on point sets for 3D classification and segmentation,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 77–85.

[9] R. Zhang, ''Research on polymorphic object semantic segmentation of complex 3D scenes based on laser point clouds,'' Ph.D. dissertation, Dept. Survey Map, PLA Strategic Support Force Inf. Eng. Univ., Zhengzhou, China, 2018.

[10] L. Wang, ''Intelligent segmentation of point cloud based on PointNet network,'' M.S. thesis, Dept. RS Inf. Eng., Wuhan Univ., Wuhan, China, 2018.

REFERENCE

REFERENCES

[1] L. Agnot, W.-J. Huang and K.-C. Liu. A 2D to 3D video and image conversion technique based on a bilateral filter. In *Proc. SPIE Three- Dimensional Image Processing and Applications*, volume 7526, Feb.2010.

[2] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *Proc. European Conf. Computer Vision*, pages 25–36,2004.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 886–893,2005.

[4] F. Durand and J. Dorsey. Fast bilateral filtering for the display of high- dynamic-range images. *ACM Trans. Graph.*, 21:257–266, July2002.

[5] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust L1 optimal camera paths. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*,2011.

[6] M. Guttmann, L. Wolf, and D. Cohen-Or. Semi-automatic stereo extraction from video footage. In *Proc. IEEE Int. Conf. Computer Vision*, pages 136–142, Oct.2009.

[7] K. Karsch, C. Liu, and S. B. Kang. Depth extraction from video using non-parametric sampling. In *Proc. European Conf. Computer Vision,* 2012.

[8] J. Konrad, G. Brown, M. Wang, P. Ishwar, C. Wu, and D. Mukherjee. Automatic 2D-to-3D image conversion using 3D examples from the Internet. In *Proc. SPIE Stereoscopic Displays and Applications*, volume 8288, Jan.2012.

[9] J. Konrad, M. Wang, and P. Ishwar. 2D-to-3D image conversion by learning depth from examples. In *3D Cinematography Workshop (3DCINE'12) at CVPR'12*, pages 16–22, June2012.

[10] M. Liao, J. Gao, R. Yang, and M. Gong. Video stereolization: Combining motion analysis with user interaction. *IEEE Trans. Visualization and Computer Graphics*, 18(7):1079–1088, July2012.

[11] B. Liu, S. Gould, and D. Koller. Single image depth estimation from predicted semantic labels. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 1253–1260, June2010.

[12] R. Phan, R. Rzeszutek, and D. Androutsos. Semi-automatic 2D to 3D image conversion using scale-space random walks and a graph cuts based depth prior. In *Proc. IEEE Int. Conf. Image Processing*, Sept. 2011.

[13] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *NIPS*,2005.

[14] A. Saxena, M. Sun, and A. Ng. Make3D: Learning 3D scene structure from a single still image. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(5):824 –840, May2009.

[15] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light

sensor. In *Proc. Int. Conf. on Computer Vision - Workshop on 3D Represent. and Recogn.*,2011.

[16] M. Subbarao and G. Surya. Depth from defocus: A spatial domain approach. *Intern. J. Comput. Vis.*, 13:271–294,1994.

[17] R. Szeliski and P. H. S. Torr. Geometrically constrained structure from motion: Points on planes. In *European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, page 171–186, 1998.

[18] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Trans. Pattern Anal. Machine Intell.*, 30(11):1958–1970, Nov.2008.

[19] M. Wang, J. Konrad, P. Ishwar, K. Jing, and H. Rowley. Image saliency: From intrinsic to extrinsic context. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 417–424, June2011.

[20] R. Zhang, P. S. Tsai, J. Cryer, and M. Shah. Shape-from-shading: A survey. *IEEE Trans. Pattern Anal. Machine Intell.*, 21(8):690–706, Aug. 1999.