DENSE 3D POINT CLOUD REPRESENTATION OF A SCENE USING
UNCALIBRATED MONOCULAR VISION

Thesis
Submitted to
The School of Engineering of the
UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for
The Degree of
Master of Science in Electrical Engineering

By
Yakov Diskin
Dayton, OH
May, 2013
DENSE 3D POINT CLOUD REPRESENTATION OF A SCENE USING
UNCALIBRATED MONOCULAR VISION

Name: Diskin, Yakov

APPROVED BY:

__________________________
Raúl Ordóñez, Ph.D.
Advisory Committee Member
Associate Professor, Electrical and
Computer Engineering

__________________________
Eric Balster, Ph.D.
Advisory Committee Member
Assistant Professor, Electrical and
Computer Engineering

__________________________
Vijayan K. Asari, Ph.D.
Advisory Committee Chairman
Professor, Electrical and Computer
Engineering

__________________________
John G. Weber, Ph.D.
Associate Dean
School of Engineering

__________________________
Tony E. Saliba, Ph.D.
Dean, School of Engineering
& Wilke Distinguished Professor
© Copyright by

Yakov Diskin

All rights reserved

2013
ABSTRACT

DENSE 3D POINT CLOUD REPRESENTATION OF A SCENE USING UNCALIBRATED MONOCULAR VISION

Name: Diskin, Yakov
University of Dayton

Advisor: Dr. Vijayan K. Asari

We present a 3D reconstruction algorithm designed to support various automation and navigation applications. The algorithm presented focuses on the 3D reconstruction of a scene using only a single moving camera. Utilizing video frames captured at different points in time allows us to determine the depths of a scene. In this way, the system can be used to construct a point cloud model of its unknown surroundings. In this thesis, we present the step by step methodology of the development of a reconstruction technique. The original reconstruction process, resulting with a point cloud was computed based on feature matching and depth triangulation analysis. In an improved version of the algorithm, we utilized optical flow features to create an extremely dense representation model. Although dense, this model is hindered due to its low disparity resolution. As feature points were matched from frame to frame, the resolution of the input images and the discrete nature of disparities limited the depth computations within a scene. With the third algorithmic modification, we introduce the addition of the preprocessing step of nonlinear super resolution. With this addition, the accuracy of the point cloud which
relies on precise disparity measurement has significantly increased. Using a pixel by pixel approach, the super resolution technique computes the phase congruency of each pixel’s neighborhood and produces nonlinearly interpolated high resolution input frames. Thus, a feature point travels a more precise discrete disparity. Also, the quantity of points within the 3D point cloud model is significantly increased since the number of features is directly proportional to the resolution and high frequencies of the input image. Our final contribution of additional preprocessing steps is designed to filter noise points and mismatched features, giving birth to the complete Dense Point-cloud Representation (DPR) technique. We measure the success of DPR by evaluating the visual appeal, density, accuracy and computational expense of the reconstruction technique and compare with two state-of-the-arts techniques. After the presentation of rigorous analysis and comparison, we conclude by presenting the future direction of development and its plans for deployment in real-world applications.
This thesis is dedicated to:

My parents, Boris and Olga Diskin, who have supported me through all circumstances and have given me everything they could to ensure my success in life.

My family, who have taught me determination, focus and will power throughout life and have challenged my arguments, thoughts and patience.

My dearest friends, on whom I frequently rely on to clear my head, relieve stress or simply to find a reason to procrastinate, and who bring constant joy and never ending memories.

Dr. Vijayan Asari and the Vision Lab crew, by guiding me as a beacon into the professional world, who have advised me and surrounded me with greatest intellectuals in our research field.

“If I have seen further than others, it is by standing upon the shoulders of giants.”

-  Sir Isaac Newton

“The true sign of intelligence is not knowledge but imagination.”

“If you can’t explain it simply, you don’t understand it well enough.”

-  Albert Einstein
# TABLE OF CONTENTS

ABSTRACT ........................................................................................................ iv
DEDICATION ....................................................................................................... vi
TABLE OF CONTENTS ..................................................................................... vii
LIST OF FIGURES ............................................................................................ ix
LIST OF TABLES ................................................................................................ x

## CHAPTER I: INTRODUCTION ................................................................. 1
   I.1 Overview ................................................................................................. 2

## CHAPTER II: BACKGROUND AND RELATED WORK ......................... 4
   II.1 Data Type and Acquisition ............................................................... 5
   II.2 Monocular Vision ............................................................................. 8
   II.3 Visual Structure from Motion ......................................................... 10
   II.4 Probabilistic Volumetric Representation .................................... 14

## CHAPTER III: SCENE RECONSTRUCTION FRAMEWORK ..................... 22
   III.1 SURF Extraction ........................................................................ 23
   III.2 Feature Matching and Disparities .............................................. 25
   III.3 Depth Triangulation .................................................................. 29
   III.4 Point Localization ....................................................................... 31
   III.5 Point Cloud Model ...................................................................... 32
   III.6 Evaluation ..................................................................................... 35
<table>
<thead>
<tr>
<th>CHAPTER IV: DENSE 3D RECONSTRUCTION</th>
<th>38</th>
</tr>
</thead>
<tbody>
<tr>
<td>IV.1 Optical Flow Points</td>
<td>40</td>
</tr>
<tr>
<td>IV.2 Dense Point Cloud Models</td>
<td>46</td>
</tr>
<tr>
<td>IV.3 Metrics, Analysis, and Evaluation</td>
<td>56</td>
</tr>
<tr>
<td>CHAPTER V: DEPTH RESOLUTION ENHANCEMENT</td>
<td>61</td>
</tr>
<tr>
<td>V.1 Super Resolution</td>
<td>62</td>
</tr>
<tr>
<td>V.2 Analysis of Super Resolution</td>
<td>66</td>
</tr>
<tr>
<td>V.3 Enhanced Dense Reconstruction Results</td>
<td>74</td>
</tr>
<tr>
<td>CHAPTER VI: NOISE SUPRESSION OF POINT CLOUDS</td>
<td>78</td>
</tr>
<tr>
<td>VI.1 Preprocessing for Noise Removal</td>
<td>79</td>
</tr>
<tr>
<td>VI.2 Post-processing for Spiky Noise Suppression</td>
<td>85</td>
</tr>
<tr>
<td>VI.3 Filtered Point Cloud Results</td>
<td>89</td>
</tr>
<tr>
<td>CHAPTER VII: COMPARISON AND DISCUSSIONS</td>
<td>91</td>
</tr>
<tr>
<td>VII.1 Experimental Description</td>
<td>92</td>
</tr>
<tr>
<td>VII.2 Comparison, Metrics and Analysis</td>
<td>94</td>
</tr>
<tr>
<td>CHAPTER VIII: CONCLUSION AND FUTURE WORK</td>
<td>106</td>
</tr>
<tr>
<td>LIST OF PUBLICATIONS</td>
<td>109</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>112</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

| II.1 | Data captured in (a-b) laboratory environment, (c-d) urban areas with a medium altitude aerial vehicle. | 6 |
| II.2 | Data extremes: (a) Data captured at extreme altitudes and (b) low angles of perceptive... | 8 |
| II.3 | The human brain conducts continuous depth triangulation analysis... | 9 |
| II.4 | Monocular vision from a moving camera produces various viewing perspectives at different points in time... | 10 |
| II.5 | Image of the gradient orientation and magnitudes... | 13 |
| II.6 | Four different views and associated point cloud models. The reconstruction consists of 4,585 images and 2,662,981 3D points with 11,839,682 observed SIFT features [1]... | 14 |
| II.7 | Voxel notation. $X$ is voxel on the world surface $S$ lying on the ray $R(X)$ and projecting into image $I$ at pixel $I_{C(X)}$. However $V_{R(X)}$ is the voxel that actually produced color $I_{C(X)}$... | 17 |
| II.8 | PVR proposed by Pollard and Mundy [69]. (a) Initialization of voxels (b) explains the voxel notation, a pixel $I_X$ back projects into a ray of voxels $R_X$, $V$ is the unique voxel along $R_X$ that produces the intensity $I_X$ [71]... | 20 |
| II.9 | Resulting PRV models. (a) color model of Providence, RI (b) surface model of Providence, RI (c) surface model of the U.S. Capitol building... | 21 |
III.1 Steps to create a 3D model from two-dimensional video data captured by a single moving camera……………………………………………………………………………………………………23

III.2 Left to Right: the discretized and cropped Gaussian second order partial derivatives in y-direction and xy-direction, and Bay, Tuytelaars and Gool’s approximation thereof using box filters. The grey regions are equal to zero…………………………………………………………………………………………………………………………24

III.3 Subset of calculated and trackable SURF points with a frame. Only points of a large scale are shown………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………
III.8 An illustration of an increasing field of view as the captured data moves away from the camera sensor. 

III.9 (a) An original frame from a video recorded on a desk in a laboratory setting. The closest objects to the camera are the flags followed by the green water bottle and lastly the red bottle. (b) The single window sparse 3D reconstruction model of the scene presented in Fig. 5a. (c) An original frame from an aerial UAS flight of a railroad bridge over a river. (d) A sparse 3D reconstruction of the bridge scene. It is difficult to see any structure within the model due to the small number of points.

III.10 When the reconstruction process begins, new points are registered. On advanced frames, the same features points are recognized and not repeatedly added. Only new points are added to the global list from proceeding point clouds.

III.11 SURF alone produces a sparse point cloud, as illustrated a resulting registered point cloud of an city block is barely identifiable.

III.12 (a) A perpendicular view of the point cloud with indicators on distances used in the accuracy analysis. (b) A top view of the point cloud. The measurements made for the evaluation in the x-, y-, and z-axis are indicated with numbered lines.

IV.1 Steps to create a 3D model from two-dimensional video data captured by a single moving camera. The more advanced technique generates dense point cloud models with the marked additional steps of Horn-Schunck Optical Flow.
The basic rate of change of image brightness equation constrains the optical flow velocity. The velocity \((u, v)\) has to lie along a line perpendicular to the brightness gradient vector \((E_x, E_y)\). The distance of this line from the origin equals \(E_t\) divided by the magnitude of \((E_x, E_y)\).

The three partial derivatives of images brightness at the center of the cube are each estimated from the average of first differences along four parallel edges of the cube. Here the column index \(j\) corresponds to the \(x\) direction in the image, the row index \(i\) to the \(y\) direction, while \(k\) lies in the time direction.

The Laplacian is estimated by subtracting the value at a point from a weighted average of the values at neighboring points. Shown here are suitable weights by which values can be multiplied.

A disparity map of optical flow points demonstrates the depth of feature points from the camera. An outdoor street scene shows that objects closest to the camera plane such as the trees exhibit the long disparities, while objects further away such as the buildings exhibit shorter disparities.

The following are point clouds generated from the Horn-Schunck Optical Flow method. (a) and (c) show the original input frame while (b) and (d) are reconstruction shown at a perpendicular angle as the camera travels along a city street.

(a) Original video frame. (b) Reconstructed model viewed from the same angle. (c) When the model is rotated the incline of the hill becomes visible with the railroad horizontal in the background.

Sample frames in sequential order (a) to (d) from a video capture by a mobile platform.
IV.9  The images in sequential order (a) to (d) correspond to the video frames in Figure IV.8. These images illustrate the disparity map of the optical flow points. These disparity maps are color coded, where red (long arrows) represents points closest to the camera followed by green (medium length arrows) and blue (short arrows), respectively.  

IV.10 (a) A subsection of original frame showing an elevator call buttons.  (b) The same subsection of an elevator call buttons within the point cloud model 

IV.11 (a) Several frame from the input video concatenated together to illustrate the real-world scene.  (b) The 3D point cloud model of the scene created from the reconstruction framework described.  

IV.12 (a) A view from the top of the 3D point cloud model. Note that different objects are correctly placed in different depth layers.  (b) Several views of the 3D model illustrating the density of the point cloud.  

IV.13 A comparison between the original frame and the reconstructed model. A illustration of the density of the model allows the visibility of the labels on the trash bins.  

IV.14 Different views of a point cloud model generated from a street scene depicting dense point clusters as objects at various depths.  

IV.15 Details of the point cloud model illustrating the density and depth layers.  

IV.16 Illustration of the point cloud perpendicular to the plane of the camera.  

IV.17 The orange lines represent the vertical measurements made within the model.  

xiii
IV.18 We illustrate the point cloud view from above. The orange lines represent the depth measurements that were compared to the real-world values.

IV.19 (a) An original frame from a video recorded of N. Main St in downtown Dayton, OH. The scene contains numerous objects of different color and texture at varying depths. (b) This model is computed from two consecutive frames of video. (c) A view of the same 3D reconstruction from the top. Note that due to the discrete nature of feature disparities points are dispersed into layers.

V.1 The algorithmic architecture with the addition of super resolution designed to enhance the depth resolution of the point cloud models.

V.2 Structure of super resolution module.

V.3 There are four directional variance bins used for the computations of the covariance matrix.

V.4 Super resolution results in a higher resolution input image.

V.5 Several images to illustrate the effects of super resolution. Increase in resolution from (a) 100x100 to (b) 400x400. Second example, (c) 119x211 on the left to (d) 796x844 on the right.

V.6 Harris Corners for Columbus Large Format Imagery. Left images (a) (c) (e) are of the original resolution, while images on the right (b) (d) (f) are processed after the applied super resolution.

V.7 A vehicle contains more feature points in the higher resolution image.

V.8 The effects of super resolution are illustrated. (a) The original input frame. (b) Resolution is increased by a factor of four after super resolution applied to the
original input. The input image has become 2440 x 1440. (c) Resolution is increased by a factor of four after super resolution applied to the original input. The input image has become 4880 × 2880.

V.9 The effects of super resolution on the point cloud models. (a) 3D point cloud model from the original input image (b) 3D point cloud model from the super resolution input image. The resolution has been increased sixteen times that of the original creating more points in the point cloud and more unique layers.

V.10 (a) Frontal view of the 3D reconstruction model created from the original resolution input frames. (b) Top view of the 3D reconstruction model created from the original resolution input frames. (c) Frontal view of the 3D reconstruction model created from the super resolution input frames. (d) Top view of the 3D reconstruction model created from the super resolution input frames. Note the difference in the number of layers and density of the point cloud.

VI.1 The algorithmic flow diagram of Dense Point-Cloud Representation technique. Additional noise removal steps presented in this chapter are highlighted in green.

VI.2 Original image on the top (a) contains non-uniform lighting. The enhanced image (b) on the right contains more trackable feature points.

VI.3 Top, original frame exhibiting a subtle barrel distortion. Bottom, effects of the distortion correction algorithm implementation.

VI.4 (a) Illustration of textureless regions which cause mismatches leading to noise points within the point cloud model. These regions are very similar to one another and cause confusion when comparing the descriptors of the feature points.
within. (b) The same scene as illustrated in part ‘a’ but the textureless regions are removed from the image … ................................................................. 86

VI.5 (a) By clustering multiple depth layers into one we are able to obtain a 2D image in which spiky noise appear as small speckles (b) We illustrate the clustering of multiple layers into one which contains true points as well as many noise points. (c) The morphological “opening” operation is erosion, in which as structuring element removes all the noise and edge of larger objects.  (d) The second stage of the morphological “opening” operation is dilation................................. ................. 88

VI.6 (a) The super resolution enhancement produces an extremely dense point cloud model. In this model the previously unnoticed noisy is also enhance. The noise suppression technique described eliminates the majority of the noisy point from this model. (b) Over head view of super resolution point cloud. (c) The same point cloud model as presented in part ‘a’ but post noise suppression processing. All the major features of the scene are retained but little noise speckles are eliminated using the technique. (d) A view of the new clean point cloud model. This model features a variety of depth layers and dense reconstructed objects from the input scene.................. ......................................................... 90

VII.1 Comparison of stages in DPR (Dense Point-cloud Representation), PVR (Probabilistic Volumetric Representation) and VSFM (Visual Structure From Motion). The most visually appealing output for each technique is marked in red........................ ................................................................. 92

VII.2 Frames from the Low Altitude Downtown Dayton dataset......... ....................... 95

VII.3 The 3D reconstruction results using the Low Altitude (Dayton, OH) dataset. (a) VSFM results (b) DPR results.......................... ......................................................... 95
VII.4 Frames from the Medium Altitude Providence, RI dataset. .......................... 97
VII.5 The 3D reconstruction results using the Medium Altitude (Providence, RI) dataset. (a) VSFM results (b) PVR results and (c) DRP results. ......................... 97
VII.6 Frames from the High Altitude over Columbus, OH dataset. ......................... 99
VII.7 The 3D reconstruction results using the High Altitude (Columbus, OH) dataset.
(a) VSFM results (b) PVR results and (c) DRP results ................................. 99
VII.8 A summary of the visual assessment ratings. .............................................. 100
VII.9 The thresholded models of the three techniques illustrate the model as they would be used in application. (a) VSFM (b) PVR (c) DPR ................................. 104
LIST OF TABLES

III.1 The table contains measured and computed values of a laboratory desk scene. .......... 37

IV.1 Summarizing the accuracy of the point cloud model. For each direction, the
model unit to real-world units conversion and average error (in feet) are shown. .......... 59

V.1 Number of Harris Corners extracted in three datasets after applying various
interpolation techniques. The super resolution algorithm described provides the
highest number of extracted points. .......................................................... 71

V.2 Comparison of point cloud models created from the original input models to
models created from the super resolution inputs........................................ 74

VI.1 Comparison of point cloud models created from the original input models to
models created from the enhancement (top) and distortion correction (bottom)
inputs. .......................................................... ........................................... 82

VII.1 Metrics for each evaluation technique. The green highlighted areas indicate areas
of superior performance while red highlights indicate weak performances. DNF
(Did Not Finish) corresponds to the inability to handle certain types of scenes... .......101
CHAPTER I
INTRODUCTION

The challenge of replicating the human ability to analyze a scene or environment has recently been put on the forefront. Humans have an extraordinary skill to recognize objects and determine depth in a variety of lighting conditions and relatively large distances. Sensors, such as traditional televisions and modern digital CCD cameras, simulate the human eye by outputting projective images. As technology continues to develop, automation of machines and devices will continue encroaching even more into everyday life.

As with most engineering research theses, this Thesis presents a novel idea and its implementation results and analysis in a friendly and approachable tone. The optimistic perception that many readers will begin to read this book has pushed me to write in a more readable and personalized tone. The technical writing within is illustrated and explained thoroughly so that readers of various technical backgrounds can find interest in the material.

The aim of this research is to aid in scene understanding for autonomous machines such as unmanned aerial vehicles and other unmanned surface vehicles. In order for an unmanned aerial system (UAS) to be able to navigate itself through an unknown environment a procedure analyzing the scene must be developed. Other work has already been attempted and somewhat successfully conducted using systems with multiple sensors include depth finders [25, 26, 41, 51, 70]. In this thesis, we utilize a minimal number of sensors to construct a model of a scene. Our experiments are conducted using only ordinary electro-optic cameras, such as a Canon EOS 5D
Mark II Digital SLR Camera and a Canon EF 24-70mm f/2.8L USM AutoFocus Wide Angle Telephoto Zoom Lens. All analysis is computed from imagery, more specifically, we have constrained ourselves from utilizing laser depth sensors, GPS or orientation devices. These devices and sensors could be used to evaluate the accuracy of the reconstructed model, but not for generating or computing of the model itself.

We present a system capable of generating a 3D scene of an unknown environment using only a single high resolution moving camera. In addition, we propose the development of several computer vision techniques to enhance the reconstruction architecture. The framework for this reconstruction technique serves as the basis of this scene creation algorithm. The algorithm generates a model obtained by correctly positioning millions of points into a three-dimensional Cartesian coordinate system. Each point in the model corresponds to a point in the real world environment. As the camera travels through a scene, the same points are seen from a variety of slightly differing locations and orientations. By determining the camera position and orientation corresponding to every frame, we are able to perform triangulation techniques to compute the \((x, y, z)\) coordinates for each point in the scene.

I.1 OVERVIEW

The thesis is structured as follows. We begin by describing the data capture process and the various type of data involved in different 3D reconstruction research groups. In Chapter II, we present a variety of data and discuss the challenges associated with altitude variations, viewpoint variations, frame rate and resolution variations. We are going to describe the current state-of-the-arts reconstruction techniques, Visual Structure from Motion (VSFM) and Probabilistic Volumetric Representation (PVR). These techniques have been used in enormous applications and have a proven record in building appealing models using certain datasets. Prior to our
rigorous comparison between our technique versus VSFM and PVR, we present an in depth algorithmic development of our Dense Point-cloud Representation (DPR) technique.

In Chapter III, we outline the initial thoughts and framework of depth computations from uncalibrated monocular vision. Furthermore, we identify the areas in the algorithm that need improvement. In Chapter IV, we improve the algorithm by increasing the number of features used and as a result create an extremely dense point cloud model. Once again, we evaluate our result and algorithmic architecture and determine a way to enhance it. In Chapter V, we present the technology and results generated using a novel single image super resolution technique that allows for us to enhance the depth resolution of the scene. We conclude our development and self-evaluation in Chapter VI, after presenting adaptive noise suppression techniques that clean the model and increase the usability.

After a detailed presentation of our DPR technique, in Chapter VII, we perform an in depth comparison between our method and the state of the art methods. We impartially evaluate the techniques based visual appeal, density, computational expense and usability. Metrics, figures and table illustrate the areas in which each technique performs its best. We conclude our algorithm presentation with a discussion on the potential applications of 3D reconstruction and the future plans for the development and deployment of our system in Chapter VIII.
CHAPTER II

BACKGROUND AND RELATED WORK

In this chapter, we describe the challenges and existing partly-satisfying solutions within the research community regarding reconstructing a scene using a single camera. We begin by exploring and understanding the data collection process in Section II.1. Imagery is affected by a variety of components. From the electro-optic sensors to the image resolution, contrast, exposure and blurriness variables, all add to the complexity of analyzing a scene and processing the imagery. The platform that carries the camera sensor also plays a significant role in the type of imagery and reconstruction we are able to obtain. In this chapter, we describe a variety of data ranging in a variety of altitudes, environments, frame rates, stability and speed of platform. We further discuss the concepts of monocular vision and how it compares to human binocular vision in Section II.2. As mentioned in Chapter I, stereo vision triangulation algorithms have existed much longer and have very sophisticated systems already commercialized. The challenge of optimizing the number of cameras needed to reconstruct a scene accurately and efficiently is solved by accurately computing and estimating the locations and orientations of a single moving camera through time.

In addition, we outline the works of Changchang Wu, Sameer Agarwal, Joseph Mundi and Thomas Pollard as the measure of the current state-of-the-arts technology in representing a scene captured by a moving camera in a 3 dimensional model. These algorithms have been built on the work of simultaneous localization and mapping research [13, 14, 23] as well as
stereoscopy work [21, 22, 37]. In Section II.3 we describe the works from the University of Washington [1, 83, 85], called Visual Structure from Motion (VisualSFM). This cutting edge technique utilizes the basic components of the Scale Invariant Feature Transform [52, 53] (SIFT) keypoints to perform point matching from frame-to-frame. Similar to [5, 77], they used the matched points to compute the location and orientation of the camera associated with each frame as described in [84]. These computations allow for advanced schematic surface reconstructions [82] and visually appeal 3D models.

We conclude our background survey with the most visually appealing scene representation technique. In Section II.4, Probabilistic Volumetric Representation [69, 71] from Brown University creates a volumetric representation of the scene. Furthermore, each voxel contains tremendous amounts of information, such as reflection, intensity and probability of a surface point, all with respect to the viewing angle. This technique allows for the creation of non-point cloud 3D representations that allows for object recognition within the scene [11, 32, 46]. Before taking an in depth look at these techniques we begin by understanding the challenges associated with different types of data and its acquisition.

II.1 DATA COLLECTION AND ACQUISITION

The research field of automatic 3D reconstruction of a scene is relatively young, and therefore, contains no standardized datasets for all to use. Each reconstruction technique collects its own data, and produces models in its own format. No standardized evaluation method has been generally accepted. Many techniques evaluate the visual appeal of the final product, while other techniques focus on the density and usability of the final model. In order to understand what makes a reconstruction technique perform extremely well versus extremely poorly, we must understand the challenges caused by the input imagery. In Figure II.1, we illustrate a variety of
Figure II.1: Data captured in (a-b) laboratory environment, (c-d) urban areas with a medium altitude aerial vehicle. Aerial data captured by Brown University (c) and IDCAST (d) systems.

sequences. In Figure II.1a and Figure II.1b, we capture a scene in a laboratory environment. This environment allow us to capture data in ideal conditions, where the lighting was uniformly distributed over the object of interest, as well as the camera traveling with known speed and
known orientation in relation to the scene. The objects, such as colorful water bottles, flags, and faces, were chosen to test the colorization capabilities and the depth computations of the reconstruction technique. In Figure II.1c and Figure II.1d, we illustrate a different kind of scene captured from an aerial system. This dataset contains extremely unstable sequences that make feature matching extremely complicated. During the data collection for this research, we have tested aerial scene capture over Providence Rhode Island, Dayton Ohio, Columbus Ohio, Gary Indiana, and many other locations. Aerial imagery is challenging due to the constant changes in depth and image resolution of the scene.

We also consider several extreme scenarios presented in Figure II.2. The scene presented in Figure II.2a is captured at extremely high altitude where no significant height differences are noticeable between the buildings, trees, roads, and vehicles in the scene. The scene depicted is captured over Columbus, Ohio and referred to as the Columbus Large Image Format (CLIF) [10] dataset. Images in this dataset capture an extremely wide area for an aerial platform traveling at 7000+ feet.

In the second extreme scenario, we introduce a real-world scene of downtown Dayton, OH that contains moving pedestrians, cars, and many ordinary street objects. The scene is captured from a vehicle traveling at a somewhat constant speed with the camera perpendicular to the scene. One of the challenges of this scene is that the background sky regions are at an infinite distance from the camera and therefore make complications when reconstruction algorithms attempt to map these regions in a model. In the following section, we examine the underlining principles of depth evaluation and triangulation. We explain how a single camera can determine the depth of objects in a scene.
Figure II.2: Data extremes: (a) Data captured at extreme altitudes [10] and (b) low angles of perceptive.
Video data provided by the AFRL (a) and IDCAST (b).

II.2 MONOCULAR VISION

Traditional stereoscopy systems, such as human visual system, utilize multiple viewing angles of the same object in order to do triangulation and get a depth perception. The human eyeball is maintained in position in the orbit by six muscles, which move it to direct the gaze to any position and give convergence of the two eyes for depth analysis. They are under continuous tension and form a delicately balanced system [33]. In Figure II.3, we illustrate the eye sensor and depth perception component of human vision. As Gregory describes in his work, the optic nerve divides at the chiasma, the right half of each retina being represented on the right side of the occipital cortex and the left side on the left half. The lateral geniculate bodies are relay
Figure II.3: The human brain conducts continuous depth triangulation analysis at all times. Figure is reproduced from [33].

stations between the eyes and the visual index. As we describe our monocular vision system, many similarities to human eye vision will be discovered.

Monocular vision, better described as vision through a single camera source, presents new challenges when compared to stereo vision or a multi-camera system. In a stereo system, similar to human vision, distances between cameras (the baseline) and their orientation is known and in most circumstances remains constant. In order to generate various viewing angles with a monocular system, the camera must continuously be moving. With a moving camera, the system obtains two different viewing angels from two points in time. The challenge becomes to accurately compute the distance the camera has traveled or the exact location of the camera at each frame of video. In addition, the orientation of the camera at each point in time must be computed from the scene. The concepts of monocular vision are illustrated in Figure II.4.
Figure II.4: Monocular vision from a moving camera produces various viewing perspectives at different points in time.

II.3 VISUAL STRUCTURE FROM MOTION

In his work [1, 82, 83, 84, 85], Wu et al. describes the methodology of Visual Structure from Motion (VSFM) as following. Using a set of image feature locations and correspondences, the goal of bundle adjustment [83] is to find 3D point positions and camera parameters that minimize the re-projection error [80]. This optimization problem is constructed as a non-linear least squares problem, where the error is the squared $L_2$ norm of the difference between the observed feature location and the projection of the corresponding 3D point on the image plane of the camera.

Wu explains by letting $x$ be a vector of parameters and $f(x) = [f_1(x), ..., f_k(x)]$ be the vector of residuals errors for a 3D reconstruction. Then the optimization problem he wishes to solve is the non-linear least squares problem shown in Equation II.1.

$$x^* = \arg \min_x \sum_{i=1}^{k} \|f_i(x)\|^2.$$  \hspace{1cm} (II.1)
The Levenberg-Marquardt (LM) algorithm [67] is an extremely popular algorithm for solving non-linear least squares problems and is the algorithm of choice for bundle adjustment. LM operates by computing a series of regularized linear approximations to the original nonlinear problem. Let \( f(x) \) be the Jacobian of \( f(x) \), then in each iteration LM solves a linear least squares problem of the form

\[
\delta^* = \arg \min_{\delta} \| J(x) \delta + f(x) \|^2 + \lambda \| D(x) \delta \|^2,
\]  

(II.2)

and updates \( x \leftarrow x + \delta^* \) if \( \| f(x + \delta^*) \| < \| f(x) \| \). Here, \( D(x) \) is a non-negative diagonal matrix, typically the square root of the diagonal of the matrix \( J(x)^T J(x) \) and \( \lambda \) is a nonnegative parameter that controls the strength of regularization. Wu explains that the regularization is needed to ensure a convergent algorithm. LM updates the value of \( \lambda \) at each step based on how well the Jacobian \( J(x) \) approximates \( f(x) \) [67].

Solving Equation II.2 is equivalent to solving the normal equations

\[
(J^T J + \lambda D^T D) \delta = -J^T f.
\]  

(II.3)

where we have dropped the dependence on \( x \) for notational convenience. The matrix \( H_\lambda = J^T J + \lambda D^T D \) is known as the augmented Hessian matrix.

Within the bundle adjustment technique, the parameter vector is typically organized as \( x = [x_c; x_p] \), where \( x_c \) is the camera parameter vector and \( x_p \) the point parameter vector. For \( D \), \( \delta \), and \( J \), Wu uses subscripts \( c \) and \( p \) to denote the camera part and the point part respectively. Let \( U = J^T J_c \), \( V = J^T J_p \), \( U_\lambda = U + \lambda D^T_c D_c \), \( V_\lambda = V + \lambda D^T_p D_p \), and \( W = J^T J_p \), then Equation 3 can be re-written as the block structured linear system

\[
\begin{bmatrix}
U_\lambda & W \\
W^T & V_\lambda
\end{bmatrix}
\begin{bmatrix}
\delta_c \\
\delta_p
\end{bmatrix}
= \begin{bmatrix}
J^T_c f \\
J^T_p f
\end{bmatrix}.
\]  

(II.4)

It is worth noting that for most bundle adjustment problems, \( U_\lambda \) and \( V_\lambda \) are block diagonal matrices. Interestingly, this observation lies at the heart of the Schur complement trick used to
solve this linear system efficiently, where, by applying Gaussian elimination to the point parameters, we obtain a linear system consisting of just the camera parameters:

\[
(U_{\lambda} - WV_{\lambda}^{-1}W^T)\delta_{c} = -f^T_c f = WV_{\lambda}^{-1}f_p^T f.
\] (II.5)

The matrix \( U_{\lambda} - WV_{\lambda}^{-1}W^T \) is the Schur complement or the reduced camera matrix. Given the solution to Equation II.5, \( \delta_p \), the point parameters vector can be obtained by back substitution:

\[
\delta_p = -V_{\lambda}^{-1}(f_p^T f + W^T \delta_c).
\] (II.6)

**SIFT Features**

The SIFT algorithm can be broken down into four main stages: (1) scale-space peak selection; (2) point localization; (3) orientation assignment; and (4) point descriptor. The first stage is to search for interest points over location and scale. The image is constructed in a Gaussian Pyramid, where the image is downsampled and blurred at each level [42]. These blurred images at each level are used to compute the Difference of Gaussians (DoG), which locate edges and corners within an image. Interesting points are then extracted in stage 2 by locating the maxima/minima pixels within the different scales of the DoG at sub-pixel accuracy. Once interest points have been located, an orientation is assigned based on the gradient orientation of the pixels around the interest point. Below in Figure II.5, an example image of the gradient orientation and magnitudes is shown. The size of the window depends on the scale value where the interest point is detected.
Once orientation and scale have been addressed, the final stage is the generation of point descriptors. A descriptor is generated by breaking up an image into different bins and using gradient orientation and magnitude values to describe each bin. These values create a feature vector, which are now rotation and scaling invariant. Feature vectors can be used for many things, such as object recognition, tracking and image alignment.

This covers one of the more commonly used feature detection methods for image registration. There have been many other papers that present improvements to the SIFT method. Ke and Sukthankar proposed a SIFT algorithm that uses Principle Component Analysis (PCA) to normalize gradient patches, instead of a smoothed weighted histogram [42]. A more robust matching of SIFT features for remote sensing images and the SIFT algorithm improved and adjusted to function on SAR image registration is described in [86]. A SIFT Flow algorithm, proposed in [49], produces a highly dense flow field of SIFT descriptors that are matched between frames. The point cloud result of VSFM are presented in [1] and can be observed in Figure II.6.
II.4 PROBABLISTIC VOLUMETRIC REPRESENTATION

In this section we present the Probabilistic Volumetric Representation (PVR) algorithm. We begin by understanding the algorithmic flow to reconstruct a voxelated representation of the scene. The Marr and Poggio [56, 57] algorithm works under the assumptions that the disparity maps have unique values and are continuous almost everywhere throughout the scene. The Marr and Poggio approach simultaneously represented and altered multiple disparities allowing for initial consideration of several hypotheses. Kanade and Zitnick [87] have used the algorithm [57] and enhanced it to use 3D local support and continuous match likelihood values. This also allowed Kanade et al to explicitly detect occluded areas as regions with low likelihood values.

As Agarwal and Davis [2] describe using an iterative scheme [77], and later on [69], they have a set of voxels which are known to be surface points at each step of the algorithm. Using this idea and computing a visibility map. A visibility map indicates whether a given camera can see a voxel. The color consistency of a voxel along views in which it is visible is used to select winners at each stage that are added to the set of known surface points. In order to take into account the

Figure II.6: Four different views and associated point cloud models. The reconstruction consists of 4,585 images and 2,662,981 3D points with 11,839,682 observed SIFT features [1]. Input imagery and reconstruction results are regenerated using [1] and [83].
fact that voxels on the boundaries of objects are only partially occupied, they use real valued transparencies to represent voxels which are partially occupied by opaque objects. [2]

In other works by assuming a large number of views are available, \( N \geq 10 \), from video or from multiple cameras observing a scene. [21] The cameras have been coarsely calibrated, typically with a 2 – 5 pixel error. The goal of these methods is to produce a dense collection of unorganized 3D curve fragments which reflect the underlying geometry arising from a combination of 3D surface geometry and viewing arrangements, occluding contours, reflectance contours, shadow curves, shade curves, ridges, etc. [71] Also they use this reconstruction to refine the cameras so that a better and denser 3D curve sketch can be obtained, and so that a better distinction can be made between occluding contours and view-stationary contours such as reflectance curves and ridges. The approach [21] is divided into two stages. In the first stage, where cameras are calibrated coarsely, the goal is to reconstruct a partial, but reliable core 3D curve sketch to be used in the second stage for refining calibration and for obtaining a more complete 3D curve sketch. [2] Mundy and Pollard form a 3D curve fragment hypothesis by pairing two long curve fragments, each from a different view, with sufficient epipolar overlap. The algorithm calls the two views from which a curve pair hypothesis is formed as the hypothesis views. These views rotate in the reconstruction process among available view pairs. Each curve pair hypothesis is reprojected onto a set of other views, the confirmation views, and rated for consistency with the image and curve differential geometry. The approach can be thought of as an automated version of the curve-based CAD system from [28, 29, 30]. Those hypotheses with enough evidence in confirmation views are then reconstructed to form the initial core 3D curve sketch.

The core 3D curve sketch enables a curve-based measurement of calibration accuracy, by summing up the distances between projected curves and supporting image curves. This allows for
a fine-tuning of cameras through bundle adjustment, resulting in subpixel calibration errors. The cameras in turn allow for extra sensitivity so that smaller curve fragment pair hypotheses can be confirmed or discarded.

**Algorithm**

This subsection described by Pollard and Mundy contains a description of the voxel model of the world, an algorithm for updating it as new images come in, and an algorithm for detecting changes in new images. The modeling process involves simultaneous estimation of both the surface of the world and a color model at each point on the surface.

To begin, a bounded volume containing the entire 3D world of interest is partitioned into voxels, see Figure II.7. A voxel $X$ has two possible states a surface in the model or in empty space or inside a solid object. The state of $X$ being a surface voxel is denoted by $X \in S$. It will prove beneficial to cast the surface occupancy problem into a Bayesian framework, so it makes sense to talk about probabilities like $P(X \in S)$ which measure one’s belief that $X$ is a surface voxel. This described in [69].

For any image $I$ with known camera $C$ the voxel volume can be partitioned into rays of voxels that project into common locations in the image. The ray to which voxel $X$ belongs is denoted by $R(X)$ and the color intensity to which $X$ maps in the image by $I_C(X)$. For any ray the voxels $X' \in R(X)$ can be ordered with the inequality $X'' > X'$, when $X''$ is further along the ray from the camera center than $X'$. Also for each ray there is exactly one voxel which produced the intensity seen in the image and this voxel is denoted by $V_{R(X)}$ or simply by $V$ when the ray is understood.

Next, the prediction of pixel colors observed in the images is based on the Gaussian mixture color models, however there exists the complication that for any pixel in the image there are many voxels in the world that can potentially produce the observed color. The solution is to
maintain a mixture of Gaussians model at each voxel to be used when it is time to consider the possibility that this voxel produced the observed color. The probability density \( P(I_{C(X)}|V_{R(X)} = X) \) has a mixture of Gaussians distribution for voxel \( X \). This is from [69].

Given a sequence of images \( \{I^t\} \) and cameras \( \{C^t\} \), for the first image the surface probability, \( P^0(X \in S) \), for each voxel \( X \) is initialized to some common constant and the mixture of Gaussians model is initialized with the pixel color observed, \( I^0_{C^0(X)} \). After observing image \( I^{t+1} \) the surface estimates \( P^t(X \in S) \) and color distributions \( P^t(I_{C(X)}|V_{R(X)} = X) \) for each voxel \( X \) must be updated.

Figure II.7: Voxel notation. \( X \) is voxel on the world surface \( S \) lying on the ray \( R(X) \) and projecting into image \( I \) at pixel \( I_{C(X)} \). However \( V_{R(X)} \) is the voxel that actually produced color \( I_{C(X)} \). This figure is reproduced based on a figure in [69].
Updating the Surface Probabilities

Pollard and Mundy describe the procedure for updating the surface probability \( P^t(X \in S) \) for voxel \( X \) with information from \( I^{t+1} \) [69]. Equations and their descriptions are originally described by [2, 69]. The assumption is made that \( I^{t+1}_{C(X)} \) (from here on simply \( I_X \)) is the only relevant pixel in \( I^{t+1} \) for updating voxel \( X \), as pixel neighborhoods are not used. [69] With this assumption it is a consequence of Bayes rule that:

\[
P^{t+1}(X \in S) = P^t(X \in S) \left( \frac{P^t(I_X | X \in S)}{P^t(I_X)} \right)
\]  

(II.7)

After expanding the color probability terms by voxels along \( R(X) \) that actually produced the color, the multiplier for \( P^t(X \in S) \) in equation 1 equals:

\[
\frac{\sum_{X' \in R(X)} P^t(I_X | V = X') P^t(V = X'|X \in S)}{\sum_{X' \in R(X)} P^t(I_X | V = X')} P^t(V = X')
\]

(II.8)

\( P^t(I_X | V = X') \) is computed using the mixture of Gaussians color model stored in voxel \( X' \) after training on the first \( t \) images. The only question left is how to compute \( P^t(V = X') \) and \( P^t(V = X'|X \in S) \).

Pollard and Mundy consider \( P^t(V = X') \), which is the probability that voxel \( X' \) produced the pixel color in the image considering only the voxel surface probabilities along the ray, and not any color information. From a geometric viewpoint, the voxel will produce the color if and only if the voxel is on the surface and it is not occluded:

\[
P^t(V = X') = P^t(X \in S) P^t(X' \text{ is not occluded})
\]

(II.9)

\[
P^t(X' \text{ is not occluded}) = \prod_{X'' \subset X'} (1 - P^t(X'' \in S))
\]

(II.10)

which is the probability that all voxels between \( X' \) and the camera contain empty space. This completes the definition of \( P^t(V = X') \). The equation for \( P^t(V = X'|X \in S) \) is the same except that any instances of \( P^t(X \in S) \) above have probability 1. Note that the update equations for
have been derived using only the current surface and color probabilities for voxels lying in the same ray, and so are computable. [69]

**Updating the Color Model**

In this popular implementation a Gaussian mixture model is a combination of \( K \) Gaussian distributions with means \( \mu_k \), standard deviations \( \sigma_k \), and weights \( w_k \) that are discussed later in this subsection. Equations and their descriptions are originally described by [2, 69]. Each mode has distribution:

\[
\eta_k(y) = \frac{1}{\sqrt{2\pi\sigma_k}} e^{-\frac{(y-\mu_k)^2}{2\sigma_k^2}} \quad \text{(II.11)}
\]

and the full mixture distribution is:

\[
p(y) = \sum_{k=1}^{K} \frac{w_k}{W} \eta_k(y), \quad W = \sum_{k=1}^{K} w_k \quad \text{(II.12)}
\]

modes are ranked by \( w_k/\sigma_k \) and to train the distribution on a new color \( c \) and weight \( dw \) the first mode for which \( c \) lies within 2.5 standard deviations of the mean is updated with the following equations:

\[
w_k^{n+1} = w_k^n + dw \quad \text{(II.13)}
\]

\[
\mu_k^{n+1} = \mu_k^n + \frac{dw}{\sigma_k^n + w_k^n} (c - \mu_k^n) \quad \text{(II.14)}
\]

\[
\left(\sigma_k^{n+1}\right)^2 = \left(\sigma_k^n\right)^2 + \frac{dw}{\sigma_k^n + w_k^n} ((c - \mu_k^n)^2 - (\sigma_k^n)^2) \quad \text{(II.15)}
\]

These equations are the same as the EM-algorithm derived result in [69] in the case where the weights are constant. If \( c \) is not within 2.5 standard deviations of any mode, the least probable mode is destroyed and replaced with a high variance mode with mean \( c \) and weight \( dw \).

The algorithm converges following the Convergence Theorem [69], which states, a voxel \( X \) lying on a world surface will converge provided that the surface images a constant color in all
views, and that all voxels $X'$ near $X$ lying beneath the surface frequently project into some pixels of sufficiently different color from the true surface color at $X$. The probabilistic volumetric representation is summarized in Figure II.8. In Figure II.9, we present the resulting models using PVR.

![Figure II.8: PVR proposed by Pollard and Mundy [69].](image)

Figure II.8: PVR proposed by Pollard and Mundy [69]. (a) Initialization of voxels (b) explains the voxel notation, a pixel $I_X$ back projects into a ray of voxels $R_X$, $V$ is the unique voxel along $R_X$ that produces the intensity $I_X$. Demonstration of results is originally illustrated in [71].

In this chapter, we have presented two state of the art reconstruction algorithms. We have demonstrated the computations involved in produce a 3D representation of scene. In the next four chapters, Chapter III through Chapter VI, we describe our proposed methodology and reasoning for each algorithmic step. Our technique focuses on reducing the computational expenses and enhancing the density, usability and visual appeal over the technique presented in this chapter. Furthermore, we conclude with a rigorous comparison and analysis between the three techniques.
Figure II.9: Resulting PRV models. (a) color model of Providence, RI (b) surface model of Providence, RI (c) surface model of the U.S. Capitol building. Demonstration of results is originally illustrated in [71].
CHAPTER III
SCENE RECONSTRUCTION FRAMEWORK

We begin by examining the framework and concepts for the presented 3D scene creation technique that reconstructs an unknown environment using only a single high resolution moving camera. To test the framework, we assume a linear constant speed camera path with the camera oriented perpendicular to the scene. The assumptions have been temporarily placed on the reconstruction procedure to analyze its effectiveness and efficiency. To utilize the reconstruction technique on a real-time real-world unmanned aerial system (UAS), these assumptions must be dropped and accurate computations should replace those values. Further details on computations of camera speed and orientation can be found in Chapter VIII.

Once a scene model is created, the distances and sizes within the model are relative. In order to determine the scale factor of the model, one must know the speed or position of the camera as well as the focal length of the camera lens for each frame. By knowing the camera specifications and speed, a disparity array can be constructed. In a disparity array various disparity values are assigned real-world depth distances and computed features’ disparities are interpolated into the array to determine their corresponding depth.

This chapter describes the algorithm that generates a model obtained by correctly positioning thousands of points into a three-dimensional Cartesian coordinate system. Each point in the model corresponds to a point in the real world environment. As the camera travels through a scene, the same points are seen from a variety of slightly differing locations and orientations.
By determining the camera position and orientation corresponding to every frame, we are able to perform triangulation techniques, described in our previous work [19], to compute the \((x, y, z)\) coordinates for each point in the scene. We begin by describing the algorithmic flow of the reconstruction framework presented in Figure III.1. In this chapter a step by step procedure of the reconstruction is described, followed by an evaluation and accuracy metrics of the model.

The scene reconstruction process is broken into seven sequential steps. In our experiments we utilize our own camera keeping the focal length at 24mm throughout the video sequence. The figures in this section contain images from test videos recorded in a laboratory setting as well as street data from a moving car and UAS aerial data.

### III.1 SURF EXTRACTION

The first algorithmic step is (1) SURF Extraction, wherein we locate stable feature points within each frame using the Speeded-Up Robust Features (SURF) algorithm. Related applications of feature matching are described by Lucas and Kanade [54]. The SURF algorithm uses a Fast - Hessian Detector described by Bay, Tuytelaars and Gool [4] to identify distinct
feature points within an image. By computing the determinant of the Hessian we are able to
determine the location and scale of a feature. For example, given a point \( x = (x, y) \) in an
image \( I \), the Hessian matrix \( H(x, \sigma) \) in \( x \) at scale \( \sigma \) is defined as follows

\[
H(x, \sigma) = \begin{bmatrix}
L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\
L_{xy}(x, \sigma) & L_{yy}(x, \sigma)
\end{bmatrix}
\]  

(III.1)

where \( L_{xx}(x, \sigma) \) is the convolution of the Gaussian second order derivative \( \frac{\partial^2}{\partial x^2} g(\sigma) \) with the
image \( I \) in point \( x \), and similarly for \( L_{xy}(x, \sigma) \) and \( L_{yy}(x, \sigma) \).

As Gaussian filters are non-ideal in any case, and given success with LoG approximations
[52,53], Bay et al push the approximation even further with box filters as shown in Figure III.2.
These approximate second order Gaussian derivatives, and can be evaluated very fast using
integral images, independently of size.

SURF is a local invariant interest point detector-descriptor, similar to its predecessor the
accurate but slower SIFT (Scale Invariant Feature Transform described in [53, 73]) algorithm.
We utilize the SURF algorithm due to its unique rich descriptors and relatively quick processing
time allowing us to track and register points from subsequent frames. A SURF point is described
by a 128 element vector, containing information regarding the feature’s size, location and
orientation. The Figure III.3, shows a sample image with indicated SURF points from a video
sequence used for 3D reconstruction.

![Figure III.2: Left to Right: the discretized and cropped Gaussian second order partial derivatives in y-
direction and xy-direction, and Bay, Tuytelaars and Gool’s approximation thereof using box filters. The
grey regions are equal to zero. [4]](image-url)
III.2 FEATURE MATCHING AND DISPARITIES

Once SURF points have been identified, we examine the distance each point has traveled between frames to generate a disparity value for each feature point as described in our previous work [19, 20, 79]. The (2) Feature Matching step compares the SURF point descriptors between adjacent frames to identify nearest matching descriptors[61]. The feature matcher is implemented using squared Euclidean distance to enable a “bailout” threshold to maximize performance. In order to determine which SURF point matches are reliable and which are spurious, we track the path of a feature across a window of frames (in this case we use five consecutive frame to as the window size), retaining only those points which maintain an uninterrupted path. As a feature travels from frame to frame, we set a threshold based on the previous magnitude and direction of
the feature’s path. The filtering process described eliminates spuriously matched features. Specifically, we establish match relationships between SURF points in adjacent frames $A$ and $B$ and we identify the matching points $a \in A$ and $b = \arg \beta \in B \min |a - \beta|$ where the points are separated by a maximum feature value distance, $\delta > |a - \beta|$, as well as a maximum spatial distance. Here we use the Euclidean distance, both for the purpose of feature matching and disparity calculation; the Manhattan distance [24] has been shown to provide robust results, as well.

Then given a window size $n$, the set of all frame to frame correspondences $A$ has size $|A| = n - 1$ where

$$A_i = \cup_{a \in A_i, b \in A_{i+1}} (a, b)$$

(III.2)

for all $a \in A_i$ where

$$b = \arg \beta \in A_{i+1} \min |a - \beta|.$$  

(III.3)

In order to determine which SURF feature matches are reliable and which may be spuriously represented in the disparity map, we track the path of each feature across a window of $n$ frames, retaining only those points offering an uninterrupted tracked path across an entire given window. This is, for all $A_i \in \{A_1, A_2, ..., A_{n-2}\}$ we identify those point pairs $(a_i, b_i) \in A_i$ and $(a_{i+1}, b_{i+1}) \in A_{i+1}$ where $b_i = a_{i+1}$. This path tracking can be thought of as a pruning process, where point not on the uninterrupted path form $A_1$ to $A_n$ are removed from the sets of matched points $\{B_1, B_2, ..., B_{n-1}\}$.

For each feature point an array of coordinates corresponding to the feature’s location $(row, col)$ within each of the frames of a window is outputted and a “distance travelled” metric for the feature is computed. This distance value is referred to as a feature point’s disparity. When each tracked feature point has been associated with a disparity, a (3) Disparity Map...
illustrates the depths within a scene [45, 65, 74, 75]. In Figure III.4, we demonstrate the matching process between two frames. As the scene shifts through the video, points are tracked from frame to frame by matching their SURF descriptors. Figure III.5 illustrates a disparity maps for two scenes. Examining the disparities illustrates that feature points closer to the camera plane have large disparities, while feature points far away from the camera plane exhibit a much smaller disparity.

In Figure III.5a, we demonstrate laboratory setting with the camera traveling perpendicular the scene illustrating the disparity map of SURF points. Each line consists of four smaller segments representing the path of each feature point from frame to frame within a window. Figure III.5b shows the camera traveling in an outdoor scene with a disparity map of optical flow points. This disparity map is color coded, where green represents points closest to the camera followed by blue and black, respectively.

Figure III.4: SURF points are extracted in two consecutive video frames. Using the matching procedure described, feature points are matched and tracked from frame to frame. Input imagery captured by ATE and authorized by PRCI.
Figure III.5: A disparity map demonstrates the depth calculation concept of this framework. (a) A desk scene shows that objects closest to the camera plain such as the flags exhibit the largest disparities, while distant objects such as the bottle exhibit smaller disparities. (b) A frame from a UAS video demonstrates that the bridge appears closer to the camera than the surrounding banks. Input imagery provided by IDCAST.
III.3 DEPTH TRIANGULATION

In the fourth reconstruction step, we convert from two dimensional disparities to three dimensional Cartesian coordinates. We begin by converting the disparity map into a (4) Depth Map by assigning appropriate depth values to each feature. The depth is displayed along the z-axis in the 3D model. The conversion is done according to the following equation,

$$\text{Depth} = \frac{(\text{baseline}) \times (\text{focal length})}{\text{Disparity}} \quad \text{(III.4)}$$

With the assumptions of constant speed and focal length equation, Equation III.4 describes the inverse relationship between feature disparities and their depth. Depicted in [45] Figure III.6 illustrates the relationships described by Equation III.4 When a depth, z-coordinate, for each feature point is determined, we compute the x and y coordinates using the focal length information. The x and y coordinates in the model are dependent on the x and y image coordinates and their location relative to the center of the image. The current model’s world coordinates \((x_w, y_w, z_w)\) are determined from the following equations, where \((x_i, y_i)\) points are the image coordinates.

$$x_w = x_i - \frac{M}{2} \quad \text{(III.5)}$$

$$y_w = \frac{N}{2} - y_i \quad \text{(III.6)}$$

$$z_w = \text{depth} = f(\text{disparity}) \quad \text{(III.7)}$$

Here N represents the image height and M represents the image width. Figure III.7 illustrates the relationship between the axes, 3D model and imagery.
Figure III.6: Illustration of the relationship between focal length, the baseline and the disparities. (a) When using the same camera from two different perspectives, we can compute the disparities using the focal length and baseline (b) Disparity and depth have an inverse relationship.

Figure III.7: The x and y coordinates are computed in relation to the center of the image. In order to determine how much of the scene captured (size of the image) within the frame we utilize the focal length and computer depth.
III.4 POINT LOCALIZATION

In the fifth step of reconstruction, (5) Point Localization, we use the depth value computed for each feature point to determine the horizontal and vertical field of view at that particular depth. Furthermore, the image coordinates (row, column) of the feature are converted into appropriate Cartesian coordinates \((x, y)\). The computed sizes of a small object close to the camera and a large object far away from the camera, while appearing similar in the image, will significantly differ due to the field of view values increasing with depth. Figure III.8 contains an illustration of the camera’s focal length and its relation to the depth computations. This is a key step in determining the 3D model coordinates from an image.

![Figure III.8 An illustration of an increasing field of view as the captured data moves away from the camera sensor.](image)
III.5 POINT CLOUD MODEL

At the sixth step of the reconstruction, the initial 3D model appears as a single (6) Point Cloud. It is a collection of the (x,y,z) points, computed in the earlier steps, displayed in our own OpenGL implemented viewer. Rotation matrices around each axes [70] are as follows,

\[
Rotation\ around\ X: \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix} \quad (III.8)
\]

\[
Rotation\ around\ Y: \begin{bmatrix} \cos \beta & 0 & -\sin \beta \\ 0 & 1 & 0 \\ \sin \beta & 0 & \cos \beta \end{bmatrix} \quad (III.9)
\]

\[
Rotation\ around\ Z: \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (III.10)
\]

Although initial point clouds did not contain color, a Colorization sub-step process was implemented, in which the RGB components of the original image coordinates for each feature point were retrieved. Those values are passed along with the initial feature coordinates \((x_1,y_1)\) and final feature coordinates \((x_2,y_2)\). The colored point clouds presented in Figure III.9 show a reconstruction from a single window of frames. When comparing the original frames shown in Figure III.9a and Figure III.9c to the generated point clouds of the scenes in Figure III.9b and Figure III.9d, similarities can be observed. Each frame contains approximately 8000 detected SURF point, however only 400-800 remain after the feature matching filter is applied. The
Figure III.9: (a) An original frame from a video recorded on a desk in a laboratory setting. The closest objects to the camera are the flags followed by the green water bottle and lastly the red bottle. (b) The single window sparse 3D reconstruction model of the scene presented in Fig. 5a. (c) An original frame from an aerial UAS flight of a railroad bridge over a river. (d) A sparse 3D reconstruction of the bridge scene. It is difficult to see any structure within the model due to the small number of points.

resulting point clouds in Figure III.9b and Figure III.9d are very sparse, but as later result will show, the SURF points proved high accuracy with little noise.

In order to produce denser point clouds a Point Registration sub-step is required. In this sub-step, point clouds generated from two different windows are meshed together into on larger point cloud. This multi-window point cloud contains points from every set of frames in the video sequence. The registration process compares features of various single-window point clouds to
determine which features are new to the scene and which have already been registered through previous frames. The registration of features is done by comparing the feature descriptors of the SURF points. Figure III.10 depicts the registration processes. As the camera travels, most of the scene is unchanged from frame to frame and therefore only few new points are added as each new window is registered. As a result, a single dense multi-window point cloud is created which represent the 3D model of the recorded scene. The registration process creates a global list of points from the scene; however it also introduces new noise elements. Similarly to the feature matching procedure, these noise elements are caused by wrongly registered points introducing errors into the depth computations and by multiple registrations of the same features causing a point cloud blur. Results of a registered point cloud created using video captured from a moving vehicle are shown in Figure III.11.

![Figure III.10: When the reconstruction process begins, new points are registered. On advanced frames, the same features points are recognized and not repeatedly added. Only new points are added to the global list from proceeding point clouds.](image-url)
III.6 EVALUATION

The SURF extraction implementation has been evaluated on a variety of sample videos including UAS data and other scenarios. The video quality varies between 720p and 1080p, which yields on the order of 8,000 to 9,000 SURF points per frame, however only a subset of those points can be used for tracking the features. SURF performance is relatively slow, on the order of ~4 fps on a single core and ~24 fps on a dual quadcore. SURF calculations on separate frames are perfectly parallelizable. CUDASURF and other GPU-enabled implementation exist for additional performance gains and to enable a real-time reconstruction system.

Figure III.12 illustrates how we have evaluated the accuracy of the reconstruction model. The accuracy is measured in all three directions (x-, y- and z- axes). Figure III.12a shows the point cloud perpendicular to the scene similar to the camera’s point of view. The yellow lines represent the vertical measurements we made within the model to be compared to the real world values. The blue lines represent the horizontal measurements. Figure III.12b shows the point cloud view from above, with the flags located closest to the camera plain, followed by the green bottle and the red bottle. The orange lines represent the depth measures. Table III.1 show the measured model units, the model units converted into inches, the real world values (truth values) and the difference between the model and the real world.
Figure III.12: (a) A perpendicular view of the point cloud with indicators on distances used in the accuracy analysis. (b) A top view of the point cloud. The measurements made for the evaluation in the x-, y-, and z-axis are indicated with numbered lines.

From the Table III.1, we can conclude that the x and y-direction computations are accurate to within an inch to the real-world values, while the z-direction contains the most error and uncertainty. Note that the measurements in the horizontal and vertical axis contain subinch precision. This due to the fact that the point localization step contains less ambiguity and estimation comparison the disparity and depth computation steps. In the z-axis (depth) the table above indicates accuracy precision to within 1 to 3 inches.
In summary, we have described the reconstraction framework. From this chapter, we conclude that although SURF points are accurate and extremely distinctive, the matching process eliminates too many points. The resulting point cloud is not suitable for our applications. A more rigorous technique must focus on creating a point cloud containing more points. In the following chapters, we introduce additional steps into the framework technique to create a denser point clouds. Our solution to the sparcity of our reconstructed model is, as described in Chapter IV, Optical Flow feature points.

<table>
<thead>
<tr>
<th></th>
<th>Model (normalized units)</th>
<th>Model (inches)</th>
<th>Real World (inches)</th>
<th>Model - Real World (inches)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Horizontal</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Measurement Blue 1</td>
<td>0.1129</td>
<td>12.8706</td>
<td>13.1</td>
<td><strong>0.2294</strong></td>
</tr>
<tr>
<td>Measurement Blue 2</td>
<td>0.0342</td>
<td>3.8988</td>
<td>3.7</td>
<td><strong>-0.1988</strong></td>
</tr>
<tr>
<td>Measurement Blue 3</td>
<td>0.0324</td>
<td>3.6936</td>
<td>3.7</td>
<td><strong>0.0064</strong></td>
</tr>
<tr>
<td><strong>Vertical</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Measurement Yellow 1</td>
<td>0.0842</td>
<td>9.5988</td>
<td>9.5</td>
<td><strong>-0.0988</strong></td>
</tr>
<tr>
<td>Measurement Yellow 2</td>
<td>0.0132</td>
<td>1.5048</td>
<td>1.5</td>
<td><strong>-0.0048</strong></td>
</tr>
<tr>
<td>Measurement Yellow 3</td>
<td>0.0751</td>
<td>8.5614</td>
<td>8.3</td>
<td><strong>-0.2614</strong></td>
</tr>
<tr>
<td><strong>Depth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Measurement Orange 1</td>
<td>0.1195</td>
<td>19.0005</td>
<td>22.5</td>
<td><strong>3.4995</strong></td>
</tr>
<tr>
<td>Measurement Orange 2</td>
<td>0.0291</td>
<td>4.6269</td>
<td>6</td>
<td><strong>1.3731</strong></td>
</tr>
<tr>
<td>Measurement Orange 3</td>
<td>0.0904</td>
<td>14.3736</td>
<td>16.5</td>
<td><strong>2.1264</strong></td>
</tr>
</tbody>
</table>

Table III.1: The table contains measured and computed values of a laboratory desk scene.
A number of methods have been proposed to generate depth information using two-dimensional video from stereoscopic systems by using disparity map, or a map in which the depth information is calculated from spatially offset images of the same scene. In the case of a monocular system, they may be temporally offset, as well. These algorithms differ in the features they use to generate disparities, and the manner in which these features are matched from one frame to another. Broadly speaking, they may be divided into two categories: the local case, in which every pixel or feature point is matched independently to a point in the next frame using some feature descriptor or neighborhood related measure, and the global case, in which a depth map is iteratively calculated and updated so as to minimize some energy function.

In the local case, many algorithms are correlation based, using a distance measure between matching neighborhoods as the disparity values; more robust correlation methods have used M-estimators as described by Black and Anandan [6] and least medium squares as described by Roy and Cox [72]. Among global methods, the primary difference is in the choice of energy function. Some methods independently minimize a 1-dimensional energy function along the rows and columns of the depth map as discussed by Ishikaw and Geiger [37]. A common technique for energy minimization is simulated annealing [27], in which each point in the depth map is analogous to the state of some physical system, and the minimization function is analogous to the internal energy of the system in that state, Boykov et al. use graph cuts to compute a local
minimum in the depth map such that the energy function is piecewise smoothed using only pairs of pixels, yielding a depth map more tolerant of discontinuities than correlation techniques, and less prone to noise than simulated annealing techniques [7]. Zitnick and Kanade describe a global, cooperative method not using energy minimization, in which several feature enforce various physical constraints. For example, a uniqueness constraint, requires that a non-occluded pixel in one frame should map to a unique pixel in the next frame [87].

In order to create a model with more points, an additional algorithmic step needs to be included. We propose a technique in which a point cloud is generated using both global and local information. Specifically, we generate optical flow disparities using the Horn-Shunck optical flow estimation technique [35, 36] and evaluate the quality of these features for disparity calculations using the SURF keypoint detection method. Figure IV.1 shows where the optical flow points fit into the system flow discussed in the previous chapter.

Figure IV.1: Steps to create a 3D model from two-dimensional video data captured by a single moving camera. The more advanced technique generates dense point cloud models with the marked additional steps of Horn-Schunck Optical Flow.
IV.1 OPTICAL FLOW POINTS

We utilize the Horn-Schunck Optical Flow Points method to generate a denser point cloud model. As described by Lucas and Kanade [54], this method for finding the optical flow patterns is extremely accurate when the apparent variation of brightness patterns is smooth almost everywhere in the image. In the case of a UAS, this optical flow method is robust and accurate when a sufficient amount of the scene is present in consecutive frames. Using this algorithmic step, we generate on the order of 1.5 to 2 million points from a single pair of frames. In addition to enriching the collection of points, this optical flow method computes instantaneous velocities for all visible points replacing the Feature Extraction/Matching steps for optical flow points.

Out of several known optical flow methodologies, we choose to utilize the Horn Schunck implementation. Derivations and methodology descriptions are outlined from the Horn-Schunck publication “Determining Optical Flow” [35]. Optical flow cannot be computed locally, since only one independent measurement is available from the image sequence at a point, while the flow velocity has two components. A second constraint is needed. A method for finding the optical flow pattern is presented which assumes that the apparent velocity of the brightness pattern varies smoothly almost everywhere in the image. An iterative implementation which successfully computes the optical flow for a number of image sequences used in our reconstruction technique. The algorithm is robust in that it can handle image sequences that are quantized rather coarsely in space and time. It is also insensitive to quantization of brightness levels and additive noise.

Optical flow is the distribution of apparent velocities of movement of brightness patterns in an image. Equations and their descriptions are originally described by [35]. Optical flow can arise from relative motion of objects and the viewer [28, 29]. Consequently, optical flow can
give important information about the spatial arrangement of the objects viewed and the rate of change of this arrangement [30]. Discontinuities in the optical flow can help in segmenting images into regions that correspond to different objects [65]. The derivation that relates the change in image brightness at a point to the motion of the brightness pattern is presented below.

[35] Let the image brightness at the point $(x, y)$ in the image plane at time $t$ be denoted by $E(x, y, t)$. Now consider what happens when the pattern moves. The brightness of a particular point in the pattern is constant, so that

$$\frac{dE}{dt} = 0. \quad (IV.1)$$

Consider a patch of the brightness pattern that is displaced a distance $\delta x$ in the $x$-direction and $\delta y$ in the $y$-direction in time $\delta t$. The brightness of the patch is assumed to remain constant so that

$$E(x, y, t) = E(x + \delta x, y + \delta y, t + \delta t). \quad (IV.2)$$

Expanding the right-hand side about the point $(x, y, t)$ we get,

$$E(x, y, t) = E(x, y, t) + \delta x \frac{\partial E}{\partial x} + \delta y \frac{\partial E}{\partial y} + \delta t \frac{\partial E}{\partial t} + \varepsilon \quad (IV.3)$$

Where $\varepsilon$ contains second and higher order terms in $\delta x, \delta y, \delta t$. After subtracting $E(x, y, t)$ from both sides and dividing through by $\delta t$

$$\frac{\partial E}{\partial x} \frac{\delta x}{\delta t} + \frac{\partial E}{\partial y} \frac{\delta y}{\delta t} + \frac{\partial E}{\partial t} + O(\delta t) = 0 \quad (IV.4)$$

where $O(\delta t)$ is a term of order $\delta t$ (we assume that $\delta x$ and $\delta y$ vary as $\delta t$). In the limit as $\delta t \to 0$ becomes

$$\frac{\partial E}{\partial x} \frac{dx}{dt} + \frac{\partial E}{\partial y} \frac{dy}{dt} + \frac{\partial E}{\partial t} = 0. \quad (IV.5)$$

If we let

$$u = \frac{dx}{dt} \text{ and } v = \frac{dy}{dt},$$

41
Then it is easy to see that we have a single linear equation in the two unknowns $u$ and $v$,

$$E_x u + E_y v + E_t = 0,$$  \hspace{1cm} (IV. 6)

where we have also introduced the additional abbreviations $E_x$, $E_y$, and $E_t$, for the partial derivatives of image brightness with respect to $x$, $y$ and $t$, respectively. The constraint on the local flow velocity expressed by this equation is illustrated in Figure IV.2. Writing the equation in another way,

$$(E_x, E_y) \cdot (u, v) = -E_t.$$  \hspace{1cm} (IV. 7)

Thus the component of the movement in the direction of the brightness gradient $(E_x, E_y)$ equals

$$-\frac{E_t}{\sqrt{E_x^2 + E_y^2}}.$$  \hspace{1cm} (IV. 8)

The component of the movement is undetermined in the direction of the iso-brightness contours, at right angles to the brightness gradient. As a consequence, the flow velocity $(u, v)$ cannot be computed locally without introducing additional constraints.

Figure IV.2: The basic rate of change of image brightness equation constrains the optical flow velocity. The velocity $(u, v)$ has to lie along a line perpendicular to the brightness gradient vector $(E_x, E_y)$. The distance of this line from the origin equals $E_t$ divided by the magnitude of $(E_x, E_y)$. 

42
The Smoothness Constraint

Horn describes, if every point of the brightness pattern can move independently, there is little hope of recovering the velocities. More commonly we view opaque objects of finite size undergoing rigid motion or deformation. In this case neighboring points on the objects have similar velocities and the velocity field of the brightness patterns in the image varies smoothly almost everywhere. Discontinuities in flow can be expected where one object occludes another. An algorithm based on a smoothness constraint is likely to have difficulties with occluding edges as a result. [35]

One way to express the additional constraint is to minimize the square of the magnitude of the gradient of the optical flow velocity:

\[
\left( \frac{\partial u}{\partial x} \right)^2 + \left( \frac{\partial u}{\partial y} \right)^2 \quad \text{and} \quad \left( \frac{\partial v}{\partial x} \right)^2 + \left( \frac{\partial v}{\partial y} \right)^2. \tag{IV.9}
\]

Another measure of the smoothness of the optical flow field is the sum of the squares of the Laplacians of the x – and y – components of the flow. The Laplacians of \( u \) and \( v \) are defined as

\[
\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \quad \text{and} \quad \nabla^2 v = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}. \tag{IV.10}
\]

In simple situations, both Laplacians are zero. If the viewer translates parallel to a flat object, rotates about a line perpendicular to the surface or travels orthogonally to the surface, then the second partial derivatives of both \( u \) and \( v \) vanish (assuming perspective projection in the image formation). Here the square of the magnitude of the gradient as smoothness measure.

Estimating the Derivatives of Brightness

To estimate the derivatives of brightness from the discrete set of image brightness measurements available. It is important that the estimates of \( E_x, E_y, \) and \( E_t \), be consistent. [35] They should all refer to the same point in the image at the same time. While there are many formulas for approximate differentiation, a set which gives an estimate of \( E_x, E_y, E_t \) at a point in
the center of a cube formed by eight measurements. The relationship in space and time between these measurements is shown in Figure IV.3. Each of the estimates is the average of four first differences taken over adjacent measurements in the cube.

\[
E_x \approx \frac{1}{4} \left\{ E_{i,j+1,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i+1,j,k+1} \right\},
\]

(IV.11)

\[
E_y \approx \frac{1}{4} \left\{ E_{i+1,j,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i,j+1,k} + E_{i+1,j,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i,j+1,k+1} \right\},
\]

(IV.12)

\[
E_t \approx \frac{1}{4} \left\{ E_{i,j,k+1} - E_{i,j,k} + E_{i+1,j,k+1} - E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i+1,j,k+1} \right\},
\]

(IV.13)

Figure IV.3: The three partial derivatives of images brightness at the center of the cube are each estimated from the average of first differences along four parallel edges of the cube. Here the column index \( j \) corresponds to the \( x \) direction in the image, the row index \( i \) to the \( y \) direction, while \( k \) lies in the \( time \) direction. Figure regenerated based on [35].
Estimating the Laplacian of the Flow Velocities

To approximate the Laplacians of $u$ and $v$. The approximation takes the following form

$$
\nabla^2 u \approx k (\bar{u}_{i,j,k} - u_{i,j,k}) \quad \text{and} \quad \nabla^2 v \approx k (\bar{v}_{i,j,k} - v_{i,j,k}),
$$

where the local averages $\bar{t}$- and $i]$ are defined as follows

$$
\bar{u}_{i,j,k} \approx \frac{1}{6} \{u_{i-1,j,k} + u_{i,j+1,k} + u_{i+1,j,k} + u_{i,j-1,k} \} + \frac{1}{12} \{u_{i-1,j-1,k} + u_{i-1,j+1,k} + u_{i+1,j+1,k} + u_{i+1,j-1,k}\},
$$

$$
\bar{v}_{i,j,k} \approx \frac{1}{6} \{v_{i-1,j,k} + v_{i,j+1,k} + v_{i+1,j,k} + v_{i,j-1,k} \} + \frac{1}{12} \{v_{i-1,j-1,k} + v_{i-1,j+1,k} + v_{i+1,j+1,k} + v_{i+1,j-1,k}\},
$$

The proportionality factor $k$ equals 3 if the average is computed as shown and that the unit of length equals the grid spacing interval [35]. Figure IV.4 illustrates the assignment of weights to neighboring points.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{1}{12}$</td>
<td>$\frac{1}{6}$</td>
<td>$\frac{1}{12}$</td>
</tr>
<tr>
<td>$\frac{1}{6}$</td>
<td>-1</td>
<td>$\frac{1}{6}$</td>
</tr>
<tr>
<td>$\frac{1}{12}$</td>
<td>$\frac{1}{6}$</td>
<td>$\frac{1}{12}$</td>
</tr>
</tbody>
</table>

Figure IV.4: The Laplacian is estimated by subtracting the value at a point from a weighted average of the values at neighboring points. Shown here are suitable weights by which values can be multiplied. Figure regenerated based on [35].
Constrained Minimization

In practice the image brightness measurements will be corrupted by quantization error and noise so that the sum of the errors to be zero. This quantity will tend to have an error magnitude that is proportional to the noise in the measurement. This fact guides us in choosing a suitable weighting factor, denoted by $a^2$. When $a^2$ tends to zero we obtain the solution to a constrained minimization problem [35]. Approximating the Laplacian by the difference of the velocity at a point and the average of its neighbors then gives

\[
(E_x^2 + E_y^2)(u - \bar{u}) = -E_x[E_x \bar{u} + E_y \bar{v} + E_t], \tag{IV.17}
\]

\[
(E_x^2 + E_y^2)(v - \bar{v}) = -E_y[E_x \bar{u} + E_y \bar{v} + E_t]. \tag{IV.18}
\]

IV.2 DENSE POINT CLOUD MODELS

In this section we illustrate the results of optical flow on real world images and demonstrated the enhanced effect created by adding optical flow into the reconstruction algorithm.

In parts of the image where the brightness gradient is zero, the velocity estimates will simply be averages of the neighboring velocity estimates. There is no local information to constrain the apparent velocity of motion of the brightness pattern in these areas. Eventually the values around such a region will propagate inwards. If the velocities on the border of the region are all equal to the same value, then points in the region will be assigned that value too, after a sufficient number of iterations. Velocity information is thus filled in from the boundary of a region of constant brightness. Results of the optical flow implementation are shown in Figure IV.5, Figure IV.6 and Figure IV.7. In Figure IV.5, we observe a frame from a street scene. Each optical flow point is associated with a direction and magnitude. The points are also color-coded based on the magnitude of the disparities. Long disparities are indicated in red, slightly shorter
disparities are indicated by green, followed by blue and a single pixel disparity is indicated in black. The disparities show a strong indication of depth of particular objects in the scene. Parts of the road closest to the camera are marked in red, the trees are green, the building behind is blue, and the background is black.

These new optical flow features points, in addition to the earlier SURF points, create a dense 3D model. Figure IV.6a depicts trees along the side of a road. The image on the left illustrates the original frame while the image on the right shows the reconstructed model. Note the density and details within the trees in that scene. Similarly, Figure IV.6a demonstrates a bank building which produces a model dense enough to be able to read the name on the building.

Figure IV.5: A disparity map of optical flow points demonstrates the depth of feature points from the camera. An outdoor street scene shows that objects closest to the camera plane such as the trees exhibit the long disparities, while object further away such as the buildings exhibit shorter disparities.
Figure IV.6: The following are point clouds generated from the Horn-Schunck Optical Flow method. (a) and (c) show the original input frame while (b) and (d) are reconstruction shown at a perpendicular angle as the camera travels along a city street.

**Reconstruction from Aerial Imagery**

Additionally, Figure IV.7a shows an aerial scene. This frame comes from an UAS recorded video of a railroad. On the left, we see the original frame from the video, while in the right we observe the model viewed from the same angle as the original image. Note the visibility of the tracks and depressions that occur in the scene. Figure IV.7b shows the same point cloud model rotated 90 degrees. From the profile view, we can see that the green grassy part of the scene represents a hill and the incline is noticeable from the model. As the point cloud rotates, the elevation for the hill becomes apparent as well as the variation in size of columns supporting the railroad.
Figure IV.7: (a) Original video frame. (b) Reconstructed model viewed from the same angle. (c) When the model is rotated the incline of the hill becomes visible with the railroad horizontal in the background.

**Interior Reconstruction Models**

In another test scenario, a set of indoor video sequences have been captured from mobile robotic platform are used to reconstruct an indoor scene. An illustration of the scene captured is shown in Figure IV.8. In the video frames, we illustrate the front lobby of Kettering Laboratories at the University of Dayton. In the scene, the major areas of interest are the two elevator doors which are slightly indented into the surrounding walls. The elevators are further away from the camera plane than the walls. Another interesting area is where the trash and recycle bins, as one is occluding the other. In the next figure, Figure IV.9, we describe the optical flow patterns associated with each frame. Finally, we discuss and illustrate the detailed point cloud generated
Figure IV.8: Sample frames in sequential order (a) to (d) from a video capture by a mobile platform.

Figure IV.9: The images in sequential order (a) to (d) correspond to the video frames in Figure IV.8. These images illustrate the disparity map of the optical flow points. These disparity maps are color coded, where red (long arrows) represents points closest to the camera followed by green (medium length arrows) and blue (short arrows), respectively.
Figure IV.10: (a) A subsection of original frame showing an elevator call buttons. (b) The same subsection of an elevator call buttons within the point cloud model.

from the input imagery. The input videos of a scene are captured by first moving in one direction with the camera positioned nearly perpendicularly to the direction of motion.

The colorful point cloud presented in Figure IV.10 show a reconstruction from a subsection in a single window of frames. Figure IV.10a illustrates the original input frame while the reconstruction can been seen in Figure IV.10b. This figure shows a dense point cloud containing tracked and matched features points of a specific object.

The 3D reconstruction model of an interior hallway is presented below. We show the success of the reconstruction framework in determine the depth of feature points within the scene. Figure IV.11a illustrates the compilation of the original frames to create a sense for the entire real-world scene. Figure IV.11b demonstrates a point cloud composed of 435447 points of the elevator scene. When observing the reconstruction model, several things stand out. First, there
are large vacant spaces where a wall should be. This is caused due to the textureless nature of some object in the scene. For example, the walls around the elevator doors contain no texture. Therefore, feature from parts of the wall are unable to correctly match in subsequent frames. Similarly, parts of the elevator door contain no identifiable texture and therefore the disparity values of those features are inaccurately small. Extremely small disparity values cause the feature points to appear extremely far away.

Figure IV.11: (a) Several frame from the input video concatenated together to illustrate the real-world scene. (b) The 3D point cloud model of the scene created from the reconstruction framework described.

To illustrate the accurate depth computation within the model, we focus on the small detailed depth differences in the scene. First, we observe the difference between the elevator doors and the surrounding walls. Notice that in the real-world scene the elevator doors are indented several inches into the wall. In Figure IV.12a, we illustrate a view from above the
model. The figure focuses on the corners between the elevator doors and the surrounding wall, where it clear that the two are correctly placed in separate depth “layers.” In Figure IV.12b, we show two examples of the wall corner being dense enough to block the view of the elevator doors when the model rotated.

To illustrate the details a dense point cloud model provides we compare the original image to the reconstruction model in Figure IV.13. We have selected a scene with writings to show the legibility and density of the cloud. Note that the labels on the trash bin are clearly readable within the point cloud model.

Figure IV.12: (a) A view from the top of the 3D point cloud model. Note that different objects are correctly placed in different depth layers. (b) Several views of the 3D model illustrating the density of the point cloud.
Figure IV.13: A comparison between the original frame and the reconstructed model. A illustration of the density of the model allows the visibility of the labels on the trash bins.

Reconstruction from Ground Imagery

We conclude the results demonstrations with what has become the signature model of the reconstruction algorithm done produced by the University of Dayton. This model is constructed using a video recorded from a car driving on Main St. in downtown Dayton. The captured scene contains various details, including buildings, trees, benches, light poles, a parking lot, etc. The model displayed in Figure IV.14a consists of over two million feature points. These points are dense enough to make out distinct objects within the scene. Figure IV.14b and Figure IV.14c demonstrate an angled view of the street and an overhead view of the street, respectively.

We all illustrate the added density and depth variation. For example, in Figure IV.15a a zoomed out view of the building allows the viewer to clearly distinguish the “Performance Place” sign as well as the plant pots, light pole and fire hydrant closer to the camera. To illustrate depths more clearly, we focus on the white car parked in the parking lot behind the trees. In Figure IV.15b the car is marked by a red box. From a perpendicular view it is difficult to distinguish the depth at which the car is located relative to the trees in front, however when we rotate the point cloud notice how the car disappears behind the trees. Similar to a human’s daily
experience of obstruction of view when closer object obstructs one from seeing the scene behind
the object, this model can be rotate and examined in any angle.

Figure IV.14: Different views of a point cloud model generated from a street scene depicting dense point
clusters as objects at various depths.
IV.3 METRICS, ANALYSIS AND EVALUATION

The reconstruction algorithm has been evaluated on a variety of sample videos including data captured by a UAS, UGV, indoor, outdoor and other scenarios. We plan to compute the accuracy of the reconstruction model by measuring the real-world (RW) values in all three directions ($x, y$, and $z$ axes). We begin by evaluating the horizontal direction of the street point cloud. We made ten RW measurements and compare them with the same ten measurements within the model. As expected due to the eye appeal model angle, the horizontal direction within
the model exhibits almost the exact values as measured in the RW. Figure IV.16 marks the ten measurements made within the point cloud and charts the differences between the model and RW values. Similarly, Figure IV.17 illustrates the measurement conducted in the vertical direction.

Figure IV.16: Illustration of the point cloud perpendicular to the plane of the camera.

Figure IV.17: The orange lines represent the vertical measurements made within the model.
The third accuracy metric is evaluated in the depth direction. As shown and charted in Figure IV.18, ten RW measures were taken and compared with the same measurements within the point cloud. Unlike the horizontal and vertical direction, the depth exhibits slightly higher error measurements. The conversion factor and average error in feet for each direction are shown in Table IV.1.

Figure IV.18: We illustrate the point cloud view from above. The orange lines represent the depth measurements that were compared to the real-world values.
Table IV.1: Summarizing the accuracy of the point cloud model. For each direction, the model unit to real-world units conversion and average error (in feet) are shown.

<table>
<thead>
<tr>
<th>Direction</th>
<th>Conversion</th>
<th>Average Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal</td>
<td>1 Model Unit = 0.0206 feet</td>
<td>0.7612 feet</td>
</tr>
<tr>
<td>Vertical</td>
<td>1 Model Unit = 0.0204 feet</td>
<td>1.4750 feet</td>
</tr>
<tr>
<td>Depth</td>
<td>1 Model Unit = 0.0146 feet</td>
<td>4.7809 feet</td>
</tr>
</tbody>
</table>

The Layer Effect

When analyzing and evaluating the performance of the reconstruction, we consider the accuracy with relation to real-world values. As indicated in Table IV.1, the horizontal and vertical measurements experience nearly perfect accuracy. However, the depth experiences up to 4.7 feet of error. By taking a closer look at the reason for this increase, we determine that the depth of the point cloud model is directly related the variance of the disparities. (Note that disparities are directly related to the image resolution and baseline between frames.) When we the resolution of the image does allow to distinguish between, for example, 2 feet and 4 feet of depth both point get assigned the same z coordinate. As a result, when multiple point are assigned the same depth coordinate the point cloud experiences a discrete number of depth layers. We define the “layer effect” as a state in which the point cloud model experiences discrete depth layers. This is clearly visible in figures that display an overhead view of the model, as demonstrated in Figure IV.19c.
Figure IV.19: (a) An original frame from a video recorded of N. Main St in downtown Dayton, OH. The scene contains numerous objects of different color and texture at varying depths. (b) This model is computed from two consecutive frames of video. (c) A view of the same 3D reconstruction from the top. Note that due to the discrete nature of feature disparities points are dispersed into layers.

We conclude the chapter by highlighting the solution the problem posed at the end of Chapter III. By adding optical flow features in the algorithmic framework, we were able to create a dense point cloud model. As the results indicated, the new point clouds allow the user to distinguish the scene by clearly identifying buildings, cars and trees. With the introduction of a denser point cloud, we also introduce the “layer effect” in which numerous point contain the same depth coordinate. In the next chapter we propose a solution to this issue by increasing the depth resolution. We do this by addition of a resolution enhancement techniques applied to the input imagery.
CHAPTER V
DEPTH RESOLUTION ENHANCEMENT

In this chapter we introduce the novel concept of depth resolution enhancement. In order to eliminate the “layer effect” caused by discrete and limited disparities described at the end of Chapter IV, we apply a super resolution technique to the original frame. By increasing the resolution of the input frames, we create more “layers” thus eliminating the discreteness. Due to the increased resolution, more feature points are obtained to create a more dense point cloud as well as a larger number of layers. In general, a super resolution technique [9, 31, 38, 39, 66] is the process of obtaining a high resolution image from an image or a sequence of low resolution images. Traditionally, this process takes place by using multiple cameras mounted closely together and oriented in the same direction. Thus, a higher resolution image can be obtained by combining pixel information from each of the camera’s imagery. Our method uses a single frame to improve the quality of the image by interpolating the image. Unlike the standard linear interpolation [58] and bicubic interpolation[43], this super resolution technique uses the image directional variance to determine the inserted pixel values.

Although computationally expensive, the addition of super resolution will create more disparity resolution and thus more depth resolution. In Figure V.1, we illustrate the addition preprocessing step into the existing architecture. We determined that the ideal fit of super resolution within our algorithm would be in the very first step prior to any feature extraction. In this chapter, we will describe the super resolution technique, prove that super resolution enhances
the performance of each of the algorithmic steps and thus improves the point cloud model. We conclude with an evaluation of the resulting point cloud models.

Figure V.1: The algorithmic architecture with the addition of super resolution designed to enhance the depth resolution of the point cloud models.

V.1 SUPER RESOLUTION

Super-resolution is a process of image enhancement by which low quality, low resolution images are used to generate a high quality, high resolution image. There have been several techniques for super-resolution presented in the literature [9, 31, 39, 66]. These can be classified in two categories: single image super resolution and super resolution from several frames. In the first case, there is no additional information available to enhance the resolution [38].
As describe by Islam et al, Kernel regression analysis is a nonparametric regression method to estimate the value of an unknown function $f(x)$ at any given point based on the observations. For two dimensional cases, the regression model is

$$Y_i = f(x_i) + \epsilon_i, \quad i = 1,2,\ldots,N, \quad x_i = [x_{1i}, x_{2i}]^T$$  \hspace{1cm} (V.1)

where $\{(x_i), i = 1,2,\ldots,N\}$ are the design points in terms of pixel position, $\{Y_i, i = 1,2,\ldots,N\}$ are observations also in pixel values of the response variable $Y$, $f$ is a regression function and $\{\epsilon_i, i = 1,2,\ldots,N\}$ are independent identically distributed random errors and $N$ is the number of samples in terms of frames. The generalization of kernel estimate $\hat{f}(x)$ is given by solving the following minimization problem.

$$\min_{q_0,q_1,\ldots,q_l} \sum_{i=1}^{N} \left[ Y_i - \{q_0 + q_1(x_i - x) + \cdots + q_l(x_i - x)^l\} \right]^2 K \left( \frac{x_i - x}{h} \right)$$  \hspace{1cm} (V.2)

where $K(\cdot)$ is the kernel function with bandwidth $h$ and $l$ is a positive integer which determines the order of the kernel estimator. The parameter $h$ is also known as smoothing parameter since the value of it determines the smoothness of final output. Above equation is solved to determine the unknown regression coefficients $q_0, q_1, q_2, \ldots, q_l$. Islam et al used the Gaussian kernel throughout the paper.

In Figure V.2, we shows basic block diagram of the super resolution technique. The technique uses feature based covariance estimation for adaptive kernel regression. The adaptive kernel regression function is

$$K_{H_i}(x_i - x) = \frac{\sqrt{\det(C_i)}}{2\pi h^2 \mu_i^2} \exp \left\{ -\frac{(x_i - x)C_i(x_i - x)}{2h^2 \mu_i^2} \right\}$$  \hspace{1cm} (V.3)

Where $h$ is the global smoothing parameter and $\mu$ is the scalar constant set to 1. The only data dependent parameter in the kernel function is covariance matrix $C_i$, which is estimated locally by decomposing image features and the geometry of the image. Traditionally super resolution is
done in spatial domain, where edge or gradient information is used, but in case of texture features image gradient contain hardly any useful information. This super resolution technique uses image feature in space frequency domain to reconstruct high resolution images. When taking the Fourier transform $F(u, v)$ of an image, $I_i(x_1, x_2)$. The real and imaginary components of this transformation can be expressed in terms of magnitude and phase as follows:

$$Re(u, v) = |F(u, v)| \times \cos[\varphi(u, v)]$$  \hspace{1cm} (V.4)

$$Im(u, v) = |F(u, v)| \times \sin[\varphi(u, v)]$$  \hspace{1cm} (V.5)

The magnitude spectrum does not contain any significant feature, so phase spectrum is utilized in future computations. The phase components corresponding to real and imaginary parts of the transformation are

$$P_\varphi = \cos[\varphi(u, v)] \hspace{1cm} (V.6)$$

$$Q_\varphi = \sin[\varphi(u, v)]. \hspace{1cm} (V.7)$$

The covariance matrix is estimated separately for the $P_\varphi$ and the $Q_\varphi$ components. Therefore, the local covariance matrix is calculated as following:

$$C_i = C_{1i} + C_{2i} \hspace{1cm} (V.8)$$

![Super Resolution Analysis](image)

**Figure V.2:** Structure of super resolution module
Furthermore, the directional variance is computed using the following equations and illustrated in Figure V.3.

\begin{align*}
\text{Var}(I^0_h) &= \frac{1}{n} \sum_{k=1}^{n} \left( I^0_h(K) - \mu^0_h \right)^2 \quad \text{where } \mu^0_h = E[I^0_h] \quad (V.9) \\
\text{Var}(I^{45}_h) &= \frac{1}{n} \sum_{k=1}^{n} \left( I^{45}_h(K) - \mu^{45}_h \right)^2 \quad \text{where } \mu^{45}_h = E[I^{45}_h] \quad (V.10) \\
\text{Var}(I^{90}_h) &= \frac{1}{n} \sum_{k=1}^{n} \left( I^{90}_h(K) - \mu^{90}_h \right)^2 \quad \text{where } \mu^{90}_h = E[I^{90}_h] \quad (V.11) \\
\text{Var}(I^{135}_h) &= \frac{1}{n} \sum_{k=1}^{n} \left( I^{135}_h(K) - \mu^{135}_h \right)^2 \quad \text{where } \mu^{135}_h = E[I^{135}_h] \quad (V.12)
\end{align*}

The covariance matrix is computed as follows:

\begin{align*}
v_h &= \left[ \text{Var}(I^0_h), \text{Var}(I^{45}_h), \text{Var}(I^{90}_h), \text{Var}(I^{135}_h) \right]^T \quad (V.13) \\
v_l &= \left[ \text{Var}(I^0_l), \text{Var}(I^{45}_l), \text{Var}(I^{90}_l), \text{Var}(I^{135}_l) \right]^T \quad (V.14) \\
C_{fi} &= \begin{bmatrix}
\text{Var}(v_h) & c\sqrt{\text{Var}(v_h) \cdot \text{Var}(v_l)} \\
c\sqrt{\text{Var}(v_h) \cdot \text{Var}(v_l)} & \text{Var}(v_l)
\end{bmatrix} \quad (V.15)
\end{align*}

Figure V.3: There are four directional variance bins used for the computations of the covariance matrix.
The algorithm can be summarized as follows. The original resolution image is enhanced by a resolution factor $r$. For each $9 \times 9$ neighborhood patch in the image, we calculate the real and imaginary components of the Fourier phase angle and normalize them. Next, the algorithm calculates four variances, $4 \times 1$ vector, taking all 9 pixels along horizontal, vertical and two diagonal directions about the center pixel for both real and imaginary components. For the same patch, the algorithm estimates four variances, also $4 \times 1$ vector, taking only the $r^{th}$ pixels along the same respective directions for real and imaginary components. Next the covariance of the two vectors is calculated. This gives covariance matrix at the center pixel. By adding the two covariance matrices found, kernel function is able to be computed. The algorithm estimates unknown pixels using kernel regression. The result is a high resolution image of order $r$. In the following section we analyze the effect of super resolution on the effectiveness of various components in our reconstruction technique.

V.2 ANALYSIS OF SUPER RESOLUTION

As described in the referenced algorithm [38, 39], we use feature based covariance learning for adaptive kernel regression. As a result a low resolution input image becomes a higher resolution image through nonlinear interpolation of the original pixel intensities. Figure V.4 illustrates the idea of super resolution on low resolution images.

Figure V.4: Super resolution results in a higher resolution input image.
We begin the analysis by visually inspecting the results of the super resolution. In Figure V.5, we illustrate two aerial scene to which super resolution has been applied. The images on the left are of their native resolution, while the images on the right are the results of the super resolution.

Figure V.5: Several images to illustrate the effects of super resolution. Increase in resolution from (a) 100x100 to (b) 400x400. Second example, (c) 119x211 on the left to (d) 796x844 on the right. Imagery provided by AFRL.
Figure V.5b is perhaps a tough example to analyze the difference between the original and the enhanced, but the effects are visible on the edges of the airplane wings. Note that in Figure V.5b the increase is clearly visibility in the calibration strips example. Super resolution has provided a “deeper” look into the scene. The strips are blurred three measures above the arrows in the original image, while the on the right the separate strips are distinguishable at the arrow marks.

**Evaluation Strategies**

We evaluate the effects of super resolution by observing the effects on feature extraction [61], such as Harris Corners [34], SIFT [52, 53] and SURF points [4]. In the images below, we illustrate the tremendous improvement in the number and quality of features as a result of super resolution.

Significant work was put forth into evaluating single image super resolution in an academic sense; that is evaluating it in the same manner super resolution algorithms are often investigated and compared in published literature. A common way to compare super resolution algorithms in literature is to downsample the original image, perform super resolution and compare to the original image (and other methods). In our case, the comparison included mean square error and number of Harris Corners generated. This was done for several scenes including Blue Devil [55], Columbus Large Image Format (CLIF) [10] and Full Motion Video (FMV) datasets. Figure V.6 the left side shows the Harris Corners generated from a CLIF image while the right side shows the Harris Corners generated from a \( r = 2 \) super resolution image. An example of a car from Wide Area Motion Imagery (WAMI) data is shown in Figure V.7. Figure V.7a shows the original image and Figure V.7b shows the \( r = 2 \) super resolution image. In all cases, the super resolution images produce more Harris Corners and features.
Figure V.6: Harris Corners for Columbus Large Format Imagery. Left images (a) (c) (e) are of the original resolution, while images on the right (b) (d) (f) are processed after the applied super resolution.
Results were generally positive for single image super resolution and the feature extraction matrix can be observed in Table V.1, however upon further review it was decided that the common mean squared error evaluation method was not the best way to evaluate super resolution performance for this project since the reference image is in application unknown. Table V.1 shows the number of Harris Corner features extracted. When compared with the original value, the super resolution technique we decided to use produces the largest number of points. The next step forward is to determine the proper experiments and evaluation techniques needed to evaluate the size ($r = 2$ or $r = 4$ of original image size) super resolution imagery and its contributions to the reconstruction problem.

Figure V.7: A vehicle contains more feature points in the higher resolution image.
<table>
<thead>
<tr>
<th>Corner Extraction (X2)</th>
<th>Down Sampled</th>
<th>Super Resolution</th>
<th>Bicubic Interpolation</th>
<th>Linear Interpolation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration Strips</td>
<td>12</td>
<td>23</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>Full Motion</td>
<td>18</td>
<td>48</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Blue Devil</td>
<td>14</td>
<td>26</td>
<td>14</td>
<td>4</td>
</tr>
</tbody>
</table>

Table V.1: Number of Harris Corners extracted in three datasets after applying various interpolation techniques. The super resolution algorithm described provides the highest number of extracted points.

We continue the analysis by observing the effect of super resolution on an original input frame. In Figure V.8, we have cropped and scaled a small portion on the original scene to highlight the effects of the super resolution. Figure V.8a illustrates the original image. Note effects of super resolution edges between light and dark areas of the image. In Figure V.8b, we have increased the resolution by a scale of 4. That is, by increasing the height and width of the image by two to 2440 x 1440 pixels, we have created an image of four times the number of pixels as the original. In Figure V.8c, we observe the effects of super resolution as we increase the resolution by sixteen times the original. That is, the height and width of the original image are increased by four to a 4880 x 2880 input frame.

As observed in Figure V.8, the number of pixels in the input increases while maintaining the spatial and pixel intensity relationships of the original input. As a result the disparity resolution of tracked features also increases. Previously, the limited resolution created discrete disparities that produced the “layer effect.” When multiple feature points traveled the same discrete distance from frame to frame, they were associated with the same disparity value and
eventually depth layer. With the super resolution technique, the feature matching procedure was able to more precisely match the new location of the features. Therefore, while previously a feature point was matched with one particular location after applying super resolution that final location was represented by sixteen pixels instead of one. This more precise point matching procedure created more unique disparities and therefore more layers within the point cloud model.

Figure V.8: The effects of super resolution are illustrated. (a) The original input frame. (b) Resolution is increased by a factor of four after super resolution applied to the original input. The input image has become 2440 x 1440. (c) Resolution is increased by a factor of four after super resolution applied to the original input. The input image has become 4880 × 2880.
In Figure V.9, observe the effects of super resolution onto a single window point cloud. In this scene, the building labeled “Performance Place” has a rounded convex frontal wall and overhang. In Figure V.9a, we have reconstructed the scene using the original resolution. Observe that the convex wall and overhang are depicted as a flat surface. This is caused by all parts of the wall experiencing very similar disparities and all parts of the overhand also experiencing very similar disparities. After applying super resolution to the input image, we obtain the point cloud depicted in Figure V.9b. Note that the red overhang is now experiencing a convex curvature as well as the frontal wall of the theater building.

![Figure V.9: The effects of super resolution on the point cloud models. (a) 3D point cloud model from the original input image (b) 3D point cloud model from the super resolution input image. The resolution has been increased sixteen times that of the original creating more points in the point cloud and more unique layers.](image)

In the following section, we will illustrate the results of super resolution apply to the reconstruction procedure and analyze of the effects with relation to model’s density and the elimination of the layer effect.
V.3 ENHANCED DENSE RECONSTRUCTION RESULTS

The results are evaluated by comparing the point cloud model created from the original frame versus the model created from the super resolution frames. In this section we observe that the point cloud has more discrete disparities and the number of layers in the model increases when super resolution is applied to the input. In Table V.2, we analyze two scenes. Scene 1 corresponds to the single window point cloud created from the scene shown in Figure IV.8. This scene contains a theater building, a light pole, decoration bushes, fire hydrant, etc. Scene 2 represent the entire city block illustrated in Figure V.10a. This scene contains several buildings, trees, parked cars, light poles, etc. As shown in Table V.2, for Scene 1 super resolution technique increased the number of points within the model by a scale of 6.56. It also increased the number of layers by a scale of 4.06. In the larger Scene 2, the increase in points was more subtle. As shown, the number of points within the model increased by a scale of 2.28 and the number of layers increased by a scale of 3.93. The increase in the number of points is affected by the cloud registration procedure. As point clouds continue to get rendered together, fewer new layers are introduced with each new rendering.

<table>
<thead>
<tr>
<th>Scene</th>
<th>Number of Points</th>
<th>Number of Layers</th>
<th>Super Resolution Input</th>
<th>Number of Points</th>
<th>Number of Layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene 1</td>
<td>230,402</td>
<td>221</td>
<td>1,512,776</td>
<td>898</td>
<td></td>
</tr>
<tr>
<td>Scene 2</td>
<td>1,152,012</td>
<td>248</td>
<td>2,632,834</td>
<td>976</td>
<td></td>
</tr>
</tbody>
</table>

Table V.2: Comparison of point cloud models created from the original input models to models created from the super resolution inputs.
We examine the difference between the original point cloud model and the new point cloud model created from the super resolution input images. The models below are of an entire city block. In Figure V.10a and Figure V.10b, we observe the point cloud model created from the original frames. In Figure V.10c and V.10d we observe the point cloud model created from the super resolution input images.

The super resolution algorithm presented produces high resolution input frames that feed into a 3D reconstruction algorithm. This algorithm is able to create dense 3D point cloud models of the surrounding scene using a single moving camera. As we have shown in the sections above, super resolution has significantly improved the density of the point cloud models as well as elimination of the “layer effect” caused by limited discrete disparity values. Although the super resolution algorithm has enhanced the important feature of the model it has also magnified the noise of the model. This observation can be clearly observed in Figure V.10d. The overhead view illustrates that increase in depth layers, a tremendous improvement over its predecessor in Figure V.10b. However as mentioned with regard to noise, notice the elongated halos around objects. This noise is present in the previous versions of the point cloud, but much less noticeable. In the previous version, Figure V.10a and Figure V.10b, little noise speckle are present through the scene. With the addition of super resolution these noise speckles have also been enhance and appear much denser and over several layers. We discuss the development noise suppression in the next chapter.
Figure V.10: (a) Frontal view of the 3D reconstruction model created from the original resolution input frames. (b) Top view of the 3D reconstruction model created from the original resolution input frames. (c) Frontal view of the 3D reconstruction model created from the super resolution input frames. (d) Top view of the 3D reconstruction model created from the super resolution input frames. Note the difference in the number of layers and density of the point cloud.
In this chapter, we have presented and evaluated a super resolution technique that is designed to enhance the depth resolution of the resulting point cloud models. After assessing the effects of super resolution on various types of imagery, we concluded that by including super resolution into our algorithmic architecture we would increase the density (number of points tracked) and the depth resolution via an increase in the number of discrete disparities (increase the number of layers). We observed the effects of super resolution on the reconstruction process and observe the increase in depth layers as well. In our evaluation, we conclude that the depth resolution layers issue is no longer a prominent part of the model. However, the noise that previously was unnoticeable has also been enhanced to do the super resolution. In the next chapter, we describe our final steps to remove noise while maintaining a dense model. Our noise suppression efforts add steps to the algorithm on the preprocessing end as well as the post processing end. We further enhance the input imagery using distortion correction, nonlinear image enhancement and video stabilization to allow for better feature matching. On the post processing end, we identify the commonality in the noise speckles within the point cloud model and use dynamic filters to suppress that noise in the model.
CHAPTER VI

NOISE SUPPRESSION OF POINT CLOUD

In this chapter we presented several noise suppression techniques used to remove or reposition incorrectly placed points within the 3D point cloud model. We approach this task from several angles. On the imagery side, we understand that improving the image quality will provide better matches which in return prove more accurate point locations within the model. This is referred as preprocessing for noise removal. We present three techniques aimed to improve the ability to identify and track feature points. On the other hand, once the point cloud model is produces we would like a way to eliminate points based on their 3D neighborhood surroundings. This is referred to as post-processing noise suppression. In 3D space points belonging to the same object will experience similarities with regards to its color and texture information. Using this texture and location information, we can filter out outliers that appear as noise. In this chapter we describe the Preprocessing for Noise Removal techniques in Section VI.1 and go on to present the Post Processing Spiky Noise Suppression in Section VI.2. Figure VI.1 illustrates how the additional algorithmic steps fit into the reconstruction architecture. These techniques conclude our presentation on the methodology that composes the Dense Point-Cloud Representation (DPR) technique presented in this thesis.
VI.1 PREPROCESSING FOR NOISE REMOVAL

We implement additional structured steps to enhance the current model and relieve the system of many path, velocity or orientation constrains. The proposed ideas consist of three separate project directions: nonlinear video enhancement, barrel distortion correction and video

Figure VI.1: The algorithmic flow diagram of Dense Point-Cloud Representation technique. Additional noise removal steps presented in this chapter are highlighted in green.
stabilization,. By applying preprocessing techniques to the video frame prior to applying the reconstruction algorithm, we enhance image characterizes that eventually benefit our 3D model. First, by processing the video data with nonlinear image enhancement the reconstruction algorithm will allow for more features to be extracted. In addition, the preprocessing steps of lens distortion correction and video stabilization can eliminate many mismatched features as well as distorted disparities.

**Nonlinear Video Enhancement**

We first take a closer look at the video enhancement. Various enhancement techniques exist capable of performing non-linear enhancement. The University of Dayton Vision Lab’s image enhancement algorithms include [3, 68, 78, 81]. These image enhancement algorithms are based on a neighborhood dependent nonlinear model used to improve visual quality of images or video captured under extremely non-uniform lighting conditions. Analysis and testing of these algorithms determines which method will increase the number of extracted trackable feature points while remaining computationally inexpensive. Figure VI.2 demonstrates the effect of enhancement on a single video frame. On the top, shadowy regions behind the building and trees where feature points may not be detect are enhanced on the bottom. Regions containing shadows may limit the number of point we are able to extract, but with the use of enhancement those regions contain extractable points as well.

In conclusion, we determined the computational costs of enhancement out weight the small increases presented in Table VI.1. Although it is important to point out that feature extraction and tracking capabilities are extremely scene dependent and although it has proven to have little effect on the two scenes we tested on in this experiment, video enhancement is a proven technique for extracting more information from images.
Figure VI.2: Original image on the top (a) contains non-uniform lighting. The enhanced image (b) on the right contains more trackable feature points.
<table>
<thead>
<tr>
<th>Scene</th>
<th>Original Input</th>
<th>Enhancement Input</th>
<th>Original Input</th>
<th>Corrected Input</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of SURF Points</td>
<td>Number of Tracks (across 5 frames)</td>
<td>Numbers of SURF Points</td>
<td>Number of Tracks (across 5 frames)</td>
</tr>
<tr>
<td>Scene 1</td>
<td>156</td>
<td>23</td>
<td>172</td>
<td>24</td>
</tr>
<tr>
<td>Scene 2</td>
<td>257</td>
<td>45</td>
<td>289</td>
<td>51</td>
</tr>
<tr>
<td>Scene 3</td>
<td>498</td>
<td>135</td>
<td>492</td>
<td>131</td>
</tr>
<tr>
<td>Scene 4</td>
<td>316</td>
<td>109</td>
<td>290</td>
<td>109</td>
</tr>
</tbody>
</table>

Table VI.1: Comparison of point cloud models created from the original input models to models created from the enhancement (top) and distortion correction (bottom) inputs.

**Barrel Distortion Correction**

In addition to restricting lighting conditions, every camera lens has a slight mount of barrel distortion. In Figure VI.3, this subtle is illustrated by drawing a straight red line along the railroad tracks. The distorted image on the left produces a slight curve in relation to the railroad, while the image on the right illustrates an undistorted scene. Distortion is a systematic variation of magnification with position $m$ described as,

$$m = 1 + dr^2$$  \hspace{1cm} (VI. 1)

where $d$ is the fractional distortion and $r$ is the radius from the center. In the case of barrel distortion, the image magnification decreases with distance from the optical axis.

Table VI.1 shows the metrics changes in the number of SURF points extracted and tracked. Similarly to the enhancement result, these metrics are extremely scene and camera lens dependent and although they show little to no improvement in the tracking of SURF points,
distortion correction is an important tool to consider measuring disparity directions. In cases when the SURF filtering process is set to strict directional limits a significant changes can be seen in the number of tracked points. As shown in the table, in our case we utilize barrel distortion correction on in cases of known fish eye lens used for video capture.

Figure VI.3: Top, original frame exhibiting a subtle barrel distortion. Bottom, effects of the distortion correction algorithm implementation.
Video Stabilization

The third preprocessing technique of video stabilization will allow the retention of more feature points in the feature matching step. Many feature points are classified as “mismatched” due to slight vibrations of the camera. These vibrations cause feature points to obtain non-linear non-equal-spaced paths and therefore they are eliminated within the filtering. Video stabilization described in [81] is a SURF based stabilization, wherein a similar feature matching procedure determines the translation needed for each frame. As a result, objects within the scene remain and share similar image coordinates from frame to frame.

Video stabilization increases the number of matched points within SURF and the optical flow. Many similarities between the SURF registration and video stabilization existed, however in our algorithm pipeline the stabilization proved to be redundant. In the sense that SURF extraction and matching was used to stabilize the video with incredible computational costs by applying hymnographies to each frame of video, then SURF extraction, matching and optical flow was conducted to reconstruct the scene. Although an optimization is most certainly possible, this 3D reconstruction algorithm only utilized the stabilization capabilities in extremely shaky situations.

The effects of these techniques are illustrated in Section VI.3. In the following section, Section VI.3, we focus our attention on a post-processing noise suppression technique. All filtering and point cloud manipulation is done on the \((x, y, z)\) point \(s\) using their color \(RGB\) values and location information.
VI.2 POST-PROCESSING FOR SPIKY NOISE SUPPRESSION

In this section we present a noise suppression technique which eliminates unwanted and mismatched points from the point cloud model [16]. We begin by identifying the reason and source of noisy points. Noise is created when a feature point is mismatch from frame to frame. When a mismatch happens, the disparity value of the feature point is an outlier with respect to its neighbors. This causes the point to appear isolated in the point cloud. Due to the effects of super resolution the number of unwanted and mismatched points as increase due to the direction relation with the input image resolution. A mismatch occurs when the matching algorithm determines that the “best” match for a particular feature is not the same feature in the subsequent frame. Mismatches have a high frequency of occurrence in regions with little texture. These regions have feature points with very similar descriptors. In Figure VI.4a we highlight some of these regions. Notice that the texture in these regions is virtually the same. Points from one region can easily be mismatched within the region as well as with points in other similar regions. In order maintain only regions of uniqueness, we perform Canny edge detection [8, 56] on the image. In this way we maintain the “important” and unique aspects of the scene while eliminating the regions that cause mismatches. We can fill in these regions once the depth values of the edge have been established. In Figure VI.4b we illustrate those effects. Using edge information alone, we are still able to correctly interpret the scene and distinguish depths. We present a technique that suppressed the “spikes” and scattered points from textureless regions.
Figure VI.4: (a) Illustration of textureless regions which cause mismatches leading to noise points within the point cloud model. These regions are very similar to one another and cause confusion when comparing the descriptors of the feature points within. (b) The same scene as illustrated in part ‘a’ but the textureless regions are removed from the image.
The noise associated small mismatched regions is called spiky noise. Due to the small size of region which is being mismatch, the points within the region follow a curvature which peaks in the center of the region. Notice these spikes in the point cloud model presented in Figure IV.25d. The technique presented below eliminates those spikes from the model.

We illustrate the removal process in Figure VI.5. We have created a filter to cluster the point cloud model into layers. By clustering the model into $N$ distinct layers, we are able to perform image noise removal techniques. In Figure VI.5a, we illustrate one of those layers. It is composed of thousands of point clustered together at a particular depth within the model. The resulting image is mostly black; however $(X,Y)$ coordinates for points within the specified region are displayed. We show the extent of these points in Figure VI.5b by applying dilation. Notice that small speckles are floating between larger object in the scene. By performing the “opening” morphological operation we are able to eliminate those noise points. The “opening” operation is composed of first eroding the image with a structural element shown in Figure VI.5c. To regain the losses of the erosion we apply dilation with the same size structuring element. Noise small then half the structuring element are eliminated through this technique. The final result is shown in Figure VI.5d. Our final step uses the original depth information to return the model to 3D space. Only unfiltered points are plotted back in the model. The noise suppressed resulting point cloud models are presented in the following section.
Figure VI.5: (a) By clustering multiple depth layers into one we are able to obtain a 2D image in which spiky noise appear as small speckles (b) We illustrate the clustering of multiple layers into one which contains true points as well as many noise points. (c) The morphological “opening” operation is erosion, in which as structuring element removes all the noise and edge of larger objects. (d) The second stage of the morphological “opening” operation is dilation.
VI.3 FILTERED POINT CLOUD RESULTS

In this section, we present the final point cloud model results. These models represent the resulting work of the Dense Point-cloud Representation (DPR). These evaluation as best conducted by observing the effects on the actually point cloud. In Figure VI.6a and Figure VI.6b, we illustrate the earlier model of the dense point cloud reconstruction. We analysis the 3D point cloud models to evaluate the effects of the noise suppression techniques. The same model is processed through the noise suppression is shown the previous chapter. It noticeably lease noisy in areas prone to spiky noise. This technique is able to handle noise scatter over many depth layers. The attractive features generated from the reconstruction framework and super resolution enhancement model are still present in this noise suppression model. Objects in the scene are correctly dispersed across multiple depth layers. For instance, the road and sidewalk in the scene are flat in comparison to the trees and buildings. The density of the point cloud remains over 2 million points.

In order to appreciate the results of DPR, we will compare the reconstruction model to other representation presented in Chapter II. In the following chapter, we describe the experimental design and evaluate our proposed technology versus the state of the art techniques. A detailed discussion will summarize the advantages of the proposed method and potential use for this technology.
Figure VI.6: (a) The super resolution enhancement produces an extremely dense point cloud model. In this model the previously unnoticed noisy is also enhance. The noise suppression technique described eliminates the majority of the noisy point from this model. (b) Over head view of super resolution point cloud. (c) The same point cloud model as presented in part ‘a’ but post noise suppression processing. All the major features of the scene are retained but little noise speckles are eliminated using the technique. (d) A view of the new clean point cloud model. This model features a variety of depth layers and dense reconstructed objects from the input scene.
CHAPTER VII
COMPARISON AND DISCUSSION

In this chapter we revert back to some of the techniques discussed in Chapter 2 and compare the results of our Dense Point-cloud Representation (DPR) algorithm with two rivaling techniques. All these 3D modeling techniques have the promise of aiding systems in the areas of change detection, contextual information such as elevation, roads, georegistration, detection and elimination of shadows, autonomous navigation, visual global positioning and many more. The three methods under consideration are feature-based reconstruction are Dense Point-Cloud Representation (DPR) [16, 17, 19, 20], Volumetric Probabilistic Representation (VPR) [69, 71] and Visual Structure from Motion (VSFM) [82, 84, 85]. As this comparison section will show our feature-based reconstruction method is the fastest of the three methods and produces the densest point cloud, however it contains noise and is prone to a layer effect due to limited resolution as mentioned in earlier chapters. The VRP, otherwise called Voxel-based Modeling, produces the most complete reconstruction and is quite accurate (visually), however it is computationally expensive and requires occasional user interphase to produce it visually appeal models. The VSFM method is also a feature based method and is compatible with a variety of data types. An additional layer of complexity in evaluating the methods is that each has a different output, as shown in Figure VII.1. Our methodology, DPR, focus on producing a dense accurate point cloud model. PVR computes a SIFT-based point cloud and continue to build a volumetric scene using Bayesian estimations of intensities, reflections, etc. Finally, VSFM also
Figure VII.1: Comparison of stages in DPR (Dense Point-cloud Representation), PVR (Probabilistic Volumetric Representation) and VSFM (Visual Structure From Motion). The most visually appealing output for each technique is marked in red.

produces a SIFT-base point cloud and continue with surface refinement techniques to make the model more visually appealing.

VII.1 EXPERIMENTAL DESCRIPTION

As 3D reconstruction is still an emerging research field no standardized dataset for reconstruction have been used in publications. On the contrary, each technique is design their algorithms and display their results of local environments. In order to get a fair assessment of the capabilities of each technology, we conduct a series of tests and create an adaptive criteria by which we evaluate. Our first comparison treats each technique as a complete system. We evaluate the visual appeal of the final output of each technique. Although this metric is subjective, similar to image enhancement, visual assessment of 3D modal is also the one of most
used evaluation methods and simplest to understand. This evaluation is also useful for certain applications, which utilize 3D models to create a real-world feel. Our second comparison deals with metric values of the model. Since the only stage present in three techniques is the Point Cloud algorithmic stage, we use compare the densities of the point cloud stages. This is evaluation provides as with valuable insight, since each technique heavily relays on the points within the point cloud stage to proceed with further processing. The more points a model contains the more information is available for further algorithmic stages. The final comparison deals with the computational expense associated with each technique. Since these three techniques rival one another in potential deployment onto a real-time system, it is important to consider the computational speed and the abilities to accelerate the algorithms.

Since each of these techniques has been developed and modified based on evaluations of a specific dataset, it is natural for each technique to perform at its best on its own data set. For our evaluation we consider 3 datasets that represent a variety of altitudes. Hence, the Pixel Per Inch (PPI) of each environment is different as well as the change in view point. Our first dataset is a low altitude (0-100ft) street level environment of downtown Dayton in which buildings, trees and sky is visible in the scene. This is an important scenario because some of the techniques are not able to handle object infinitely far from the camera plane. These techniques are designed for aerial imagery that always contains the ground in view. The second dataset is a medium altitude (100-1000ft) set of Providence, Rhode Island. This is a dataset preferred by PVR and has been used to illustrate impressive models. The third and final dataset is of high altitude (1000+ ft) CLIF data captured of Columbus, Ohio. This dataset is capture at 7000+ feet with a frame rate of 2 frames per second. We will show a comparison of the three techniques across all three datasets.
VII.2 COMPARISON, METRICS AND ANALYSIS

In this section, we will demonstrate the result from the three reconstruction techniques. We will begin with the visual evaluation of the resulting models. Note, each technique produces a different output type, therefore this evaluation will be strictly based on visual appeal. Furthermore, we present the metrics that correspond to the density of the point-cloud reconstruction and computational expenses associated with each technique.

Visual Evaluation

We present this evaluation in a series of figures. In each figure we illustrate the outputted model from each technique. We go on to analyze the resulting models and highlight the advantages and disadvantages of each technique. We begin by evaluating the three technique on low altitude imagery. We use the downtown Dayton, Ohio dataset. Several frames from the scene are depicted in Figure VII.2.

This data is recorded in extremely low altitude from a vehicle travelling approximately 25mph and record at a perpendicular to the motion of the vehicle. This experiment best simulates a low flying UAV or an unmanned ground vehicle (UGV) performing reconstruction indoors. Notice the Downtown Dayton scene contains roads, buildings and trees as well as sky image regions. In Figure VII.3, we illustrate the resulting point clouds. We begin by analyzing the reconstruction by VSFM. As Figure VII.3a indicates the scene model is well defined and correctly colored. It does not contain a tremendous amount of points. The surface refinement appears to create a blurring effect of the actual point locations. Overall, however, this is an impressive result considering VSFM had no adjustment made to fit the dataset.
Next, we observe the reconstruction from DPR in Figure VII.3b. As shown in previous chapters, the model is incredibly dense and accurately represents the scene. The density of the model allows for us to read labels on signs and buildings. Overall, this comes as no surprise that the technique outperforms other algorithm on its prime dataset. Important to note that the PRV representation is not able to construct a model to due the fact there was finitely distant objects in the scene, such as sky. Due to that scene complexity, the algorithm was not able to properly assign the initial voxel grid.
In the second scenario depicts a medium altitude scene of Providence, Rhode Island. In this scene a helicopter circles downtown Providence several times at a varying speed, view angle and altitude. Figure VII.4 illustrates several frames from the video sequence. We begin by evaluating the VSFM result in Figure VII.5a. The first observation that sticks out is the sharpness of the edges where vertical and horizontal plane meet. The buildings and the ground form almost a perfect 90° angle. Although the model does not appear dense, the smearing that was evident in the previous scenario is not obvious in the medium altitude case. Overall, VSFM again produced an acceptable model that can be enhanced further. We continue the evaluation by analyzing the model generated by PVR in Figure VII.5b. Visually this model surpass the others completely. The model does not appear as point cloud but rather as rendered 3D scene. It is comparable to manually created video game 3D environments. There exists the occasional blurriness, but what make PVR so appeal is the constant Bayesian estimation updates. The scene continues to improve as more frames are added. Overall, the PVR produced the best visually appeal results for the medium altitude dataset, however that comes with little surprise as the dataset is proved by the authors. PVR is able generate incredible results on the several dataset provided by the author, however we found that on many other datasets the algorithm fails. Our final evaluation for the medium altitude data is of DPR in Figure VII.5c. Immediately, we notice the elevation differences between the parking lot and the skyscrapers. Interestingly, the layer effect happens in an angle in this scenario. Because this dataset is capture at a 45° angle in relation to the ground as well as in a circular motion, we see the layers effect produce strips across the buildings. Also, the algorithm utilizes piece-wise linear motion to compute its disparities and depths causing more prominent noise points. Overall, the DPR by visual inspection produces a dense accurate and noise prone representation of the scene in medium altitudes.
Figure VII.4: Frames from the Medium Altitude Providence, RI dataset.

Figure VII.5: The 3D reconstruction results using the Medium Altitude (Providence, RI) dataset. (a) VSFM results (b) PVR results and (c) DRP results.
Our final dataset is the high altitude Columbus Large Format Imagery (CLIF) over Columbus, Ohio. This dataset poses unique problems because of the high altitude at which imagers are captured. Object in the scene appear is tiny resolution objects. We illustrate several frames from the CLIF dataset in Figure VII.6. We begin by evaluating the performance of VSFM in Figure VII.7a. The results indicate another sparse and accurate point cloud representation. The techniques is able to handle the low resolution of building to create sharp edges with the ground plane. The infamous “horse shoe stable” sticks out above the rest in the 3D model. Overall, VSFM again produced an acceptable model that can contain sparse points blurred to take up more of the scene. In Figure VII.7b, we observe the results produced by PVR. Similarly, to the medium attitude dataset, this technique produces an extremely visually appeal model. The model does not contain the break and hole seen in a point cloud representation. The edges of the model a extremely blurred to the lack of imagery that covers those areas. As new imagery is processed the model is continuously updated. Overall, PVR is the most visually appeal model when considering high altitude imagery. Important to note that CLIF data used for this reconstruction was acquired from the PVR data repository, hence the technique was adjusted and trained to handle that particular dataset. Our final visual evaluation is of DPR on the CLIF data. Unfortunately, this dataset did not produce great reconstruction results. Due to the low frame rate, and low resolution the optical flow feature were unable to properly match and the further mismatched points dominate the model in Figure VII.7c. Overall, as we determined earlier DPR relies on high resolution and high disparity resolution, and the high altitude CLIF dataset was not enough to produce impressive accurate results. A summary of the visual assessment is provided in Figure VII.8. The figure contains the visual assessment rating, where a 3 indicates excellent and 1 indicates very poor or nonexistant. In the following section, we move on to evaluating the models based on their content, metrics and computational costs.
Figure VII.6: Frames from the High Altitude over Columbus, OH dataset.

Figure VII.7: The 3D reconstruction results using the High Altitude (Columbus, OH) dataset. (a) VSFM results (b) PVR results and (c) DRP results.
Density and Computational Expenses Metrics

In order to get a better sense at the capabilities of each of the three techniques we begin by evaluating the number of points each technique produces. Table VII.1 summarizes the metrics of the comparison. It is broken into three separate tables, each reconstruction technique is evaluated separately. For each technique, we separate the three datasets by row. The first column indicates the number of frames used in the reconstruction and their size in column four. This is important to note, since with more frames more information become available about the scene. Next, we present the time it takes to process and compute the model. These time values vary from seconds of computations to hours. We also present the number of points that compose and present in each model.

Figure VII.8: A summary of the visual assessment ratings.
<table>
<thead>
<tr>
<th>Dataset</th>
<th># of Frames</th>
<th>Time to Compute</th>
<th># of Points</th>
<th>Frame Size</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DPR</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Downtown Dayton, OH</td>
<td>12</td>
<td>43sec</td>
<td>2,632,834</td>
<td>1280x720</td>
</tr>
<tr>
<td>Downtown Providence, RI</td>
<td>2</td>
<td>8sec</td>
<td>802,501</td>
<td>1280x720</td>
</tr>
<tr>
<td>CLIF</td>
<td>2</td>
<td>125sec</td>
<td>2,654,141</td>
<td>2672x2004</td>
</tr>
<tr>
<td><strong>VSFM</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Downtown Dayton, OH</td>
<td>133</td>
<td>21min</td>
<td>303,403</td>
<td>1280x720</td>
</tr>
<tr>
<td>Downtown Providence, RI</td>
<td>112</td>
<td>16min</td>
<td>115,824</td>
<td>1280x720</td>
</tr>
<tr>
<td>CLIF</td>
<td>101</td>
<td>30min</td>
<td>820,026</td>
<td>2672x2004</td>
</tr>
<tr>
<td><strong>PVR</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Downtown Dayton, OH</td>
<td>133</td>
<td>DNF*</td>
<td>458,351</td>
<td>1280x720</td>
</tr>
<tr>
<td>Downtown Providence, RI</td>
<td>112</td>
<td>2.5 hours</td>
<td>268,269</td>
<td>1280x720</td>
</tr>
<tr>
<td>CLIF</td>
<td>101</td>
<td>3.5 hours</td>
<td>1,257,691</td>
<td>2672x2004</td>
</tr>
</tbody>
</table>

Table VII.1: Metrics for each evaluation technique. The green highlighted areas indicate areas of superior performance while red highlights indicate weak performances.

* DNF (Did Not Finish) corresponds to the inability to handle certain types of scenes.

From the table above, we learn that the computation time is extremely different for each reconstruction technique. While PVR takes on the order of hours to process and produce a 3D model, VSMF works on the order of several minutes and DPR completes its point cloud representation on the order of seconds. These time differences reflect on the fundamental differences in the approach to the reconstruction solution. While additional time is spend on the
beautification of the model in PVR, the usability for our application does not increase. Therefore we conclude that the technique that generate a point cloud the quickest and most accurate is the optimal technique. When comparing the three techniques, our DPR is magnitudes of time faster in computation than VSFM and PVR. The complexity of the feature matching and description in VSFM is an expensive function. At the same time, PVR surface multihypothesis estimation also create a severe time lag.

We continue our evaluation by comparing the number of points present in each model. While PVR does not appear as points, it utilize SIFT feature to construct the environment. The number of points associated with PVR is the number of features extracted for their computations. When considering the number of points, DPR is again much denser than VSFM and PVR. This density allow the viewer to legibly identify signs and labels in the scene. Our final comparison evaluates the point clouds stages of each technique.

**Point Cloud Evaluation**

We have already compared the visual output of each model, the density of the model, and the computational expense. We evaluate the usability of the model. As mentioned through this thesis our application focus on using 3D model for determining occulations, elevation changes, scene changes, aid in autonomous navigation and many more, in this section will describe the type of model needed for this applications.

When considering autonomous navigation, a UGV or UAS would need to know where obstacles are located in order to navigate around them. Similarly, when considering an occulsion decetor, the system relying on firmly know where there is an object and where this is not. Therefore, to assess the usability of the 3D reconstruction system we must remove the beautification stages and bring them to a thresholded true. Then algorithms such as ray tracing [41, 48] can be applied to determine the occulsion and obstacles in the scene. These point
indicate whether an object is in the scene or not in the scene. From the reconstruction techniques, DPR and VSFM already output those concrete values. PVR, however, relies on multiple hypotheses for surface estimation and therefore does not contain concrete points until a thresholding has taken place. In Figure VII.9, we present the concrete point cloud models generated by each of the three techniques that would be used for ray tracking.

It can be observed that in pure thresholded form, VSFM and PVR appear more noise than in previous demonstrations. In Figure VII.9, we evaluate the point clouds generated using the video captured over downtown Providence, Rhode Island. From the Figure VII.9a, we determine that VSFM appear relatively accurate with distinct sharp corners and edges of buildings. The PVR point cloud, Figure VII.9b, appears to be the most noisy. This is due to the thresholding done on the voxel model. By thresholding, voxels with a surface characteristic above a certain value are labeled as points. Therefore point can appear anywhere in the model irrespective of their relative location to other points. As a result, the point cloud indicates points appearing above the buildings and the sky regions. In Figure VII.9c, we present the DPR point cloud model. This image also displays the sharp edges and corners formed at the base of the buildings. As presented throughout this section, we can conclude that DPR provides the densest and most useful point cloud model with the quickest computational timing.
Figure VII.9: The thresholded models of the three techniques illustrate the model as they would be used in application. (a) VSFM (b) PVR (c) DPR. Results are generated using our implementations of the algorithm.
We have evaluated three reconstruction techniques. The current state-of-the-arts PVR has proven the capabilities of creating extremely appealing 3D models. The PVR models contain great detail and response the user’s viewing angle. We have also evaluated an extremely accurate feature based modeling technique, VSFM. With the use of SIFT features, VSFM is able to create accurate models. In this chapter, we showed that when compared to our DPR reconstruction technique, VSFM and PRV fall short in the density comparison, in the computational expense comparison, and in the usability comparison. We have proven through metrics and analysis that contribution of DPR will lead the development of a successful and deployable reconstruction system. Each technique we have presented provides a unique perspective of solving the reconstruction problem. As we conclude this thesis and present the future direction of the research, we plan to utilize the positive capabilities of VSFM and PVR to further enhance our methods and strive for a noiseless real-time reconstruction system.
CHAPTER VIII

CONCLUSION AND FUTURE WORK

In this thesis, we have presented and compared three state of the art reconstruction algorithms. We have demonstrated the computations involved in producing a 3D representation of scenes. Our technique focuses on reducing the computational expenses and enhancing the density, usability, and visual appeal over the technique presented in this chapter. We began by describing the reconstruction framework. Based on the outputs of the soley framework architecture, we conclude that although SURF points are accurate and extremely distinctive, the matching process eliminates too many points. The resulting point cloud was not suitable for our applications. A more rigorous technique was developed focusing on creating a point cloud containing more points. Our solution to the sparcity of our reconstructed model is, as described in Chapter IV of this thesis, Optical Flow feature points.

We conclude the thesis by highlighting the solution to the problem posed earlier. By adding optical flow features in the algorithmic framework, we were able to create a dense point cloud model. As the results indicated, the new point clouds allow the user to distinguish the scene by clearly identifying buildings, cars and trees. With the introduction of a denser point cloud, we also introduce the “layer effect” in which numerous points contain the same depth coordinate.

We have presented and evaluated a super resolution technique that is designed to enhance the depth resolution of the resulting point cloud models. After assessing the effects of super
resolution on various types of imagery, we concluded that by including super resolution into our algorithmic architecture we would increase the density (number of points tracked) and the depth resolution via an increase in the number of discrete disparities (increase the number of layers). We observed the effects of super resolution on the reconstruction process and observe the increase in depth layers as well. In our evaluation, we conclude that the depth resolution layers issue is no longer a prominent part of the model. However, the noise points that in the prior model were barely noticeable have also been enhanced due to the super resolution. Our noise suppression efforts add steps to the algorithm on the preprocessing end as well as the post processing end. We further enhance the input imagery using distortion correction, nonlinear image enhancement and video stabilization to allow for better feature matching. On the post processing end, we identify the commonality in the noise speckles within the point cloud model and use dynamic filters to suppress that noise in the model. The presentation of the framework, optical flow, super resolution and noise suppression conclude our presentation of the development of our Dense Point-cloud Representation (DPR) technique.

In order to appreciate the results of DPR, we compare the reconstruction model to other representation techniques presented, Visual Structure from Motion (VSFM) and Probabilistic Volumetric Representation (PRV). In this thesis, we describe the experimental design and evaluate our proposed technology versus the state of the art techniques. A detailed discussion summarized the advantages of the proposed method and potential use for this technology.

We have evaluated the three reconstruction techniques as follows. The current state of the art PVR has proven the capabilities of creating extremely appealing 3D models. The PVR models contain great detail and respond to the user’s viewing angle. We have also evaluated an extremely accurate feature based modeling technique, VSFM. With the use of SIFT features, VSFM is able to create accurate models. In this thesis, we showed that when compared to our
DPR reconstruction technique, VSFM and PRV fall short in the density comparison, in the computational expense comparison, and in the usability comparison. We have proven through metrics and analysis that the contributions of DPR will lead the development of a successful and deployable reconstruction system. Each technique we have presented provides a unique perspective of solving the reconstruction problem. As we conclude this thesis and present the future direction of the research, we plan to utilize the positive capabilities of VSFM and PVR to further enhance our methods and strive for a noiseless real-time reconstruction system.

As we discuss the future work and potentials for the applications of this technology we focus on some of the effects we are currently working on. This technology would serve as a perfect aid to an autonomous navigation system. Using the Robust Artificial Intelligence-based Defense Electro Robot (RAIDER), we have been able to reconstruct interior scene through the University of Dayton [18, 63]. In the future, control systems will utilize this 3D information to send controls for autonomous decision making and navigation. Furthermore, biometric tracking systems [64] will enable the robot to perform people following and other surveillance tasks. Other systems [12, 15, 40, 44, 59, 60, 76] can also utilize this technique for their autonomous and scene understanding applications. Scene understanding applications include 3D change detection [50, 62], elevation analysis and shadow detections.
LIST OF PUBLICATIONS


BIBLIOGRAPHY


http://homes.cs.washington.edu/~ccwu/vsfm/
