STRUCTURE-FROM-MOTION AND RGBD DEPTH FUSION

by

Akash Chandra Shekar

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Computer Science

Charlotte

2018

Approved by:

Dr. Andrew R. Willis

Dr. Min Shin

Dr. Srinivas Akella

Dr. Hamed Tabkhi

ABSTRACT

AKASH CHANDRA SHEKAR. Structure-from-Motion and RGBD depth fusion.
(Under the direction of DR. ANDREW R. WILLIS)

This article describes a technique to augment a typical RGBD sensor by integrating depth estimates obtained via Structure-from-Motion (SfM) with depth measurements from an RGBD sensor. Limitations in the RGBD depth sensing technology prevent capturing depth measurements in four important contexts: (1) distant surfaces (>8m), (2) dark surfaces, (3) brightly lit indoor scenes and (4) sunlit outdoor scenes. SfM technology computes depth via multi-view reconstruction from the RGB image sequence alone. As such, SfM depth estimates do not suffer the same limitations and may be computed in all four of the previously listed circumstances. This work describes a novel fusion of RGBD depth data and SfM-estimated depths to generate an improved depth stream that may be processed by one of many important downstream applications such as robot localization, robot mapping, robot navigation, object tracking, pose estimation, and object recognition.This approach is demonstrated on sequences of images that transition from indoor scenes, where the RGBD depth sensor can function, to outdoor scenes, where the RGBD depth sensor fails.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Dr. Andrew R. Willis, for taking me under his wing and encouraging me at every step of the way with his exceptional understanding of the subject. His invaluable insights played a crucial role in this research. I would also like to thank my co-advisor, Dr. Min Shin, for enabling me to continue research from the Electrical and Computer Engineering Department and for guiding me in the right direction. Additionally, I appreciate the members of my committee, Dr. Srinivas Akella, and Dr. Hamed Tabkhi, for their constant support. Last but not the least, I would like to acknowledge Mr. John Papadakis for all the thought-provoking discussions.

TABLE OF CONTENTS

LIST OF FIGURES vii

LIST OF TABLES viii

LIST OF ABBREVIATIONS 1

CHAPTER 1: INTRODUCTION 1

1.1. RGBD Sensing Technology and Limitations 1

1.2. The Structure from Motion 2

1.3. Contribution 3

CHAPTER 2: RELATED WORK AND BACKGROUND INFORMATION 5

2.1. RGBD Camera 5

2.1.1. Limitation of RGBD sensors 6

2.2. Structure from Motion 7

2.2.1. Camera model 8

2.2.2. Camera Distortion and Calibration 13

2.2.3. Estimation of the Camera pose 15

2.2.4. Estimation of the 3D structure 21

2.3. Non linear Optimization 29

2.4. Visual Odometer and Graph Optimization 34

CHAPTER 3: LSD SLAM 35

3.1. Point Cloud Reconstruction 38

3.2. Point Cloud Re-Projection 38

CHAPTER 4: Methodology 40

4.1. Time and Spatial Sampling Issues 40

4.2. Image Registration Issues 42

4.3. Noise removal 45

4.4. Resolving the Unknown SfM Scale 46

4.5. RGBD Measurement Noise 47

4.6. RGBD and SfM Depth Fusion 48

CHAPTER 5: Results 50

CHAPTER 6: Conclusion and Futher work 54

REFERENCES 55

LIST OF FIGURES

FIGURE 1.1: The depth fusion introduction 3

FIGURE 2.1: ORBBEC Astra RGBD sensor 6

FIGURE 2.2: Limitations of RGBD sensors 7

FIGURE 2.3: Thin lens camera model 9

FIGURE 2.4: Pin-hole camera model 10

FIGURE 2.5: Checkerboard for camera calibration 14

FIGURE 2.6: Epipolar geometry 23

FIGURE 2.7: Disparity Computation 25

FIGURE 3.1: An overview of the LSD SLAM algorithm 37

FIGURE 3.2: Outputs of LSD SLAM 37

FIGURE 4.1: An overview of the proposed depth fusion algorithm. 41

FIGURE 4.2: Depth registration 45

FIGURE 4.3: SFM outlier detection 46

FIGURE 5.1: Results 51

LIST OF TABLES

TABLE 5.1: Percentage of occurrence the depths from difference sources 50

CHAPTER 1: INTRODUCTION

RGBD sensors are a relatively new class of image sensors. Their key novel feature is the ability to simultaneously capture color "RGB" images of the scene and depth "D" images of scene; hence the term "RGBD." RGB images are capture using a conventional visible light camera that incorporates a lens that focuses light rays from scene locations onto distinct light-sensitive pixels of image sensor. RGBD integrate the three devices: (1) an IR projector, (2) and IR camera and (3) a RGB camera in a rigid relative geometry to create a single sensor that captures color-attributed (X, Y, Z) surface data at ranges up to ~6m with frame rates up to 30 Hz. RGBD sensors have a wide range of applications which include mapping, localization, pose estimation, and object recognition. They have become popular for their ease-of-use and low cost in comparison with the other visual sensor technologies such as LIDAR, and have been incorporated into consumer products like mobile phones, gaming consoles, and automobiles [1].

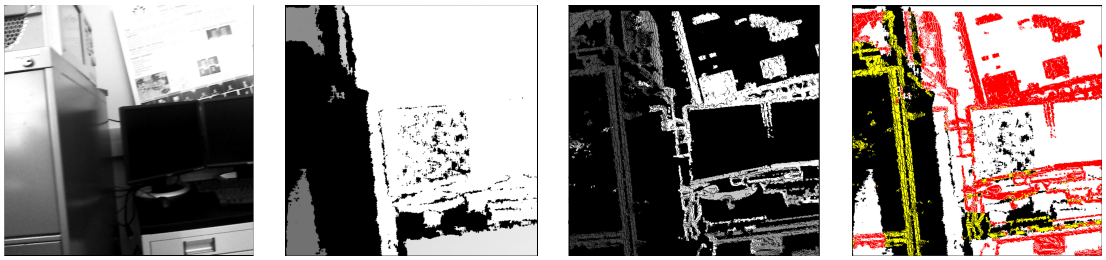
1.1 RGBD Sensing Technology and Limitations

Depth image formation is accomplished using structured light technology to measure the geometric position of viewed surfaces. This is accomplished by illuminating scene surfaces with an infrared (IR) projector having a known pattern and then using an IR camera to capture the projected pattern [2]. Deformation of the projected pattern over scene object is analyzed and used to triangulate the depth of scene surfaces with respect to the camera's optical axis. The IR projector operates outside the visible light frequencies and, as such, does not interfere with the captured RGB stream pixel values.

Despite the popularity of RGBD sensors, their utility for generic depth measurement is limited in several ways due to shortcomings associated with structured light depth estimation. One significant shortcoming is that RGBD sensors often fail to provide meaningful depth values in sunlit outdoor scenes. Here the IR radiation from sunlight interferes with the projected pattern causing the depth estimation process to fail. This phenomenon also occurs in sunlit indoor scenes. RGBD sensors also fail to collect measurements from surfaces having specific reflectance properties. This includes the following three reflectance contexts: (1) "dark" surfaces, i.e., surfaces having a low reflectance, (2) specular, i.e., mirror-like, surfaces and (3) transparent surfaces [3][4].

1.2 The Structure from Motion

The Structure from Motion (SfM) algorithm leverages ideas originally drawn from photogrammetry to estimate the three-dimensional structure of a scene from a time series of RGB images from a moving single camera. This is achieved by calibrating the camera [5] to develop a highly-accurate model to describe how 3D positions are projected into camera images. Using this image formation model, the SfM algorithm matches together pixels in separate images that correspond to projections of the same 3D location as the camera moves in the scene. Using the camera projection model and the assumption that matched pixels are measurements of the same 3D world coordinates, the SfM algorithm solves for both the pose of the camera within the global coordinate system and the set of 3D surface positions provided by matched image pixels [6]. The SfM problem is non-linear in the unknowns and is typically solved in a two-stage sequence. Stage 1 solves for the relative pose of the cameras at the instant the images were recorded. Stage 2 conditions on the estimated camera pose values and solves for the 3D scene structure. Both stages use correspondences between pixels from different images to solve the non-linear equations in the unknown variables. The camera pose tracking problem of Stage 1 is often solved by finding a



(a) (b) (c) (d)

Figure 1.1: An overview of the proposed RGBD and SfM fusion algorithm (a) shows a grayscale image of the scene. (b) shows the sensed RGBD depth image. (c) shows the SfM-estimated depth image, and (d) shows the fused image. The fused image has been color-coded as follows: (white) denotes depth locations sensed only by the RGBD sensor, (yellow) denotes depth locations only sensed via SfM, (red) denotes fused (RGBD+SfM) depth locations and (black) denotes depth locations without RGBD or SfM measurements.

map that transforms pixels from the original (x, y) coordinate field to new coordinate positions (x', y') such that both locations correspond to images of the same 3D scene point. The multi-view 3D surface reconstruction of Stage 2 is often solved using the bundle adjustment algorithm [7].

While scene reconstruction via SfM produces depth images in contexts where depth cameras fail, this modality for depth estimation also has several shortcomings. Specifically, the theoretical formulation of the SfM problem shows that the scale of the estimated 3D structure cannot be known without prior or outside information. This complicates both the mathematical and computational SfM solutions. SfM also presumes that viewed surfaces are static, i.e., they do not move, and when this assumption is violated reconstructed surfaces are highly inaccurate.

1.3 Contribution

This article seeks to leverage the strengths of RGBD-derived and SfM-derived depth measurements by fusing these measurements into an improved depth image that provides depth measurements in contexts where at least one of the two depth estimation approaches succeeds. Figure 1.1 shows an RGBD-SfM fusion result for an indoor

scene and how the fusion result (Figure 1.1(c,d)) captures more scene geometry than either approach independently. Our proposed method to fuse RGBD and SfM depth imagery includes consideration of the RGBD sensor depth image noise model, the SfM algorithm depth noise model and also copes with the inherent unknown scale and scale-drift problems intrinsic to SfM. To our knowledge these technical issues have not been discussed elsewhere in the literature.

CHAPTER 2: RELATED WORK AND BACKGROUND INFORMATION

This article proposes fusion of SfM-estimated depths with depths captured from an RGBD image sensor. This section is dedicated to discussing the relevant aspects of the SfM algorithm and the sensed RGBD measurements necessary to explain the proposed fusion method. Specifically, this section reviews the theoretical details of the SfM algorithm, methods for processing depth images including computing depth images for arbitrary camera poses, and details existing knowledge regarding the sensor measurement noise for RGBD depth measurements.

2.1 RGBD Camera

RGBD cameras are the class of sensors that capture both intensity values (RGB values) and depth values for every pixel in the frame. The RGB values are measured by a traditional camera with the thin lens model, discussed in section 2.2.1. Depths are measured with the help of infrared ray (IR) projectors and IR sensors. There are two different approaches of using the IR rays to measure the depth. The first approach relies on the principles of the structured light [8], where the known pattern of infrared light is projected onto the environment, and the deformed pattern from the environment interactions are observed and triangulated for the depth measurement. The second approach uses the concept of Time of Flight (TOF) [9], where the IR rays are continuously emitted into the environment, and the time taken for the IR rays to hit the object and travel back, along with the change in its phase, are measured to estimate depths.

For this study, the ORBBEC Astra Pro RGBD sensor was used, and it has the following specifications:

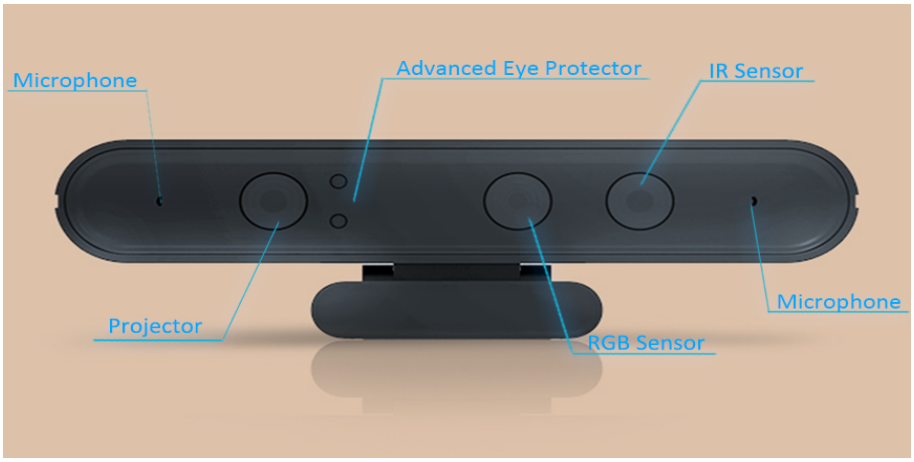


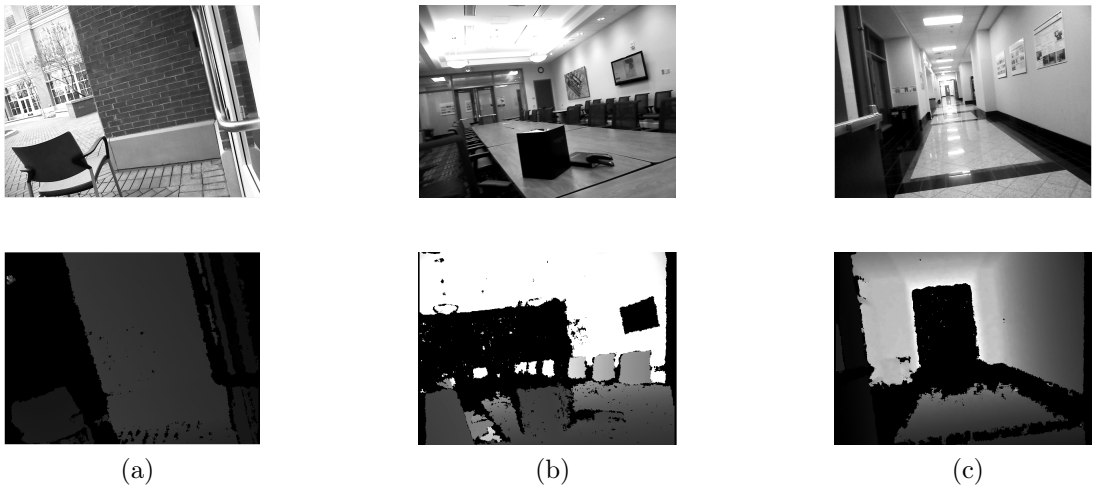
Figure 2.1: ORBBEC Astra RGBD sensor

- a field of view of 60° horiz x 49.5° vert.
- a range of 0.6 - 8 m.
- captures RGB images of dimension 640 * 480 @ 30FPS.
- captures depth images of dimension 640 * 480 @ 30FPS.

Experimental results reflects this accuracies. For which, single channel, 8 bit unsigned integer of intensity images, and 32 bit floats of depth images were recorded for various scenarios as a Robot Operating System (ROS) bag. Each bag was recorded for approximately 30 seconds @ 30FPS, which produces a ROS bag of size approximately 1.3 giga-bytes.

2.1.1 Limitation of RGBD sensors

Despite their advancements over the years, RGBD sensors have limitations. These limitations are primarily caused due to the use of infrared rays. The main limitation of RGBD sensors is that they cannot be used outdoors for depth measurements. The IR rays emitted by the Sun overwhelms the IR rays projected by RGBD sensors, corrupting the depth measurements, as shown in the figure 2.2a. For the same reason, the RGBD sensors fail to provide correct depth measurements for sunlit indoors.



(a) (b) (c)

Figure 2.2: (a) shows the RGBD failure for outdoor. (b) shows the RGBD failure for a black object and a refractive door. (c) shows the RGBD failure for specular surface and its the range limitation.

Secondly, the material property of an object in the environment affects the fidelity of depth measurements. For example, the highly specular objects like a mirror or a shiny metal reflect IR rays, as shown in the figure 2.2c. While the refractive surface like glass make them pass through, as shown in the figure 2.2b. And the diffuse black objects observe them. All this phenomenon modify the IR rays, resulting in incorrect depth measurements. Finally, RGBD sensors have a hardware limitation; they can measure depths for an only limited range. For ORBBEC Astra Pro it is 0.6 - 8 m, as shown in the figure 2.2c.

2.2 Structure from Motion

The SfM algorithm uses a time sequence of images from a moving camera to recover the 3D geometry of objects viewed by the camera. While this problem can be solved without a calibrated camera, reconstruction accuracy will adversely affected. This work assumes that the camera calibration parameters are known, estimated via camera calibration see subsection 2.2.2. The SfM algorithm can be broken down into two key steps:

1. estimation of the camera pose, i.e., position and orientation, at the time each image was recorded,

2. estimation of the 3D structure of the scene.

As previously mentioned, typical SfM systems solve (1) by computing a map that associates pixels from the original (x, y) coordinate field to new coordinate positions (x', y') such that both locations correspond to images of the same 3D scene point and (2) via multi-view 3D surface reconstruction algorithm, e.g., bundle adjustment [7]. In the following sections we provide an overview of aspects of the SfM algorithm necessary for the development of the proposed RGBD-SfM depth fusion algorithm.

2.2.1 Camera model

The SfM operates on two-dimensional images, and it is important to understand how these images are generated. An image is formed by capturing the light energy (irradiance) for every pixel. This process can be mathematically represented by the thin lens camera model, which describes the relationship between the three-dimensional coordinates and its projection onto the two-dimensional image plane. In this model, the center of the lens is called an optic center, and the line passing through the optic center ($\mathbf{o}$) is called an optic axis. The plane perpendicular to the optic axis is called the focal plane. The thin lens itself is characterized by its focal length (f_x, f_y) and diameter. The focal length is the distance from the optic center, where all the ray intersects the optic axis. The point of the intersection itself is called the focus of the lens. One of the important properties to consider is that the rays entering the lens through the optic center are undeflected, while the rays entering the lens in all the other places are refracted. With this model, the irradiance at each pixel is computed as an integration of all the energy emitted from a region of an environment, which is determined by all the rays converged at that pixel.

The fundamental equation of the thin lens is obtained using similar triangles from

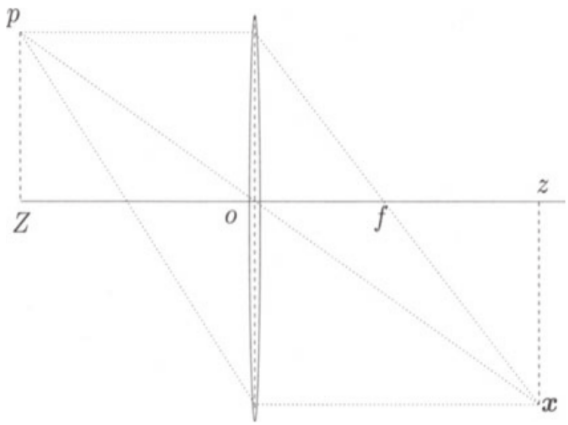


Figure 2.3: Thin lens camera model

figure 2.3

$$\frac{1}{Z} + \frac{1}{z} = \frac{1}{f}$$

For the simplification of calculation, we consider an ideal camera model called the pinhole camera model. In this case, the aperture of a thin lens is assumed to be zero, all rays are forced to go through the optical center **o**; therefore they remain undeflected. Consequently, as the aperture decreases to zero, the only points that contribute to the irradiance at the image pixel $\mathbf{p} = [x, y]$ are on a line through the center o of the lens. If a point p has coordinates $\mathbf{P} = [X, Y, Z]$ relative to a reference frame centered at the optical center **o**, with its z-axis being the optical axis of the lens, then it is immediate from the similar triangles in the figure 2.4 that the coordinates of p and its image x are related by the so-called ideal perspective projection.

$$x = -f_x \frac{X}{Z} \tag{2.1}$$

$$y = -f_y \frac{Y}{Z} \tag{2.2}$$

This mapping of 3D point to 2D is called projection and is represented

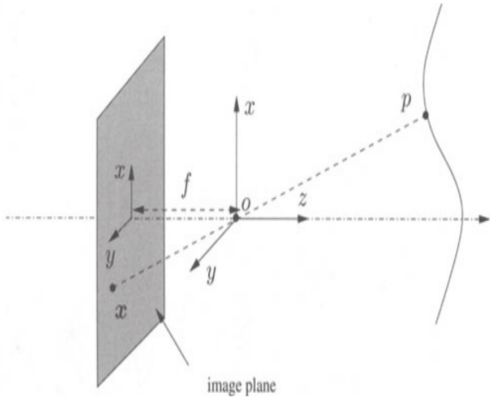


Figure 2.4: Pin-hole camera model

by $\boldsymbol{\pi}$

$$\boldsymbol{\pi} : \mathbb{R}^3 \rightarrow \mathbb{R}^2$$

This is also written as $\mathbf{p} = \boldsymbol{\pi}(\mathbf{P})$.

The negative sign in the eq 2.1 and 2.2 makes the object appear upside down on the image plane. Since we are working with digital camera, we can handle this by moving the image plane to front of the optic center to $z = (f_x, f_y)$, which will make $(x, y) \rightarrow (-x, -y)$.

This can be represented in matrix form as

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{Z} [f_x, f_y] \begin{bmatrix} X \\ Y \end{bmatrix}$$

In homogeneous coordinates, this relationship can be modified as

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.3}$$

The equation 2.3 can be further decomposed into

$$\begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \frac{1}{Z} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

with

$$\mathbf{K}_f = \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times3}, \boldsymbol{\pi} = \frac{1}{Z} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times4}$$

The matrix $\boldsymbol{\pi}$ is a projection matrix and it is a non linear operation .

With the rigid body representation for camera from the equation 2.12, we can represent the overall geometric model for an ideal camera as:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \frac{1}{Z} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.4}$$

The equation 2.4 represents the ideal camera model, where the retinal frame is centered at the optical center, and its axis aligned with the optical axis. But in practice, this does not true and the origin of the image coordinate frame typically in the upper-left corner of the image. We need to address this distortion between the retinal plane coordinate frame and the pixel array in our camera model. This distortion can be corrected by a special matrix :

$$\mathbf{K}_s = \begin{bmatrix} 1 & s_\theta & o_x \\ 0 & 1 & o_y \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{2.5}$$

$$\mathbf{K} = \mathbf{K}_s\mathbf{K}_{\boldsymbol{f}} = \begin{bmatrix} 1 & s_\theta & o_x \\ 0 & 1 & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f_x & s_\theta & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

where

- o_x: x-coordinate of the principal point in pixels,
- o_y: y-coordinate of the principal point in pixels,
- f_x : size of unit length in horizontal pixels also called as focal length in x axis,
- f_y : size of unit length in vertical pixels also called as focal length in y axis,
- $\frac{f_x}{f_y}$: aspect ratio of a pixle
- s_θ: skew of the pixel, often close to zero.

Since the parameters in matrix 2.6 are unique to every camera and are not influenced by any external factors, they are called an intrinsic parameter, and the matrix itself is called an intrinsic matrix $\mathbf{K}$. With matrix $\mathbf{K}$, the ideal camera model from equation 2.4 can be updated as

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s_\theta & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \frac{1}{Z} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.7}$$

In the matrix notation,

$$\mathbf{p} = \mathbf{K}\boldsymbol{\pi}\mathbf{Q}\mathbf{P} \tag{2.8}$$

To summarize, equation 2.8 represents the projection of three-dimensional coordinates $\mathbf{P} = [X, Y, Z]^T$ by camera with the pose $\mathbf{Q}$, as defined in the equation (2.13), (also called as extrinsic parameters) and camera calibration $\mathbf{K}$, onto two-dimensional coordinate $\mathbf{p} = [x, y]^T$.

2.2.2 Camera Distortion and Calibration

The images generated by a typical camera usually have a distortions. Since, these images are used for the estimation of intrinsic parameters, the distortions should be fixed, in order to prevent errors in camera calibration. There are mainly of two types of distortions, the radial distortion and tangential distortion. The radial distortion effects image by curving the string lines, its effect is more as we move away from the center of image. This distortion can be corrected by Brown's distortion model [10] as following.

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

where (x, y) are distorted image intensities and $(x_{corrected}, y_{corrected})$ are corrected image intensities, where k_x is the distortion coefficient of the camera to be determined, and r is the distance of pixel from the principal point.

Tangential distortion are due to misalignment of lens with imaging plane. As a consequence some areas in image may look nearer than expected. This distortion can be corrected by The Brown–Conrady [10] model as following

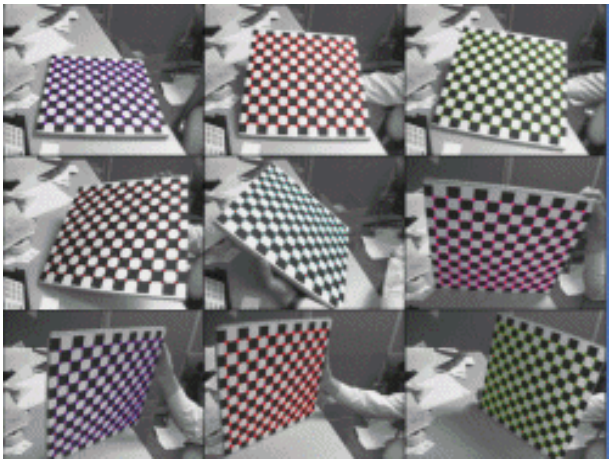


Figure 2.5: Checkerboard for camera calibration

$$x_{corrected} = x + [2p_1xy + p_2(r^2 + 2x^2)]$$

$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2xy]$$

In total there are 5 parameters, known as distortion coefficients given by:

$$D_{coff} = (k_1, k_2, p_1, p_2, k_3) \tag{2.9}$$

The camera calibration is the process of determining all intrinsic parameters of the camera. For any given camera, all the essential parameters from equation 2.6 and equation 2.9 are determined by measuring the difference between the position of key features in captured image and their supposed true position. For this process, images of known pattern are captured from different orientations. Since the estimation of corners are convenient and less prone to error, the two dimensional checkerboard pattern are often used for camera calibration [11, 5]. There are many standard library implementations of this process and we have used the one in OpenCV library [12].

2.2.3 Estimation of the Camera pose

The camera pose is represented by the rigid body motion, which is introduced in the section 2.2.3.3, these camera poses between pair of images can be estimated by solving for their alignment, as discussed in the section 2.2.3.2. Further, this process of pose estimation along with the knowledge of depth, can be applied to estimate the camera trajectory over the sequence of images, which discussed in section 2.4.

2.2.3.1 Image alignment

The image alignment is the process of transforming (warping) one image with respect to another image, with a goal to minimizing the total difference between the intensities [13].

$$\hat{\boldsymbol{\xi}} = \min_{\xi} \sum_{\mathbf{p}} \left(\mathbf{I}(\omega(\mathbf{p}, \boldsymbol{\xi})) - \mathbf{I}_{\mathbf{ref}}(\mathbf{p})\right)^2 \tag{2.10}$$

In equation (2.10), the warp function, $\omega(\mathbf{p}, \boldsymbol{\xi})$, which is a rigid body motion, maps pixel locations, $\mathbf{p}$, in the reference image $\mathbf{I}_{\mathbf{ref}}$ to pixel locations in the image $\mathbf{I}$ with the current estimate of the transformation parameters $\boldsymbol{\xi}$. With correspondence estimation, $\boldsymbol{\xi}$ is a pose transformation of the viewing camera represented as 2.15. We then seek the camera pose transformation parameters, $\hat{\boldsymbol{\xi}}$, that minimize the error in equation 2.10, which provides the camera pose change that best explain the differences in these two of images of the same scene. Given two images and the camera pose change between them, one can take the information in one image, and through the warp function map these values into the viewpoint of the other image to establish a correspondence. In this case the theoretical difference between expected and observed values for the image pair is zero if the camera pose change is known exactly and sensor noise and other outside influences are ignored. The wrap function which achieve this goal can be estimated incremental by non linear optimization. To preform an image alignment, the correspondence between images should be established, and the wrap function ω should be defined.

2.2.3.2 Solving for Image Pixel Correspondences

There are generically two different approaches for finding corresponding observations of the same 3D surface location in multiple images referred to as *direct* and *indirect* [14]. In this discussion, we refer to the correspondence problem as a source-to-target matching problem. Let $\mathbf{I}_t(x, y)$ denote an image recorded at time t and $\mathbf{I}_{t+\Delta t}(x, y)$ denote a subsequent image measured at time $t + \Delta t$. The correspondence problem seeks to find a map that transforms pixels from the original (x, y) coordinate field of $\mathbf{I}_t$ to new coordinate positions (x', y') in $\mathbf{I}_{t+\Delta t}$ such that $\mathbf{I}_t(x, y)$ and $\mathbf{I}_{t+\Delta t}(x', y')$ correspond to images of the same 3D scene point.

Indirect Methods Indirect methods compute this mapping by detecting special (x, y) locations referred to as features locations with a purpose-built feature detection algorithm, e.g., the Harris corner detector [15]. A description of the image patch in the vicinity of each detected (x, y) location is computed using some feature descriptor algorithm, e.g., Lowe's SIFT descriptor [16]. Feature descriptors seek to provide a vector of values from the image patch data that is invariant to the image variations that occur during camera motion. These include but are not limited to the following effects: illumination variation, affine and/or projective invariance, photometric invariance (brightness constancy), and scale invariance. Popular feature descriptors often prioritize scale and affine invariance as their strengths. The invariance property allows for correspondences to be computed by finding the mapping from the feature descriptor set calculated from image $\mathbf{I}_t(x, y)$ to the feature descriptor set calculated from image $\mathbf{I}_{t+\Delta t}(x, y)$.

Direct Methods Direct methods on the other hand typically iteratively solve for a set of transformation parameters that best align a pair of images by the minimization of pixel-wise errors. An image warping function, $\omega(\mathbf{x})$, maps a pixel location, $\mathbf{p} = [x, y]^t$, in the original coordinate field to new coordinate positions, $\mathbf{p}'$, such that both

locations correspond to images. A classical solution to this problem is given by the Lucas-Kanade-Tomassi (LKT) camera tracking algorithm [13].

2.2.3.3 Rigid Body Motion

The warp function in 2.10 is a camera transformation between a pair of images, which is a rigid body motion. We need an efficient model to represent these rigid body motions. The camera position is represented by a three dimensional vector in an Euclidean space $\mathbb{R}^3$, and the rigid body motion of this camera is composed of a rotation and translation.

Traditionally, rotation is represented by a 3×3 special orthogonal group called rotational matrix. Special Orthogonal matrix SO(3) is a matrix which satisfy $\mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^\mathbf{T} = \mathbf{I}$ and have a determinant of $+1$.

$$SO(3) = \{\mathbf{R} \in \mathbb{R}^{3\times3} \mid \mathbf{R}^T\mathbf{R} = \mathbf{I}, det(\mathbf{R}) = +1\}$$

The rotation transformation of a camera position $\mathbf{P}_c$ from its local coordinate frame C, to its position $\mathbf{P}_w$, in the world coordinate frame W is represented as

$$\mathbf{P}_w = \mathbf{R}_{wc}\mathbf{P}_c$$

Because the rotational matrix is orthogonal, we have $\mathbf{R}^{-1} = \mathbf{R}^T$, with this, the inverse transformation of coordinates is achieved by

$$\mathbf{P}_c = \mathbf{R}_{wc}^{-1}\mathbf{P}_w = \mathbf{R}_{wc}^{T}\mathbf{P}_w$$

The continuous rotation of a camera is described as a trajectory $\mathbf{R}(t) : t \to SO(3)$ in the space of $SO(3)$. When the start time of camera motion is not zero, the rotation of the camera from time t_1 to time t_2 is denoted by $\mathbf{R}(t_2, t_1)$. When we have more then one camera rotation, their composition is represented as

$$\mathbf{R}(t_2,t_0) = \mathbf{R}(t_2,t_1)\mathbf{R}(t_1,t_0), t_0 < t_1 < t_2 \in \mathbb{R}$$

The translation of a camera is represented by a $T \in R^3$, 1×3 vector, which accounts for the amount of translation in every dimension. With this, the complete rigid body motion is represented by

$$\mathbf{P}_w = \mathbf{R}_{wc}\mathbf{P}_\mathbf{c} + \mathbf{T}_{wc} \tag{2.11}$$

However, the above equation is not linear but affine. We may convert this to linear by using homogeneous coordinates, where we append the value 1 for 1×3 vector and make it a 1×4 vector,

$$\bar{\mathbf{P}} = \begin{bmatrix} \mathbf{P} \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \in \mathbb{R}^4$$

With this new notation for point, we can rewrite the transformation from equation 2.11 as

$$\bar{\mathbf{P}}_w = \begin{bmatrix} \mathbf{P}_w \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{wc} & \mathbf{T}_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_c \\ 1 \end{bmatrix} = \mathbf{Q}_{wc}\bar{\mathbf{P}}_c \tag{2.12}$$

where, the 4×4 matrix $\mathbf{Q}_{wc} \in \mathbb{R}^{4\times4}$ is called the homogeneous representation of the rigid-body motion.

The set of all possible rigid body transformation involving a rotation and a translation can be represented by a Special Euclidean group called SE(3)

$$SE(3) = \{\mathbf{Q} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \mid \mathbf{R} \in SO(3), \mathbf{T} \in \mathbb{R}^3\} \subset \mathbb{R}^{4\times4} \tag{2.13}$$

This representation can extend to include an variable for uniform scaling in all the dimensions. With α representing the scalar value for uniform scaling, the transformation from equation 2.12 is updated as fallowing

$$\bar{\mathbf{P}}_w = \alpha \mathbf{Q}_{wc} \bar{\mathbf{P}}_c \tag{2.14}$$

Similar to 2.13 the set of possible configurations of a rigid body with uniform scaling is represented by a Similarity transformation $Sim(3)$, which is the composition of a rotation, translation and a uniform scale, and hence this representation has 7 degrees of freedom,

$$SIM(3) = \{\mathbf{W} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & \alpha^{-1} \end{bmatrix} \mid \mathbf{R} \in SO(3), \mathbf{T} \in \mathbb{R}^3, \alpha \in \mathbb{R}\} \subset \mathbb{R}^{4\times4} \tag{2.15}$$

We can represent the pose change of the camera between two time instances as a combination of orientation and translation by $SE(3)$ as shown in the equation 2.13. We could also incorporate the scale change to $SE(3)$ by using $SIM(3)$ as shown in the equation 2.15

2.2.3.4 Exponential Map

The special orthogonal group for a three dimensional transformation is represented by a 3×3 rotation matrix $\mathbf{R} \in SO(3)$, with the constraint $\mathbf{R}^T\mathbf{R} = \mathbf{I}$, this constraint implies that the $SO(3)$ transformations does not affect the volume of the rigid body, i.e, the value of the quantity $x^2+y^2+z^2$ is not changed by the rotation transformation. The group $SO(3)$ has nine parameters, but the invariance of the volume implies six

independent constraints, making only three free parameters. From this intuition, we know that the six out of nine parameters in $SO(3)$ representation are redundant, and we can have better a representation for rigid body motion.

The $SO(3)$, $SE(3)$ and $SIM(3)$ are categorized under the special group called the Lie group, which are defined on the smooth differentiable manifolds. Every Lie group has a tangent space at identity called the Lie algebra, which is a vector space used to study the infinitesimal transformations.

For rotation group $SO(3)$ its lie algebra $so(3)$ is represented by

$$so(3) = \left\{ \hat{\boldsymbol{\omega}} \in \mathbb{R}^{3\times 3} \mid \boldsymbol{\omega} \in \mathbb{R}^3 \right\}$$

where $\hat{\omega}$ is a skew symmetric matrix representation for the vector ω

The map from the space $so(3)$ to $SO(3)$ is called the exponential map.

$$exp : so(3) \to SO(3); \hat{\boldsymbol{\omega}} \mapsto e^{\hat{\mathbf{R}}}$$

$$\hat{\boldsymbol{\omega}}(t) = e^{\hat{\mathbf{R}}t} \tag{2.16}$$

The equation 2.16 represents a rotation around the axis $\omega \in \mathbb{R}^3$ by an angle of t radians. And the inverse mapping is obtained by logarithm of SO(3)

$$log : SO(3) \to so(3); log(\mathbf{R}) \mapsto \hat{\boldsymbol{\omega}} \tag{2.17}$$

We can extend this to full rigid body motion which also involves the translation along with the rotation, for rotation group $SE(3)$ its lie algebra $se(3)$ is represented by .

$$se(3) \doteq \left\{ \hat{\boldsymbol{\xi}} = \begin{bmatrix} \hat{\boldsymbol{\omega}} & \boldsymbol{\upsilon} \\ 0 & 0 \end{bmatrix} \mid \hat{\boldsymbol{\omega}} \in so(3), \boldsymbol{\upsilon} \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times 4}$$

with $\boldsymbol{\upsilon}(t) = \dot{\mathbf{T}}(t) - \hat{\boldsymbol{\omega}}(t)\mathbf{T}(t) \in \mathbb{R}^3$

Similarly, the exponential map from the space $se(3)$ to $SE(3)$is given by

$$exp : se(3) \to SE(3); \hat{\boldsymbol{\xi}} \mapsto e^{\mathbf{Q}}$$

Similar to 2.17, the inverse to the exponential map is defined by logarithm

$$log : SE(3) \to se(3); log(\mathbf{Q}) \mapsto \hat{\boldsymbol{\xi}}$$

With the exponential map and its inverse, we could easily change states from Lie group to Lie algebra and visa-verse, which very essential to gain the performance during the non-linear optimization of image alignment as discussed in the section 2.3.

2.2.4 Estimation of the 3D structure

The image of a three-dimensional structure is generated by the principles of projection as depicted in the equation 2.8. By doing so, we lose the depth value of the structure, however, given a pair of images, generated from the same camera of known intrinsic parameters, represented in the matrix 2.5 , we could estimate the depth of structure by imposing an epiploar constraint on them.

2.2.4.1 Epipolar geometry

The figure (2.6) represents an epipolar geometry, in which a pair of cameras at their respective origins o_1 and o_2, with a relative pose of $\xi = (\mathbf{R}, \mathbf{T})$, where $\mathbf{R} \in SO(3)$ is the relative orientation and $\mathbf{T} \in \mathbb{R}^3$ is translation, are capturing two dimensional images of the same three dimensional point p. Here, the projection of a camera

center o_1 and o_2 onto their counter stereo image pair are called epipoles e_1 and e_2 respectively. The triangle formed between the camera origins o_1 , o_2 and the three dimensional point $\mathbf{P}$ is in the same plane, called an epiploar plane. The lines, l_1 and l_1, formed by intersection of epiploar plane with image plane are called epipolar lines. If $\mathbf{P}_1, \mathbf{P}_2 \in \mathbb{R}^3$ are the 3-D coordinates of a point $\mathbf{P}$ relative to the two camera frames, by the rigid-body transformation we have

$$\mathbf{P}_2 = \mathbf{R}\mathbf{P}_1 + \mathbf{T}$$

Now, let $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$ be the homogeneous coordinates of the projection of the same point $\mathbf{P}$ in the two image planes with respective unknown scales of $\alpha_1$and α_2.

$$\alpha_2 \boldsymbol{x}_2 = \mathbf{R}\alpha\boldsymbol{x}_1 + \mathbf{T}$$

By left-multiplying both the side by $\hat{T}$, where $\hat{T}$ is a skew symmetric representation of the vector T

$$\alpha_2 \hat{\mathbf{T}}\boldsymbol{x}_2 = \hat{\mathbf{T}}\mathbf{R}\alpha_1\boldsymbol{x_1}$$

By left-multiplying both the side by x_2^T

$$0 = \mathbf{x}_2^T\hat{\mathbf{T}}\mathbf{R}\alpha\boldsymbol{x}_1 \tag{2.18}$$

with

$$\mathbf{E} = \hat{\mathbf{T}}\mathbf{R} \tag{2.19}$$

Equation 2.18 is the epipolar constraint and the matrix $\mathbf{E} = \hat{\mathbf{T}}\mathbf{R}$ is called the essential matrix. It encodes the relative pose between the two cameras. Geometrically, it proves that o_1 , o_2 and p forms a triangle and, lies on the same plane, i.e., an epiploar

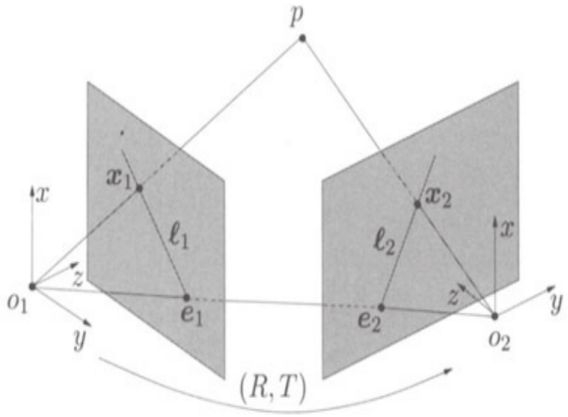


Figure 2.6: Epipolar geometry

plane, hence their triple product which measures the volume of the parallelepiped is zero.

2.2.4.2 Solving for Scene Geometry

There are generically two different approaches for 3D reconstruction referred to as *sparse* and *dense*. Sparse methods reconstruct the 3D scene geometry only for a select subset of the entire image data [17]. This subset is often corner locations or locations marked by some type of feature extraction, e.g., SIFT or SURF. [16, 18] This results in a sparse description of the 3D scene in terms of a point cloud. In contrast, dense methods [19] reconstruct as many 3D geometric locations as possible and seek to provide a complete description of the 3D scene.

Sparse reconstructions often benefit from having a lower computational cost but provide few 3D measurements. Dense reconstructions have higher computational cost but provide a much more complete description of the 3D scene. Dense reconstruction techniques have seen much recent interest, although a highly accurate, dense, and real-time SfM approach has remained elusive.

A third class of algorithms, referred to as *semi-dense* algorithms [6], seeks to strike a compromise between the sparse and dense methods. The reconstruction techniques used are most similar to dense methods, however, only a subset of all image pixels are

reconstructed. These approaches leverage the high accuracy of dense reconstruction techniques, but are sparse enough to allow for real-time operation.

Here, reconstruction is limited to those pixels which possess high intensity gradient values. These regions often correspond to scene geometries such as edges, corners, and curves and to other areas of the scene that are highly textured. The thought here is that regions of the image that possess large changes in intensity convey more information than regions that possess less, thus semi-dense reconstructions provide a compressed version of the total scene

2.2.4.3 Stereo correspondence and Disparity Estimation

We can estimate the depth of the object from a pair of images captured by the same camera with a small transnational change but the same orientation. The amount of distance that the object has shifted from the first image to the second is called the disparity [20], and this information is useful in computing the depth of the object itself. With the epiplore constraint 2.2.4.1 and known translation between the pair of images, we can efficiently compute disparity of the pixel, this approach is called the fixed baseline disparity mapping since the baseline between the pair of images are known, and they are fixed. This process starts with finding the correspondence between images. Stereo correspondence is the problem of finding which part of one image correspond to which parts of another image, where the difference is due to the movement of the camera. As discussed in 2.2.3.2 the correspondence can be either dense or sparse. In dense correspondence, image intensities are used, whereas in the sparse, only image feature are used. Also, with epiploar constraint in place, the correspondence search can be narrowed down along the epipolar lines.

Let x_L and x_R be the one of the established correspondences in the image pair 2.7 and $\mathbf{P} = (X, Y, Z)$ be the 3D location of object, from the epiploar constraint 2.18, we know that these corresponding image points and 3D position of the object forms a triangle as depicted in the image 2.7 Since the focal length f_x of the camera and

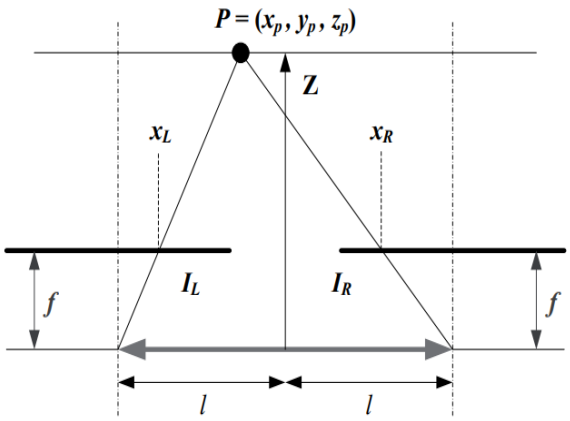


Figure 2.7: Disparity Computation

baseline $B = 2l$ are known, we can compute the disparity d, and consequently the inverse depth of object $invZ$ with properties of similarity triangles as following

$$d = x_L - x_R = f\left(\frac{X+l}{Z} - \frac{X-l}{Z}\right)$$

$$d = \frac{2lf_x}{Z}$$

$$Z^{-1} = \frac{d}{fB} \tag{2.20}$$

$$Z^{-1} \propto disparity \tag{2.21}$$

However, the problem with the fixed baseline disparity computation is that the error grows quadratically with depth, error of the depth is proportional to the length of the baseline, as a result, for the shorter baseline, the nearer depths have better accuracy but not the further depths, and vise-versa. This can addressed efficiently by using variable baseline/resolution instead of fixed baseline[21], where the baselines and focal length are selected with proportion to the depth, which helps us archive the

constant accuracy over all the depths.

Further accuracy and performance improvements could be achieved by adopting a probabilistic approach of adaptive baseline[6], which explicitly takes advantage of the fact that in a video, small baseline frames are available before large-baseline frames. In this case, instead of estimating depth for every pixel, only the subset of them which contributes to the high accuracy of disparity search were preferred. For every qualified pixel, a suitable reference frame is selected from the stack of earlier tracked frames for disparity search, in which, the one dimensional disparity search is performed along the epiploar line. This selection of the frame is based on the age of the pixel, for a given pixel, the oldest frame in which disparity search range and the observation angle was not above certain threshold was selected. If a disparity search with this frame is unsuccessful, the pixels age is increased, such that subsequent disparity search use newer frames. This adaptive baseline disparity search maximizes the stereo accuracy. Further, the prior knowledge from an earlier iteration of disparity search is utilized to minimize the disparity search range.

The disparity search is performed along the epiploar line, and it is subjected to two sources of errors [6], the geometric error, which accounts for the noise in rigid body motion $\boldsymbol{\xi}$ and the projection $\boldsymbol{\pi}(.)$. And, the photometric error, which is caused by the intensity difference between image pair. The image gradient plays a crucial role in depth estimation since it determines the extent of both the errors. The direction of the image gradient defines the geometric error, and its magnitude defines the photometric error. i.e., the geometric error on the epipolar line leads a small disparity error, if the epipolar line is parallel to the image gradient, and a large otherwise. Similarly, small image intensity errors have a large effect on the estimated disparity if the image gradient is small, and a small effect otherwise. This intuition helps us prevent the erroneous depth computation in the earlier stage itself, as soon as we have the epipolar lines constructed for given pixel, we can compare the direction and magnitude of the

epipolar line to the local gradient of the image around the pixel for a threshold. As a consequence, the region of the image with the rich pattern, which results in large gradient change, have a high probability of depth estimation, then the region with no gradient, resulting in the semi dense depth estimation.

Let l and g represent a normalized epipolar line and a normalized image gradient respectively. ϵ_l be the isotropic Gaussian noise with the variance σ_l^2 associated with the absolute position of the origin of the epipolar line l, then the variance of the geometric disparity error is

$$\sigma^2_{\lambda(\boldsymbol{\xi},\boldsymbol{\pi})} = \frac{\sigma_l^2}{dot(g,l)^2} \tag{2.22}$$

It is evident in the equation 2.22 that the direction of the image gradient and epipolar line, measured by the square of the dot product between g and l, defines this error.This error originates from the noise in camera orientation $\boldsymbol{\xi}$ and the camera calibration $\boldsymbol{\pi}(.)$ and independent of noise in intensity of image.

Let g_p representing the gradient of a image intensity on the epipolar line at disparity λ, and σ_i^2 representing the variance of the image intensity noise, we can have the variance of the photometric disparity error as

$$\sigma^2_{\lambda(I)} = \frac{2\sigma_i^2}{g_p^2} \tag{2.23}$$

Equation (2.23) shows that the photometric disparity error depends on the magnitude of gradient of image, which directly depends on the intensity of image and hence is independent of the geometric disparity error.

With the variance from equation (2.22) and (2.23), the total variance of the observed inverse depth $invZ$, estimated using equation (2.20), is

$$\sigma^2_{Z^{-1},obs} = \alpha^2(\sigma^2_{\lambda(\boldsymbol{\xi},\boldsymbol{\pi})} + \sigma^2_{\lambda(I)}) \tag{2.24}$$

where α is the proportionality constant defined for each pixel, and can be calculated as

$$\alpha = \frac{\delta_{Z^{-1}}}{\delta_\lambda}$$

where δ_d is the length of the searched inverse depth interval, and δ_λ the length of the searched epipolar line segment.

For a given pixel, the depth observed Z_{obs} is computed as the inverse of the equation (2.20), and its variance $\sigma^2_{Z,obs}$ is same as the equation (2.24) , which can be represented as Gaussian distribution

$$\mathcal{N}(Z_{obs}, \sigma^2_{Z,obs}) \tag{2.25}$$

2.2.4.4 Depth propagation and fusion

The depths estimated for every pair of images are propagated and can be used as an prior knowledge for the next iteration of depth estimation for the new pair of images. For the small camera rotation, the inverse depth and its variance from the earlier frame is propagated and is assigned to closet integer pixel position. If Z_0^{-1} is the inverse depth for a pixel from the previous frame, and the inverse depth Z_1^{-1} at the same pixel for the current frame with the camera translation of t_z along the optical axis is estimated as

$$Z_1^{-1} = (Z_0 - t_z)^{-1}$$

and its variance $\sigma^2_{invZ_1}$ is given by

$$\sigma^2_{Z_1^{-1}} = \left(\frac{Z_1^{-1}}{Z_0^{-1}}\right)^4 \sigma^2_{Z_0^{-1}} + \sigma^2_{Z_p^{-1}}$$

where, $\sigma^2_{Z_0^{-1}}$ is the variance of inverse depth from previous frame and $\sigma^2_{Z_p^{-1}}$ is the

inverse depth prediction uncertainty. These depths are fused in similar fashion of the extended Kalman filter[22] , where $\sigma^2_{Z_p^{-1}}$ corresponds to variance from to the prediction step and $\sigma^2_{Z_0^{-1}}$ corresponds to the variance from the measurement step. Finally, the new observation of inverse depth is incorporated into the prior, by multiplying two distributions i.e., if $\mathcal{N}(Z_p^{-1}, \sigma^2_{Z_p^{-1}})$ is the prior distribution and the $\mathcal{N}(invZ_0, \sigma^2_{invZ_0})$ is the noise observation, the posterior inverse depth is

$$\mathcal{N}(\frac{\sigma^2_{Z_p^{-1}} Z_0^{-1} + \sigma^2_{Z_0^{-1}} Z_p^{-1}}{\sigma^2_{Z_p^{-1}} + \sigma^2_{Z_0^{-1}}}, \frac{\sigma^2_{Z_p^{-1}} \sigma^2_{Z_0^{-1}}}{\sigma^2_{Z_p^{-1}} + \sigma^2_{Z_0^{-1}}})$$

The posterior inverse depth will have a better standard deviation then the observed inverse depth, and propagated to next frame as an prior.

2.3 Non linear Optimization

In practice, because of the noise in image correspondence and other errors, we cannot measure the actual coordinates but only their noisy versions, say

$$\widetilde{x}_1^j = x_1^j + \omega_1^j$$

$$\widetilde{x}_2^j = x_2^j + \omega_2^j$$

where $x_1^j$and x_2^j are the ideal image coordinates and $\omega_1^j = \left[\omega_{11}^j, \omega_{12}^j, 0\right]^T$and $\omega_2^j = \left[\omega_{21}^j, \omega_{22}^j, 0\right]^T$are localization errors in the correspondence called residuals. Therefore, we need a way to optimize the parameters $(x, \mathbf{R}, \mathbf{T})$ that minimize this errors.

One of the approach to optimize the error, is to minimize the squared 2-norm of residuals, if we consider the first camera frame as the reference

$$\phi(x, \mathbf{R}, \mathbf{T}, \alpha) = \sum_{j=1}^{n} \left\|\omega_1^j\right\| + \left\|\omega_2^j\right\|^2 = \sum_{j=1}^{n} \left\|\widetilde{x}_1^j - x_1^j\right\|^2 + \left\|\widetilde{x}_2^j - \boldsymbol{\pi}(\mathbf{R}\alpha^j x_1^j + \mathbf{T})\right\|^2 \quad (2.26)$$

The error in the equation 2.26 is often called the "re-projection error", since x_1^j and x_2^j are the recovered three dimensional points projected back onto the image planes. This process of minimizing the expression 2.26 for the unknowns $(\mathbf{R}, \mathbf{T}, x_1, \alpha)$ is also known as bundle adjustment[7], since, we are adjusting bundles of light rays to reduce the error.

For the purpose of simplification consider a function

$$y_i = f(x_i) \quad (2.27)$$

the goal is to find the parameters x, which minimizes the squared error between observation y and estimation $f(x)$, i.e., residual,

$$r_i(x)^2 = (f(x_i) - y_i)^2 \quad (2.28)$$

$$\mathbf{r} = [r_0(x_0), r_1(x_1), r_2(x_2).....r_{N-1}(x_{N-1})]^T$$

One of the simplest ways to minimize this squared error is the gradient descent method. It is a first-order optimization method which aims to determine a local minimum of a non-convex cost function by iteratively stepping in the direction in which the energy decreases most. Hence this method is also called as the steepest descent method.

$$x_{t+1} = x_t - \alpha \mathbf{g} \quad (2.29)$$

with step size α and gradient g

$$\mathbf{g}_j = 2\sum_i r_i \frac{\partial r_i}{\partial x_i} \tag{2.30}$$

The problem with the gradient descent is that, since it is only a first order approximation, it it may end up in local minimum which is not necessarily the global minimum. We can achieve a better results with second order approximation, and it is done by an efficient approach called the Gauss-Newton method. Gauss-Newton is a iterative method for finding the value of the variables which minimizes the sum of squares, this method achieves this by assuming that the least squares function is locally quadratic and tries to find the the minimum of the quadratic.

For the residual from the equation 2.28 we have to minimize

$$\min_x \sum_i r_i(x)^2 \tag{2.31}$$

the parameters x that minimizes the equation 2.31 is determined by differentiating it with respect to x and equating it to zero

$$2\sum_i r_i \frac{\partial r_i}{\partial x_i} = 0 \tag{2.32}$$

But the equation 2.32 cannot be solved directly. Gauss-Newton method iteratively approximate the Taylor approximation of residual function

$$r(x) \simeq r(x_t) + \mathbf{g}^T(x - x_t) + \frac{1}{2}(x - x_t)^T \mathbf{H}(x - x_t) \tag{2.33}$$

The equation 2.33 is the Taylor approximation of residual upto second derivative, where g is the gradient as defined in equation 2.30 and the Hessian H is

$$\mathbf{H_{jk}} = 2\sum_i r_i \left(\frac{\partial r_i}{\partial x_j}\frac{\partial r_i}{\partial x_k} + r_i \frac{\partial^2 r_i}{\partial x_i \partial x_k}\right) \tag{2.34}$$

The equation 2.34 is full hassian, by assuming the the problem is quadratic, we can ignore the second order term in it, and, we can approximate the Hessian matrix with Jacobian matrix as :

$$\mathbf{H}_{jk} = 2\sum_{i}\mathbf{J}_{ij}\mathbf{J}_{ik} \tag{2.35}$$

with

$$\mathbf{J}_{ij} = \frac{\partial r_i}{\partial x_j} \tag{2.36}$$

Now, the equation 2.33 can be differentiated as

$$\frac{\partial r}{\partial x} = \mathbf{g} + \mathbf{H}(x - x_t) = 0 \tag{2.37}$$

Also from the equation 2.30 and equation 2.36 , we have

$$\mathbf{g} = 2\mathbf{J}^T\mathbf{r} \tag{2.38}$$

with the Jacobian representation for gradient as shown in equation 2.38 and Hassian as in equation 2.35, equation 2.37 can be updated as

$$2\mathbf{J}^T\mathbf{r} + \mathbf{J}^T\mathbf{J}(x - x_t) = 0$$

and x_{t+1} is estimated as

$$x_{t+1} = x_t - (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{r} \tag{2.39}$$

Gauss-Newton starts with the initial guess for $x_t$and it iteratively sloves for x_{t+1} untill the minimum is approached. This method does not need the computation of second derivatives (Hessian matrix) of the of function, which is often expensive and sometimes not possible to compute, instead the Hessian is approximated with the

Jacobian matrix of the function as show in equaition 2.35.

The modification to Gauss-Newton method is the Levenberg-Marquardt (LVM)[23] algorithm,

$$x_{t+1} = x_t - ((\mathbf{J}^T\mathbf{J}) + \rho\mathbf{I})^{-1}\mathbf{J}^T\mathbf{r} \tag{2.40}$$

this is a hybrid between the Gauss-Newton and gradient decent methods. For small value of $\rho \simeq 0$, the equation 2.40 is equivalent to the Gauss-Newton method as represented in the equation 2.39, and for the large value of ρ, the identity matrix in equation 2.40 dominates the Hessian matrix $\mathbf{H}$ and the equation behaves as a gradient decent, similar to equation 2.29 with the step size of $\alpha = 1/\rho$. LVM efficiently minimizes the squared error by updating the parameters in the steepest-descent direction when the initial guess are far way from the solution, and switches to Gauss-Newton method when small updates are required for approximation. The parameter ρ is called a damping parameter, which was heuristically selected and adjusted every iteration based the previous error, and this process is also called as the damped least-squares approximation.

Further, Marquardt suggested that the convergence can be achieved quicker if there is larger movement along the directions where the gradient is smaller. To achieve this he replaced the identity matrix $\mathbf{I}$ in the equation 2.40 with diagonal values of $\mathbf{J}^T\mathbf{J}$, with the intuition of scaling each component of the gradient according to the curvature.

$$x_{t+1} = x_t - ((\mathbf{J}^T\mathbf{J}) + \rho(diag(\mathbf{J}^T\mathbf{J})))^{-1}\mathbf{J}^T\mathbf{r} \tag{2.41}$$

The Gauss-Newton and Levenberg-Marquardt are the second order approximation methods, used to find the parameters which produces the minimal residual, equation 2.28. Unlike the gradient decent method, which is a first order approximation, these methods are faster and does not end up with the local minimum.

2.4 Visual Odometer and Graph Optimization

Visual odometer is the process of estimating the camera trajectory from the stream of images. For every pair of images, we can estimate the depth as discussed in section 2.2.4, and consequently use this depth for the pose estimation from following frames. This process of pose and depth estimation is repeated for every new image, and depth is continuously propagated and refined as discussed in section 2.2.4.4

This is an nonlinear least square optimization problem, which tries to estimate the set of parameters that accurately represents the trajectory of the camera and observed 3D points of the object. The complexity of this problem can increases quickly since we are dealing with multiple features from many pairs of images and each pose estimation has a 7 degree of freedom 2.15. One efficient way to handle such a large-scale nonlinear optimization problem is to represent them in a graph[24]. There are two many advantages to this approach; Firstly, the graph representation helps with the modular representation of the problem, where each node is reference image and an edge between the nodes is a relative pose estimated. And more importantly, graph representation helps handle other essential issues like global optimization and loop closures.

CHAPTER 3: LSD SLAM

We are using LSD-SLAM [25] as our SfM implementation, it is a *semi-dense*, *direct* method which optimizes the geometry directly on the image intensities. LSD-SLAM provides as output a reconstruction of the observed 3D environment as a pose-graph of specially designated RGB image keyframes with associated semi-dense depth images. Direct correspondences are found between every RGB frame and each RGB keyframe to estimate the keyframe-to-RGB-frame relative camera pose. This is achieved by Levenberg-Marquardt optimization of the photometric error between the RGB image pair.

$$E(\boldsymbol{\xi}) = \sum_i (\mathbf{I}_{ref}(\mathbf{p}_i) - \mathbf{I}(\boldsymbol{\pi}(\mathbf{p}_i, D_{ref}(\mathbf{p}_i), \boldsymbol{\xi})))^2 \tag{3.1}$$

Equation (3.1) shows the specific image alignment object function and formalizes the form for equation (2.2.3.1). Here, the warp function relies on the current estimate of depth D_{ref} to determine the relative pose $\boldsymbol{\xi} \in sim(3)$. The reference depth D_{ref} is the depth associated with current key frame, these depth values could be initialized either with random values or with the depth measurements from a RGBD sensor to initiate the process. $\boldsymbol{\pi}(.)$ is the projection of the three dimensional point cloud to two dimensional pixel coordinates and it depends on the camera calibration.

We can apply Guass-Newton optimization, as discussed in section 2.3, to find optimal $\boldsymbol{\xi}$ which minimizes the equation 3.1, to do that we need to compute the Jacobian of the residual as required by the equation 2.39. The jacobian of 3.1 with respect to $\boldsymbol{\xi}$ is

$$\mathbf{J} = \frac{1}{z'} \begin{pmatrix} \nabla \mathbf{I}_x f_x & \nabla \mathbf{I}_y f_y \end{pmatrix} \begin{pmatrix} 1 & 0 & -\frac{x'}{z'} & -\frac{x'y'}{z'} & (z' + \frac{x'^2}{z'}) & -y' \\ 0 & 1 & -\frac{y'}{z'} & -(z' + \frac{y'^2}{z'}) & -\frac{x'y'}{z'} & x' \end{pmatrix} \tag{3.2}$$

x',$y'$and z' are the warped three dimensional points before projection

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \mathbf{R}_{\boldsymbol{\xi}} \mathbf{K}^{-1} \begin{pmatrix} x \\ y \\ \hat{z} \end{pmatrix} + \mathbf{T}_{\boldsymbol{\xi}}$$

here, $\hat{z}$ is the reference depth at the pixel (x, y). f_x, f_y are the focal lengths of the camera and K is the intrinsic matrix of the camera. $\nabla \mathbf{I}_x$ and $\nabla \mathbf{I}_y$ are the image gradients.

For every new frame tracked, the depths associated with the keyframe are continuously refined by performing the adaptive baseline stereo 3D reconstruction [6] between new tracked frames and the stack of previously tracked frames as discussed in the section 2.2.4. When there is drastic pose change between the tracked frame and keyframe, the current tracked frame is promoted as a keyframe and depth map from previous keyframe is propagated to the new keyframe and regularized. Concurrently, all the keyframes are added as a nodes to a pose graph that stores the relative pose between the keyframes as edges/constraints [24]. The pose graph stores the global trajectory of the camera in the 3D scene and an optimization algorithm processes the pose graph to improve the camera pose estimates as new image correspondences are found by image matching and loop closures.

The depth computation process involves finding the epipolar lines between new tracking frame and reference frames, along which the disparity for the pixel is determined. This process have two error sources, the error on disparity itself called the geometric disparity error and the error which encodes intensity difference called the

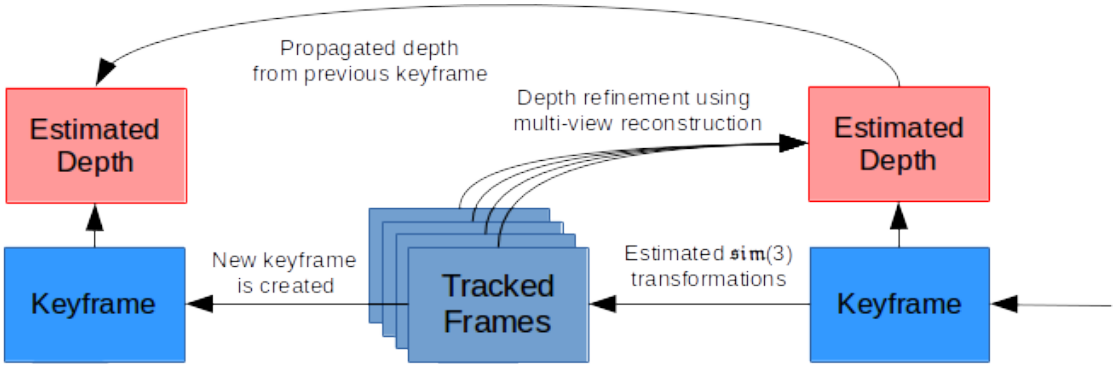


Figure 3.1: An overview of the LSD SLAM algorithm. Each incoming frame is tracked against the current keyframe. If it does not satisfy the criteria for new keyframe creation, it is used along with previous tracked frames for refinement of the estimated depth values of the keyframe. Otherwise, the frame is considered a new keyframe and depth estimates from the previous keyframe are propagated and used for initialization of the new depth estimates.

(a) (b) (c)

Figure 3.2: The outputs of the LSD SLAM (a) shows the good depth estimation for corner with rich pattern. (b) shows the good depth estimation along the edges of door and edges of poster. (c) shows the good depth estimation along the edges of table and chairs.

photometric disparity error, the former error accounts for the magnitude of the image gradient along the epipolar line , while the later one on the angle between the image gradient and the epipolar line.

We know from the section 2.2.4.3 that the depths are estimated with high fidelity for the regions having a large gradient change. This claim is further backed by the outputs of LSD-SLAM, we can see that in the figure 3.2b, that the depths are estimated for the rather plane wall, because of the presence of posters on wall, which introduce the gradient change.

3.1 Point Cloud Reconstruction

Measured 3D (X, Y, Z) positions of sensed surfaces can be directly computed from the intrinsic camera parameters and depth image values. Here, the Z coordinate is directly taken as the depth value and the 3D (X, Y) coordinates are computed using the pinhole camera model. In a typical pinhole camera model 3D (X, Y, Z) points are projected to (x, y) image locations [26], e.g., for the image columns the x image coordinate is $x = f_x \frac{X}{Z} + c_x - \delta_x$. However, for a depth image, this equation is re-organized to "back-project" the depth into the 3D scene and recover the 3D (X, Y) coordinates as shown by equation (3.3)

$$\begin{aligned} X &= (x + \delta_x - o_x)Z/f_x \\ Y &= (y + \delta_y - o_y)Z/f_y \\ Z &= Z \end{aligned} \tag{3.3}$$

where Z denotes the sensed depth at image position (x, y), (f_x, f_y) denotes the camera focal length (in pixels), (o_x, o_y) denotes the pixel coordinate of the image center, i.e., the principal point, and (δ_x, δ_y) denote adjustments of the projected pixel coordinate to correct for camera lens distortion.

3.2 Point Cloud Re-Projection

Depth images can be simulated for camera sensor in arbitrary poses by "re-projection." For discussion, assume that depth image $z(x, y)$ has been recorded in the "standard" camera/optical coordinate system where the origin corresponds to the camera focal point, the z-axis corresponds to the depth/optical axis extending out into the viewed scene, the x-axis points towards the right and spans the image columns and the y-axis points downward and spans the image rows.

Let $\mathbf{R}$ denote the 3D rotation that rotates the coordinate axes of the standard coordinate system to align with the same axes of a second camera having arbitrary

pose. Similarly, Let $\mathbf{T}$ denote the 3D translation vector describing the position of the focal point of a second camera having arbitrary pose. Using this notation, the re-projection algorithm consists of the following three steps:

1. Back-project $z(x, y)$ to create an (X, Y, Z) point cloud (as described in § 3.1),

2. Transform, i.e., rotate, translate and scale, each point $\mathbf{P}_i = [X, Y, Z]^t$ in the point cloud to generate a new point $\mathbf{P}_i^{'} = [X^{'}, Y^{'}, Z^{'}]^t$ that lies in a standard optical coordinate system centered on the second camera's focal point and having orientation that aligns with corresponding x, y, z-axes using equation (3.4),

$$\mathbf{P}_i^{'} = \mathbf{R} * \alpha * (\mathbf{P}_i - \mathbf{T}) \tag{3.4}$$

3. Re-project the (X, Y, Z) point cloud using the pinhole camera equations to compute the new depth image $z^{'}(x, y) = Z$ using equation (3.5).

$$\begin{aligned} x &= f_x(\tfrac{X^{'}}{Z^{'}}) - \delta_x + o_x \\ y &= f_y(\tfrac{Y^{'}}{Z^{'}}) - \delta_y + o_y \\ Z &= Z^{'} \end{aligned} \tag{3.5}$$

Typically, the re-projected point cloud measurements fall at non-integer locations in the new depth image and the values of $z^{'}(x, y)$ must then be interpolated via bilinear interpolation or some other interpolation scheme (nearest neighbor).

CHAPTER 4: Methodology

The proposed fusion approach applies the semi-dense monocular reconstruction approach referred to as Large Scale Direct (LSD) SLAM [25]. The LSD-SLAM algorithm solves the SfM problem using a *direct* method to compute pixel correspondences and a *semi-dense* method for 3D reconstruction. We select this approach as it does not require or impose any prior knowledge about the scene structure as required by *dense* reconstruction methods and it gives more 3D estimates than *sparse* approaches while having similar computational cost.

The LSD-SLAM SfM algorithm consists of the following three components:

1. A tracking component that estimates the pose of the camera
2. A depth map estimation component that estimates semi-dense depth images for keyframes
3. A map optimization component that seeks to create a 3D map of the environment that is self-consistent.

This work utilizes the first two components to explore fusion of RGBD depth images with SfM depth images. For this work, the map optimization component (3) is not used. Figure 4.1 depicts an overview of the proposed depth fusion algorithm.

4.1 Time and Spatial Sampling Issues

As mentioned previously, RGBD sensors measure depth at a rate of 30 frames per second (fps) and LSD-SLAM computes depth images only for *keyframes* which is a sparse subset of the measured RGB frames. Further, keyframes are not generated uniformly in time but created when the SfM algorithm detects criteria required to

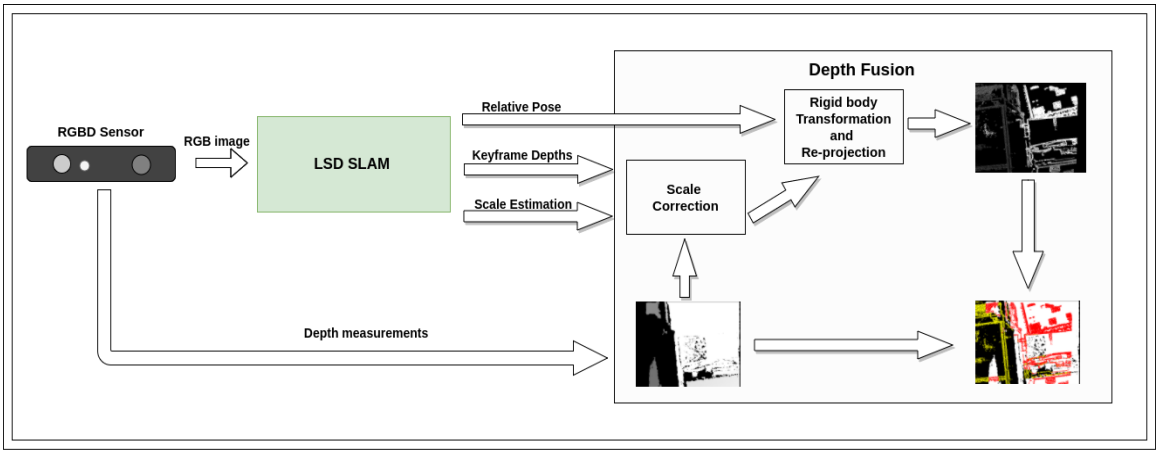


Figure 4.1: An overview of the proposed depth fusion algorithm.

create a new keyframe. This condition is triggered when the current camera pose is too far from the most recent keyframe camera pose and when the current frame tracking result is "good" in the sense that the image warping correspondence objective function suggests an accurate or low-error result. As a result, SfM-estimated depths exist only for those RGB images designated as keyframes.

Further, the spatial distribution of SfM-estimated depths within SfM keyframes are localized to only those pixels having "good" 3D reconstruction characteristics. In this sense, the quality of the depth estimate depends on accurately matching pixels along epipolar lines inscribed in the image. The matching performance here is best when there is a significant change in the image intensities along the epipolar line. Hence, 3D depth reconstruction is limited to those pixels that lie at sharp intensity changes, i.e., "edge" pixels, and further limited to those "edge" pixels that lie on edges that are roughly perpendicular to the direction of the epipolar line (see [6] for details).

The LSD-SLAM algorithm estimates depth at "good" pixel positions as a 1-dimensional Gaussian distribution specified as a mean image $\mu_{SfM}(x, y)$, i.e., the estimated depth image, and a variance image $\sigma^2_{SfM}(x, y)$ such that the RGB keyframe pixel at location $\mathbf{I}(x, y)$ is estimated to have depth $\mu_{SfM}(x, y)$ with uncertainties given by $\sigma^2_{SfM}(x, y)$. In this sense, the keyframe image $\mathbf{I}(x, y)$ augmented with the estimated depth image

$\mu_{SfM}(x, y)$ is analogous in format to sensed RGBD image data. Yet, the uncertainties for the image $\mu_{SfM}(x, y)$ are given by the image $\sigma^2_{SfM}(x, y)$ rather than the experimentally validated uncertainties discussed in § 4.5.

4.2 Image Registration Issues

Fusing depth measurements requires knowledge of the correspondence between the depth measurements generated from the RGBD sensor and the SfM algorithm. For SfM keyframes this correspondence is trivial due to the fact that RGBD sensors support hardware registration. Hardware registration co-locates the RGBD depth image measurements, $Z_{rgbd}(x, y)$, and RGB appearance values, $\mathbf{I}(x, y)$. Hence, for hardware-registered RGBD depth images, $Z_{rgbd}(x, y)$ is the measured depth of the surface having RGB pixel $\mathbf{I}(x, y)$. Similarly, SfM-estimated depths for an RGB keyframe, $\mu Z_{obs}(x, y)$, are the depths for the surface having RGB pixel $\mathbf{I}(x, y)$. Hence fusion is accomplished by fusing the measurements at corresponding (x, y) locations in the RGBD depth image, $Z_{rgbd}(x, y)$, and the SfM depth image, $\mu Z_{obs}(x, y)$.

Depth correspondences for RGB images that are not SfM keyframes must be computed from one or more SfM keyframe depth images. This article uses the most recent, i.e., closest-in-time, keyframe to generate co-registered SfM depth images for arbitrary RGB images. To do so, the depth image from the most recent keyframe, $Z_{obs}(x, y)$, is "back-projected" to create a 3D point cloud of SfM measurements. Using the estimated keyframe-to-camera pose change, the 3D measurements are then re-projected into the RGB camera image plane using the 3D projection equations for the camera provided via camera calibration. The resulting depth image, $\widetilde{Z}_{obs}(x, y)$ is then co-registered with the RGBD depth image $Z_{rgbd}(x, y)$ and RGB image $\mathbf{I}(x, y)$.

However, due to the noise in depth computation and consequent trajectory estimation, the depth registration is not always exact, this should be addressed as early as possible to avoid the drift in their computations. We try to achieve better depth registration by estimating the transformation error between the point clouds of sen-

sor depth measurements and LSD depth estimation. Since the LSD estimations are not to the scale, we also need to estimate the scale for better registration of depths. The Umeyama's Least-Square estimation [27] shows us how to estimate these $Sim(3)$ transformation parameters efficiently. In his work, Umeyama shows how to estimates the rotation, translation and scale between two point clouds with known correspondence. In order to avoid errors, we use only the valid depths in both the point cloud for estimation. By using $\mathcal{P}_{rgbd}$ and $\mathcal{P}_{obs}$ to represent the valid point clouds of sensor measurements and LSD estimation respectively, each containing N three dimensional points,

$$\mathcal{P}_{rgbd} = \{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_1, ...\mathbf{P}_{N-1}\}$$

$$\mathcal{P}_{obs} = \{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_1, ...\mathbf{P}_{N-1}\}$$

we compute the mean, variances and covariance of both them as following.

$$\boldsymbol{\mu}_{rgbd} = \frac{1}{N}\sum_{i=0}^{N-1} \mathcal{P}_{rgbd,i}$$

$$\boldsymbol{\mu}_{obs} = \frac{1}{N}\sum_{i=0}^{N-1} \mathcal{P}_{obs,i}$$

$$\sigma^2_{rgbd} = \frac{1}{N}\sum_{i=0}^{N-1} \left\|\mathcal{P}_{rgbd,i} - \boldsymbol{\mu}_{rgbd}\right\|^2$$

$$\sigma^2_{obs} = \frac{1}{N}\sum_{i=0}^{N-1} \left\|\mathcal{P}_{obs,i} - \boldsymbol{\mu}_{obs}\right\|^2$$

$$\Sigma_{obs,rgbd} = \frac{1}{N}\sum_{i=0}^{N-1} (\mathcal{P}_{rgbd,i} - \boldsymbol{\mu}_{rgbd})(\mathcal{P}_{obs,i} - \boldsymbol{\mu}_{obs})^T$$

With singular value decomposition of $\Sigma_{obs,rgbd}$ as $\mathbf{UDV}^T$,the optimum rotation matrix R which achieves the minimum error between point clouds is given by

$$\mathbf{R} = \mathbf{US}\boldsymbol{V}^T$$

$$\mathbf{S} = \begin{cases} \mathbf{I} & det(\Sigma_{obs,rgbd}) \geq 0 \\ diag(1,1,-1) & det(\Sigma_{obs,rgbd}) < 0 \end{cases}$$

The scale adjustment c is computed by

$$\alpha = \frac{1}{\sigma_{obs}^2} trace(\boldsymbol{DS})$$

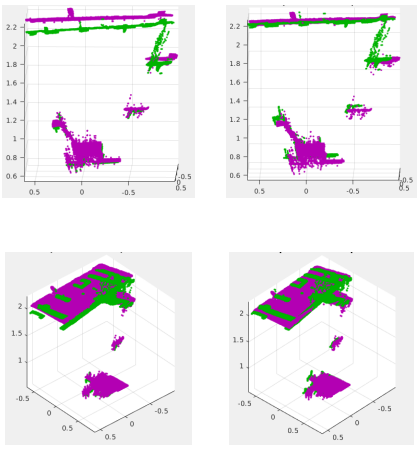
With scale and rotation established, the translation parameters are determined as

$$\mathbf{T} = \boldsymbol{\mu}_{rgbd} - \alpha \mathbf{R} \boldsymbol{\mu}_{obs}$$

with the $Sim(3)$ parameters we transform LSD point clouds before projecting them to find $\widetilde{z}_{obs}(x, y)$

$$\boldsymbol{\tilde{pc}_{obs}} = \alpha \boldsymbol{R} \boldsymbol{pc}_{obs} + \mathbf{T}$$

Using these techniques co-registered SfM depth images can be computed for a general RGBD image. When RGBD images correspond to SfM keyframes the registration is "automatic," i.e., no computation is necessary. In all other cases, a co-registered SfM depth image must be computed by reconstructing a 3D point cloud from a keyframe and then projecting the point cloud into the target RGB camera image using the estimated camera calibration and keyframe-to-camera-frame relative pose parameters.



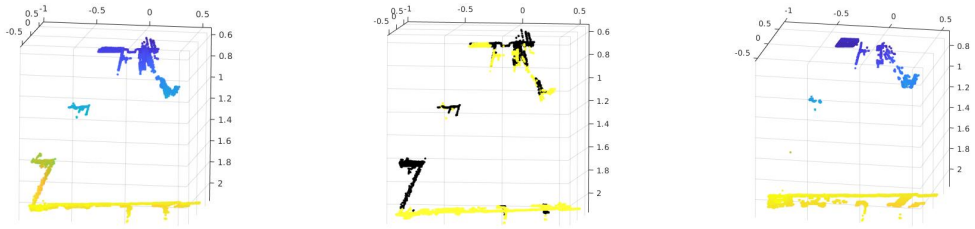
(a) (b)

Figure 4.2: (a) shows the two different views of the point clouds before depth registration. (b) shows the different views of the point clouds after depth registration. The green represents the point cloud from LSD SLAM and purple represents the point clouds from RGBD sensors

4.3 Noise removal

Noise in the image and other inaccuracies in nonlinear optimization leads to erroneous depth computation; these outlier depth values should be detected and removed to avoid further drift in pose estimation and consequent accumulation of the error in depth estimations. Fortunately, these outliers have a particular pattern as shown in the 4.3a (a), i.e., if we consider the principle point as an origin, the outlier depths form a veiling pattern along the line from the origin to the furthest depth value. Also, we know that the most distant depth is the valid depth, we can find this true depth and eliminate all other noisy depth by tracing along the line to the origin.

We can implement this by finding the line from principle point, $(0, 0, 0)$, to the furthest depth for every valid depth in depth map. From the equation 3.3 which represents the point in three dimensional space, we can obtain the line (l_x, l_y, l_z) to principle point $(0, 0, 0)$ as:



(a) (b) (c)

Figure 4.3: (a) LSD depths with veiling errors. (b) Veiling errors detected (represented in black). (c) LSD depths with veiling errors removed.

$$\begin{aligned} l_x &= ((x + \delta_x - c_x)Z/f_x) - 0 \\ l_y &= ((y + \delta_y - c_y)Z/f_y) - 0 \\ l_z &= (Z - 0) \end{aligned} \tag{4.1}$$

the slope for the line is determined by:

$$m = l_y/l_x \tag{4.2}$$

Since these depths have a veiling pattern in the z-axis, the outlier depths tend to have the same slope with small variance, and lower z value then the real depth. By traversing the line towards the origin, we find and eliminate all the outlier depths having the slope in the range of empirically determined threshold value. The experimental results are shown in the figure 4.3b, where the depth outliers are marked in black color, and result after the outlier elimination is shown if figure 4.3c.

4.4 Resolving the Unknown SfM Scale

The methods described in previous sections detail how co-registered SfM depth measurements are computed for every sensed RGBD frame. However, as discussed in previously, SfM depth images intrinsically have an unknown scale, α, which reflects the fact that the solution for the scene structure is not geometrically unique, i.e., the same scene structure can be observed at a infinite number of distinct scales. Therefore

fusion requires the scale of the SfM depth image to fit the scale of the real-world scene measured by the RGBD camera.

Given that the depth measurements for the RGBD depth image are co-registered with the SfM estimated depth image the scale parameter can be directly estimated by minimizing the sum of the squared depth errors [28] between the SfM depth image and the RGBD depth image. Let V denote the set of (x, y) positions that have valid depth measurements for "standard" fusion as described in 4.6. Equation (4.3) shows the error function used to compute the unknown scale value and equation (4.4) shows the solution $\widehat{\alpha}$ that minimizes this error.

$$e(\alpha) = \sum_{(x,y)\epsilon V} \|Z_{rgbd}(x, y) - \alpha Z_{SfM}(x, y)\|^2 \tag{4.3}$$

$$\widehat{\alpha} = \sum_{(x,y)\epsilon V} \frac{Z_{rgbd}(x, y)}{Z_{SfM}(x, y)} \tag{4.4}$$

4.5 RGBD Measurement Noise

The proposed fusion algorithm relies on experimental studies of accuracy and noise for RGBD sensor measurement, e.g., the Kinect sensor. Research in [29] shows that a Gaussian noise model provides a good fit to observed measurement errors on planar targets where the distribution parameters are mean 0 and standard deviation$\sigma_Z = \frac{m}{2f_x b} Z^2$ for depth measurements where $\frac{m}{f_x b} = -2.85e^{-3}$ is the linearized slope for the normalized disparity empirically found in [29]. Since 3D the coordinates for (X, Y) are a function of both the pixel location and the depth, their distributions are also known as shown below:

$$\begin{aligned} \sigma_X &= \tfrac{x-o_x+\delta_x}{f_x}\sigma_Z = \tfrac{x-o_x+\delta_x}{f_x}(1.425e^{-3})Z^2 \\ \sigma_Y &= \tfrac{y-o_y+\delta_y}{f_y}\sigma_Z = \tfrac{y-o_y+\delta_y}{f_y}(1.425e^{-3})Z^2 \\ \sigma_Z &= \tfrac{m}{f_x b} Z^2 \sigma_{d'} = (1.425e^{-3})Z^2 \end{aligned} \tag{4.5}$$

These equations indicate that 3D coordinate measurement uncertainty increases as a quadratic function of the depth for all three coordinate values, where Z denotes the sensed depth at image position (x, y), (f_x, f_y) denotes the camera focal length (in pixels), (o_x, o_y) denotes the pixel coordinate of the image center, i.e., the principal point, and (δ_x, δ_y) denote adjustments of the projected pixel coordinate to correct for camera lens distortion.

However, the quadratic coefficient for the (X, Y) coordinate standard deviation is at most half that in the depth direction, i.e., $(\sigma_X, \sigma_Y) \approx 0.5\sigma_Z$ at the image periphery where $\frac{x-c_x}{f} \approx 0.5$, and this value is significantly smaller for pixels close to the optical axis.

4.6 RGBD and SfM Depth Fusion

For fusing measurements we consider the structured-light measurement of the RGBD sensor to generate a distribution for the unknown true depth of the scene surfaces at each RGBD (x, y) pixel in the depth image. These measurements are considered to be independent and identically distributed to the measurements of the true unknown depth of the scene surfaces from the registered SfM estimated depths. With these assumptions, solving the depth fusion problem is equivalent to estimating the posterior distribution of the true scene depth at each (x, y) position given the distributions for the RGBD and SfM depth values.

Fortunately, previous sections show that Gaussian models are appropriate distributions for both the RGBD and SfM depth values and the parameters of these models are either known (see § 4.5) or estimated continuously (see § 4.1). When both distributions are Gaussian, the posterior distribution can be found analytically and is a well-known result used in pattern recognition and other prediction frameworks, e.g., the Kalman filter as discussed in [30]. Specifically, let the Gaussian noise for RGBD depth at position (x, y) be represented as $\mathcal{N}(Z_{rgbd}, \sigma^2_{rgbd})$ and from the section 2.2.4.3 we have the Gaussian noise for the co-registered SfM depth image at position (x, y)

as $\mathcal{N}(Z_{obs}, \sigma^2_{Z,obs})$. The posterior distribution on the unknown true depth at position (x, y) is also Gaussian and let $\mathcal{N}(\mu_{fused}, \sigma^2_{fused})$ denote the mean and variance parameters of this distribution. Equations (4.6) and (4.7) provide optimal estimates of the mean and variance of the fused depth at position (x, y). The best estimate of the fused depth is given by the highest probability value in the posterior distribution which is the mean fused image, $\mu_{fused}(x, y)$.

$$\mu_{fused} = \frac{Z_{rgbd}\sigma^2_{Z,obs} + Z_{obs}\sigma^2_{rgbd}}{\sigma^2_{Z,obs} + \sigma^2_{rgbd}} \tag{4.6}$$

$$\sigma^2_{fused} = \frac{\sigma^2_{rgbd}\sigma^2_{Z,obs}}{\sigma^2_{rgbd} + \sigma^2_{Z,obs}} \tag{4.7}$$

CHAPTER 5: Results

The experiments were conducted with different setups to test and analyze the depth fusion algorithm. Experiments recorded RGBD image sequenced from ORBBEC Astra RGBD sensors and applied LSD-SLAM to their image streams using their factory-provided intrinsic camera calibration parameters. Each experiment included approximately 60 seconds of RGBD image data at the rate of 30 fps. The recorded RGB images were processed offline by the LSD-SLAM algorithm to generate SfM depth images. Experiments initialize the LSD-SLAM algorithm with the first recorded depth image from the RGBD sensor to facilitate the initial scale approximation. The output from the LSD-SLAM algorithm consisting of the relative pose for every tracked frame and the depth map for each keyframe was then captured to disk. The fusion algorithm was then run offline on the recorded RGBD image stream and LSD-SLAM output files to generate the results shown in this section.

Experiment 1 depicts a indoor office scene at the university. This scene includes specular and dark surface structures at close range that are not measured by the RGBD sensor. Yet, the SfM algorithm estimates depths at a number of locations (on the podium) where ther are significant intensity changes. These additional depths are

Table 5.1: Percentage of occurrence the depths from difference sources

Experiment	RGBD-only depths (%)	SfM-only depths (%)	Fused depths (%)
1	67.7	6.7	25.5
2	55.5	10.0	34.3
3	68.1	15.1	16.6
4	51.0	25.7	23.1
5	68.3	8.8	22.8
Average	62.12	13.26	24.46

(a) (b) (c) (d) (e)

Figure 5.1: Results for three experiments are shown. Images shown are organized into separate columns. Column (a) shows a grayscale image of the scene (b) shows the sensed RGBD depth image (c) shows the SfM-estimated depth image (d) shows the fused image and (e) shows the standard deviation for fused depths (in m.). The fused image has been color-coded as follows: (white) denotes depth locations sensed only by the RGBD sensor, (yellow) denotes depth locations only sensed via SfM, (red) denotes fused (RGBD+SfM) depth locations and (black) denotes depth locations without RGBD or SfM measurements.

evident in the fused results, which includes SfM-only depth measurements in regions in the vicinity of image edges. The experiment demonstrates that depth fusion can improve depth images by obtaining depths from surfaces not measurable by the RGBD sensor.

Experiment 2 depicts a hallway in the UNCC EPIC building that includes both specular surfaces and high intensity illumination from overhead lights. The experiment is a second example showing that depth fusion bolsters over all depth measurement performance by providing the depths when RGBD sensors fail. The SfM takes advantage of the patterns on the surface and estimates the depths irrespective of the nature of its reflectance properties and color.

Experiment 3 depicts the UNCC faculty conference hall. Here a number of scene structures lie beyond the measurement range of the RGBD sensor. Yet, the SfM algorithm is able to estimate the depth of these scene structures (albeit at high variance) providing depths that would otherwise not be possible.

Experiment 4 depicts indoor to outdoor transition. By initializing the LSD SLAM indoor, we get the scale estimated. We can see that in outdoor, when the sensor measurements fails because of the infrared ray flooding, depth estimates from LSD SLAM are used. Often because of IR flooding the sensor measurements gets corrupted, in those scenarios we can consider replacing entire corrupted depth sensor measurements by LSD depth estimation instead of fusion for better results.

Experiment 5 depicts indoor with well lit conditions, even though the sensors are supposed work completely fine in this scenario, there are rare chances of the them failing at steep surfaces. During such failures, depth fusion comes handy.

Table 5.1 quantifies the amount of additional depth information provided via RGBD-SfM depth image fusion. When we average over the five experiments discussed approximately 62% of the fused depths originate from the RGBD sensor alone, approximately 13% original from the SfM algorithm alone, and approximately 24% of the

fused depths results from RGBD-SfM fusion. The addition of 13% of novel depth data is a significant contribution. Further, fused depths account for roughly 24% of depth image data and the measurement error for all of these measurements will be reduced by the fusion. Variance reduction will be greatest for low-variance SfM depth estimates which are typically in textured scene locations close to the camera. Yet, we note that by inspection of equation (4.7), it is theoretically impossible for the variance of any fusion result to increase. Another obvious observation from the Table 5.1 is that the SfM has a largest contribution in Experiment 4 (outdoor scenario) since sensor fails outdoors and it has least contribution in Experiment 5 (well lit indoor) where sensor is at its best.

CHAPTER 6: Conclusion and Futher work

The SfM depth estimation complements the RGBD sensor measurements and can provide depths when RGBD sensors fail. The depth fusion algorithm provides the effective way to augment the RGBD depth stream and results in improved depth images. The experiments conducted shows these improvements in the experimental results for a number of scenarios where RGBD sensors fail including successfully capturing depth for out-of-range RGBD depth locations and successfully capturing depth measurements from specular and dark objects.

As future work, the proposed depth fusion can be generalized to address depth image locations the include measurements having non-Gaussian noise distributions, especially for the outdoor scenario the error noise are not necessarily Gaussian. Also, the depth image gets corrupted by random speckles because of infrared ray flooding, these speckles could be wrongly treated as depth value, since there values are in valid depth range. Fortunately, it is observed that these speckles have particular pattern and a prepossessing logic could be implemented to get rid of them.

Also, experimental results show that the scale and trajectory estimation of SfM depths are not always accurate. Back-propagation of RGBD-SfM fused depth imagery into the SfM algorithm can be exploited to improve the camera pose estimation in the camera pose graph. We know that the fused depths have measurements where SfM fails and also they have lower variance than SfM's, propagating them to SfM leads to improvement in estimated depth values, which leads to better trajectory computation for future frames.

REFERENCES

[1] K. Litomisky, "Consumer RGB-D cameras and their applications," 2012.

[2] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE MultiMedia*, vol. 19, pp. 4–10, Apr. 2012.

[3] C. Jing, J. Potgieter, F. Noble, and R. Wang, "A comparison and analysis of rgb-d cameras' depth performance for robotics application," in *2017 24th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pp. 1–6, Nov 2017.

[4] A. Kadambi, A. Bhandari, and R. Raskar, *3D Depth Cameras in Vision: Benefits and Limitations of the Hardware*, pp. 3–26. Cham: Springer International Publishing, 2014.

[5] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1330–1334, Nov. 2000.

[6] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *2013 IEEE International Conference on Computer Vision*, pp. 1449–1456, Dec 2013.

[7] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment - a modern synthesis," in *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, (London, UK, UK), pp. 298–372, Springer-Verlag, 2000.

[8] P. Zanuttigh, G. Marin, C. Dal Mutto, F. Dominio, L. Minto, and G. M. Cortelazzo, *Operating Principles of Structured Light Depth Cameras*, pp. 43–79. Cham: Springer International Publishing, 2016.

[9] P. Zanuttigh, G. Marin, C. Dal Mutto, F. Dominio, L. Minto, and G. M. Cortelazzo, *Operating Principles of Time-of-Flight Depth Cameras*, pp. 81–113. Cham: Springer International Publishing, 2016.

[10] P. Drap and J. Lefevre, "An exact formula for calculating inverse radial lens distortions," *Sensors*, vol. 16, no. 6, 2016.

[11] Z. Zhang, *Camera Calibration*, pp. 76–77. Boston, MA: Springer US, 2014.

[12] Y. M. Wang, Y. Li, and J. B. Zheng, "A camera calibration technique based on opencv," in *The 3rd International Conference on Information Sciences and Interaction Sciences*, pp. 403–406, June 2010.

[13] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *Int. J. Comput. Vision*, vol. 56, pp. 221–255, Feb. 2004.

[14] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 611–625, March 2018.

[15] C. Harris and M. Stephens, "A combined corner and edge detector," in *In Proc. of Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[16] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, Nov 2004.

[17] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, (Nara, Japan), November 2007.

[18] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, pp. 346–359, June 2008.

[19] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for RGB-D cameras," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2100–2106, Nov 2013.

[20] J. Schmidt, H. Niemann, and S. Vogt, "Dense disparity maps in real-time with an application to augmented reality," in *WACV*, 2002.

[21] D. Gallup, J. M. Frahm, P. Mordohai, and M. Pollefeys, "Variable baseline/resolution stereo," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2008.

[22] G. Welch and G. Bishop, "An introduction to the kalman filter," tech. rep., Chapel Hill, NC, USA, 1995.

[23] Â. P. Gavin, "The levenberg-marquardt method for nonlinear least squares curve-fitting problems," 2013.

[24] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2O: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3607–3613, May 2011.

[25] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," pp. 834–849, 2014.

[26] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003.

[27] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, pp. 376–380, Apr. 1991.

[28] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[29] J. Papadakis and A. R. Willis, "Real-time surface fitting to RGBD sensor data," in *SoutheastCon 2017*, pp. 1–7, March 2017.

[30] P. S. Maybeck, "Stochastics models, estimation, and control: Introduction," 1979.