

# Learning Depth and Visual Odometry From Light Fields

School of Aeronautical, Mechanical and Mechatronic Engineering

The University of Sydney

Joseph Daniel

May 7, 2020

# **Declaration of Contribution**

# Abstract

The meteoric rise of mobile robotics has seen traditionally difficult-to-navigate environments like the home, the road and the ocean, become standard operating environments for autonomous machines. This rapid advancement has culminated in an increasing need for accurate robotic navigation, where the pinnacle of autonomy is the ability to operate adaptively in unconstrained environments. Central to this capability is the ability to perceive motion - a task that has recently drawn considerable attention from those in the computer vision community and given rise to a family of algorithms called *visual odometry*.

The primary contribution of this work is a novel, data-driven pipeline for visual odometry, which combines recent successes in plenoptic imaging with the pattern recognition capabilities of convolutional networks. Notably, we formulate our algorithm as an *unsupervised* learning problem, meaning neither ground-truth odometry, nor depth is used to train our pipeline. The self-supervising nature of our algorithm equips it with a robustness to effects like thermal expansion, and shock - effects which are all too common in mobile robotics, and likely to render equipment calibration invalid. Being self-supervised, our algorithm is capable of adapting and learning from experience, even whilst operating in-situ, building resilience to these destabilising effects.

We train and validate our algorithm on a dataset, collected and annotated as part of this thesis project. A robotic arm with a known kinematic model is used to acquire our dataset, providing valuable ground-truth data for validation.

Finally, we compare our proposed pipeline to two existing families of visual odometry algorithms. The first is a group of algorithms that directly model the geometry of multi-view imaging, solving directly for ego-motion using closed-form solutions. The second family uses data-driven approaches, but unlike this work they do not harness the power of plenoptic imaging. In doing so, we demonstrate the power of plenoptic imaging as a novel imaging modality for robotics, in terms of accuracy, computational convenience and adaptability.

# Acknowledgements

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.1.1	Visual Odometry and Robotic Perception . . . . .	1
1.1.2	Computational Imaging . . . . .	3
1.2	Problem Statement . . . . .	4
1.3	Contributions . . . . .	5
1.4	Outline . . . . .	6
<b>2</b>	<b>Background</b>	<b>7</b>
2.1	Geometry in Computer Vision . . . . .	7
2.1.1	The Pinhole Camera . . . . .	7
2.1.2	Epipolar Geometry and the Fundamental Matrix . . . . .	8
2.2	The 4D Light Field . . . . .	9
2.2.1	Visualising the Light Field . . . . .	11
2.3	Machine Learning in Computer Vision . . . . .	13
<b>3</b>	<b>Literature Survey</b>	<b>16</b>
3.1	Geometric Approaches to Visual Odometry . . . . .	16
3.1.1	A Survey of Geometric Approaches to Pose-Estimation . . . . .	17
3.1.2	Iterative Methods . . . . .	18
3.1.3	Closed-form Methods . . . . .	19
3.2	Data-driven Approaches to Visual Odometry . . . . .	20
3.2.1	A Taxonomy of Data-Driven Methods for Pose-Estimation . . . . .	20
3.2.2	Simultaneous Unsupervised Depth and Pose Estimation . . . . .	22

3.3	Convolutional Neural Networks and Light Fields . . . . .	25
<b>4</b>	<b>Methodology</b>	<b>28</b>
4.1	Data Acquisition . . . . .	28
4.1.1	Ground Truth Pose Data . . . . .	28
4.1.2	Imagery . . . . .	30
4.2	Depth and Visual Odometry Pipelines . . . . .	31
4.2.1	Pose Estimation . . . . .	31
4.2.2	Depth Estimation . . . . .	32
4.2.3	Photometric Warp Loss . . . . .	32
4.2.4	Input Methods . . . . .	34
4.3	Complete Light Field Reconstruction . . . . .	37
<b>5</b>	<b>Results</b>	<b>40</b>
5.1	Experimental Setup Details . . . . .	40
5.2	Pose Estimation and Visual Odometry . . . . .	41
5.2.1	Single-view Reconstruction Approach . . . . .	41
5.2.2	Light field Reconstruction Approach . . . . .	44
5.2.3	Monocular Approach . . . . .	47
5.2.4	Comparison between Approaches and Summary . . . . .	48
5.3	Depth and Photometric Reconstruction . . . . .	49
5.3.1	Single-view Reconstruction Approach . . . . .	50
5.3.2	Light field Reconstruction Approach . . . . .	51
5.3.3	Monocular Approach . . . . .	52
5.3.4	Comparison between Approaches and Summary . . . . .	52
<b>6</b>	<b>Discussion</b>	<b>53</b>
6.1	Comparison between Unsupervised Algorithms using Light Field Data . . . . .	53
6.1.1	Enforcing Scale Consistency . . . . .	53
6.2	Comparison between Input Methods . . . . .	53
6.2.1	Performance of Epipolar Images . . . . .	53
6.2.2	Performance of Volumetric Stacks . . . . .	53
6.2.3	Performance of Focal Stacks . . . . .	53

6.3 Comparison to existing Monocular Approaches . . . . .	53
---	----

# List of Figures

1.1 Examples of robots that operate in unconstrained spaces. . . . .	2
2.1 The pinhole model of the camera . . . . .	8
2.2 The fundamental matrix in epipolar geometry . . . . .	9
2.3 The two-plane parameterisation and the free-space assumption . . . . .	10
2.4 An example of a camera array mounted on a robotic arm. This camera array is configured as 17 sub-apertures arranged on a single plane in a cross-hair formation. Camera arrays sample several views of the same scene and are thus capable of acquiring a sparse sample of the light field. This is the camera that will be used throughout this thesis project. . . . .	12
2.5 Epipolar Plane Images (left): Shown as a slice of a volume, the images formed by dissecting the image in the $s,u$ and $t,v$ planes are characterised by sheared straight lines, with the grade of the slope encoding the amount of parallax experienced by a pixel at that $u$ or $v$ coordinate. Synthetic aperture focusing (right): taking the average of every image from the camera array yields an image where different parts are in focus depending on the alignments of the images. . . . .	12
2.6 Multilayer Perceptron (left): each output from one layer is fed into the inputs of every node in the subsequent layer. Convolutional Neural Networks (middle) on the other hand have a ‘receptive field’, taking advantage of the spatial coherence of pixels in image data. The convolutional upsampling operation (right) is frequently used to upsample a low-dimensional feature space, to a higher dimensional one. It is often employed as a ‘learned’ information decompression. . . . .	14

3.1	Machine learning approaches have demonstrated strong results in simultaneously estimating depth maps and relative poses between images. Typically, a pair of CNN's is used, one for each task. The depth CNN is provided with the first view at $t = 1$ and the pose CNN is provided both views. . . . .	22
3.2	To provide a supervisory signal to the depth and pose CNN's, Garg et al. [18] suggested using photometric reconstruction. The loss function is formulated by taking the difference between the reconstructed image $\hat{I}_{t1}$ and the actual image $I_{t1}$ , shown in equation 3.1. . . . .	24
3.3	Angular Filters (left): by fixing the coordinate $(u, v)$ and varying $(s, t)$ , the 2D grid of images can be presented to a neural network in the form of orthographic images. Spatial Filters (right): instead of fixing $(u, v)$ coordinates, $(s, t)$ is fixed while $(u, v)$ is varied. This presents information to the neural network as a grid of perspective images. . . . .	26
3.4	Orthographic projection (left) [4]: the lines of sight to all points in the scene are parallel to one other, and perpendicular to the projective plane. Perspective projection (right) [23]: lines of sight converge to a point giving the effect that distant objects appear smaller than nearby objects. . . . .	26
4.1	The experimental setup utilises a Universal Robots UR5E robotic arm to precisely measure ground-truth pose to allow effective evaluation of the projects visual odometry results. . . . .	29
4.2	Tiled $U, V$ slices (top): this is an intuitive way of slicing the light field, which tiles images from each sub-aperture side-by-side. Tiled $T, V$ slices (bottom): somewhat less intuitively, this method instead tiles the vertical epipolar-plane images side-by-side. We call these <i>wide</i> EPIs. As shown, the overall dimensionality and shape of the light field is unchanged, we simply present the information in a different order, revealing different encodings of the captured scene. We can also tile the light field as $S, U$ slices, stacking them vertically, which we refer to as <i>tall</i> EPIs. . . . .	36

4.3 Our proposed encoder module which encodes the tiled EPIs, downsampling them to the shape expected by our depth and pose networks. Notably, features which have been learned in $T$ , $V$ as well as $S$ , $U$ are concatenated with the stacked $U$ , $V$ slices (exactly as seen in our first input-method.) For brevity, we have only shown a small patch of the tall and wide EPIs, but the reader is reminded that these tilings are in fact very tall, and very wide. . . . .	36
4.4 Given knowledge of how one sub-aperture has moved through space, we can also compute how any other sub-aperture has moved by chaining the transforms in the order shown. . . . .	38
5.1 Instantaneous absolute errors when <b>tiled epipolar images</b> are the input method. . . . .	41
5.2 Instantaneous absolute errors when <b>focal stacks</b> images are the input method. . . . .	42
5.3 Instantaneous absolute errors when <b>volumetric stacks</b> are the input method. . . . .	42
5.4 Estimated and true trajectories for input sequence 16. . . . .	43
5.5 Estimated and true trajectories for input sequence 44. . . . .	43
5.6 Instantaneous absolute errors when <b>tiled epipolar images</b> are the input method. . . . .	44
5.7 Instantaneous absolute errors when <b>focal stacks</b> are the input method. . . . .	45
5.8 Instantaneous absolute errors when <b>volumetric stacks</b> are the input method. . . . .	45
5.9 Estimated and true trajectories for input sequence 16. . . . .	46
5.10 Estimated and true trajectories for input sequence 44. . . . .	46
5.11 Estimated and true trajectories using monocular imagery. . . . .	47
5.12 Instantaneous translational errors vs. input methods. . . . .	48
5.13 Cumulative translational errors vs. input methods. . . . .	48
5.14 While using light field reconstruction has a small advantage over single view reconstruction in translational terms, we find that in terms of rotation the benefits are far more significant. Once again, the tiled EPI insertion strategy continues to demonstrate the smallest error in terms of rotation on our validation dataset. . . . .	49

# List of Tables

4.1	PoseNet Convolutional Network Architecture . . . . .	31
4.2	DispNet Convolutional Network Architecture. . . . .	33
5.1	Mean Absolute Instantaneous Error and Cumulative Error . . . . .	43
5.2	Mean Absolute Instantaneous Error and Cumulative Error . . . . .	46



# Chapter 1

## Introduction

Humans have a remarkable ability to perceive visual stimuli emanating from the world around them. Not only do we identify different objects, scene depth, movement, and colour with ease, but these scenes can even elicit enjoyment and meaning for us. Man-made machines on the other hand are much more easily decomposed into a set of deterministic modules that require explicit, well defined instruction sets. This thesis is concerned with developing a set of algorithms that breaks down, and utilises the wealth of information in image data to produce valuable signals that can be employed in robotic perception.

### 1.1 Motivation

#### 1.1.1 Visual Odometry and Robotic Perception

True autonomy in mobile robotics requires the ability to perceive the world adequately. In particular, understanding the structure and size of the space inhabited by the robot, as well as its own movement through that space are vital to its ability to navigate and interact with the world. Computer vision algorithms in combination with increasingly lower-cost and higher quality imaging hardware offers a popular solution, encompassing enormous diversity in sensing modalities and precipitating the development of powerful perception systems in modern robotics. Plenoptic imaging is one such expression of evolving imaging technologies that continues to yield promising results in tasks such as mapping [30], underwater imaging [46], low light imaging [14], and classification [48].

This work is concerned with the application of **plenoptic imaging** in two intimately related

tasks: **visual odometry** which involves estimating the motion of a camera in 3D space, and **depth reconstruction** which estimates the shape of the scene in an image. We elucidate the motivation for addressing these challenges by looking at one of the most important challenges currently being tackled in robotics: localisation and navigation. A robots locale is a vital piece of information required to effectively plan paths and navigate through any environment. The challenge however, is that localisation typically requires a map of the space. In an unknown environment where a map doesn't exist, the robot must simultaneous tackle the problems of reconstructing the environment's geometry, and localising itself within that geometry. This is a challenge known as SLAM (Simultaneous Localisation and Mapping), and has proven an enigmatic robotics problem for at least 30 years [8, 30]. Odometry is a crucial component of even the most sophisticated SLAM algorithms. In this work, odometry is treated as a problem which can be tackled with visual perception and computer vision algorithms.

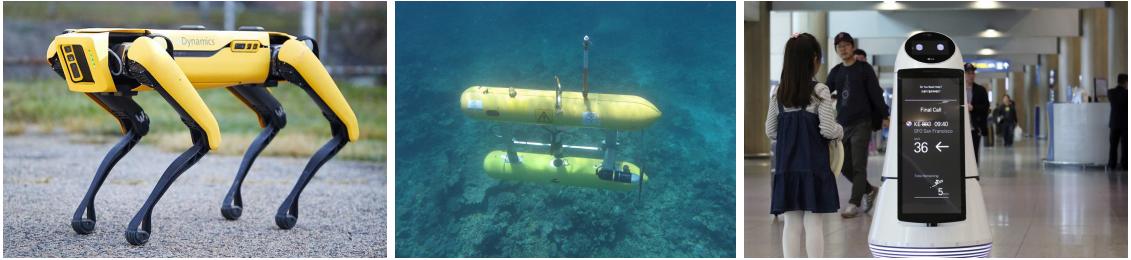


Figure 1.1: Autonomous mobile robots are increasingly operating in unconstrained environments, which can be characterised by uneven terrain, weak GPS signal, or crowds of people. An important challenge for the broader adoption of these types of mobile robots is navigational capabilities, which relies fundamentally on mapping and localisation. *Left:* Spot by Boston Dynamics. *Center:* AUV Sirius by the Australian Center for Field Robotics. *Right:* Troika by LG, currently deployed at Incheon International Airport in Seoul.

One might question the practicality of using image data for these tasks when great success has been found with more specialised sensing modalities; motion estimation for example is typically addressed with inertial sensor measurements or Global Positioning System (GPS) receivers. Other robotic systems employ lidar, acoustic range-finding, time-of-flight cameras or structured-light cameras to gain access to 3D models of the environment. Cameras however, can often offer superior qualities in size, weight, cost, power consumption, and they deliver a rich, highly detailed representation of the world. Furthermore, unlike *active* sensor technologies which sense the world by

‘illuminating’ it, cameras are *passive* sensors, meaning they do not interfere with one another and can be employed in a more diverse range of environments. Thanks to their ability to adjust their exposure period, aperture size and sensor gain, cameras are also highly capable in a variety of lighting conditions and environments.

Furthermore, the applications of visual odometry extend beyond the context of SLAM algorithms. For example, a multi-rotor Unmanned Aerial Vehicle (UAV) can fuse measurements from inertial and visual measurements to hover in place in GPS denied environments. Outside of the field of robotics, Augmented Reality (AR) applications are employing visual odometry and SLAM to render virtual 3D models in real time on top of camera feeds. Proen   [42] envisages an AR application that employs visual odometry to guide an operator through a contaminated nuclear facility, whilst rectifying existing 3D models of the facility. The adversities of installing guidance and navigation infrastructure in such facilities can be extrapolated to cases such as planetary rovers and Remotely Operated Underwater Vehicles (ROVs).

### 1.1.2 Computational Imaging

Despite the many advantages of visual perception in robotics, we should not discount the inherent trade off bound to cameras by the laws of optics. Each pixel in a modern digital camera sensor integrates the number of arriving photons in order to form an image. A lens can help concentrate rays from a larger area through the cameras aperture; the larger the aperture, the more light is received at the sensor, allowing fine control over the exposure. A larger aperture diameter however also gives rise to a shallower depth of field, which can be used to produce aesthetically pleasing images in photography, but limits the regions of the image where computer algorithms perform well. Similarly, more light can be collected by integrating photons over a longer period of time, but in robotics where the camera tends to be moving, this leads to motion blur and reduced image clarity.

In response to these limitations, *computational photography* has catapulted modern imaging into an age where optics, algorithms, sensors and illumination are designed in coordination with one another. Take modern smart-phones as an example, with their relatively slow lenses, small sensors, and cheap optics. In comparison to the enormous lenses and excellent sensors of their DSLR cousins, smart-phones should by all rights produce lower quality images. Modern smart-phones however benefit from powerful CPUs, hardware-accelerated graphics, and an entire community of software engineers working on image processing algorithms. Many of the effects that were once unique to expensive imaging devices are now being replicated on smartphones. Low light imaging is enhanced

using algorithms such as Google’s ‘Night Sight’, and more recently machine learning been used to improve clarity, colour accuracy and signal-to-noise ratios. Bokeh, an effect that is typically difficult to achieve on a wide lens is now replicated with ‘Portrait Mode’, another smartphone feature employing a range of computational imaging techniques.

This work similarly adopts computational imaging as an approach to visual odometry. Humans are exceptionally well adapted to tasks involving visual motion and depth perception - our two eyes allow us to process the 3D geometry of the scene, while our learned experiences are often able to fill in the gaps where geometric information is insufficient or unavailable. Image processing algorithms are not equipped with this same kind of human intuition, and so the deceptively complex task of estimating the 3D structure of a scene from a sequence of images continues to attract attention from the computer vision community [13, 40, 17, 53].

The price of camera components is decreasing while image quality continues to improve, not only making cameras an attractive perception module for autonomous robotics, but also spurring the popularity of multiple-view imaging. Embracing this idea, this work develops an algorithm that performs visual odometry by taking advantage of the rich geometric information exposed when using multiple views. Thus, the vocabulary and concepts from the literature on light field imaging are adopted broadly throughout this work. More broadly however, the principle underpinning much of this thesis is that novel imaging devices that break away from the traditional pinhole model of the camera have vast implications in the field of machine vision. We need not look any further than the animal kingdom to see why this is true - the biological eye is estimated to have evolved independently no fewer than 50 times, each variant acutely adapted to the particular set of challenges in their environment. The diversity and evolutionary ingenuity in biological visual perception systems prompts an important question in robotics and computer vision - how best should we equip robots to see the world given a particular set of challenges and environments? In this thesis, a camera array which is a simple yet versatile extension of the stereo camera is used to develop the algorithms that address the visual odometry and depth reconstruction problems.

## 1.2 Problem Statement

There is a well established library of solutions addressing the visual odometry and depth reconstruction problems. Some use closed form solutions that directly solve for geometry and ego-motion, while others, like this work utilise a data-driven approach to indirectly model visual odometry. These

two families of algorithms, which will be referred to as the *geometric* and *data-driven* approaches respectively throughout this work, form two crucial touchstones from which our pipeline will be evaluated.

An important challenge addressed in this work that is absent in the present literature of either family, is the ability to perform visual odometry using an *uncalibrated* camera array. Robots operating in challenging environments are frequently subjected to destabilising effects such as thermal expansion, vibration and shock, frequently rendering calibrations invalid. Multiple-view geometry, which is harnessed by the *geometric* family of algorithms, is materially dependent on knowing the orientation and position of the cameras; rotating a camera even a few arc-seconds from its calibrated position results in a disproportionately magnified error in the image being formed, especially in large, open spaces where field robotics tend to operate.

Existing *data-driven* approaches on the other hand, have typically used monocular or stereo imagery, overlooking the richness of light field imagery as a possibility for improving robustness and accuracy. Within the data-driven family of algorithms, a relatively new *unsupervised* approach has emerged, cleverly piecing together the available information to learn visual odometry from raw footage alone. The unsupervised nature of this approach means that the model calibrates itself in a fashion, learning from raw data to produce ego-motion predictions. Unsupervised machine learning makes ‘online-learning’ possible, in which models take advantage of new data becoming available in-situ, improving performance and building robustness to different environments. Most importantly however, unsupervised models are able to adapt to errors in the calibration of the equipment without requiring hand-labelled data to supervise the learning process. A self-calibrating and adaptive perception module which is robust to adverse effects is an attractive capability in robotics. Once again however, existing unsupervised algorithms have not been applied to light field data, prompting a natural, yet important next step which is the subject of this work: combining the richness of light field data with the self-calibrating capabilities of unsupervised learning algorithms to perform visual odometry.

### 1.3 Contributions

This work builds on existing data-driven approaches, suggesting an adaptation of conventional 2D machine learning algorithms to enable unsupervised learning on light field data. Specifically, we suggest two novel variants of an existing algorithm that enables the ingestion of 4D imagery. Fur-

thermore, we describe three encodings of the light field that presents 4D data as 2D slices, from which convolutional networks are able to learn the tasks of depth- and pose-estimation. The three light field encodings are compatible with both variants of the proposed algorithm, allowing for a direct comparison between input methods. We demonstrate that our algorithm out-performs existing monocular approaches for depth and visual odometry, reinforcing the benefits of light field technology in robotic imaging.

## 1.4 Outline

Concepts from computer vision, machine learning and light field imaging are used heavily throughout this thesis, and **Chapter 2** begins by providing an overview of the relevant background information. On the subject of computer vision, it describes the pinhole model of the camera, the fundamentals of multiple view geometry, and develops intuition relevant to light field imaging. Machine learning is presented as an approach to solving computer vision problems, with emphasis directed towards convolutional neural networks, as an algorithm that continues to gain popularity in image based problems.

The existing approaches for performing visual odometry and depth estimation are reviewed in **Chapter 3**, presented in two categories: geometric approaches which rely on directly modeling movement through 3D space, and machine learning approaches which take advantage of massive datasets to build resilient, data-driven solutions.

**Chapter 4** introduces a novel light-field based algorithm for simultaneously learning depth and pose. Though algorithms have been proposed in previous work that perform the two tasks either monocularly or with stereo camera pairs, this thesis differs in the use of light-field imagery to gain improved access to geometric information in the scene. While previous work has benefitted from large scale open source datasets, similar pose-stamped footage does not exist publicly for camera arrays. The acquisition of such a dataset thus forms one of the objectives of this thesis, the methodology for which is discussed in the chapter.

**Chapter 5**, presents the results from initial experiments using the pipeline described in Chapter 4, performed on both the early dataset and the open source KITTI dataset [19]. The early challenges and milestones are discussed, and a qualitative discussion on the successes, failure modes, and future directions is presented.

# Chapter 2

## Background

### 2.1 Geometry in Computer Vision

The pixel is the building block of digital imagery. Thanks to our ability to fabricate advanced circuits on the scale of nanometers, digital camera sensors have become ubiquitous - making their way into all manner of consumer devices. Because the photodiodes in camera sensors are typically arranged on a 2D plane, most of our operations on digital images manipulate the 2D structures present in that pixel data. However, it is important to remember that the 2D shapes and structures that appear on the sensor plane are projected versions of a 3D world.

#### 2.1.1 The Pinhole Camera

The most basic kind of camera, the pinhole camera projects a 3D world point  $[X, Y, Z]$  to a homogeneous coordinate on a 2D plane  $[u, v, w]$ . This relationship is captured in the  $3 \times 3$  camera intrinsics matrix and is typically written as

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \quad (2.1)$$

The intrinsics matrix can alternatively be thought of as mapping ray directions to pixel coordinates:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tan(\theta) \\ \tan(\phi) \\ 1 \end{bmatrix}. \quad (2.2)$$

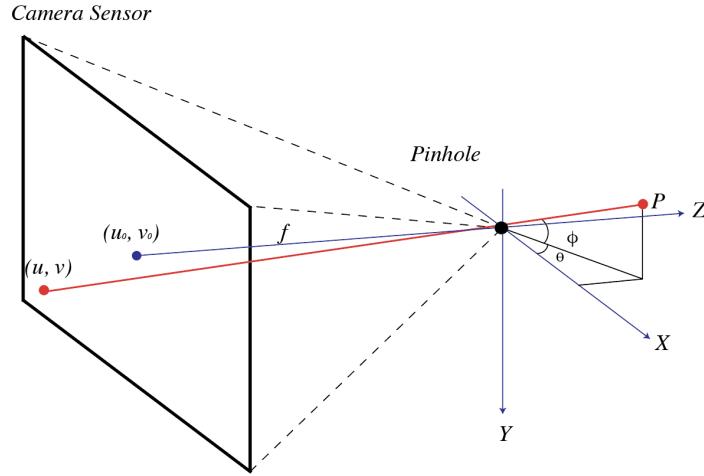


Figure 2.1: The pinhole model describes how a light ray originating from world coordinate  $P$  is received at pixel coordinate  $(u, v)$ . Equivalently, the pinhole model can be thought of as associating the light ray of elevation  $\phi$  and azimuth  $\theta$  with the specific pixel coordinate  $(u, v)$ . Since the relationship is a one-to-one mapping between ray directions and pixel coordinates, the inverse of the intrinsics matrix  $K^{-1}$  associates each pixel with a corresponding ray direction.

### 2.1.2 Epipolar Geometry and the Fundamental Matrix

Epipolar geometry describes the relationship between two cameras, and imposes a set of constraints which makes it possible for us to draw meaningful geometric information when we have two views of the same scene. This relationship is encapsulated in a  $3 \times 3$  matrix called the Fundamental matrix  $F$ . The culmination of the epipolar constraint is that any point in 3 dimensional space that appears at pixel  $[u_1, v_1, 1]$  in one view and  $[u_2, v_2, 1]$  in the second view must satisfy the relationship:

$$[u_1, v_1, 1]^T F [u_2, v_2, 1] = 0. \quad (2.3)$$

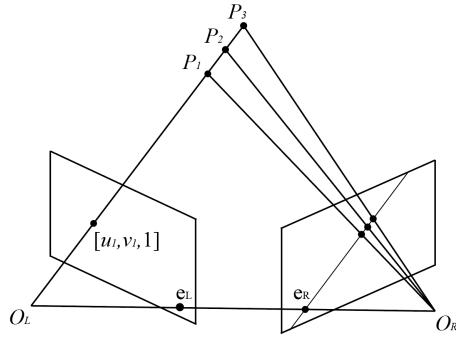


Figure 2.2: Epipolar geometry is the geometry of stereo camera pairs. Without knowing depth, the point  $[u, v, 1]$  can be projected to any number of points  $P_1, P_2, P_3\dots$  in 3D space, having many possible projections on the sensor plane of the right camera. The Fundamental matrix however constrains the set of possible projections to lie on a straight line called the epipolar line.

The Fundamental matrix thus describes the rotation and translation between two cameras up to scale. Any point projected from pixel coordinate  $[u_1, v_1, 1]$  must lie on the epipolar line in the second image, which is formed by the intersection of the epipolar plane and the imaging plane.

## 2.2 The 4D Light Field

Light field imaging has emerged as a powerful tool in computer vision for robotics, offering a rich higher-dimensional representation than what can be captured by conventional optics. The underlying principle used to describe the light field is the plenoptic function, a 7-dimensional mapping that assigns radiance values to the light rays at every position in space, in every orientation, at all wavelengths, throughout all of time [1]. This can be formally expressed as  $L(x, y, z, \theta, \phi, \lambda, t)$ , and measured in  $W/m^2/sr/nm/s$ .

Levoy et al. [32] shows that with the addition of practical constraints however, the plenoptic function can be expressed more concisely as a parameterisation of 4 variables. Pixels on camera sensors integrate the number of photons arriving at them over a finite period of time removing the temporal dimension, and each colour channel can be thought of as a monochromatic sampling of the light field, removing the spectral dimension. Additionally, and importantly, is the constraint that the radiance of light rays propagating through a vacuum do not change if samples are restricted to the convex hull of the scene, thus reducing the overall dimensionality of the plenoptic function by

one additional parameter [32]. To illustrate this, one could think of the light rays leaving the inside of a bowl sat upright on a table. Many of those rays may only travel a small distance before being blocked by the inside of the bowl itself, meaning those rays will never be registered by any practical measurement device. As shown in Figure 2.3 if we consider only the convex hull of the bowl however, we are required only to represent the value of the plenoptic function on the encapsulating surface of the object [22].

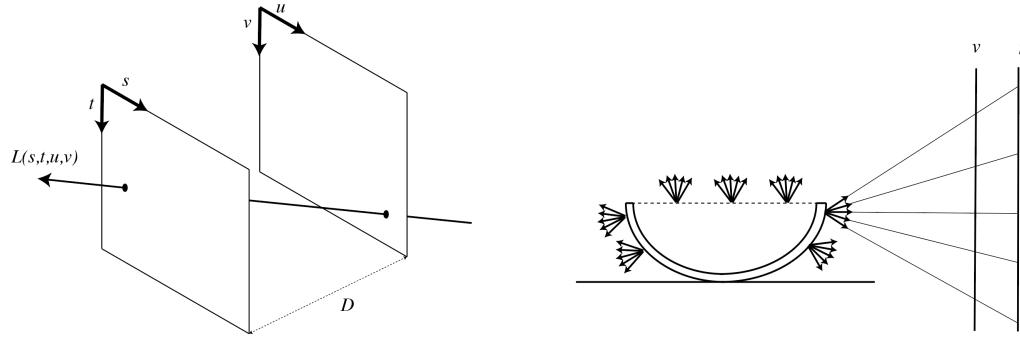


Figure 2.3: Two-plane parameterisation (left): the plenoptic function can be described by the radiance along a ray passing through two parallel planes. The free space assumption (right): if we consider only the bundle of rays leaving from the convex hull of the object at a particular instance in time, in a single colour channel, we can parameterise the light rays as a function of 4 variables rather than 7.

Also illustrated is a common convention for describing light rays in this 4 dimensional space called the two plane parameterisation [22]. In this parameterisation, two parallel planes are used to fix both the position and orientation of each ray by fixing their points of intersection with two parallel planes. By convention, the plane closest to the scene is termed  $u, v$  and the plane closest to the camera sensor is the  $s, t$  plane.

This 4D realisation of the light field originated as a model of rendering 3D computer graphics, one which shifted emphasis from notions of texture and geometric primitives to modeling the behaviour of light rays permeating space. Since then however, the conceptual framework of the light field has drawn a following of researchers at an intersecting region of signal processing, computer vision and robotics [12].

### 2.2.1 Visualising the Light Field

#### **TODO: 2D Grids of 2D Images**

Light field imaging finds a foothold in this work through the utilisation of camera arrays, which are devices that sample multiple views of the same scene. Using a camera array is a simple method for acquiring a sparse sample of the light field, where the position of the camera determines  $(s, t)$  while the location of the pixel determines  $(u, v)$  [49]. The images captured from a camera array are mapped easily to the 4D light field.

#### **TODO: Show a figure**

#### **Epipolar Plane Images**

One way that this geometric information can be easily visualised is by taking slices of the light field image in the  $s, u$  or  $t, v$  axes as shown in Figure 2.5. While the idea of taking a 2D slice from the 4D image can seem complex, the task of generating a so called ‘epipolar plane image’ from a camera array is deceptively simple. Images captured from camera arrays can be stacked to form a solid volume, from which 2D slices can be sampled. Each of these slices yields an image characterised by sheared straight lines, encoding information about the geometry of the scene, including depth and occlusions [7].

#### **Focal Stacks**

The light field can alternatively be visualised by processing the image into a ‘focal stack’. Focal stacks closely resemble images with shallow depth of field such as those that can be captured from a commercial DSLR camera. Light field focal stacks differ from focus in the optical sense however in that they are synthetic and can be computed after the image has been taken, allowing control over depth of field and the focal plane in post-processing. Focal stacks can be computed from camera array images by layering images over one another and taking the average value for each pixel. The result is that parts of the scene that closely overlap appear in focus while areas with poor overlap create a ‘bokeh’ effect.

These representations of the light field play an important role in this thesis project as we experiment with different methods for feeding light field images to the machine learning pipeline. An important consideration in any machine learning algorithm is the feature space - based on what particular inputs will the algorithm be making its decision? Raw images contain millions of mea-

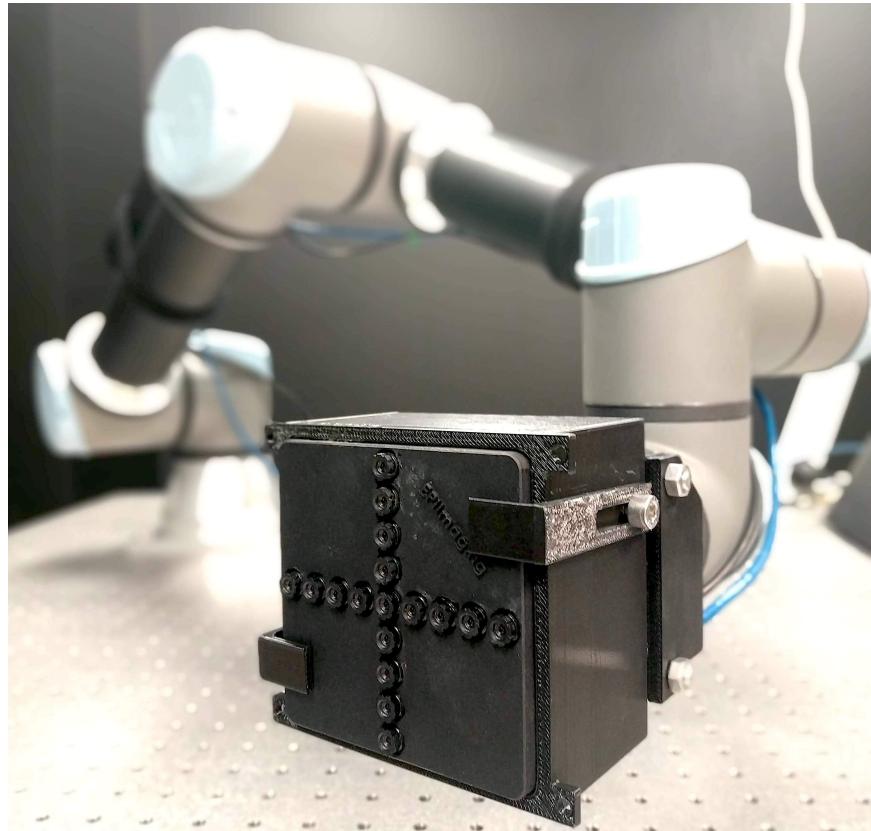


Figure 2.4: An example of a camera array mounted on a robotic arm. This camera array is configured as 17 sub-apertures arranged on a single plane in a cross-hair formation. Camera arrays sample several views of the same scene and are thus capable of acquiring a sparse sample of the light field. This is the camera that will be used throughout this thesis project.

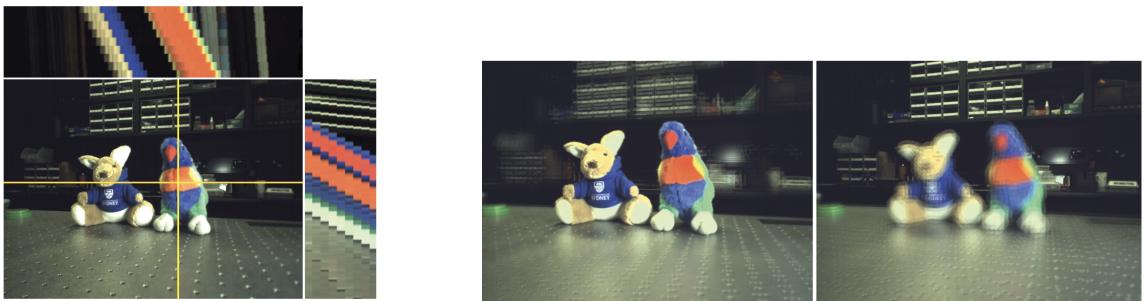


Figure 2.5: Epipolar Plane Images (left): Shown as a slice of a volume, the images formed by dissecting the image in the  $s,u$  and  $t,v$  planes are characterised by sheared straight lines, with the grade of the slope encoding the amount of parallax experienced by a pixel at that  $u$  or  $v$  coordinate. Synthetic aperture focusing (right): taking the average of every image from the camera array yields an image where different parts are in focus depending on the alignments of the images.

surements and thus represent an incredibly high-dimensional feature space for neural networks to process. Light field images are several times larger, and thus it is important that some form of dimensionality reduction is used to ease the training process. With the goal of investigating effective methods of feeding light fields to neural networks, this thesis will explore the use of three different light field formats as the entry point to the machine learning pipeline. The first two will be the focal stack, and epipolar plane image described above, interpreting the images as a 3 dimensional volume created by stacking 2 dimensional images on top of one another. The third will treat the light field as a 4 dimensional volume, requiring a 4D signal processing pipeline to fully take advantage of the dimensionality.

## 2.3 Machine Learning in Computer Vision

An oft-quoted anecdote in computer vision tells of MIT researcher Seymour Papert, who in 1966 assigned a summer project that sounded simple enough, namely to construct a ‘visual system’ that could describe what objects it saw by name [41]. While the regimes of computer vision have evolved substantially since 1966, many of the ideas, and challenges have persisted. This is embodied in the popularity of projects such as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [45], drawing researchers from institutions around the world. It was at the ILSVRC annual challenge where in 2012, a convolutional neural network achieved a top-5 error rate of 15.3%, outperforming all previous submissions by 10.8% [29]. Until ‘AlexNet’ in 2012, the competition consisted of competitors introducing algorithms that produced marginal improvements year on year. Needless to say, an improvement of over 10% generated noise in the computer vision community, drawing attention to the powerful capabilities of deep learning.

While deep neural networks for computer vision have gained massive popularity since the success of AlexNet, the history of neural architecture models begins much earlier, with the perceptron as described by Frank Rosenblatt in [43]. The fundamental building block of neural networks, the ‘perceptron’ is a module that accepts several inputs, and produces a single output computed as a weighted sum of each input - allowing complex functions to be approximated when several perceptrons are layered together as a ‘multilayer perceptron’ [38]. The process of finding the optimal set of weights that produce the desired output given a set of inputs is referred to as training, and in practice is usually found by optimising some cost function using the backpropagation algorithm [44]. Backpropagation uses the chain rule of calculus to calculate partial derivatives of each weight

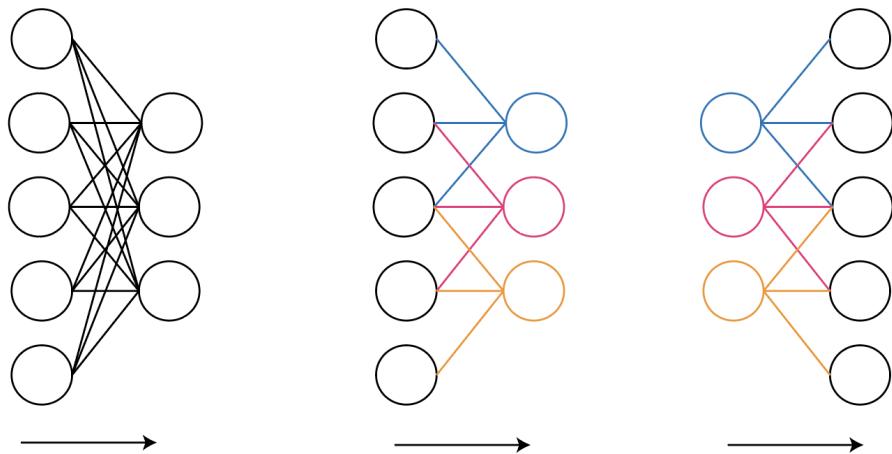


Figure 2.6: Multilayer Perceptron (left): each output from one layer is fed into the inputs of every node in the subsequent layer. Convolutional Neural Networks (middle) on the other hand have a ‘receptive field’, taking advantage of the spatial coherence of pixels in image data. The convolutional upsampling operation (right) is frequently used to upsample a low-dimensional feature space, to a higher dimensional one. It is often employed as a ‘learned’ information decompression.

in the network with respect to the cost function. A gradient based optimisation algorithm such as stochastic gradient descent (SGD) can then be employed to minimise the cost function. Thus, it is important that each step in the computation of a neural networks output be differentiable - that is, it must support the backpropagation of gradients or else gradient-based optimisation will fail. The ‘fully-differentiable’ requirement for neural networks is an important consideration in this work; a novel cost function is described in Chapter 4, where each step in its computation must be differentiable with respect to the output of the previous step.

Convolutional neural networks (CNN’s) are similar to multilayer perceptrons, but introduce a spatial invariance that makes them particularly well suited for extracting high-level features from image data. Where multilayer perceptrons are densely connected, the connectivity pattern of each node in a CNN takes advantage of the hierarchical organisation of patterns in image data by using a ‘receptive field’. Biologically inspired, these nodes respond to stimuli only in their receptive field, and so they typically learn to identify salient features in the image - combinations of pixels that represent some kind of underlying structure. As these types of networks grow deeper, the features that they learn typically become more complex [31]. Early layers are provided with a small region of the image, typically a window between 3 and 11 pixels wide, and so will usually learn primitive

features such as lines and corners. Deeper layers however may learn to combine low-level geometries, recognising higher-level features such as eyes and mouths, and eventually even human and animal faces.

CNN's are thus popular 'feature extractors' in computer vision - their ability to learn to respond to different types of stimuli mean that they have been used as a dimensionality downsampling tool, taking the millions of dimensions present in digital images and compressing them to a feature space with a much smaller number of parameters. Closely related is the convolutional upsampling operation which performs the inverse - taking a feature space and learning to upsample that feature space into something meaningful [34]. This has given rise to a particular topology of CNN called the encoder-decoder architecture, which we take advantage of to predict depth (described in detail in Chapter 4). The encoder part of a CNN is composed of a series of convolutional downsampling operations - this can be thought of as finding an efficient compression of the information stored in the image. The decoder subsequently uses this compressed form of the information to extract some meaningful information about it. One example where this architecture has been used is in [34], which outputs a classification for each pixel in the image. In this thesis, a fully convolutional encoder-decoder architecture will be used for a similar purpose, but rather than discretely classifying each pixel, we regress their depth values (more precisely, their  $z$ -coordinate).

# Chapter 3

## Literature Survey

### 3.1 Geometric Approaches to Visual Odometry

Since the invention of the pinhole camera, most cameras have captured 2D representations of a 3D world, meaning information about physical structure is lost. With the addition of a second viewpoint however we can learn a little more about the shape of the scene as governed by epipolar geometry [25]. Meanwhile, a video is a sequence of images, taken in very quick succession, and so one could think of video footage as a multi-view camera where each image is separated not only spatially but also temporally. In this work, identifying the amount of motion between two temporal frames of the video camera is referred to as ‘pose-estimation’. Doing this over several frames produces a trajectory, a process which we call ‘visual odometry’.

Due to the diversity of applications, sensors being used, and requirements, the literature on visual odometry has produced a huge variety of approaches, some which utilise features such as SURF [3], SIFT [36], or Harris corners [24], while others take featureless approaches. Some approaches use monocular imagery by studying the epipolar relationship between two views of the same scene [35], while others use Random Sample Consensus (RANSAC) to correspond a known 3D model of a scene to its 2D projection [16], while yet other approaches solve directly for pose using closed-form solutions given two sets of matched 3D points [13]. Naturally, there is a need to distinguish between these different approaches; in this literature survey, we classify approaches to pose-estimation into one of three main categories, described in detail the following section.

### 3.1.1 A Survey of Geometric Approaches to Pose-Estimation

**2D-to-2D:** Performing monocular visual odometry is an example of estimating pose using 2D-to-2D projections. In these cases, no 3D information is available and so geometry must be inferred using the epipolar constraint, which was described in Chapter 2. The 2D-to-2D approach is problematic because with a single camera there is often no way of concretely discerning the magnitude of the movement based on pixel data alone, meaning some kind of scale factor needs to be estimated based on characteristics of the image [17, 40, 52, 53]. In fact, this scale ambiguity is often exploited by film makers - what appears as a sweeping shot of a vast landscape or monument on the big screen is often modeled as a miniature film set in the studio. Because the image is monocular, there is no way to ground our measurements of scale in real world units, and so we resort to our imaginations and learned experiences to fill in the gaps. What *is* preserved in these monocular setups however is the overall structure of the scene and motion of the camera - we may not know how large the object is or how far the camera has moved, but we *can* compute the shape of the object as well as the direction of camera motion.

**3D-to-2D:** Known as the Perspective-n-Point (PnP) problem, one 3D-to-2D approach requires solving for pose using a known set of 3D points and their corresponding 2D projections in the image. Fischler & Bolles [7] in their work pioneering the PnP problem [16] applied the RANSAC algorithm to solve for camera pose using  $n = 3$  (abbreviated to P3P), producing 4 algebraic solutions for camera pose. They additionally showed that the P6P problem produces a unique algebraic solution.

**3D-to-3D:** The 3D-to-3D approach is possible when the sensor provides range data, such as in the case of stereo and RGB-D cameras. The task is simplified to finding the rigid alignment of two point clouds. This can be solved for using closed form solutions if point correspondences are known, such as Horn's method using unit-quaternions [26] or Arun *et al.*'s method using the singular value decomposition of a  $[3 \times 3]$  matrix [2]. Alternatively, iterative methods such as Iterative Closest Point (ICP) [5, 9, 51] can be used to register the point clouds if point correspondences are not known.

While prior work within each of these approaches has demonstrated impressive results, we observe two limitations of these existing methods. The first limitation applies to the first two categories, which both employ 2D imagery. With increasingly capable compute and potential data throughput, we believe that the barriers to multiple-view imaging are rapidly diminishing, representing the emergence of a promising imaging modality that can provide a more robust mechanism to tackling the pose-estimation problem. Specifically, we believe that multiple-view imaging can and should be used when dealing with problems relating to geometry in computer vision. The second limitation

applies to the 3D-to-3D visual pose-estimation approach, where, to the best of our knowledge, all methods rely on a calibrated camera array, stereo pair or RGB-D camera. While these types of sensors are typically excellent at measuring 3D geometry at short ranges, even small changes to their calibration parameters are a punishing blow to their accuracy. The remainder of this section will summarise some of the key learnings in pose-estimation presented in the works mentioned above.

### 3.1.2 Iterative Methods

Iterative methods are typically formulated as non-linear least-squares minimisation problems, which can be solved by linearising the problem and iteratively stepping to a minimum. Local minima represent a challenge in these minimisation problems as the global minimum may be overlooked. In overcoming this, we typically make the reasonable assumption that a standard video camera operating with real-time constraints on a mobile robotic platform generally produces frames several times per second, meaning inter-frame motion is relatively small. Iterative methods may therefore initialise their parameters to initially estimate zero movement, meaning there is a high likelihood that the iterative solution converges to the global minimum. Alternatively, a closed-form solution may be used to initialise the estimate, followed by iterative refinement.

Another iterative approach pioneered by Fischler & Bolles [16] uses random sampling to iteratively fit a model to observed data-points through a guess-and-check procedure which is robust to outliers and noisy data.

*Reprojection error* is an important concept in both iterative and closed-form methods of pose-estimation. It describes the photometric error when a 3D point (obtained either through estimation or range sensing) is projected onto the sensor plane of a camera. The distance between the actual observed point and the projected point is the photometric error. As a motivating example, consider an RGB-D camera which has provided a range measurement for each pixel, allowing the reconstruction of a point-cloud representation of the scene. After the camera has moved a small amount, we are interested in finding the translation  ${}^2t_1$  and rotation  ${}^2R_1$  from the first frame to the second. A point  $P$  from the point-cloud should be seen by the camera in the second frame at the coordinate  $K({}^2R_1P + {}^2t_1)$ . Comparing this with the actual observed coordinate of that point, we compute the reprojection error. More generally, given the homogenous transformation from the first pose to the second  $[R|t]$ , for every 3D point observed in the first frame,  $P_i$ , and their (actual) observed 2D projections in the second frame,  $p_i$ , the reprojection error is computed as

$$\sum_{i=0}^n ||p_i - K[R|t]P_i||^2.$$

Hartley & Zisserman [25] formulate a pose-estimation pipeline that uses the *8-point-algorithm* to estimate an initial pose, from which point correspondences are triangulated to form a 3D representation of the scene. This 3D representation is used to iteratively refine both the 3D reconstruction and the pose estimate by minimising the reprojection error.

### 3.1.3 Closed-form Methods

Treating the pose-estimation problem with a closed-form solution has many advantages over iterative methods. In particular, closed-form solutions attempt to estimate pose without any iterative refinement by solving directly for the global minimum. Furthermore, the computation-time required by closed-form solutions is independent of the scenery itself, running in constant-time - an attractive characteristic in real-time systems, where consistent run-time can greatly simplify system design.

In Section 2.1.2 we studied the relationship between multiple cameras, where we described the Fundamental Matrix as the rotation and translation between two cameras, up to scale. Closely related is the Essential Matrix, which similarly describes this scale-ambiguous relationship between two cameras, with the key difference being that while the Fundamental Matrix is defined in the pixel-space of the camera, the Essential Matrix is defined instead in terms of the normalized coordinate frame. If the intrinsic parameters for the camera is known, it is a straightforward conversion between the two representations. The essential matrix can be decomposed to find the rotation and translation of the two cameras up to scale, yielding four possible solutions. The interested reader is directed to [25] for the full derivation.

*Plenoptic Flow* is one technique suggested by Dansereau *et al.* [13] that takes a modular approach to visual odometry, whilst exploiting the richness of the signals in the full light-field extended into the time-domain. The modules include a *depth estimation* component, a 4D analogue to the traditionally 2D *optical flow* problem, and a closed-form point-cloud registration algorithm [26]. The first step exploits the *gradient-depth* constraint, fixing each pixel's depth in the scene and allowing the reconstruction of a point-cloud from the first frame of video. Now, a naive approach is to subsequently perform the same operation using a second frame of video, resulting in two point-clouds that can be registered to compute the relative pose. Unfortunately, this approach produces two independent point-clouds lacking known point-to-point correspondences, meaning an iterative

algorithm is required to find the rigid alignment between them. Given the approach’s emphasis on closed-form solutions, an alternative idea is proposed that exploits the time-domain behaviour of light-fields. The *plenoptic flow* algorithm estimates a 3D velocity for each point from light field derivatives, producing known point-to-point correspondences, which allows registration using Horn’s method for absolute orientation [26].

## 3.2 Data-driven Approaches to Visual Odometry

A relatively new approach that has driven a large body of research is the use machine learning and statistical modeling to perform pose-estimation, utilising the image-processing power of convolutional neural networks to learn a non-linear mapping directly from a pair of images to their relative pose [15, 18, 20, 33, 54]. Combining the spatial awareness of the convolutional down sampling operation with a neural network’s ability to learn accurate approximations for complex, non linear functions, these approaches have found success in both supervised [33, 15], and unsupervised configurations [18, 20, 54].

We first establish a taxonomy of deep-learning methods that have emerged for approaching the pose-estimation problem in Section 3.2.1. In Section 3.2.2 a more extensive discussion of existing methods is provided, with emphasis on the methods the most closely align with our proposed approach, and the limitations of current methods which have guided this work’s hypotheses.

### 3.2.1 A Taxonomy of Data-Driven Methods for Pose-Estimation

All data-driven methods covered in this section relate specifically to *deep-learning* approaches for pose-estimation. In image-processing algorithms of late, deep-learning almost invariably includes the use of a convolutional neural network, due to the powerful regularising characteristics of the convolutional down-sampling operation and their hierarchical feature-extraction capabilities.

Similarly to the geometric approaches described in Section 3.1, the very closely related task of depth-estimation and 3D scene reconstruction is an important step in many of the methods described here. Thus, this section will provide an overview of methods including both depth-estimation and pose-estimation.

**Datasets:** Data-driven algorithms require data, and their surge has been helpfully facilitated by the availability of massive online datasets such as KITTI [19], NYU-v2 [39], and CityScapes [10]. Such datasets are typically accompanied with suites of validation data collected from a variety of

sensor sources, such as RGB-D cameras, stereo camera pairs, lidar, inertial sensors and GPS. The availability of such datasets has been crucial to the training and validation of many of the works described in this section.

**Supervised Methods for Depth:** In the supervised family of algorithms, Eigen et al. [15] and Liu et al. [33] take advantage of the ground truth depth maps available in the KITTI dataset to learn depth monocularly; this is an enigmatic challenge for non-learning-based methods as there is insufficient information in monocular imagery to infer depth.

**Supervised Methods for Pose-Estimation:** On the problem of pose-estimation and visual odometry, both Wang *et al.* [47] and Jiao *et al.* [28] combine the image-processing capabilities of CNNs with the time-domain strengths of Recurrent Neural Networks (RNNs) to estimate 6-degree-of-freedom poses over short sequences of video. In these approaches, a Long-Short-Term-Memory network (LSTM) is used to output temporally cohesive pose estimates, recognising that the inertial stability of a moving vehicle means its velocity and direction is unlikely to quickly change direction over the course of a few frames of video. These approaches have demonstrated strong results in visual odometry, however they require ground-truth pose data to supervise their networks.

**Unsupervised Methods:** Unsupervised methods have arisen from the recognition of two main facts. Firstly, that ground truth data is typically expensive, and time-consuming to collect. Secondly, that a visual-perception module that is capable of improving performance in-situ without direct supervision, is an attractive capability in robotics. Unsupervised approaches [18, 54] have taken advantage of the epipolar constraints imposed on moving cameras, cleverly piecing together information to simultaneously estimate both a 3D scene reconstruction, and the motion of the camera. Many of these unsupervised methods suffer from a scale ambiguity due to insufficient information, which we have discussed in Section 3.1.1. Furthermore, some methods may even suffer from scale *inconsistency*, a more concerning issue where there are no constraints enforcing pose is estimated consistently at the same scale. For example, the pose estimation module may estimate a motion of only a few centimeters in one pair of frames, while the next pair produces an estimate of a few meters, despite the actual inter-frame motion being very similar. A more extensive overview of these types of methods is provided in Section 3.2.2, with some discussion around how the scale-ambiguity has been resolved.

**Semi- and Weakly-Supervised Methods:** A more recent paradigm has been to treat the problem as one that can be approached using *some*, but not *all* of the ground-truth information that is available. In general, the same approach is used as unsupervised methods, simultaneously

estimating depth and pose, with the addition of certain supervision signals. The thinking behind such approaches is that while sometimes equipment such as lidar might be too expensive or infeasible to attach to a mobile robotic platform, certain parameters about the equipment is known ahead-of-time. For example, Godard *et al.* [20] and Zhan *et al.* [50] both propose using the known distance between a stereo-pair of cameras to enforce scale-consistency and scale-awareness in their predictions.

### 3.2.2 Simultaneous Unsupervised Depth and Pose Estimation

This section closely follows the derivation of simultaneous depth and pose estimation proposed by Zhou *et al.* [54], although many works have adopted the same pipeline. The general pipeline utilises two CNN's, one who's purpose is to predict dense per-pixel depth maps given a single image, and the other who's job is to estimate the 6 degree-of-freedom pose from the first frame to the second given both images.

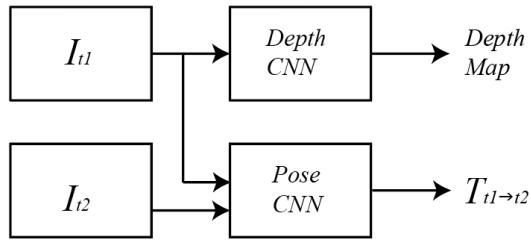


Figure 3.1: Machine learning approaches have demonstrated strong results in simultaneously estimating depth maps and relative poses between images. Typically, a pair of CNN's is used, one for each task. The depth CNN is provided with the first view at  $t = 1$  and the pose CNN is provided both views.

While the training pipeline is unsupervised in the sense that labelled data is not required, some form of supervision signal is nevertheless required to optimise the parameters of the pair of networks. These papers take advantage of the fact that if the physical structure of a scene is known, then a novel rendering of that scene from a different viewpoint is achievable. Thus, [18] suggested using photometric reconstruction as a supervision signal - from  $I_{t2}$ , reconstruct  $I_{t1}$ . The difference between the reconstruction and the real image can form a supervisory signal for the pair of networks. This work similarly uses photometric reconstruction as the principal supervisory signal in training a pair

of networks to perform visual odometry and depth estimation. The photometric reconstruction loss is

$$L_{photometric} = \sum_n^W \sum_m^H |I_{t1}(m, n) - \hat{I}_{t1}(m, n)|. \quad (3.1)$$

Humans can easily imagine what a scene will look like if we move our heads slightly, but only if we know the overall shape of the object we are looking at. Similarly, given two images separated by a small interval of time  $I_{t1}$  and  $I_{t2}$ , one could synthesise what  $I_{t1}$  would look like by sampling the pixels from  $I_{t2}$ , if the relative pose  $T_{t1 \rightarrow t2}$  between the two cameras and a pixel-wise depth of the scene  $D_{t1}(p)$  is known. In practice, this can be done by obtaining the coordinates of the pixels in the first image  $p_{t1}$  projected on to  $I_{t2}$ 's camera sensor. Assuming a pinhole model, the complete expression for  $p_{t1}$ 's projected location on the second camera,  $\hat{p}_{t2}$  is

$$\hat{p}_{t2} = K T_{t1 \rightarrow t2} D(p_{t1}) K^{-1} p_{t1}. \quad (3.2)$$

Here  $K$  represents the intrinsics matrix. In right-to-left order, this transform first maps each pixel to a ray direction using the inverse camera intrinsics  $K^{-1}$ . Each ray is then assigned a z-coordinate using the per-pixel depth map  $D(p_{t1})$ , producing a 3D point cloud. The transform  $T_{t1 \rightarrow t2}$  transforms each point to the coordinate frame of the second camera, then the matrix  $K$  projects each of those points onto the camera sensor of the second camera. The obtained coordinate  $\hat{p}_{t2}$  is continuous, while pixel coordinates are discrete. A pixel value thus needs to be interpolated, keeping in mind that each step of a neural network pipeline must be differentiable to support the backpropagation of gradients.

It was suggested by Zhou et al. [54] that adopting bilinear sampling could be used as a fully differentiable sampling mechanism. The use of bilinear interpolation as a differentiable sampling pipeline was first proposed Jaderberg et al. [27], and adapted by Zhou et al. [54] to perform the differentiable image warp. A bilinear sampling kernel is described by

$$V = \sum_n^W \sum_m^H U_{nm} \max(0, 1 - |x - n|) \max(0, 1 - |y - m|). \quad (3.3)$$

$V$  is the output of the sampling kernel,  $U$  is the source image being sampled,  $n$  and  $m$  index over the columns and rows of the kernel respectively,  $H$  and  $W$  are the height and width of the sampling kernel, and  $x$  and  $y$  are the local coordinates of the sampling location. The  $2 \times 2$  sampling kernel used to perform the photometric reconstruction is, in essence a weighted sum of the 4 nearest

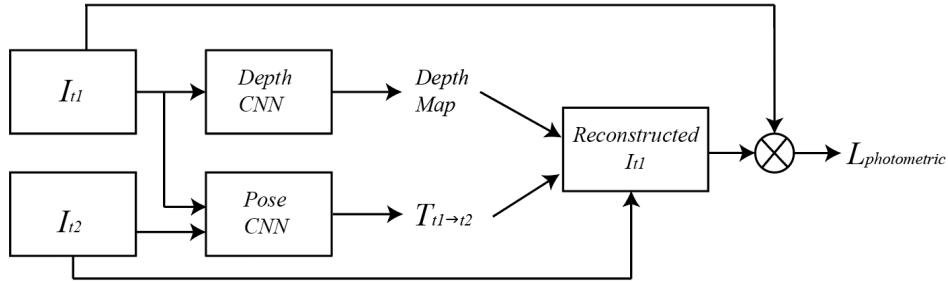


Figure 3.2: To provide a supervisory signal to the depth and pose CNN’s, Garg et al. [18] suggested using photometric reconstruction. The loss function is formulated by taking the difference between the reconstructed image  $\hat{I}_{t1}$  and the actual image  $I_{t1}$ , shown in equation 3.1.

neighbour pixels, based on its proximity to those pixels. The equation is differentiable with respect to both  $x$  and  $y$ :

$$\frac{\partial V}{\partial x} = \sum_n^W \sum_m^H U_{nm} \max(0, 1 - |y - m|). \quad (3.4)$$

$$\frac{\partial V}{\partial y} = \sum_n^W \sum_m^H U_{nm} \max(0, 1 - |x - n|). \quad (3.5)$$

Using this bilinear sampling kernel, we can thus sample pixels from  $I_{t2}$  to reconstruct  $I_{t1}$  in a fully differentiable manner, allowing the backpropagation of gradients through the networks. The pipeline with photometric reconstruction as the supervision signal can be illustrated as shown in Figure 3.2. Bilinear interpolation is similarly employed in this work to compute the photometric reconstruction loss of the estimated depth and pose. However, because this work operates on light fields, the output of photometric reconstruction won’t be a single image, but a whole array of images.

In addition to the photometric reconstruction loss providing the main supervision signal to the network, Zhou et al. [54] employs a smoothness loss to ensure that the produced depth map is globally smooth. The smoothness penalises the second order gradient of the image - i.e. the depth network is encouraged to produce a depth map characterised mostly by low-frequency components, and penalised for high-frequency components. Recognising that depth discontinuities frequently occur in parts of the image where a strong edge appears, Godard et al. [20] on the other hand suggested the use of an edge-aware smoothness loss that also penalises large gradients in the depth map  $\partial d$ , but lowers the weight of the loss in regions where the image gradient  $\partial I$  is large.

$$L_{smooth} = \sum_n^W \sum_m^H |\partial_x d_{nm} e^{-|\delta_x I_{nm}|} + \partial_y d_{nm} e^{-|\delta_y I_{nm}|}| \quad (3.6)$$

There have been several iterations [6, 20, 21, 50] of this pipeline, most of which have focussed on introducing novel loss functions to improve the quality of the depth and pose estimates. To the best of our knowledge however, this pipeline has only been applied to monocular and stereo camera imagery. While monocular and stereo cameras are ubiquitous in modern digital devices, this thesis drives the investigation further, studying the potential for the use of similar pipelines to achieve improved results using a broader, more generalised family of imaging devices.

Breaking free of the pinhole principle from which human eyes have developed and which commercial cameras have adopted, has wide implications in the field of machine vision. Reinforcing this idea is the rise in popularity of imaging techniques such as multi-spectral imaging, plenoptic cameras and polydioptric cameras, however interpreting ray directions and geometry for new cameras may not always be apparent, giving rise to the data-driven approach. Working on the principle that machine vision does not necessitate mimicking the human eye to capture imagery, this thesis investigates the capabilities of machine learning for querying depth and pose using one such novel imaging device - a camera array.

### 3.3 Convolutional Neural Networks and Light Fields

Crafting a data-driven pipeline for visual odometry and depth requires a neural architecture that is capable of ingesting light field data. Importantly, the shape of the data must be considered. Our parameterisation of the light field has four dimensions, denoted by  $S$ ,  $U$ ,  $T$ ,  $V$ . Convolutional neural networks normally operate on 2D image data, meaning the search for a suitable neural architecture is an important step in testing this work's hypotheses (convolutions in 3 and 4 dimensions are also possible, however they incur significant computational overhead, and for reasons that will become apparent in the next chapter, are difficult to apply to our specific image data).

Wang et al. [48] exploit the rich textural information contained in the light field, applying a convolutional network to the task of material classification using light field imagery. In addition to reporting significantly better accuracy compared to 2D imagery (with gains of over 7%), [48] suggests 5 novel methods for interpreting light field data for convolutional networks. One of Wang's most promising architectures, which is referred to in their paper as using interleaved 'spatial' and 'angular' kernels, exploits the fact that presenting light field data in a different order exposes new

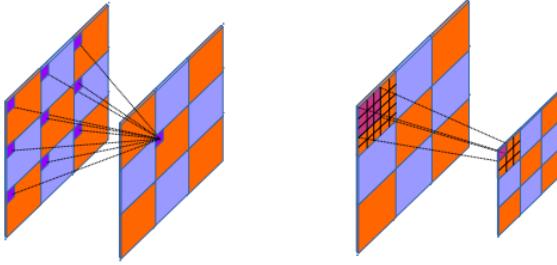


Figure 3.3: Angular Filters (left): by fixing the coordinate  $(u, v)$  and varying  $(s, t)$ , the 2D grid of images can be presented to a neural network in the form of orthographic images. Spatial Filters (right): instead of fixing  $(u, v)$  coordinates,  $(s, t)$  is fixed while  $(u, v)$  is varied. This presents information to the neural network as a grid of perspective images.

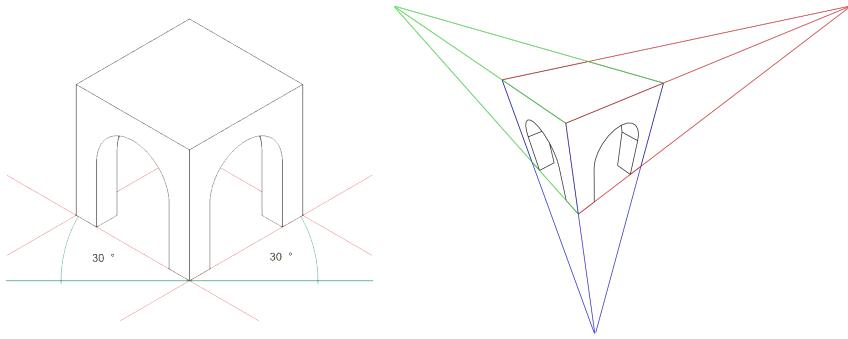


Figure 3.4: Orthographic projection (left) [4]: the lines of sight to all points in the scene are parallel to one other, and perpendicular to the projective plane. Perspective projection (right) [23]: lines of sight converge to a point giving the effect that distant objects appear smaller than nearby objects.

kinds of information.

As discussed in [12], a 4D light field image may be tiled as a 2D grid of 2D projective images (called  $(u, v)$  slices). Alternatively, the light field may be tiled as a 2D grid of 2D *orthographic* images. By presenting the image data to their network in the two different forms, Wang et al. approximates convolution in 4 dimensions, allowing their network to learn a 4D signal processing function without the computational overhead of convolving in 4 dimensions.

There are some hurdles in this approach that make it challenging to apply in our depth and visual odometry pipeline. Most importantly, arranging the light field as orthographic images is impossible with our specific camera without introducing significant redundant data. The challenge arises from the shape of the camera array. As shown in Figure 2.4, our camera has sub-apertures arranged in

a cross-hair formation, meaning the complete 2D grid of 2D images is unavailable to us. In the following chapter, we will describe a novel technique that, similar to Wang et al. approximates convolution in 4 dimensions by presenting light field data in a different order to the ordinary  $(u, v)$  slices that we are familiar with.

Wang et al. suggests another suitable architecture for application in this work, which involves stacking the images along their colour-channels, feeding the data to a convolutional network as a cuboidal volume. This approach is a simple and intuitive way of arranging the light field data, however neural networks may not be able to take full advantage of the 4D signals in light field data this way. Nevertheless, it is a useful way of representing the light field, especially as a format which is easily prepared for ingestion by CNNs, and is thus also explored as an input format in this work's experiments.

# Chapter 4

## Methodology

In Chapter 3, we saw that previous work has produced a variety of approaches to depth estimation and visual odometry, using both data-driven and hand crafted solutions. In this chapter, we propose a novel pipeline that behaves similarly to the pipelines described in Chapter 3, with the key distinction of operating on 4D, as opposed to 2D data. The contributions of previous work focus on the use of monocular and stereo image data, while this chapter focuses on developing a pipeline that employs the full breadth of geometric information in the light field. The first section of this chapter describes the tools and methodologies of acquiring a suitable dataset, while the second part derives the pipeline used for our experiments.

### 4.1 Data Acquisition

#### 4.1.1 Ground Truth Pose Data

Important to this work is a strong evaluation framework. Existing datasets such as KITTI and CityScapes benefit from state-of-the-art sensor suites including inertial sensors and lidar, allowing researchers to effectively benchmark their results. Similarly, this thesis places a strong emphasis on validation using ground truth data. One of the primary tasks for evaluation is visual odometry, and so ground-truth pose data for each image is a valuable resource. In this project, pose is collected by attaching the camera to a Universal Robots UR5E robotic manipulator, which is capable of sensing to a high degree of accuracy the pose of the end effector. Furthermore, the UR5E is able to perform accurate movements and trajectories with payloads of up to 5 kilograms, making it a suitable robotic

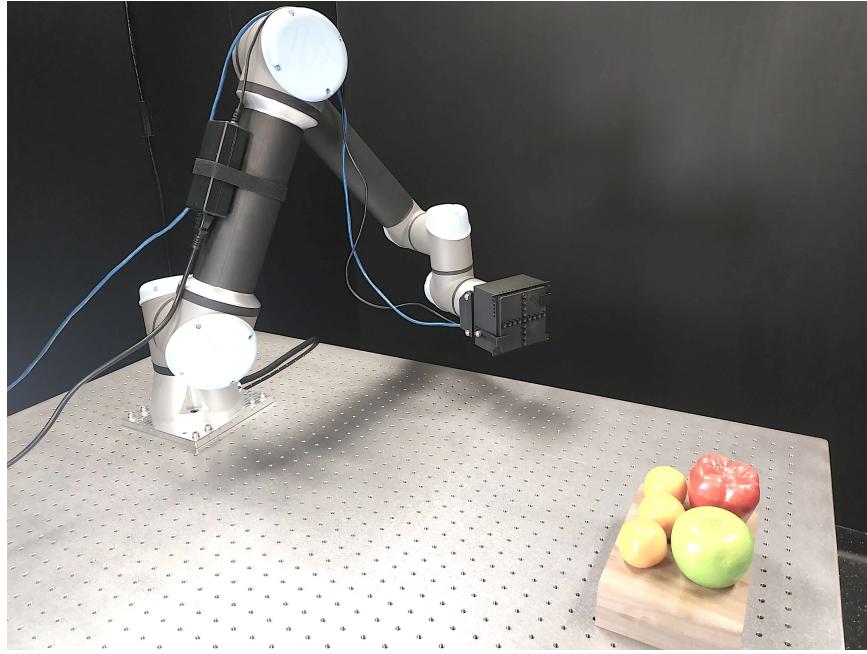


Figure 4.1: The experimental setup utilises a Universal Robots UR5E robotic arm to precisely measure ground-truth pose to allow effective evaluation of the projects visual odometry results.

mount for the camera.

### Manipulator-Camera Interface Adapter

The UR5E uses a standard interface plate for attaching end-effectors such as gripping tools. The camera on the other hand, despite its excellent construction, is an early prototype provided by EPI-Imaging, without a physical adapter interface for the UR5E manipulator. We therefore fabricate our own mount for the device, taking into consideration the camera's ventilation and cabling requirements. A two-piece adapter was designed in SolidWorks and fabricated using a fused deposition modelling (FDM) platform with polylactic acid (PLA) filament.

### Manipulator Control

A client library for communicating with the UR5E was developed in python, establishing a TCP/IP connection with the robot over the local network, allowing both movement commands to be sent to the robot, and feedback data about the end-effector's pose to be streamed back to the client PC. The client library also provides functionality for computing trajectories and corresponding joint

angles, using the PyBullet [11] physics engine and the known kinematic model. One useful trajectory function from our library procedurally generates waypoints while keeping the camera’s field-of-view trained on the scene, using a stringent collision-checking mechanism to enable autonomous data-capture.

### 4.1.2 Imagery

The specific imaging device being used is manufactured by EPIImaging, and consists of 17 subapertures. The communication interface with the camera is a network connection serving image data using the HTTP protocol via a REST API. Both rectified and un-rectified imagery can be retrieved, and the format of the returned data is a  $17 \times 3 \times 1280 \times 960$  stream of raw bytes representing 8-bit pixel values from each of the 17 sensors. A simple client library was developed in python to automate API requests, and perform decoding of the pixel data.

## 4.2 Depth and Visual Odometry Pipelines

In this section, we derive a pipeline for simultaneous depth and visual odometry using light field data. Our pipeline is constructed modularly, so to build a precise intuition for how our pipeline operates, we begin by describing the functions of each individual module. We first introduce the CNN modules used to estimate pose and depth. Subsequently we'll discuss strategies for forming meaningful loss-functions to train our pair of networks using a photometric-warp loss-module. Next, we launch a discussion on the input formats, altering how light field data is fed to our networks, discussing the comparable advantages and disadvantages of each. Finally, we introduce some modifications to the photometric-warp loss-module that sacrifices some flexibility in exchange for more robust depth and pose estimates that more-fully exploit information in the light field.

### 4.2.1 Pose Estimation

We treat pose estimation as a regression problem, estimating three translational components [X, Y, Z] and three rotational components [Rx, Ry, Rz]. We perform this regressions with a convolutional network, which we call ‘PoseNet’, in line with previous works. This network architecture is fully convolutional, meaning that in principle it produces the correct number of outputs for any 2D input.

Table 4.1: PoseNet Convolutional Network Architecture

Layer	Kernel Size	Output Channels	Stride	Padding	Activation
Conv1	7	16	2	3	ReLU
Conv2	5	32	2	2	ReLU
Conv3	3	64	2	1	ReLU
Conv4	3	128	2	1	ReLU
Conv5	3	256	2	1	ReLU
Conv6	3	256	2	1	ReLU
Conv7	3	256	2	1	ReLU
Output	1	1	1	1	-

Importantly, the translation and rotation are regressed relative to the *camera* coordinate frame of the first image, whereas the UR5E returns absolute position and orientation in the *world* coordinate frame. To validate our predictions we therefore need to compute the relative translation  $P_{t1 \rightarrow t2}$  and

rotation  $R_{t1 \rightarrow t2}$  of the two images, given their absolute positions  $P_{t1}, P_{t2}$  and orientation  $R_{t1}, R_{t2}$  in the world frame. Concatenating translation and rotation into a single transform matrix  $T = [R|t]$  allows us compute both relative rotation and translation in a single step, as:

$$T_{t1 \rightarrow 2} = T_{t1}^{-1} T_{t2}. \quad (4.1)$$

Intuitively, this computation first computes the transform from pose 1 to the world frame, then from the world frame to pose 2. We can decompose this matrix into 3 translational and 3 rotational components.

### 4.2.2 Depth Estimation

Depth estimation is similarly modeled as a regression problem, with the same number of outputs as inputs. The depth estimation module uses the fully-convolutional encoder-decoder ‘DispNet’ architecture [37]. Skip connections from the encoder to the decoder mean the decoding layers are able to access both high level features and low level features on either side of the latent space. Prior work [54] has demonstrated that predicting depth maps at multiple scales improves photometric reconstruction error by enforcing some level of local smoothness. In practice, this helps the network learn to predict depth in challenging, textureless regions of the image.

### 4.2.3 Photometric Warp Loss

We have now described one CNN tasked with estimating a per-pixel depth map of the scene, and another which estimates the 6-degree-of-freedom pose between two frames of video. The challenge is now to provide a supervision signal that sufficiently constrains these networks to actually perform the tasks of depth and pose estimation.

#### A Differentiable Loss Function with Image Based Rendering

As described in Chapter 3, given an estimate for these two pieces of information, we are able to synthesise approximate images of the scene as seen from a nearby viewpoint. We take advantage of this fact to form a photometric-warp loss, similar to [54].

As described in Chapter 3, the procedure for this is to first project each pixel from the first light field  $LF_1$  to a 3D point cloud using the known camera intrinsics and estimated depth values. If we

Table 4.2: DispNet Convolutional Network Architecture.

	Layer	Kernel Size	Concatenate	Output Channels	Stride	Padding	Activation
Encoder	Conv1	7	-	32	2	3	ReLU
	Conv2	5	-	64	2	2	ReLU
	Conv3	3	-	128	2	1	ReLU
	Conv4	3	-	256	2	1	ReLU
	Conv5	3	-	512	2	1	ReLU
	Conv6	3	-	512	2	1	ReLU
	Conv7	3	-	512	2	1	ReLU
Decoder	UpConv7	3	Conv6	512	2	1	ReLU
	UpConv6	3	Conv5	512	2	1	ReLU
	UpConv5	3	Conv4	256	2	1	ReLU
	UpConv4	3	Conv3	128	2	1	ReLU
	UpConv3	3	Conv2	64	2	1	ReLU
	UpConv2	3	Conv1	32	2	1	ReLU
	Output2	3	-	1	1	1	Sigmoid
	UpConv1	3	-	32	2	1	ReLU
	Output1	3	-	1	1	1	Sigmoid

Layers prefixed with ‘Conv’ or ‘Output’ are convolutional layers, and those prefixed with ‘UpConv’ use the transposed-2D-convolution to upsample the preceding layer.

have regressed a depth map  $D(s_0, t_0, u, v)$  for the sub-aperture  $s_0, t_0$ , we can for each pixel  $[u, v]^T$  project its position to a 3D point cloud  $Q$ .

$$Q = D(s_0, t_0, u, v) K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad \forall u, v. \quad (4.2)$$

The 3D coordinate  $Q$  is then projected onto the sensor plane of  $LF_2$ , at the pixel coordinate  $[s_0, t_0, \hat{u}, \hat{v}]$ . This relies on our estimate of the relative pose  $[R|t]$  from the first to the second camera origin.

$$[s_0, t_0, \hat{u}, \hat{v}] = K[R|t]Q \quad \forall Q \quad (4.3)$$

The pixels  $[s_0, t_0, \hat{u}, \hat{v}]$  are subsequently sampled to populate a new array, forming our synthesised image. This transform is equivalent to constructing a 3D model of the scene using our depth values, moving an imaginary camera by  $[R|t]$  and taking a virtual snapshot of the scene from this new angle, the snapshot being the resulting photometric warp.

More broadly, the practice we have described here is referred to as ‘image based rendering’, which is a related, but slightly different paradigm from the rasterization, shaders and textures typically used in graphics-based rendering engines. In image based rendering, we instead synthesise viewpoints using existing image data. In a sense, image based rendering simply shuffles the pixels of an existing image around, according to some pre-defined rules. In our case, we derive the shuffling-rules using 3D data inferred by our depth and pose networks. The perceptive reader may have noticed that the pixel value  $p_2 = [s_0, t_0, \hat{u}, \hat{v}]$  is continuous in  $u$  and  $v$ , while pixel coordinates should be discretely valued. A naive solution is to discretise the coordinate  $p_2$  by sampling its nearest-neighbouring pixel. We must remember however that neural networks rely on the chain rule of calculus to perform back-propagation, and so whatever sampling regime we choose must be differentiable. We therefore turn to bilinear sampling, as described in Chapter 3 as a differentiable sampling mechanism. Its operation and derivatives are described in equations 3.3-3.5.

#### 4.2.4 Input Methods

This work tackles an interesting and unique challenge, of how to feed sparse light field data to convolutional neural networks. We summarise this challenge as:

*Given sparse 4 dimensional data, how best can we arrange that data as 2 dimensional slices in a valuable and informative way to a convolutional network for the purposes of visual odometry and depth estimation.*

As we have seen in Chapter 3, this problem has been approached in prior work, usually with the assumption that a complete 2D grid of 2D images is available. In this work however, the specific camera configuration prohibits arranging our data as a 2D grid of images without introducing significant redundant data. In a way, this forces us to consider more flexible and generalisable methods for processing light fields into constructive forms of 2 dimensional data. In this section, we suggest three experimental methods for ingesting light field data through neural networks. In the next chapter, we will present visual odometry and depth results using each of these methods.

### Volumetric Images

The most straightforward of our three methods, this technique involves taking  $U, V$  slices (perspective-projection images) of the light field, and stacking them along a 3rd axis (usually reserved for the colour-channels). The resulting image is an  $H \times W \times N$  volume, where  $H$  and  $W$  are the height and width of the  $U, V$  slice, and  $N$  is the number of sub-apertures being sampled. This method is suggested, and tested by Wang et al. [48] for material classification, who report improved results over conventional 2D imagery. Internally, we expect a convolutional network to learn features relating to parallax, occlusion and depth when exposed to data of this format.

### Focal Stacks

Focal stacks are constructed as a superposition of 2D images from several sub-apertures, creating interference in regions of the image that are ‘out-of-focus’, while keeping ‘in-focus’ elements of the image crisp. A focal stack specifically encodes depth in the form of interference at each region of the image. Thus, we might expect a CNN trained on this type of imagery to learn to decode in- and out-of-focus parts of the image, using this information to estimate depth and pose. Importantly, we must consider that our spatial resolution (number of sub-apertures) is significantly lower than our angular resolution (number of pixels-per-sensor). As a result our focal stacks will exhibit significant aliasing effects which may impact training.

### Tiled Epipolar Plane Images

In Chapter 3 we saw that Wang et al. [48] sliced the 4D light field in two different ways, first as perspective-projection images, and secondly as orthographic-projection images. By showing slices in  $S, T$  and  $U, V$ , Wang et al. suggests that a convolutional network is able to learn to approximate 4D signal processing functions without the computational overhead of convolving in 4 dimensions. Due to the specific constraints of the camera array being used (and indeed of any camera array not arranged as a regular grid of sub-apertures), there are challenges associated with slicing our light field data as orthographic images. We suggest that there is indeed another meaningful way of slicing the 4D light field into 2D arrays - namely as epipolar plane images. As we discussed in Chapter 2, epipolar plane images are  $S, U$  or  $T, V$  slices of the light field, encoding depth and occlusion in the slope of their characteristic sheared lines. With our input strategy, we tile  $T, V$  slices side-by-side, and  $S, U$  slices top-to-bottom.

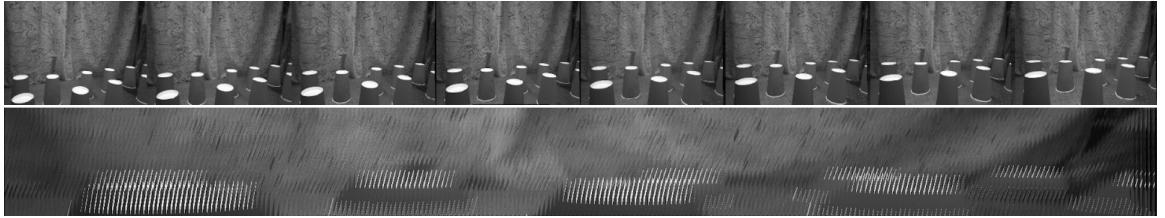


Figure 4.2: Tiled  $U, V$  slices (top): this is an intuitive way of slicing the light field, which tiles images from each sub-aperture side-by-side. Tiled  $T, V$  slices (bottom): somewhat less intuitively, this method instead tiles the vertical epipolar-plane images side-by-side. We call these *wide* EPIs. As shown, the overall dimensionality and shape of the light field is unchanged, we simply present the information in a different order, revealing different encodings of the captured scene. We can also tile the light field as  $S, U$  slices, stacking them vertically, which we refer to as *tall* EPIs.

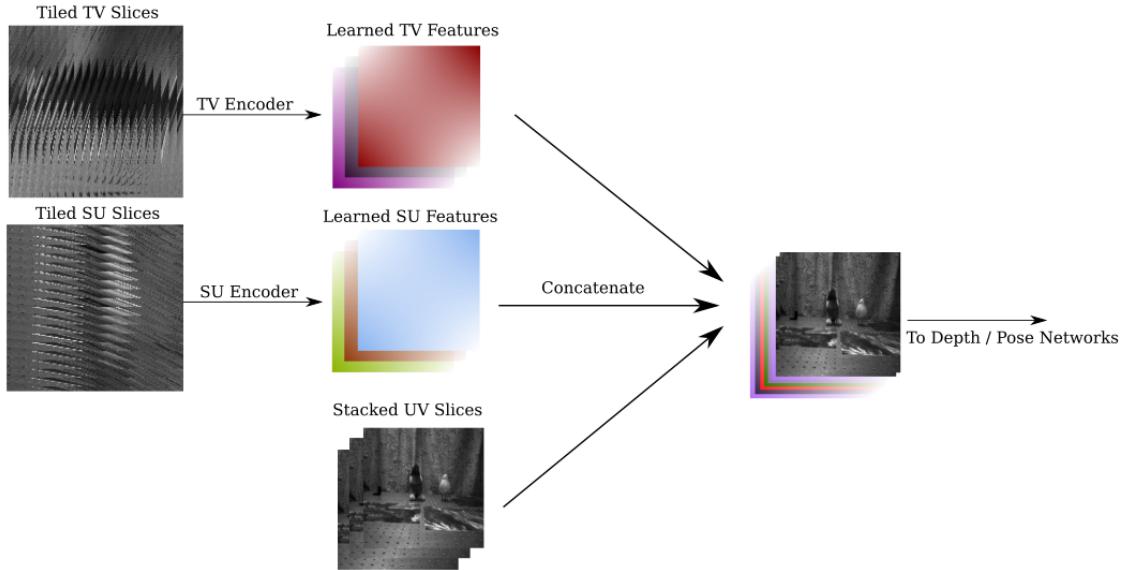


Figure 4.3: Our proposed encoder module which encodes the tiled EPIs, downsampling them to the shape expected by our depth and pose networks. Notably, features which have been learned in  $T, V$  as well as  $S, U$  are concatenated with the stacked  $U, V$  slices (exactly as seen in our first input-method.) For brevity, we have only shown a small patch of the tall and wide EPIs, but the reader is reminded that these tilings are in fact very tall, and very wide.

One challenge we face in ingesting this data, is that both previously described methods have input shapes  $H \times W$ . The tiled images from Figure 2.4 on the other hand have shape  $H \times (W \times N)$  where  $N$  is the number of sampled sub-apertures. In practice, these images are much wider than either of our previous input formats. We therefore suggest the use of an additional convolutional encoding module that down-samples the tiled EPIs to the shape expected by our depth and pose networks.

The encoding module consists of one convolutional layer followed by a ReLU activation layer. For the *wide* EPIs we convolve with a kernel of size  $N \times N$ , and a horizontal stride of  $N$ . Zero-padding is applied to the top and bottom of the tiled EPI. The effect of this convolution is that the wide EPI is down-sampled with the same output size as a single  $U, V$  slice. Similarly, we apply an  $N \times N$  filter with a vertical stride of  $N$  to the tall EPIs, also reducing their shape to that of a  $U, V$  slice. The resulting feature-maps are combined with the  $U, V$  slices, and provided to our depth and pose networks. Thus, our networks are able to learn features in three different feature-spaces. We hypothesise that with this ingestion strategy, our networks are able to learn an approximation for 4D signal processing functions, without the computational overhead of convolving in 4 dimensions, and without the requirement for a complete 2D grid of images.

### 4.3 Complete Light Field Reconstruction

In this section, we modify our pipeline to more-fully take advantage of the information in the light field. One important weakness of the pipeline we have described so far, is that there are no constraints on our networks to estimate scale with the same magnitude between video frames. Given three frames of video of the same scene, and stable camera motion, the depth network will happily estimate a room that is twice as large in the second pair of frames than the first, if of course the pose network also estimates a twice-as-large translation. This is the scale ambiguity that we have discussed in Chapter 2. With the modifications suggested in this section, we take advantage of the fact that we have multiple sub-apertures capturing the scene at the same time, using this knowledge to enforce scale consistency across video frames.

The modification described here is straightforward, but it sacrifices some flexibility. The pipeline as we have described it so far requires only one vital piece of information to learn depth and visual odometry - the focal length of the camera being used (assuming all sub-apertures have very similar focal lengths). With this modification, we are required to at least know the arrangement of the

sub-apertures in relation to one-another (although the absolute scale of their layout is not required). In our case, we know that our sub-apertures are arranged as a cross-hair.

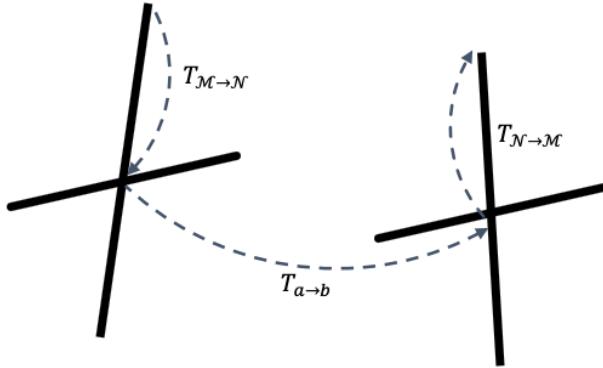


Figure 4.4: Given knowledge of how one sub-aperture has moved through space, we can also compute how any other sub-aperture has moved by chaining the transforms in the order shown.

Instead of using photometric warp to reconstruct the image of a single sub-aperture, this modification adjusts the pipeline such that instead, we now reconstruct the entire light field. The camera array is a rigid body, meaning that as it rotates and translates through space, the sub-apertures also rotate and translate relative to each other, but in a very well defined way. Say we know the homogenous transform  $T_{a \rightarrow b}$  of sub-aperture  $\mathcal{N}$  between two frames of video, we can compute the global transform of any other sub-aperture  $\mathcal{M}$  by chaining together the homogenous transforms as shown in Figure 4.4.

To exploit this knowledge, we modify the depth prediction network to now output  $N$  depth maps. Each depth map should be aligned with exactly one sub-aperture, so we can reconstruct its corresponding image. The pose network continues to estimate only the pose of the central sub-aperture, but the frame-to-frame pose for each pinhole is computed from the chained transform shown in Figure 4.4 (where  $T_{a \rightarrow b}$  is the PoseNet output). For each sub-aperture we perform the same photometric warp with bilinear interpolation to synthesise  $N$  viewpoints.

In addition to imposing a stronger supervision signal, this pipeline guarantees to a certain extent that our pose and depth modules predict depth at a consistent scale between video frames. We have enforced a photometric warp that requires prior knowledge of the shape of the camera, and with this information now making up a part of the photometric-warp pipeline, our neural networks are

forced to learn to use this information to estimate depth and pose at the correct scale. To see why this is true, we should consider the effect on the loss function for an incorrectly estimated scale-factor. We may find that estimating a twice-as-large room for a twice-as-large translation works to photometrically warp the center-view well, but this will fail immensely for each of the other  $N - 1$  sub-apertures being warped. Thus, our networks are penalised harshly for incorrect scale estimates.

# Chapter 5

# Results

In this chapter, we describe the results from our pose and depth experiments. In Chapter 3 we described an unsupervised pipeline for learning depth and visual odometry from monocular imagery. Similarly, in Chapter 4 we adapted conventional 2D machine learning techniques to apply 4D image data to learning these tasks. Specifically, we recommended two variants of an algorithm for learning depth and visual odometry from light fields - the first uses all the information from the light field to generate a novel rendering of a single  $U, V$  slice, while the second method uses the known configuration of the camera array to create a rendering of the complete light field. In this chapter, we contrast results for three algorithms - each of the two novel pipelines described in Chapter 4, and the existing monocular approach described in Chapter 3. For the two algorithms that we have suggested, we also evaluate three ingestion methods: focal-stacks, volumetric images, and tiled EPI images.

First, we evaluate each algorithm's performance in pose-estimation, and demonstrate results for cumulated trajectories for several input-sequences. Subsequently, we'll evaluate the algorithms' performance in depth-prediction and photometric reconstruction.

## 5.1 Experimental Setup Details

In each of our experiments, we use the same training dataset of 9000 grayscale input images consisting of 48 short video-sequences. The testing set, which is also the same across all experiments, consists of 400 input images over 4 video-sequences. For training we use a batch size of 2 images, and train for a total of 300 epochs.

## 5.2 Pose Estimation and Visual Odometry

An important metric of error that we use to evaluate pose prediction, is the *absolute instantaneous error* for each of the 6 degrees-of-freedom that are estimated. This metric describes the magnitude of the error between the predicted and the true component. As described in Chapter 4, we have collected ground-truth pose data using a robotic manipulator platform with a repeatability of  $\pm 0.01$  millimeters, thus allowing a strong evaluation of pose-estimation. Meanwhile, we can evaluate each pipeline’s effectiveness in estimating the *average cumulative error* by comparing the true trajectory with the predicted trajectory.

### 5.2.1 Single-view Reconstruction Approach

The following histograms summarise 400 observations of the absolute instantaneous error for each of the 6 degrees of freedom (translation and rotation in X, Y and Z), over three input methods (tiled epipolar images, focal stacks, and volumetric stacks).

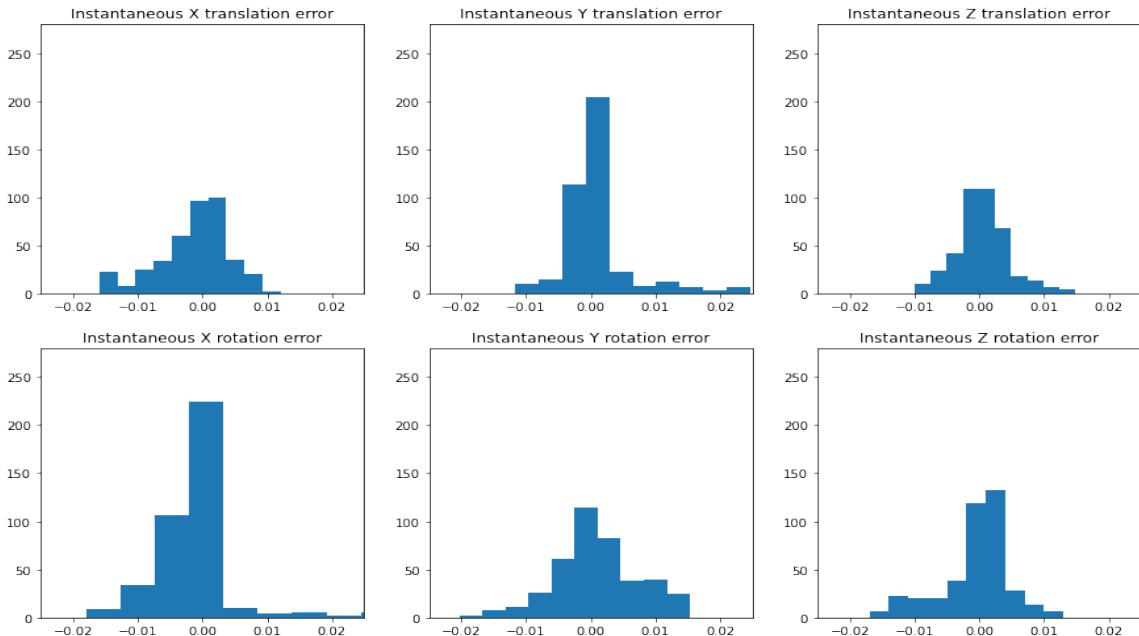


Figure 5.1: Instantaneous absolute errors when **tiled epipolar images** are the input method.

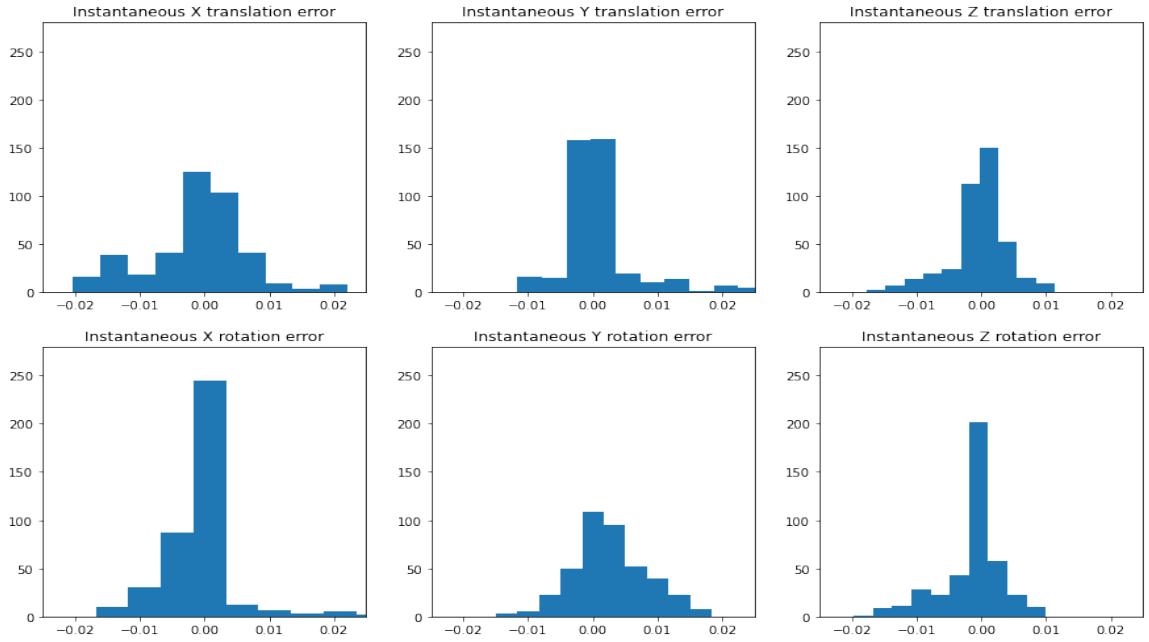


Figure 5.2: Instantaneous absolute errors when **focal stacks** images are the input method.

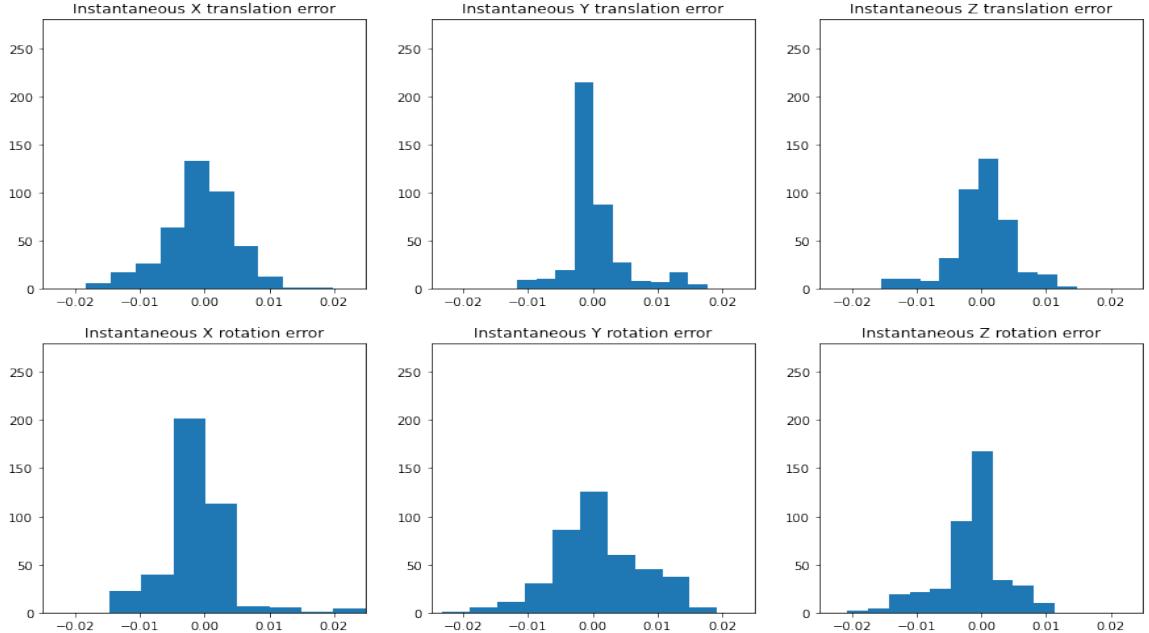


Figure 5.3: Instantaneous absolute errors when **volumetric stacks** are the input method.

Figure 5.5 shows 2 examples of true and estimated trajectories for the three input methods.

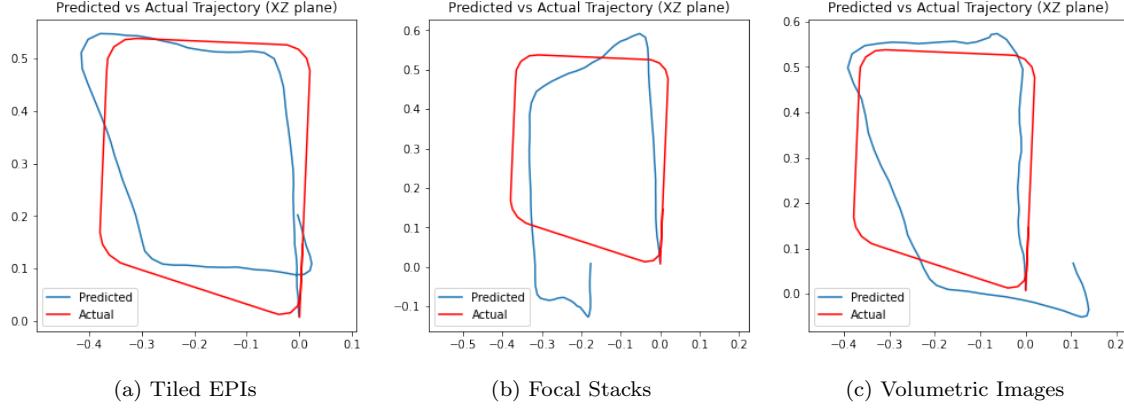


Figure 5.4: Estimated and true trajectories for input sequence 16.

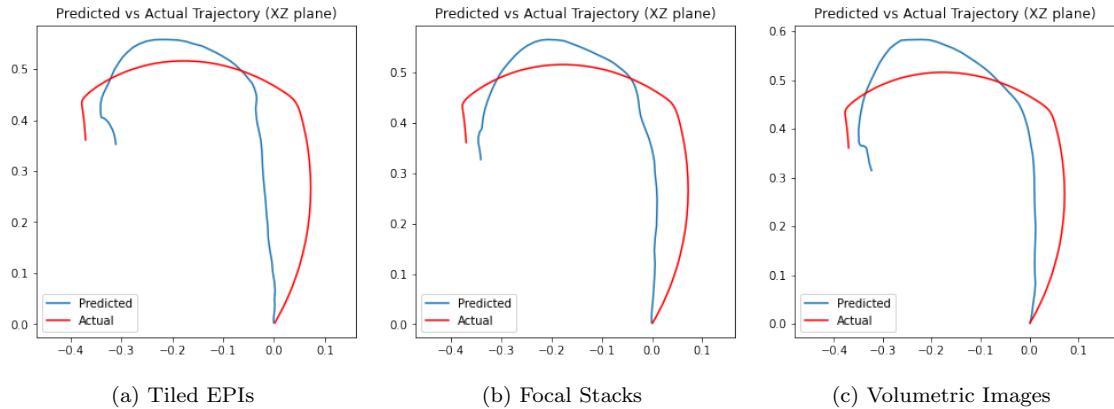


Figure 5.5: Estimated and true trajectories for input sequence 44.

Table 5.1 summarises the instantaneous and cumulative errors for the three input methods.

Table 5.1: Mean Absolute Instantaneous Error and Cumulative Error

Input Method	Absolute Instantaneous Error	Cumulative Error
Tiled EPI Polar Images	TODO	TODO
Focal Stacks	TODO	TODO
Volumetric Stacks	TODO	TODO

### 5.2.2 Light field Reconstruction Approach

Here we describe the performance of the second variant of our suggested pipeline, using the known camera configuration to photometrically warp the complete light field. Because this pipeline incorporates known information about the relationship between individual sub-apertures on the camera module, we expect this pipeline to demonstrate improved performance in pose estimation and scale-awareness.

We carry this expectation because this variant of the pipeline generates a stronger supervision signal - the larger number of pixels being warped results in a significantly heavier loss for the same error, compared with the single-view reconstruction approach. Additionally, this variant carries a much smaller tolerance to scale inconsistencies due to the constraints imposed when incorporating multiple sub-apertures and their known relative spacing.

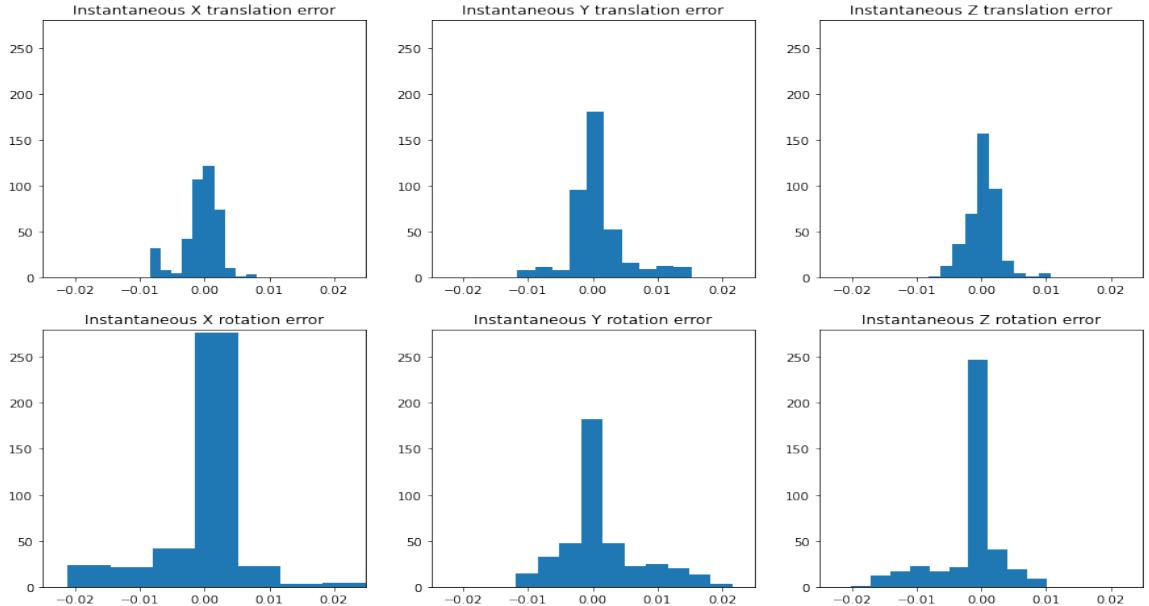


Figure 5.6: Instantaneous absolute errors when **tiled epipolar images** are the input method.

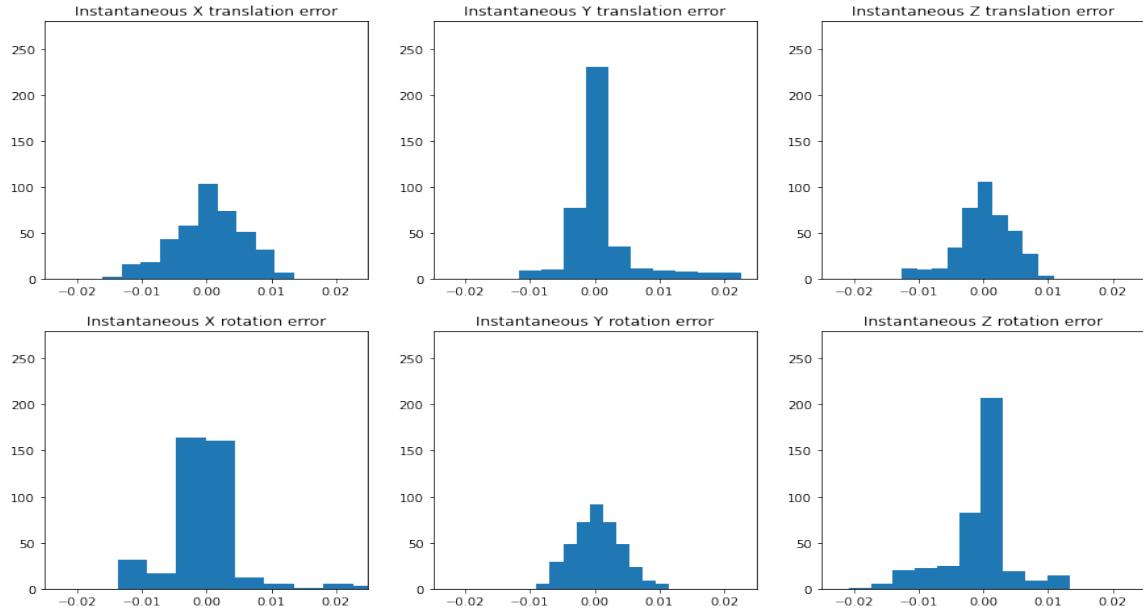


Figure 5.7: Instantaneous absolute errors when **focal stacks** are the input method.

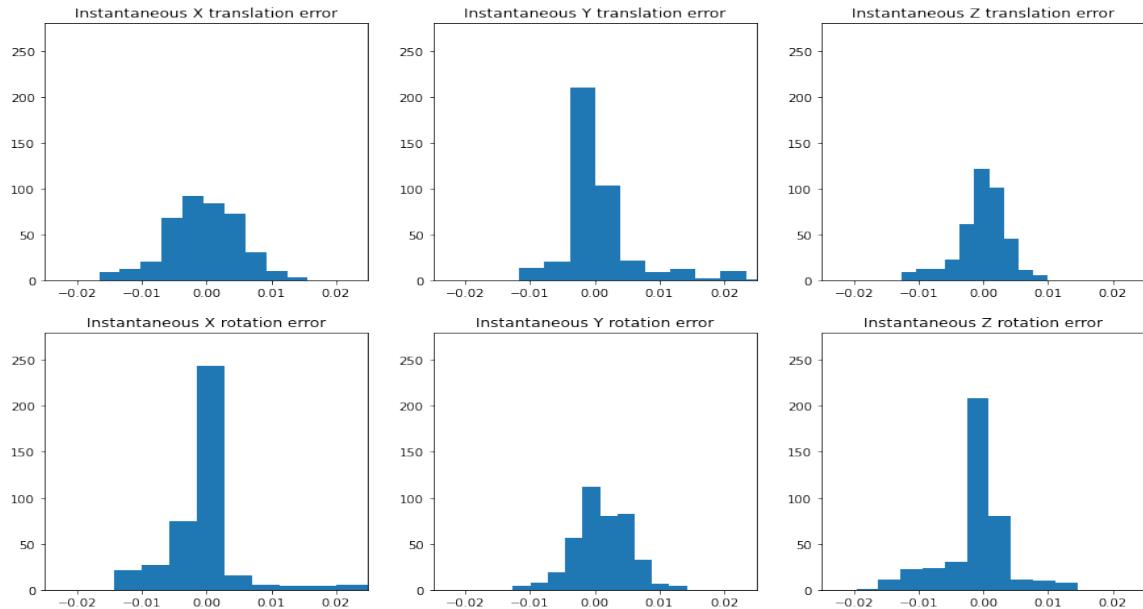


Figure 5.8: Instantaneous absolute errors when **volumetric stacks** are the input method.

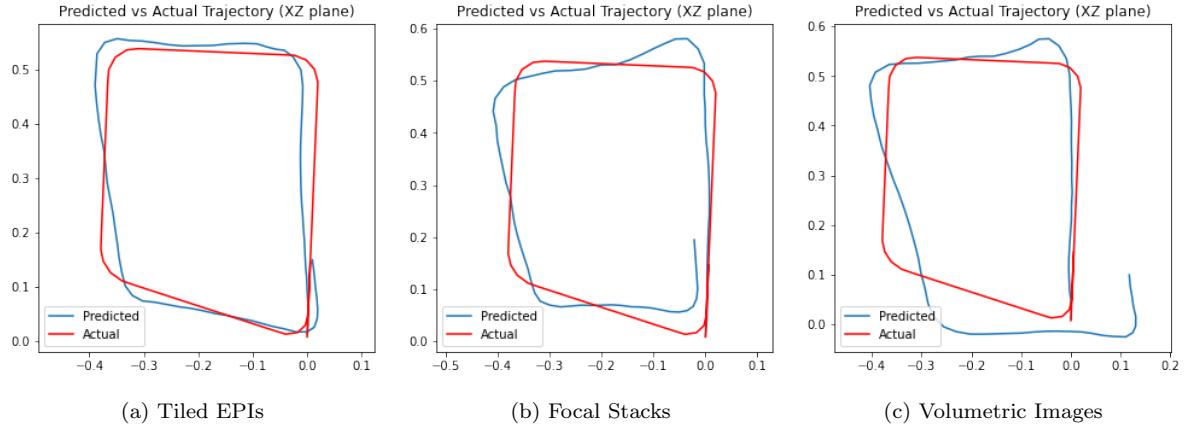


Figure 5.9: Estimated and true trajectories for input sequence 16.

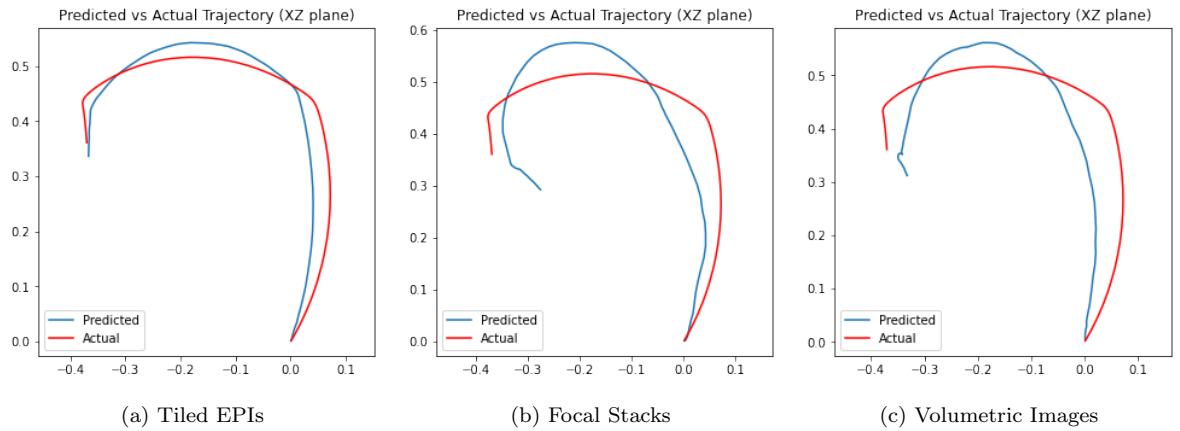


Figure 5.10: Estimated and true trajectories for input sequence 44.

Table 5.2: Mean Absolute Instantaneous Error and Cumulative Error

Input Method	Absolute Instantaneous Error	Cumulative Error
Tiled EPI Polar Images	TODO	TODO
Focal Stacks	TODO	TODO
Volumetric Stacks	TODO	TODO

### 5.2.3 Monocular Approach

TODO: histograms of absolute instantaneous error

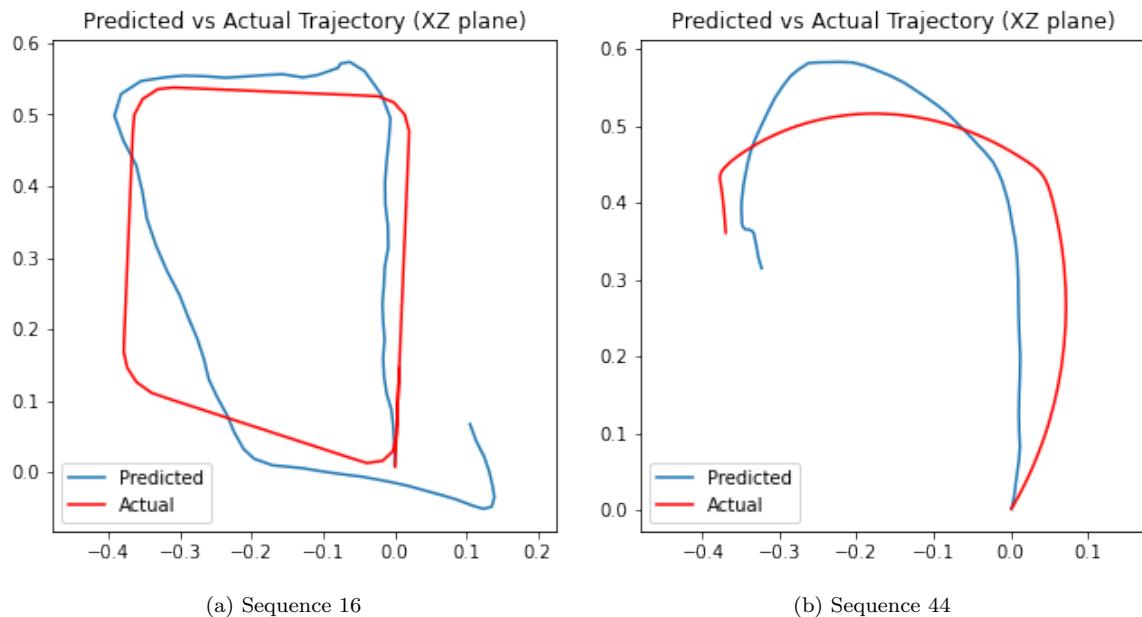


Figure 5.11: Estimated and true trajectories using monocular imagery.

### 5.2.4 Comparison between Approaches and Summary

Instantaneous Absolute Translational Error vs. Input Method

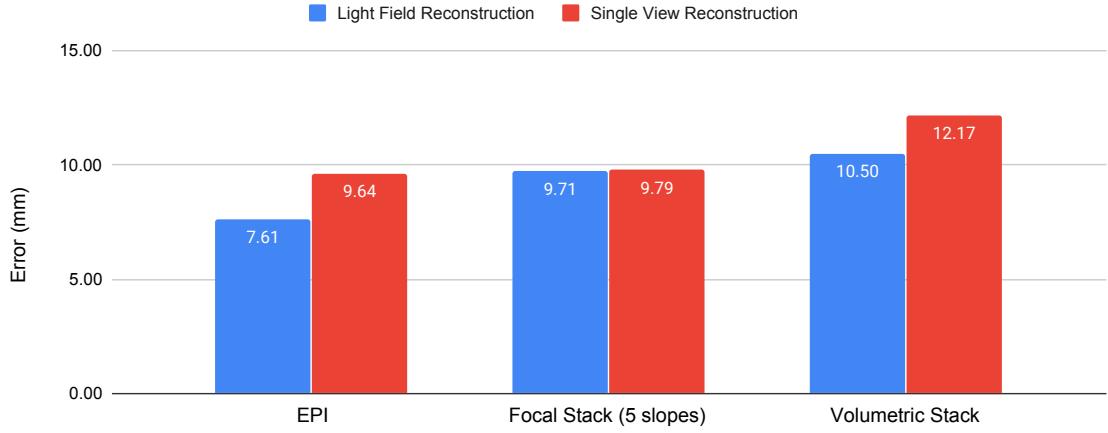


Figure 5.12: We find that in terms of translational error, the light field reconstruction pipeline has an edge over the single-view warp variant in all three cases. Additionally, using tiled EPI's demonstrates the smallest translational error of the three input methods.

Cumulative Translation Error vs. Input method

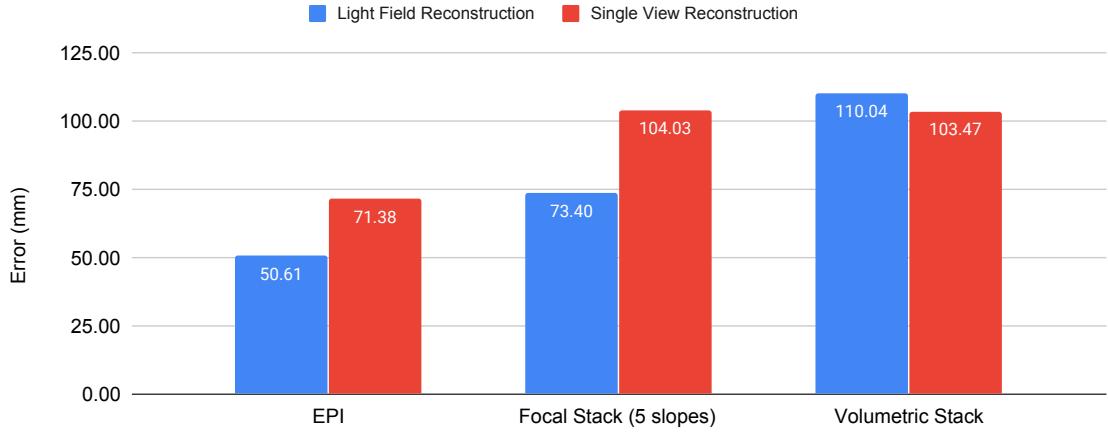


Figure 5.13: Similarly, we find that over the course of an entire video-sequence, tiled EPI's once again outperform all other input strategies. Over an entire video-sequence, the light field reconstruction pipeline generally outperforms the single-warp pipeline.

Instantaneous Absolute Rotation Error vs. Input Method

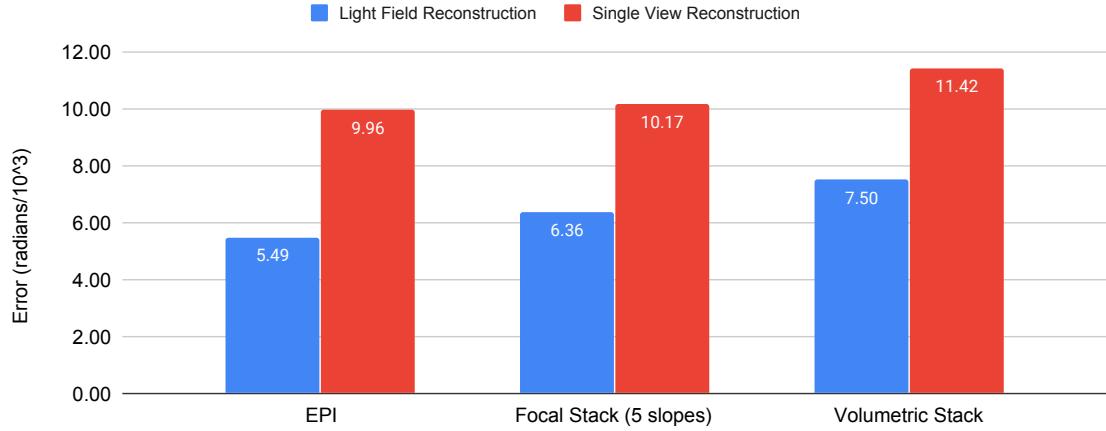


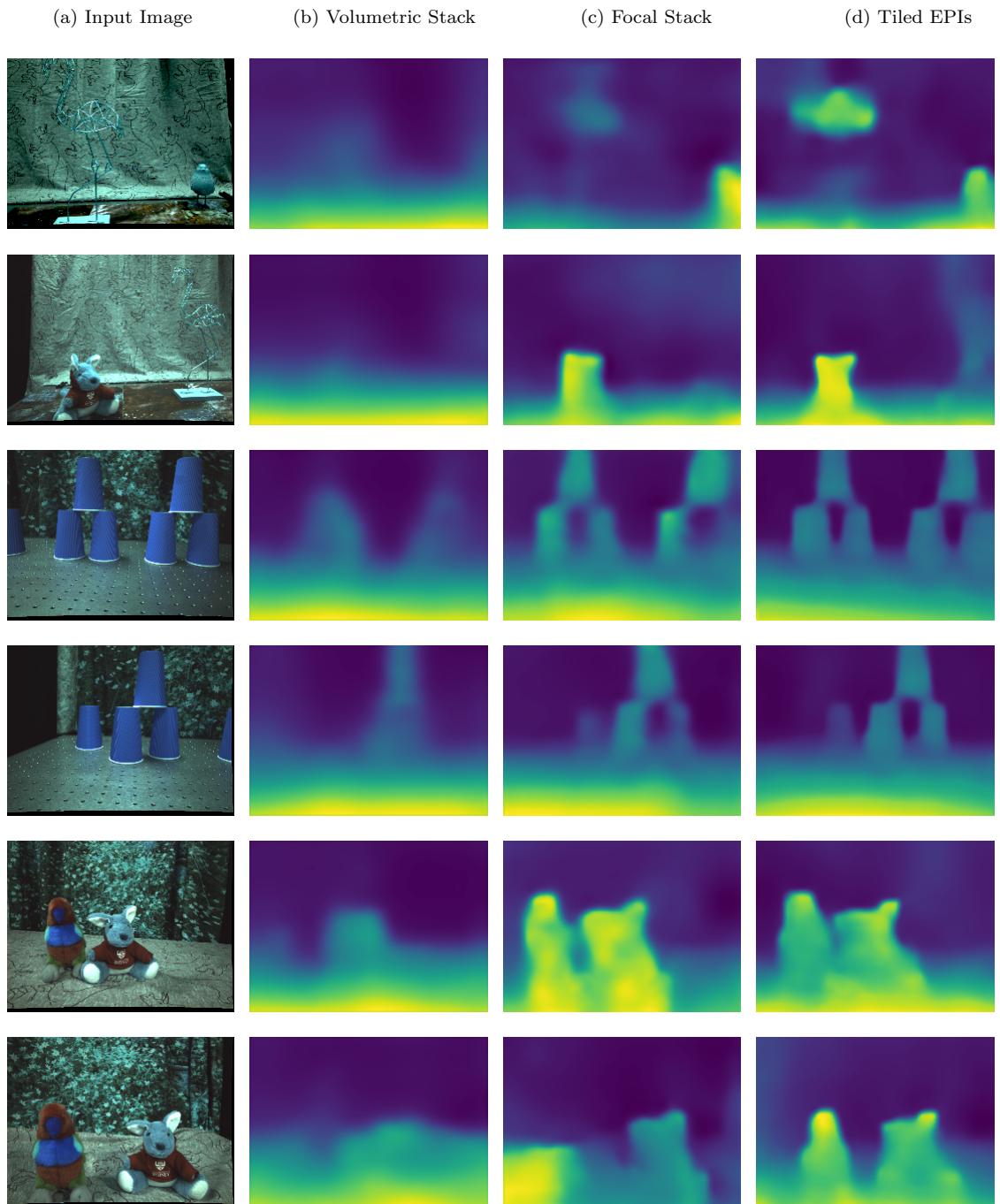
Figure 5.14: While using light field reconstruction has a small advantage over single view reconstruction in translational terms, we find that in terms of rotation the benefits are far more significant. Once again, the tiled EPI insertion strategy continues to demonstrate the smallest error in terms of rotation on our validation dataset.

### 5.3 Depth and Photometric Reconstruction

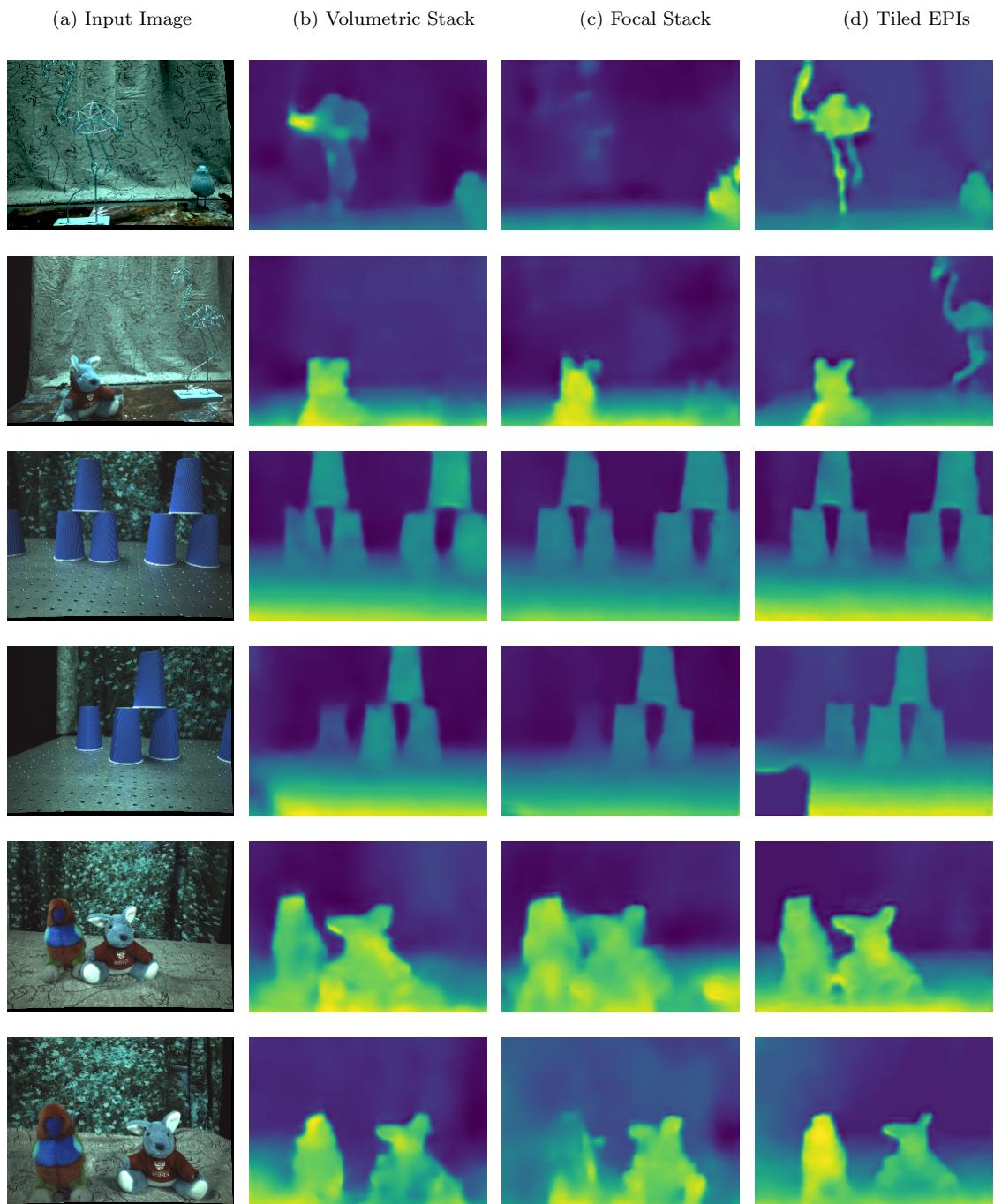
Aside from pose predictions and trajectories, our pipeline also generates depth-map predictions. In this section, we evaluate these outputs. Because our data acquisition did not include the collection of ground-truth depth data, we must find more creative ways of evaluating depth. One way we can do this is by inspecting the visual quality of the depth maps, observing the artifacts and details resolved. For example, in the depth maps shown, we include a particularly challenging scenario that includes a wire-frame model of a flamingo.

Another way we can evaluate the quality of our depth maps is to compare the photometric loss from image synthesis. Comparing the synthesised image with the ground-truth image, we can compute a difference-image which shows the magnitude of the error in rendering the synthesised image.

### 5.3.1 Single-view Reconstruction Approach



### 5.3.2 Light field Reconstruction Approach



### 5.3.3 Monocular Approach

### 5.3.4 Comparison between Approaches and Summary

# **Chapter 6**

## **Discussion**

### **6.1 Comparison between Unsupervised Algorithms using Light Field Data**

#### **6.1.1 Enforcing Scale Consistency**

### **6.2 Comparison between Input Methods**

#### **6.2.1 Performance of Epipolar Images**

#### **6.2.2 Performance of Volumetric Stacks**

#### **6.2.3 Performance of Focal Stacks**

### **6.3 Comparison to existing Monocular Approaches**

# Bibliography

- [1] Edward H. Adelson and James R. Bergen. The plenoptic function and the elements of early vision. In *Computational Models of Visual Processing*, pages 3–20. MIT Press, 1991.
- [2] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(5):698–700, May 1987.
- [3] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.
- [4] Stefan Behrendt. *Isometric projection*. Aug 2006.
- [5] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992.
- [6] Jia-Wang Bian, Zhichao Li, Naiyan Wang, Huangying Zhan, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. 2019.
- [7] Robert C. Bolles and H. Harlyn Baker. Readings in computer vision: Issues, problems, principles, and paradigms. chapter Epipolar-plane Image Analysis: A Technique for Analyzing Motion Sequences, pages 26–36. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987.
- [8] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian D. Reid, and John J. Leonard. Simultaneous localization and mapping: Present, future, and the robust-perception age. *The Computing Research Repository*, 2016.
- [9] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145 – 155, 1992. Range Image Understanding.

- [10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [11] Erwin Coumans and Yunfei Bai. *pybullet, a Python module for physics simulation in robotics, games and machine learning*. 2017.
- [12] Donald Dansereau. *Plenoptic Signal Processing for Robust Vision in Field Robotics*. PhD thesis, Australian Centre for Field Robotics, 2014.
- [13] Donald Dansereau, Ian Mahon, Oscar Pizarro, and Stephan B. Williams. Plenoptic flow: Closed-form visual odometry for light field cameras. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4455–4462, 2011.
- [14] Donald Dansereau, Oscar Pizarro, and Stefan Williams. Linear volumetric focus for light field cameras. *ACM Transactions on Graphics*, 34:1–20, 03 2015.
- [15] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *The Computing Research Repository*, abs/1406.2283, 2014.
- [16] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
- [17] Paul Verlaine Gakne and Kyle O’Keefe. Tackling the scale factor issue in a monocular visual odometry using a 3D city model. 2018.
- [18] Ravi Garg, Vijay Kumar B. G, and Ian D. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. *The Computing Research Repository*, 2016.
- [19] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 2013.
- [20] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *The Computing Research Repository*, 2016.
- [21] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. *The Computing Research Repository*, 2018.

- [22] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 43–54, New York, NY, USA, 1996. ACM.
- [23] Wolfram Gothe. *Three-point perspective projection*. Jul 2009.
- [24] Chris Harris and Mike Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.
- [25] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.
- [26] Berthold Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society A*, 4:629–642, 04 1987.
- [27] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial transformer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2017–2025. Curran Associates, Inc., 2015.
- [28] Jian Jiao, Jichao Jiao, Yaokai Mo, Weilun Liu, and Zhongliang Deng. Magicvo: End-to-end monocular visual odometry through deep bi-directional recurrent convolutional neural network. *CoRR*, abs/1811.10964, 2018.
- [29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.
- [30] Andreas Kuehnefuss, Niclas Zeller, Franz Quint, and Uwe Stilla. Feature based rgb-d slam for a plenoptic camera. 2016.
- [31] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.
- [32] Mark Levoy and Pat Hanrahan. Light field rendering. In *Proc. ACM SIGGRAPH*, 1995.
- [33] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian D. Reid. Learning depth from single monocular images using deep convolutional neural fields. *The Computing Research Repository*, 2015.

- [34] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *The Computing Research Repository*, 2014.
- [35] H. C. Longuet-Higgins. *A Computer Algorithm for Reconstructing a Scene from Two Projections*, pages 61–62. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987.
- [36] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [37] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. *The Computing Research Repository*, abs/1512.02134, 2015.
- [38] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA, 1969.
- [39] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.
- [40] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–652–I–659 Vol.1, Los Alamitos, CA, USA, 2004. IEEE Computer Society.
- [41] Seymour Papert. The summer vision project. 1966.
- [42] Pedro F. Proença. *Robust RGB-D Odometry under Depth Uncertainty for Structured Environments*. PhD thesis, University of Surrey, 2018.
- [43] Frank Rosenblatt. *Principles of Neurodynamics*. Spartan Books, 1959.
- [44] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.
- [45] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

- [46] K. A. Skinner and M. Johnson-Roberson. Towards real-time underwater 3D reconstruction with plenoptic cameras. In *2016 International Conference on Intelligent Robots and Systems*, pages 2014–2021, Oct 2016.
- [47] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. DeepVO: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. *CoRR*, abs/1709.08429, 2017.
- [48] Ting-Chun Wang, Jun-Yan Zhu, Hiroaki Ebi, Manmohan Chandraker, Alexei A. Efros, and Ravi Ramamoorthi. A 4D light-field dataset and CNN architectures for material recognition. *The Computing Research Repository*, 2016.
- [49] Li Yao, Yunjian Liu, and Weixin Xu. Real-time virtual view synthesis using light field. *EURASIP Journal on Image and Video Processing*, 2016(1):25, 2016.
- [50] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian D. Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. *The Computing Research Repository*, 2018.
- [51] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces, 1994.
- [52] Dingfu Zhou, Y. Dai, and Hongdong Li. Reliable scale estimation and correction for monocular visual odometry. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 490–495, 2016.
- [53] Dingfu Zhou, Yuchao Dai, and Hongdong Li. Ground plane based absolute scale estimation for monocular visual odometry. *The Computing Research Repository*, 2019.
- [54] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017.