ABSTRACT

Title of dissertation: COMPUTER VISION IN THE SPACE OF LIGHT RAYS: PLENOPTIC VIDEOGEOMETRY AND POLYDIOPTRIC CAMERA DESIGN

Jan Neumann, Doctor of Philosophy, 2004

Dissertation directed by: Professor Yiannis Aloimonos
Department of Computer Science

Most of the cameras used in computer vision, computer graphics, and image processing applications are designed to capture images that are similar to the images we see with our eyes. This enables an easy interpretation of the visual information by a human observer. Nowadays though, more and more processing of visual information is done by computers. Thus, it is worth questioning if these human inspired "eyes" are the optimal choice for processing visual information using a machine.

In this thesis I will describe how one can study problems in computer vision without reference to a specific camera model by studying the geometry and statistics of the space of light rays that surrounds us. The study of the geometry will allow us to determine all the possible constraints that exist in the visual input and could be utilized if we had a perfect sensor. Since no perfect sensor exists we use signal processing techniques to examine how well the constraints between different sets of light rays can be exploited given a specific camera model. A camera is modeled as a spatio-temporal filter in the space of light rays which lets us express the image formation process in a function approximation framework. This framework then allows us to relate the geometry of the imaging camera to the performance of the vision system with regard to the given task.

In this thesis I apply this framework to problem of camera motion estimation. I show how by choosing the right camera design we can solve for the camera motion using linear, scene-independent constraints that allow for robust solutions. This is compared to motion estimation using conventional cameras. In addition we show how we can extract spatio-temporal models from multiple video sequences using multi-resolution subdivison surfaces.

COMPUTER VISION IN THE SPACE OF LIGHT RAYS : PLENOPTIC VIDEO GEOMETRY AND POLYDIOPTRIC CAMERA DESIGN

by

Jan Neumann

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2004

Advisory Commmittee:

Professor Yiannis Aloimonos, Chair and Advisor
Professor Rama Chellappa, Dean's Representative
Professor Larry Davis
Professor Hanan Samet
Professor Amitabh Varshney

ACKNOWLEDGMENTS

I owe my gratitude to all the people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever. First and foremost I'd like to thank my advisor, Professor Yiannis Aloimonos for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past seven years, while still being free to follow my own paths. He has always given me help and advice and there has never been an occasion when he hasn't given me time to discuss personal and research questions with him.

I also want to thank Professor Rama Chellappa, Professor Larry Davis, Professor Amitabh Varshney, and Professor Hanan Samet for agreeing to serve on my thesis committee and for sparing their invaluable time reviewing the manuscript. I also want to thank the late Professor Azriel Rosenfeld whose guidance and enthusiasm for the field of computer vision made the computer vision lab at the University of Maryland the inspiring place that it is.

My colleagues and friends at the computer vision lab have enriched my graduate life in many ways and deserve a special mention, Patrick Baker, Tomas Brodsky, Filip Defoort, Cornelia Fermüller, Gutemberg Guerra, Abhijit Ogale, and Robert Pless. I also want to thank my friends Yoni Wexler, Darko Sedej, Gene Chipmann, Nizar Habash, Adam Lopez, Supathorn Phongikaroon and Theodoros Salonidis for sharing the graduate student experience with all its highs and lows with me, and understanding if I could not always spend as much time with them as they deserve. I also want to thank Kostas

Daniilidis for the many illuminating discussions about the structure of the space of light rays.

I dedicate this thesis to my family - my mother Insa, father Joachim and brother Lars,who have always stood by me and guided me through my career, and especially to my wife Danitza, who has supported me during the most stressful times of my PhD and was willing to accept the negative consequences of being with a PhD student, thank you.

In the end, I would also like to acknowledge the financial support from the Department of Computer Science and the Graduate School for their two fellowships and the Deutsche Studienstiftung for their summer schools that truly enriched my studies during the early years of my graduate school career.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

For most of us vision is a very natural process that happens unconsciously and allows us to navigate accurately through the world, detect moving objects and judge their speeds, recognize objects and actions, and interact with the world using visual feedback. Unfortunately, up to now nobody has been able to decipher how humans are able to perform their advanced vision tasks. Therefore, when we think about vision algorithms, we look beyond the human example and utilize mathematical disciplines such as geometry, calculus, and statistics to find solutions to the tasks mentioned, although many of these methods do not directly map into any existing biological circuits. Despite the advances in algorithm development, when we think about vision hardware, we still usually think of cameras that capture images similar to the images taken by (two) eyes such as our own - that is, images acquired by camera-type eyes based on the pinhole principle.

Basically all commercial photo and video cameras used in computer vision research, with the exception of a few examples we will study later, are primarily designed to capture images that are as similar as possible to the images that a human eye would capture if placed at the camera's position. This is advantageous because it enables an easy interpretation of the visual information by a human observer. This advantage though seems to be of less and less importance nowadays, because more and more image interpretation tasks become automated and thus humans will interpret less and less of the raw data that is coming from the cameras, but rather the processed output. Therefore, I believe that we should look beyond the pinhole camera principle, and design a camera in dependence on the task we want to perform. Human eyes are not the only types of eyes that exist, the biological world reveals a large variety of eye designs many of which are not based on the pinhole principle.

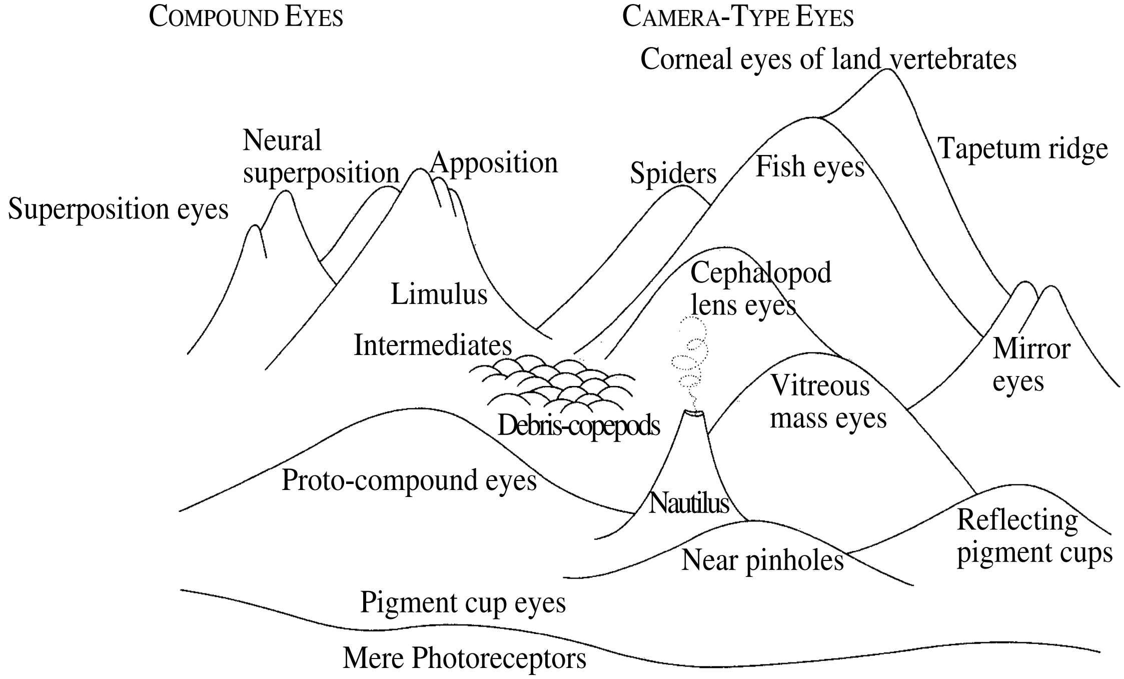


Figure 1.1: Michael Land's landscape of eye evolution.

The biological world gives a good example of task specific eye design. It has been estimated that eyes have evolved no fewer than forty times, independently, in diverse parts of the animal kingdom [37], and these eye designs, and therefore the images they capture, are highly adapted to the tasks the animal has to perform. This evolution of eyes is nicely illustrated in Michael Land's landscape of eye evolution (Fig. 1.1) where every hill and mountain denotes a different independent eye design. This suggests that we should not just focus our efforts on designing algorithms that optimally process a given visual input, but also optimize the design of the imaging sensor with regard to the task at hand, so that the subsequent processing of the visual information is facilitated. The notion that we need to build accurate models of the information processing pipeline to optimize our choice of algorithm in dependence on the statistics of the environment has become more prominent lately ([32], [149]).

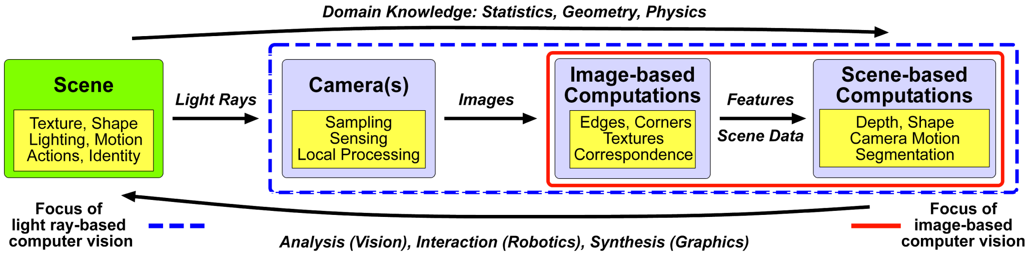


Figure 1.2: The information pipeline in computer vision. Usually computer vision starts from images, but this already assumes a choice of camera. To abstract the notion of a specific sensing device, we need to analyze vision problems in the space of light rays

The exploration of new sensor designs has already begun. Technological advances make it possible to construct integrated imaging devices using electronic and mechanical micro-assembly, micro-optics, and advanced on-chip processing. These devices are not only of the kind that exists in nature such as log-polar retinas [24], but also of many other kinds such as catadioptric cameras [100, 50]. Some initial work has begun on custom mirror-design where a mirror is machined such that the imaging geometry of the mirror-camera combination is optimized to approximate a predefined scene to image mapping [59, 141]. Nevertheless, a general framework to relate the design of an imaging sensor to its usefulness for a given task is still missing. As seen in Fig. 1.2 if we phrase our analysis of vision algorithms in terms of images, then we already have chosen a specific camera model to form an image. To abstract from a specific camera geometry, we need to base our analysis on the most general representation for visual information, that is the space of light rays and its functional representation the plenoptic function. By analyzing vision problems in the space of light rays, we can optimize over both the sensor and the algorithm to find an optimal solution. In this thesis I develop such a framework by studying the relationship between the subset of light rays captured by a camera and its performance with regard to a task.

In this thesis I will use the term polydioptric camera that was introduced in [104] to denote a generalized camera that captures a multi-view point subset of the space of light rays. (*dioptric*: assisting vision by refracting and focusing light). The essential question of polydioptric camera design is how to find the camera design that allows us to perform the tasks of interest as well as possible. For this we need to assess the relative importance of spatial resolution and depth resolution with regard to the problem at hand, and how the best compromise between the two can be found.

This framework is able to simultaneously address the three problems crucial to the design of next generation imaging systems:

- **The sampling problem.**

  Which rays of light should be sampled to yield optimal information for a particular function whether the task at hand is high resolution image reconstruction, imaging from different views, recovery of a scene's 3D structure, or detection of various salient features and targets? How can the appropriate sampling strategies be implemented in the prescribed form and fit? Generally, we are searching for a transform

Π that captures a set of light rays $\mathcal{L}$ and generates signal measurements $\mathcal{I}$.

$$\Pi(\mathcal{L}) = \mathcal{I}$$

Π could be a compound transform to account for many optical centers and many different (potentially overlapping) fields of view. In general the transform Π has a null space; therefore, direct inversion based on the signal measurements will not be possible.

- **The sensing problem.** How should the optical signals be sensed as to yield an optimal SNR, dynamic range, possibly salience-ordered pixel/region readout, programmable spatial resolution, and possible detector properties adaptation? How do the selected sensor functions contribute to the prescribed system function?

- **The processing problem.** What kind of scene features can different camera designs extract? Some can only extract features based on texture cues (2D), others in addition also features based on shape cues (3D) as we discussed above. What are the mathematical tools that need to be developed? Or conversely, what sampling and sensing strategy is necessary to achieve efficient processing for a prescribed function? What processing hardware will be necessary to achieve these operations within a prescribed time? How can we analyze how much these features will help us?

These three problems are not orthogonal and must be addressed simultaneously. For example, efficient mathematics will demand a particular sampling of the light rays. Sensors may not be able to deliver sufficient SNR for a given sampling strategy. Therefore, the optimal solution will lie somewhere in the ”space of compromises” across the three areas. In this thesis I will focus on the processing and sampling questions because they are the two properties that are much easier to modify for the camera designer then the underlying electronics on the sensor chip itself. Thus, we assume that the properties of the sensor element are fixed and can be described by a simple model.

To find the optimal solution and design a task specific camera, we need to answer the following two questions:

1. How is the relevant visual information that we need to extract to solve our task encoded in the visual data that a camera can capture?

2. What is the camera design and image representation that optimally facilitates the extraction of the relevant information?

To answer the first question, we first have to think about what we mean by visual information. When we think about vision, we usually think of interpreting the images taken by (two) eyes such as our own - that is, perspective images acquired by camera-type eyes based on the pinhole principle. These images enable an easy interpretation of the visual information by a human observer. Therefore, most work on sensor design has focused on designing cameras that would result in pictures with higher fidelity (e.g.[66]). Image fidelity has a strong impact on the accuracy with which we can make quantitative measurements of the world, but the qualitative nature of the image we capture (e.g., single versus multiple view point images) also has a major impact on the accuracy of measurements which cannot be measured by a display-based fidelity measure. Since nowadays most processing of visual information is done by machines, there is no need to confine oneself to the usual perspective images. Instead, as motivated at the start of the introduction, we propose to study how the relevant information is encoded in the geometry of the time-varying space of light rays which allows us to determine how well we can perform a task given *any* set of light ray measurements. In Chapter 2 we will the examine the structure of the space of light rays and analyze what kind of information can be extracted about a scene. To answer the second question we have to determine how well a given eye can capture the necessary information. We can interpret this as an approximation problem where we need to assess how well the relevant subset of the space of light rays can be reconstructed based on the samples captured by the eye, our knowledge of the transfer function of the optical apparatus, and our choice of function space to represent the image. In Chapter 4 we will model eyes as spatio-temporal sampling patterns in the space of light rays which allows to use well developed tools from signal processing and approximation theory to evaluate the suitability of a given eye design for the proposed task and determine the optimal design. The answers to these two questions then allow us to define a fitness function for different camera designs with regard to a given task.

1.1 Why study cameras in the space of light rays?

A conventional camera is observing the world only from a single effective view point. It is well known that due to the camera projection the range information about the world gets lost. That means one can only capture an image of the two-dimensional properties of the object surfaces in the scene. Therefore, any information that is not uniquely determined by the scene texture itself cannot be accurately recovered without assumptions about the world. As an example, if we want to segment the captured image into image regions corresponding to the different objects in the scene, we have to make assumptions beforehand to which texture belongs to which object.

In contrast, if a camera captures light from many view points the correlations between the different images can be used to infer information about the three-dimensional structure of the world, that is for example shape estimation and the segmentation of the scene into distinct objects using occlusion events. In this case if we want to segment the multi-perspective image according to the views of the objects we observed, we can utilize the intrinsic structure of the captured multi-perspective image to detect occlusion events and segment the image according to a combination of two-dimensional texture and three-dimensional depth cues.

If the camera is moving, thus instead of images we capture image sequences, we can use the correlations between different images that observe the same scene to infer more information about the world even with conventional cameras. The problem of structure from motion has been studied in depth [61, 55, 88], but it is still not sufficiently solved to allow for fool-proof algorithms. Thus, if we only rely on texture cues to detect independently moving objects or track objects, it is difficult to distinguish between changes in images due to moving objects or parallax-inducing 3D structures. It is essential that we estimate the motion of the camera which is in general a highly non-linear problem since it involves the estimation of the 3D structure of the scene as well. The non-linear nature of the problem causes ambiguities in the solution and due to the complexity of the estimation it has not been possible up to now to process the motion estimation directly on a camera chip.

In contrast, a multi-perspective camera allows us to compute the rigid motion of a camera solely based on spatio-temporal image derivative measurements by solving a system of linear equations with few (six) unknowns as we will show in Chapter 5. Efficient solvers for these problems have been recently implemented on cheap graphics hardware chips, thus allowing us to solve for the 3D motion of the camera directly inside the box using the camera hardware. In addition, the low-dimensionality and scene independent formulation of the changes in the images due to the camera motion allow us to apply simple motion segmentation algorithms to the captured polydioptric images to detect and track moving objects.

These superior abilities of polydioptric imaging with regard to 3D scene structure estimation and segmentation, can also be utilized to improve the resolution of the sensor through super-resolution techniques. In nearly all super-resolution techniques it is assumed that the scene consists of a single fronto-parallel plane and that there are no occlusions between different views. In reality this is not necessarily satisfied. A polydioptric camera is able to first compute an estimate of the depth structure of the scene and detect occlusion events, before the super-resolution system is solved with appropriate masking of the input and adaption of the parameters modeling the imaging process.

1.2 Example Application: Dynamic 3D photography

To evaluate and compare different eye designs in a mathematical framework, I choose the recovery of spatio-temporal scene descriptions from image sequences, that is structure from motion, as the problem of interest. There are many approaches to structure from motion (e.g. see [54, 88] for an overview), but essentially all these approaches disregard the fact that the way images are acquired, already determines to a large degree how difficult it is to solve for the structure and motion of the scene. Since systems have to cope with limited resources, their cameras should be designed to optimize subsequent image processing.

Of most practical importance are the following two specific subcases of the structure from motion problem:

1. **Static Structure from Dynamic Cameras:** We are given a single (or a set of) camera(s) that move rigidly in space. Based on the recovered image sequences we would like to estimate the camera motion, the properties of the static objects in scene (such as shape and textures). If the scene properties are not invariant over time, then we need to use robust statistics to differentiate between the static and non-static parts of the scene.

2. **Dynamic Structure from Static Cameras:** Given a number of calibrated cameras in a known configuration that capture a sequence of images, find the shape of the objects, their surface properties, and the motion field on the object surface. For some example results see Chapter 6.

1.3 Plenoptic video geometry: How is information encoded in the space of light rays?

In this outline we will illustrate the concept of the polydioptric camera design by examining the plenoptic video geometry for the case of 3D ego-motion estimation. This analysis will be further refined in Chapter 5. At each location $\boldsymbol{x}$ in free space the plenoptic function $\mathcal{L}(\boldsymbol{x}; \boldsymbol{r}; t);\ \mathcal{L} : \mathbb{R}^3 \times \mathbb{S}^2 \times \mathbb{R}_+ \rightarrow \Gamma$ measures the radiance, that is the light intensity or color from a given direction $\boldsymbol{r}$ at time t. Γ denotes here the spectral energy, and equals $\mathbb{R}$ for monochromatic light, $\mathbb{R}^n$ for arbitrary discrete spectra, or could be a function space for a continuous spectrum. $\mathbb{S}^2$ is the unit sphere of directions in $\mathbb{R}^3$.

Let us assume that the albedo of every scene point is invariant over time and that we observe a static world under constant illumination. In this case, the radiance of a light ray does not change over time which implies that the total time derivative of the

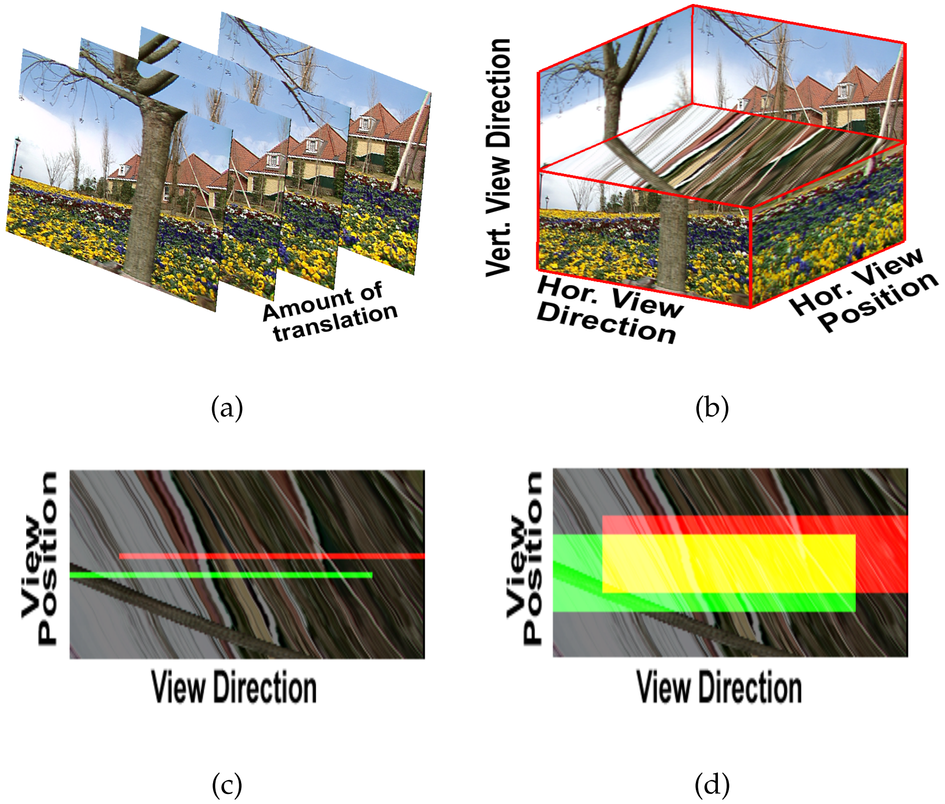


Figure 1.3: (a) Sequence of images captured by a horizontally translating camera. (b) Epipolar image volume formed by the image sequence where each voxel corresponds to a unique light ray. The top half of the volume has been cut away to show how a row of the image changes when the camera translates. (c) A row of an image taken by a pinhole camera at two time instants (red and green) corresponds to two non-overlapping horizontal line segments in the epipolar plane image, while in (d) the collection of corresponding "rows" of a polydioptric camera at two time instants corresponds to two rectangular regions of the epipolar image that do overlap (yellow region). This overlap enables us to estimate the rigid motion of the camera purely based on the visual information recorded.

plenoptic function vanishes:

$$\frac{d}{dt}\mathcal{L}(\boldsymbol{x};\boldsymbol{r};t) = 0.$$

The set of imaging elements that make up a camera each capture the radiance at a given position coming from a given direction. If the camera undergoes a rigid motion, then we can describe this motion by an opposite rigid coordinate transformation of the ambient space of light rays in the camera coordinate system. This rigid transformation, parameterized by the rotation matrix $R(t)$ and a translation vector $\boldsymbol{q}(t)$ which maps the time-invariant space of light rays upon itself. Thus we have the following *exact* equality which we call the *discrete plenoptic motion constraint*

$$\mathcal{L}(R(t)\boldsymbol{x} + \boldsymbol{q}(t); R(t)\boldsymbol{r}; t) = \mathcal{L}(\boldsymbol{x}; \boldsymbol{r}; 0) \tag{1.1}$$

We see that if a sensor is able to capture a continuous non-degenerate subset of the plenoptic function, then the problem of estimating the rigid motion of this sensor has become an image registration problem that is *independent of the scene*. Therefore the only free parameters are the six degrees of freedom of the rigid motion. This global parametrization of the plenoptic motion field by only six parameters leads to a highly constrained estimation problem that can be solved with any multi-dimensional image registration criterion.

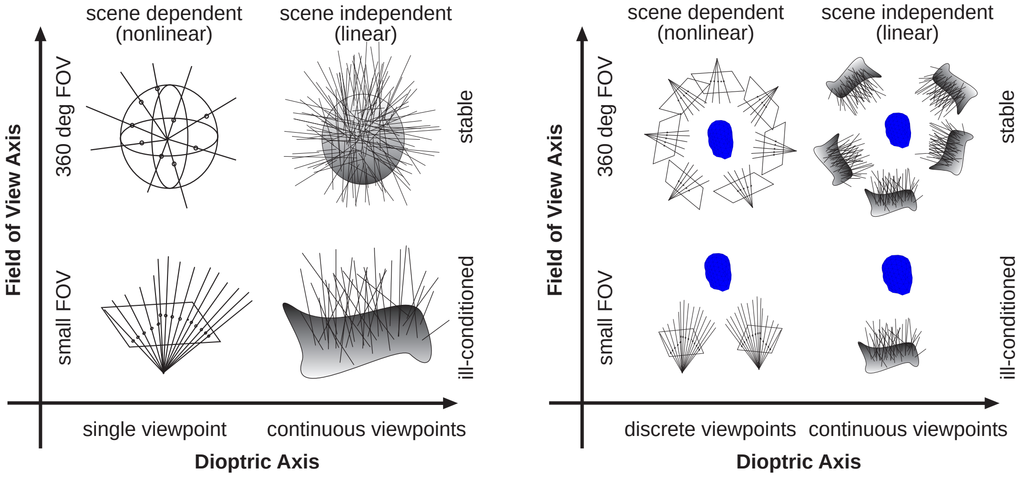
To illustrate this idea with an example, a camera is translated along the horizontal image axis and the images of the sequence are stacked to form an image volume (Figs. 1.3a-1.3b). Due to the horizontal translation scene points always project into the same row in each of the images. Such an image volume is known as an epipolar image volume [14] since corresponding rows lie all in the same epipolar plane. Each pixel in this volume corresponds to a unique light ray.

A horizontal slice through the image volume (Figs. 1.3c-1.3d) is called an epipolar plane image and contains the light rays lying in this epipolar plane parameterized by view position and direction. A row of an image frame taken by a pinhole camera corresponds to a horizontal line segment in the epipolar plane image (Fig. 1.3c) because we observe the light rays passing through a single point in space (single view point). In contrast, a polydioptric camera captures a rectangular area (multiple view points) of the epipolar image (Fig. 1.3d) because it captures light rays passing through a range of view points. Here we assumed that the viewpoint axis of the polydioptric camera is aligned with the direction of translation used to define the epipolar image volume. If this is not the case, the images can be warped as necessary. We see that a camera rotation around an axis perpendicular to the epipolar image plane corresponds to a horizontal shift of the camera image (change of view direction), while a translation of the camera parallel to an image row, causes a vertical shift (change of view point). These shifts can be different for each pixel depending on the rigid motion of the camera.

If we want to recover this rigid transformation based on the images captured using a pinhole camera, we see in (Fig. 1.3c) that we have to match two non-overlapping sets of light rays (shown as a bright green and a dark red line) since each time a pinhole camera captures by definition only the view from a single viewpoint. Therefore, it is necessary for an accurate recovery of the rigid motion that we have a depth estimate of the scene, since the correspondence between pixels in image rows taken from different view points depends on the local depth of the scene.

In contrast, we see in (Fig. 1.3d) that for a polydioptric camera the matching can be based purely on the captured image information, since the sets of light rays captured at consecutive times (bright yellow region) overlap. Disregarding sampling issues at the moment, we have a ”true” brightness constancy in the region of overlap, because we match a light ray with itself (Eq. (1.1)). This also implies that polydioptric matching is invariant to occlusions and view-dependent visual events such as specularities. We conclude that the correspondence of light rays using a polydioptric camera depends only on the motion of the camera, not on any properties of the scene, thus enabling us to estimate the rigid motion of the camera in a completely scene-independent manner. A more detailed analysis of the structure of the space of light rays can be found in Chapter 2.

1.4 Polydioptric Camera Design



(a) Static World - Moving Cameras

(b) Moving Object - Static Cameras

Figure 1.4: a) Hierarchy of Cameras for 3D Motion Estimation. The different camera models are classified according to the field of view (FOV) and the number and proximity of the different viewpoints that are captured (Dioptric Axis). The camera models are clockwise from the lower left: small FOV pinhole camera, spherical pinhole camera, spherical polydioptric camera , and small FOV polydioptric camera.

It is known that the stability of the structure from motion estimation depends on the collective field of view of all the "sub"-sensors making up the polydioptric camera. This relationship has been studied for example in [3, 33, 35, 70, 96, 47]. Some of these results can be found in Chapter 5.

Combining this result with the plenoptic motion equations (1.1) in [104] we can define a coordinate system on the space of camera designs (as shown in Fig. 1.4). The different camera models are classified according to the field of view (FOV) and the number and proximity of the different viewpoints that are captured (Dioptric Axis). This in turn determines if structure from motion estimation is a well-posed or an ill-posed problem, and if the estimation is scene dependent or independent (thus implying that the motion parameters are related linearly or non-linearly to the image measurements if we have differential motion as shown in Section 1.3).

One can see in the figure that the conventional pinhole camera is at the bottom of the hierarchy because the small field of view makes the motion estimation ill-posed and it is necessary to estimate depth and motion simultaneously. Although the estimation of structure and motion for a single-viewpoint spherical camera is stable and robust, it is still scene-dependent, and the algorithms which give the most accurate results are search techniques, and thus rather elaborate. One can conclude that a spherical polydioptric camera is the camera of choice to solve the structure from motion problem since it combines the stability of full field of view motion estimation with the linearity and scene independence of the polydioptric motion estimation. Such a camera would enable us to utilize new scene-independent constraints between the structure of the plenoptic function and the parameters describing the rigid motion of a polydioptric imaging sensor. Using tools of polydioptric sampling theory as described in Chapter 4 enables us to extend this qualitative analysis to a quantitative analysis.

A polydioptric camera can be implemented in many ways. The simplest design is an array of ordinary cameras very close to each other (see Fig. 1.5 or [158]) or one could use specialized optics or lens systems such as described in [2, 42, 99]. In Section 7.1, some examples of these designs are presented.

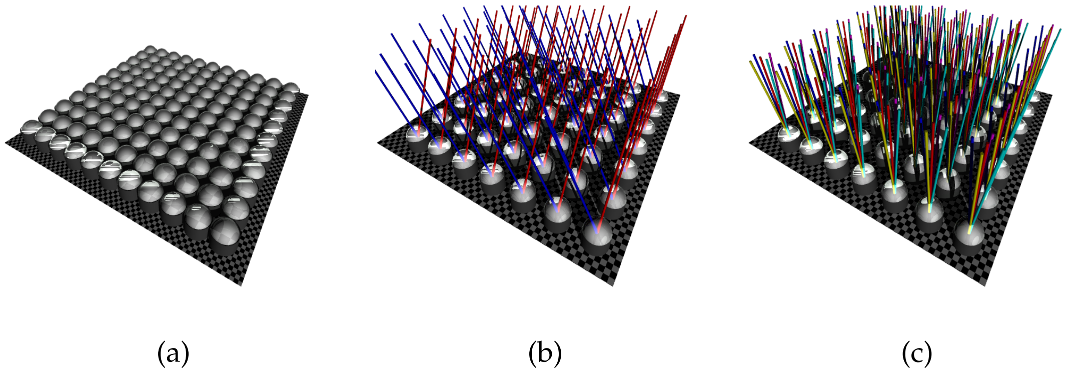


Figure 1.5: (a) Design of a Polydioptric Camera (b) capturing Parallel Rays and (c) simultaneously capturing Pencil of Rays.

Whatever design one uses, it is not possible to capture the plenoptic function with arbitrary precision. If we want to use the plenoptic motion constraints, we need to reconstruct a continuous light field from discrete samples. In this thesis I study the implementation of a polydioptric camera using a regular array of densely spaced pinhole cameras. This problem has been studied in the context of light field rendering [27, 162] where the authors examined which rays of a densely captured light field need to be to retained to reconstruct the continuous light field. In chapter 4 we will examine how we can estimate the approximation error between the time-varying light field and a reconstruction based on the images captured. Specifically, we determine how we can compute an estimate of the approximation error given a description of the camera and the statistics of the scene.

1.5 Overview of the thesis

Inspired by how Argus, the hundred-eyed guardian of Hera, the goddess of Olympus, alone defeated a whole army of Cyclops, the mythical monocular giants, and the impressive navigational feats of insects with compound eyes, I will show in this thesis how one can analyze vision problems in the space of light rays, and thereby find solutions that are optimal with a respect to the joint design of vision sensors and algorithms. Based on the geometrical and statistical properties of these spaces, I introduce a framework to systematically study the relationship between the shape of an imaging sensor and the task performance of the entity using this sensor. I illustrate this concept by analyzing the structure of the time-varying plenoptic function and identifying the important camera design parameters for the case of 3D motion estimation. This analysis is then used to form a hierarchy of camera designs with regard to the task under study. The hierarchy implies that large field of view polydioptric cameras are the optimal cameras for 3D ego motion estimation because they allow for stable and scene-independent motion estimation algorithms. Polydioptric cameras are generalized cameras which capture a multi-perspective subset of the plenoptic function. Using a combination of multi-dimensional sampling analysis, statistical image modeling I then improve upon the qualitative hierarchy of camera design by defining a metric on the space of cameras. This metric is based on how well a given polydioptric camera is able to capture the plenoptic function, and how the approximation errors propagate through the motion estimation to the final parameter estimate. This allows us to determine the best camera arrangement for a task.

Although camera motion estimation is important for tasks in navigation, it is often only part of the information that we want to extract about the world. Thus, at the end of this thesis I also examine the problem of estimating the shape and 3D motion of non-rigidly moving objects in a scene using a distributed camera network.

Chapter 2

Plenoptic Video Geometry: The Structure of the Space of Light Rays

2.1 Preliminaries

All the visual information that can be captured in a volume of space by an imaging sensor is described by the intensity function defined on the space of light rays surrounding us. This function is known as the plenoptic function[1] [1]. For each position in space it records the intensity of a light ray for every direction, time, wave length, and polarization, thus providing a complete description of all uninterpreted visual information. The idea of a function that contains all images of an object was already described by Leonardo Da Vinci in his notebooks. Similar functions were used later by Mehmke in 1898 and by Gershun in 1936 to describe the reflection of light on objects. Gershun was the first one to use the term light field for the vector irradiance field [49]. One can find more about the historical study of the space of light rays in the book by Moon and Spencer about the scalar irradiance field [98].

The time-varying shapes of the objects in a scene, their surface reflectance properties, the illumination of the scene, and the transmittance properties of the ambient space

[1] From the Greek words *plenus* full and *optic* view.

determine the structure and properties of the visual space. The objects cannot transmit their shape and surface properties directly to an observer. Without actively interacting with the object, the observer can only record the intensities of a set of light rays that reflect or emit from the object surface and infer the properties of the objects in the scene based on these ”images”. This is possible for example by utilizing the geometric ray model of light transport which relates the geometric and reflection properties of the object surface and the illumination to the observed image intensity values. An image can be defined as a collection of rays with a certain intensity where each ray captured has a position, orientation, time, wavelength and domain of integration (scale). Such a *ray element* was called a raxel by Grossberg and Nayar [52]. In computer graphics the scene parameters are known and the goal is to generate the view of a scene from a given view point by determining the ray properties for all the rays that make up an image (e.g. using standard computer graphics techniques such as ray tracing and global illumination). The quality that can be achieved is often nearly indistinguishable from actual views as evidenced by the ubiquitous use of computer generated imagery in today’s cinema.

In comparison, computer vision attempts to extract a description of the world based on images, thus one could say that computer graphics and computer vision are ”inverse problems” of each other. We are given the images of a scene (in general a scene in the real world) and based on the information in the images, we would like to find the set of variables that describe the scene. Since it is impossible to find an infinite number of variables, we will use models to approximate the true scene parameters and to recreate the scene. The model parameters can be found based on the images captured given some a priori assumptions about the scene. In 3D photography we use the samples of the visual space captured by the cameras to determine the most likely set of variables that give rise to the observed structure of the visual space. For segmentation we assign labels corresponding to different objects in the scene or for recognition we try to find the best label for an image region given a set of classes. To solve these tasks we have to analyze how the scene and object properties manifest themselves in the global and local structure of the plenoptic function.

The structure of the visual space is generated by the objects and their properties in the scene. We have a distance function $\mathcal{D}(\boldsymbol{x};t) : \mathbb{R}^3 \times \mathbb{R}^+ \rightarrow \mathbb{R}$ defined on the four-dimensional spatio-temporal volume (three space dimensions and one time dimension) that describes the space that is occupied by the objects in the scene. The surface of the objects is defined by the zero level set of the distance function, that is $\mathcal{S} := \{(\boldsymbol{x};t)|\mathcal{D}(\boldsymbol{x};t) = 0\}$. The geometric properties of this function describe the shape of the objects and their temporal evolution. We will use the term object surface to denote this zero level set $\mathcal{S}$.

The change of the distance function over time $d\mathcal{D}/dt$ captures only deformations of the shape along its normal which is given by

$$\nabla_x \mathcal{D}(\boldsymbol{x};t) = \begin{pmatrix} \partial\mathcal{D}(\boldsymbol{x};t)/\partial x_1 \\ \partial\mathcal{D}(\boldsymbol{x};t)/\partial x_2 \\ \partial\mathcal{D}(\boldsymbol{x};t)/\partial x_3 \end{pmatrix} \text{ where } \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \tag{2.1}$$

We can also define a vector field $\mathcal{M}(\boldsymbol{x};t) : \mathcal{S} \rightarrow \mathbb{R}^4$ on the object surface that describes the trajectories of unique points on the object surface, the 3D motion flow or scene flow [152]. Usually we restrict the time component of this vector field to be of unit magnitude (or whichever time step is small enough to allow an accurate first-order description of the deformations of shape and surface properties). This vector field is restricted to transport points only on the surface $\mathcal{S}$, therefore we have the constraint $[\mathcal{M}^{\mathrm{T}}\nabla\mathcal{D}] = 0$

where $\nabla\mathcal{D} = [\nabla_x\mathcal{D}; \mathcal{D}/\partial t]$.

On the surface $\mathcal{S}$ we have the reflection properties of the objects given by the surface light field $\mathcal{L}_{\mathcal{S}} : \mathcal{S} \times \mathbb{S}^2 \times \mathbb{R}_+ \rightarrow \Gamma$. Γ denotes here the spectral energy, and equals $\mathbb{R}$ for monochromatic light, $\mathbb{R}^n$ for arbitrary discrete spectra, or could be a function space for a continuous spectrum. $\mathbb{S}^2$ is the unit sphere of directions in $\mathbb{R}^3$.

If the object is not emitting any light, then a surface light fields can often be factored into a number of physically motivated components such as an illumination component and a component describing the surface reflection. A popular representation in graphics expresses the radiance $\mathcal{L}_{\mathcal{S}}(\boldsymbol{x}; \boldsymbol{r}; t)$ leaving the surface in direction $-\boldsymbol{r}$ in terms of the surface irradiance $I_{\mathcal{S}}(\boldsymbol{x}; \boldsymbol{m}; t) = \mathcal{L}(\boldsymbol{x}; \boldsymbol{m}; t)(\boldsymbol{n}^{\mathrm{T}}\boldsymbol{m})$ measured from direction $\boldsymbol{m}$ through the rendering equation [72]

$$\mathcal{L}_{\mathcal{S}}(\boldsymbol{x}; \boldsymbol{r}; t) = \int_{H(n)} B(\boldsymbol{x}; \boldsymbol{r}; \boldsymbol{m}; t)\ I_{\mathcal{S}}(\boldsymbol{x}; \boldsymbol{m}; t) d\boldsymbol{m} \tag{2.2}$$

where $B(\boldsymbol{x}; \boldsymbol{r}; \boldsymbol{m}; t)$ is the *bidirectional reflection distribution function*(BRDF) [108] defined on the surface $\mathcal{S}$. $H(\boldsymbol{n})$ is the hemisphere of directions $H(\boldsymbol{n}) = \{\boldsymbol{m} : \|\boldsymbol{m}\| = 1 \text{ and } \boldsymbol{m}^{\mathrm{T}}\boldsymbol{n} > 0\}$.

Another more complex example is the *Bidirectional Surface Scattering Reflectance Distribution Function* [108], in short BSSRDF. The BRDF only describes the interaction of light that enters and exits the surface at the same point, therefore it does not describe the scattering of light inside the surface as seen in marble, milky fluids or human skin. The BSSRDF is thus parameterized by two location vectors, one for the entrance and one for the exit location of the light rays. For some nice examples that were rendered using this model see [69]. Even such a general function is not able to adequately describe all the surface reflection phenomena, since we left out the effect of polarization for example. In computer vision we usually only use simple reflection models, because the noise in the image acquisition process makes it difficult to find the parameters of the more complex models under non-laboratory conditions. In computer vision, we are more interested in a representation that can describe the intrinsic complexity of a reflection model. A nice measure to assess the complexity of a surface light field is described in Jin et al. [71] where they describe the concept of the radiance tensor, that is the matrix that is constructed from the observed intensities of a number of surface points in a local neighborhood on the object surface that are observed from a number of viewing positions. Each column contains the different intensity values of the scene points as seen from a single view point, while the rows of the matrix contain the different views of a single scene point. If the local surface patch is small compared to the distance to the observing cameras and to the light sources in the scene, then this radiance tensor will have a rank of two or less. This is easy to show. Due to the small distance between the points in comparison to the distance to the light sources and cameras, we can assume that the surface irradiance is the same for each point in the patch, that the vector from each scene point to a single camera center is also nearly constant, and that the object surface is locally planar so that the normals and tangent directories are also constant across the points. Then we can factor the surface light field observed at each point as

$$\mathcal{L}_{\mathcal{S}}(\boldsymbol{x}, \boldsymbol{r}) = \sum_{i=1}^{N} \rho_i(\boldsymbol{x}, \boldsymbol{r}_0) B_i(\boldsymbol{x}_0, \boldsymbol{r}) \tag{2.3}$$

where N is usually smaller then 4. This decomposition was also used before by Chen et al. in the context of surface light field compression [28].

The simplest example of this factorization is the case of Lambertian reflectance of the surface with albedo $\rho(\boldsymbol{x}; t)$ which describes a perfectly diffuse reflector. In this case, we can conveniently express $\mathcal{L}_{\mathcal{S}}(\boldsymbol{x};\boldsymbol{r};t)$ for all directions $\boldsymbol{r} \in H(-\boldsymbol{n})$ by

$$\mathcal{L}(\boldsymbol{x};\boldsymbol{r};t) = \rho(\boldsymbol{x};t)[\boldsymbol{n}(\boldsymbol{x};t)^{\mathrm{T}}\boldsymbol{s}(\boldsymbol{x};t)] \tag{2.4}$$

where $\boldsymbol{s}(\boldsymbol{x};t) = \int_{H(\boldsymbol{n})} \mathcal{L}(\boldsymbol{m};\boldsymbol{x};t)(\boldsymbol{n}^{\mathrm{T}}\boldsymbol{m})d\boldsymbol{m}$ is the net directional irradiance on the surface point, given by the surface integral over $H(\boldsymbol{n})$, (for more details see [62]).

If we factor the popular diffuse plus specular model we will get:

$$\begin{aligned}
\mathcal{L}_{\mathcal{S}}(\boldsymbol{x};\boldsymbol{r}) &= \int_{H(n)} B(\boldsymbol{x};\boldsymbol{r};\boldsymbol{m})\ I_{\mathcal{S}}(\boldsymbol{x};\boldsymbol{m})d\boldsymbol{m} \\
&= \int_{H(n)} (\rho_d(\boldsymbol{x}) + \rho_s(\boldsymbol{x})B(\boldsymbol{x}_0;\boldsymbol{r};\boldsymbol{m}))\ I_{\mathcal{S}}(\boldsymbol{x}_0;\boldsymbol{m})d\boldsymbol{m} \\
&= \rho_d(\boldsymbol{x}) \int_{H(n)} I_{\mathcal{S}}(\boldsymbol{x}_0;\boldsymbol{m})d\boldsymbol{m} \\
&+ \rho_s(\boldsymbol{x}) \int_{H(n)} B(\boldsymbol{x}_{\mathbf{0}};\boldsymbol{r};\boldsymbol{m})\ I_{\mathcal{S}}(\boldsymbol{x}_{\mathbf{0}};\boldsymbol{m})d\boldsymbol{m} \\
&= \rho_d(\boldsymbol{x})B_d(\boldsymbol{x}_{\mathbf{0}}) + \rho_s(\boldsymbol{x})B_s(\boldsymbol{r};\boldsymbol{x}_{\mathbf{0}})
\end{aligned} \tag{2.5}$$

This factorization indicates that under the correct assumptions the radiance tensor has only rank two. This rank constraint can then be used as a plausibility criterion to decide if the scene parameters are estimated correctly.

A realistic model for the reflection properties that can be split in a diffuse and a specular component is given by Ward's anisotropic (elliptical) Gaussian model [156]. At every point on the surface we can locally define at each location $\boldsymbol{x}$ an orthonormal coordinate system $\boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{n}$ and define a half-way vector $\boldsymbol{h}$ between the incoming $\boldsymbol{m}$ and outgoing rays $\boldsymbol{r}$: $\boldsymbol{h} = (\boldsymbol{m}+\boldsymbol{r})/\|\boldsymbol{m}+\boldsymbol{r}\|$. Then we can write the BRDF for this model as:

$$\begin{aligned}
B(\boldsymbol{x};\boldsymbol{r};\boldsymbol{m}) &= \rho_d(\boldsymbol{x}) + \rho_s(\boldsymbol{x})B_s(\boldsymbol{x};\boldsymbol{r};\boldsymbol{m}) \qquad (2.6) \\
&= \frac{\rho_d(\boldsymbol{x})}{\pi} + \rho_s \frac{\exp\left[\left(1 - \frac{1}{(\boldsymbol{h}^{\mathrm{T}}\boldsymbol{n})^2}\right)\left(\left(\frac{\boldsymbol{h}^{\mathrm{T}}\boldsymbol{t}_1}{\alpha_1}\right)^2 + \left(\frac{\boldsymbol{h}^{\mathrm{T}}\boldsymbol{t}_2}{\alpha_2}\right)^2\right)\right]}{4\pi\alpha_1\alpha_2\sqrt{(\boldsymbol{m}^{\mathrm{T}}\boldsymbol{n})(\boldsymbol{r}^{\mathrm{T}}\boldsymbol{n})}}
\end{aligned}$$

where ρ_d is the diffuse reflection coefficient, ρ_s is the specular reflectance coefficient and α_1 and α_2 are the standard deviations of the microscopic surface slope (as a measure of surface roughness) in the tangent directions $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$. These 4 parameters together define a physically valid reflection model.

2.2 Representations for the space of light rays

2.2.1 Plenoptic Parametrization

We define the light field as the extension of the surface light field, which is only defined on the object surfaces $\mathcal{S}$, to the free space of $\mathbb{R}^3$, that is $\{\boldsymbol{x}|\mathcal{D}(\boldsymbol{x}, t) > 0\}$.

We will assume that the ambient space is a transparent medium, such as air for example, which does not change the color or intensity of light along its path. Thus, the light field along the view direction $\boldsymbol{r}$ is constant and the following equations hold:

$$\mathcal{L}(\boldsymbol{x}; \boldsymbol{r}; t) = \mathcal{L}(\boldsymbol{x} + \lambda\boldsymbol{r}; \boldsymbol{r}; t)\ \forall\lambda \text{ s.t. } \mathcal{D}(\boldsymbol{x} + \mu\boldsymbol{r}) > 0\ \forall\mu \in [0, \lambda] \tag{2.7}$$

which implies that

$$\nabla_{\boldsymbol{x}}\mathcal{L}^{\mathrm{T}}\boldsymbol{r} = \nabla_{\boldsymbol{r}}\mathcal{L}^{\mathrm{T}}\boldsymbol{r} = 0\ \forall\boldsymbol{x} \in \mathbb{R}^3, f(\boldsymbol{x}; t) > 0 \tag{2.8}$$

Here $\nabla_{\boldsymbol{x}}\mathcal{L}$ and $\nabla_{\boldsymbol{r}}\mathcal{L}$ are the partial derivatives of $\mathcal{L}$ with respect to $\boldsymbol{x}$ and $\boldsymbol{r}$:

$$\nabla_{\boldsymbol{x}} = (\partial/\partial x_1, \partial/\partial x_2, \partial/\partial x_3)^{\mathrm{T}}\text{ , and}$$

$$\nabla_{\boldsymbol{r}} = (\partial/\partial r_1, \partial/\partial r_2, \partial/\partial r_3)^{\mathrm{T}}.$$

To initialize this "differential equation" we define the following equality on the object surface

$$\mathcal{L}(\boldsymbol{x}; \boldsymbol{r}; t) = \mathcal{L}_{\mathcal{S}}(\boldsymbol{x}; \boldsymbol{r}; t) \text{ where } (\boldsymbol{x}; t) \in \mathcal{S}. \tag{2.9}$$

Since $\mathcal{L}$ is constant along $\boldsymbol{r}$, the set of functions $\mathcal{L}_{\mathcal{S}}$ defined on $\mathcal{S}$ generate the structure of the visual space, that is the geometry of the plenoptic function, completely.

Due to the brightness invariance along the ray direction the plenoptic function in free space reduces locally to five dimensions – the time-varying space of directed lines for which many representations have been presented (for a comparison of different representations see Camahort and Fussel [22] or the book by Pottmann and Wallner [118] about line geometry). If we deal with smog, fog, transmitting or partially transparent objects, then this invariance will not hold, but we will not consider these cases in this work.

The dependence of the structure of the plenoptic space on the shape of the scene objects can be analyzed using a directional distance function that describes the shortest distance between a point location and a scene surface in a given direction:

$$\mathcal{Z}(\boldsymbol{x};\boldsymbol{r};t) = \min_{\lambda} \mathcal{D}(\boldsymbol{x} + \lambda \boldsymbol{r};t) = 0 \tag{2.10}$$

Using this function, we can rewrite the equality Eq. (2.9) as

$$\mathcal{L}(\boldsymbol{x},\boldsymbol{r},t) = \mathcal{L}_{\mathcal{S}}(\boldsymbol{x} + \mathcal{Z}(\boldsymbol{x},\boldsymbol{r},t)\boldsymbol{r},\boldsymbol{r},t) \tag{2.11}$$

The space of light rays can now be parameterized in a number of different ways which offer different advantages with regard to completeness, redundancy and ease to describe geometric transformations. The question of representation is very important because the space of light rays has a specific structure since it is parameterized by the geometric and signal properties of the objects in the scene which introduce redundancy in the representation. In addition since all physical measurements of light are done with a finite aperture, we always need to account for the specific scale of the measurement in our later computations. This introduces a continuity on the space of measurements which also needs to be reflected in the space and metric that we will use for computations in the space of light rays. We call the parametrization that we used in this section the *plenoptic* parametrization because it is possible to assign a brightness value for each location, direction and time, thus capturing the full, global structure of the light rays.

2.2.2 Plücker coordinates

Sometimes we are interested in a representation for the space of light rays that does not exhibit the redundancy intrinsic to the plenoptic parametrization. As said in the last section, since light rays do not change their color in a transparent medium like air, we can represent light rays in the space of directed lines in three-dimensional space without losing any information. Linear subspaces are best treated in the context of Grassmann coordinates [118]. The Grassmann coordinates for lines in three-dimensional space are known as Plücker coordinates and are a very useful to describe lines and their motions in space. The Plücker coordinates of the line $l := \{\boldsymbol{x} + \lambda \boldsymbol{r}, \lambda \in \mathbb{R}\}$ are given by the tuple $(\boldsymbol{r}, \boldsymbol{m}) \in \mathbb{R}^6$, containing the direction vector $\boldsymbol{r}$ and the moment vector $\boldsymbol{m}$ defined by $\boldsymbol{m} = \boldsymbol{x} \times \boldsymbol{r}$. We see that the Plücker moment is the normal to the plane that contains the line and the origin, and its magnitude is equal to the distance of the line to the origin. If we normalize $\boldsymbol{r}$ to unit length, $\|\boldsymbol{r}\| = 1$, then we have a representation for an oriented line. This representation was described bye Study [17] to describe the space of oriented lines. Since the norm of all the line elements in the Study representation is of unit length, one also says the that all the coordinates representing oriented lines lie on the Study sphere.

This parametrization is equivalent in terms of its sampling properties in line space to the sphere-plane representation described in [23]. Here the line l is defined by its direction $\boldsymbol{r}$ and the intersection of l with the plane perpendicular to $\boldsymbol{r}$ through the origin. Since

$\boldsymbol{r} \cdot \boldsymbol{m} = 0$ by definition, $\boldsymbol{m}$ lies in this plane, although not on the line. The intersection point $\boldsymbol{x}$ of the line l with the plane $\boldsymbol{x} \cdot \boldsymbol{r} = 0$ is given by $\boldsymbol{x} = \boldsymbol{m} \times \boldsymbol{r}$.

Comparing the plenoptic and the Plücker parameterizations, we see that, locally, for each plane through the origin of the coordinate system perpendicular to a fixed direction $\boldsymbol{r}$, the plane $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{r} = 0$, they differ only by a rotation of 90 degrees around $\boldsymbol{r}$ since $\boldsymbol{m} = \boldsymbol{r} \times \boldsymbol{x}$. Thus we can define an intensity function on the space of Plücker lines using the identity:

$$L(\boldsymbol{m}, \boldsymbol{r}, t) = \mathcal{L}((\boldsymbol{m} \times \boldsymbol{r}, \boldsymbol{r}, t)$$

$\mathcal{L}$ can be used to describe the plenoptic function *globally* while L can be used to describe the plenoptic function *locally*, since it chooses a single plane to describe rays and thus cannot account for occlusions or the changes in the differential structure of the plenoptic function due to a motion of the observer. For example when an observer moves along the main view direction then the image expands/contracts radially. Such a forward motion does not change the brightness of the light ray along the axis of forward motion since we have the brightness invariance along the ray, but it changes the magnitude of the derivatives $\nabla_r \mathcal{L}$ and $\nabla_r L$ since they depend on the distance to the scene. In this case we have $\nabla_r \mathcal{L}(\boldsymbol{x}, \boldsymbol{r}, t) = \mathcal{Z}(\boldsymbol{x}, \boldsymbol{r}, t) \nabla_x \mathcal{L}(\boldsymbol{x}, \boldsymbol{r}, t)$ and $\nabla_r L(\boldsymbol{m}, \boldsymbol{r}, t) = \mathcal{Z}(\boldsymbol{m}, \boldsymbol{r}, t) \nabla_m L(\boldsymbol{m}, \boldsymbol{r}, t)$.

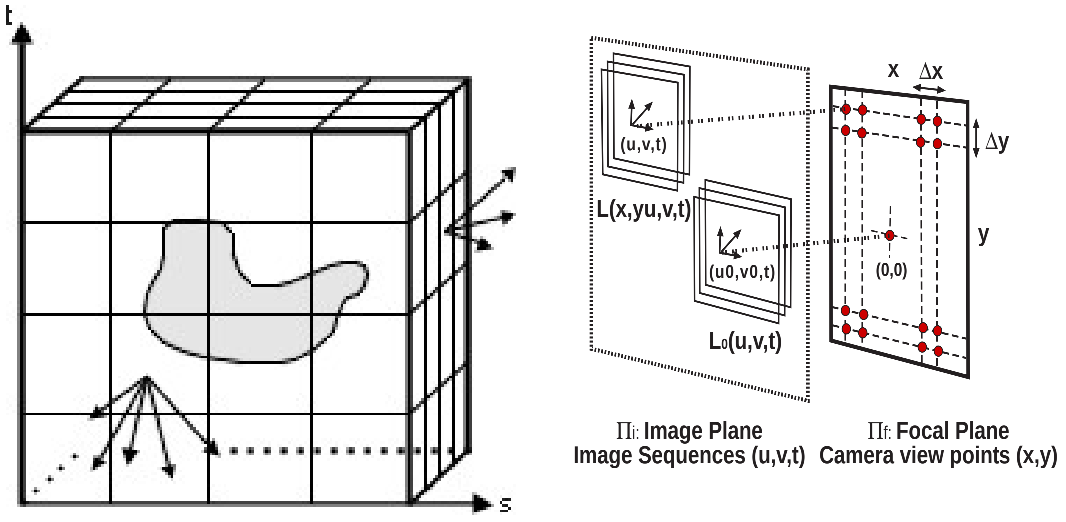
The main advantage of the Plücker parameterization is that we can express the motion of lines in space very elegantly in terms of a matrix multiplication. If a line is undergoing a rigid motion described by the rotation matrix R and the translation $\boldsymbol{t}$ around the origin of the coordinate system, then this can be expressed by multiplying the line coordinate vector $\boldsymbol{l} = (\boldsymbol{r}, \boldsymbol{m})$ with a 6×6 motion matrix Q:

$$\begin{pmatrix} \boldsymbol{r}' \\ \boldsymbol{m}' \end{pmatrix} = \begin{pmatrix} R & 0 \\ -[\boldsymbol{t}]_x R & R \end{pmatrix} \begin{pmatrix} \boldsymbol{r} \\ \boldsymbol{m} \end{pmatrix} = Q \begin{pmatrix} \boldsymbol{r} \\ \boldsymbol{m} \end{pmatrix}. \tag{2.12}$$

This motion equation can be derived easily by moving two points on the line rigidly in space and then recomputing the Plücker line coordinates with respect to the new position of the two points.

2.2.3 Light field parameterization

One difficulty with the previous two parameterization is that although we can define a metric between lines, it is difficult to express this distance directly in terms of distances between the line coordinates since they do not lie in a Euclidean space, but on a non-linear manifold. The light field parameterization is a convenient representation for the space of light rays, because it is closed under affine combinations of coordinate vectors and we can approximate the geodesic distances between points on the non-linear manifold of lines by the Euclidean distance between the coordinate vectors that represent the lines in the light field parameterization. As described by Peternell and Pottman [114], we can divide the space of lines into caps of directions, that are sets of directions close to a central axis c_i that defines each cap i. We can then choose two parallel planes Z_i^+ and Z_i^- perpendicular to the central axis c_i of each cap and represent all lines not perpendicular c_i by their intersection with the two planes Z_i^+ and Z_i^-. This defines now an affine space where the distance between two lines in 3D can be approximated by the Euclidean point distance between the affine coordinates of the intersections of the lines with the two planes. These new coordinates are equivalent to the coordinates of the line directions after being stereographically projected onto a plane. The different caps are glued together using interpolation weights so that we end up with a continuous representations. The extent of the caps should be chosen such that we do minimize the error in the directional distances between the rays. This two-plane parameterization was used by [51, 80] to represent the space of light rays. All the lines passing through some space of interest can be parameterized by surrounding this space (that could contain either a camera or an object) with two nested cubes and then recording the intersection of the light rays entering the camera or leaving the object with the planar faces of the two cubes. We only describe the parameterization of the rays passing through one pair of faces, the extension to the other pairs of the cube is straight forward. Without loss of generality we choose both planes to be perpendicular to the z-axis and separated by a distance of f. We denote one plane as *focal plane* Π_f indexed by coordinates (x, y) and the other plane as *image plane* Π_i indexed by (u, v), where (u, v) is defined in a local coordinate system with respect to (x, y) (see Fig. 2.1a). Both (x, y) and (u, v) are aligned with the (X, Y)-axes of the world coordinates and Π_f is at a distance of Z_Π from the origin of the world coordinate system.



(a) (b)

Figure 2.1: (a) Parameterization of light rays passing through a volume in space by recording the intersection of the light rays with the faces of a surrounding cube. (b) Lightfield Parameterization.

This enables us to parameterize the light rays that pass through both planes at any time t using the tuples (x, y, u, v, t) and we can record their intensity in the time-varying light field $L(x, y, u, v, t)$.

Since the properties of the generating functions determine the intensity (or color) distribution in the space of light rays as described by the plenoptic function, we can construct a catalog of the relations between these properties and the local differential structure of the plenoptic function. This catalog then relates all the observable visual events such as colors, textures, occlusions, and glares to the shape, surface reflectance and illumination of the scene. In this section we will study the plenoptic function directly without any reference to an imaging system. The properties of imaging systems will be studied later in chapter 3.

2.3 Information content of plenoptic subspaces

Recently, computer graphics and computer vision took interest in non-perspective subsets of the plenoptic function to represent visual information to be used for image-based rendering. Some examples are light fields [80] and lumigraphs [51], multiple centers of projection images [120] which have been used in cell animation already for quite some time [159], or multi-perspective panoramas [113]. For an overview over the use of multi-perspective images for image-based rendering see Zhang and Chen [163].

Non-perspective images have also been used by several researchers to reconstruct the observed scene from video sequences (for some examples see [14, 132, 26]). The descriptions of non-perspective images have been formalized lately by the work of Swaminathan et al [141] and Hicks [59, 58]. For more information you can also see the page of catadioptric sensor design `http://www.mcs.drexel.edu/~ahicks/design/design.html`.

In their seminal paper [1] Adelson and Bergen demonstrate with examples from early vision, how the importance of different subspaces of the space of light rays is reflected in the resolution by which a subspace is sampled and the amount of processing that is applied to them. A similar criterion should be applied to design of artificial sensors. Adelson and Bergen defined a set of ”periodic tables for early vision” where they relate first and second order derivative measurements in different two-dimensional subspaces to visual events in the world. In this section we will extend their results by examining how the correlation between different dimensions can be utilized to improve the imaging process.

If we compute the gradient of the spatio-temporal lightfield ∇L, we can analyze the eigenvalue structure of the tensor that is formed by the outer product of the gradients to extract information about the scene (see [56] for the analysis for perspective images).

2.3.1 One-dimensional Subspaces

If we study one-dimensional subspaces, we can measure the change in static radiance for linearly varying orientations and constant view points, or linearly varying view points and a fixed view direction. If we fix both view direction and view orientation and only the temporal dimension is varying, then we can measure the change in irradiance of a light ray over time. It is to note that the time domain is different from the spatial domain, because it only allows for causal filtering, since we cannot make measurements in the future. Linear subspaces where two or more dimensions are simultaneously varying allow us to fixate on a specific point in the world and measure a slice of the surface light field at that point.

2.3.2 Two-dimensional Subspaces

In two dimension things become interesting because we can observe correlated behavior between different domains which allows us to extract more complex information about the world.

Texture Information

The subsets $L(x, y) = L(x, y, \cdot, \cdot, \cdot)$ or $L(u, v) = L(\cdot, \cdot, u, v, \cdot)$ correspond to the standard perspective or orthographic images which enable us to measure the colors and texture of all the visible surfaces of the objects in the scene. The complexity of the textures can be classified by looking at the eigenvalues λ_1 and λ_2 of $\nabla L \nabla L^{\mathrm{T}}$. If both the eigenvalues are zero then the scene consists of a homogeneous region. If only one eigenvalue is non-zero, then we have a linear structure such as a brightness edge in the texture, and if both eigenvalues are large, then the surface texture is too complex to be described by first order derivative operators. An example would be a texture that is isotropically changing, or one that consists of a corner or a point feature. Each point in the world is only imaged by a single light ray, thus for a given number of imaging elements we get the most information about the surface textures in the world, but a single static image does not give us any information about the shape of the objects in the world.

Shape and Segmentation Information

Any set of light rays that lies on a doubly ruled surface allows us to infer spatial information about the scene since the light rays intersect in space [112, 129]. Especially the case where the orientation vector and the position vectors vary in the same plane is easy to analyze. These subsets of light rays are known as epi-polar plane images (EPIs) [15]. Examples are $L(x,u) = L(x,\cdot,u,\cdot,\cdot)$ or $L(y,v) = L(\cdot,y,\cdot,v,\cdot)$ and will be analyzed in more detail in Section 2.5. Since the points in the world are imaged by multiple imaging elements we can relate the spatial arrangement of the imaging elements to the correlation between visual measurements at different elements to extract spatial information about the scene in view. Since we use the correlation of textural elements to infer spatial structure we cannot always extract spatial information, for example if we compute the structure tensor in an EPI and we have only vanishing eigenvalues, then this corresponds to a texture-less region of the scene where we cannot extract any shape information. Each line in an EPI corresponds to the light passing through a single point in space. If this point lies on the scene surface, then the changes in color and intensity are due to reflection properties at this scene point and should vary slowly (unless it is specular). If the line corresponds to a scene point that does not lie on the surface of an object then it corresponds to a virtual pinhole camera image of the scene and the intensities can vary arbitrarily. If the reflection properties of the scene texture are dominated by the Lambertian component, that means that the light intensity reflected is independent of the angle at which we look at the surface, then if we fixate on a scene point by varying the rate of change of view point and view direction appropriately, the observed radiance of the scene point will not change and thus form a line in the EPI of uniform color. Thus for most scenes with non-specular reflection properties, we can identify the lines corresponding to scene points by finding the lines where the intensity varies only very little.

The slope of this line structure encodes the depth of the scene because the depth is proportional to the ratio of the necessary changes in view direction and view position to fixate on a scene point. This equivalent to triangulating the scene points from a continuum of view positions where correspondence is established by following the tracks of uniform color formed by the projection of a scene point using different centers of projection. We can use the structure tensor to estimate the depth accurately. If we have one vanishing eigenvalue indicating an edge in the EPI, then the depth of the scene can be computed from the direction of the eigenvector corresponding to the non-vanishing eigenvalue. The changes in slope across the EPI can be used to infer properties of the surface shape such as surface orientation and surface curvature.

In case that both eigenvalues are non-zero, then we have either a specularity or an occlusion in the scene. The difference in dimensionality of the image structure between texture and occlusion discontinuities can be used to easily segment the image captured with the sensor into regions belonging to different objects in the scene. Depth occlusions form y-junctions in the EPI and the angle of the junction can be used to differentiate between self-occlusions due to surface curvature where the intersection angle between the tracks of different features is very small and object occlusions where the angle is larger (corresponding to the magnitude of difference in depth between the objects).

Linear Motion

Any two-dimensional subset that involves simultaneously time and one of the spatial dimensions allows us to detect and measure the planar motion of the camera or an object in the scene (see Table 2.1). For perspective views (u, t) or (v, t), we can measure the two components of the motion along the optical axis and the coordinate axis, but have to estimate the depth of the scene at the same time, while for orthographic views we can only measure the displacement along the spatial dimension, but we can do this accurately without any additional depth estimation.

2.3.3 Three-dimensional Subspaces

Structure from Motion: Simultaneous 3D Motion and Depth Estimation

If we the take conventional perspective image sequences we can estimate the 3D motion of the camera and the depth of scene, a problem known as structure from motion. This problem has been studied in depth and many solutions have been proposed. Unless we have pure rotation or are imaging a plane scene, this estimation is often very difficult since the temporal changes in the image depend not just on the camera and object motions, but also on the depth structure of the scene(Tables 2.1 and 2.2). This forces us to solve the motion and depth segmentation simultaneously with the motion estimation leading to a high dimensional problem which is very sensitive to noise.

In this case the dynamic images $L(x, y, t) = L(x, y, \cdot, \cdot, t)$ or $L(u, v) = L(\cdot, \cdot, u, v, t)$ are the subsets of interest. We can have the following cases for the eigenvalues $\lambda_1, \ldots, \lambda_3$ of the structure tensor $\nabla L \nabla L^{\mathrm{T}}$. As before, if all eigenvalues are of equal value (either non-zero or zero) we cannot extract any information, since the scene is either too homogeneous or too random. If we have 1 non-zero eigenvalue, we have the motion of a line in the image, resulting in a plane in spatio-temporal space. Here we can only determine the component of the motion parallel to the edge in the image, the normal flow, due to the aperture problem. If we have two non-vanishing eigenvalues, then this corresponds to the motion of a point feature along a line in spatio-temporal space and the eigenvector corresponding to the vanishing eigenvector is parallel to the direction of the line. For the orthographic images, this flow is corresponding to the actual motion of the scene point, while in the perspective images, this motion also depends on the distance to the scene and the motion along the optical axes.

Scene-independent Planar Motion Estimation and Motion Segmentation

If we look at the three-dimensional subspaces that are formed by $L(x, u, t)$ and $L(y, v, t)$, then we can extract the motion of a camera in the epipolar plane in a scene independent manner (see Table 2.2). Since changes in the epipolar image over time due to the camera motion form only a low three-dimensional subspace parameterized by the motion parameters it is easy to detect independently moving objects since their trajectory will give rise to gradients in the EPI that are not compatible with the estimated rigid motion. In the future I plan to study if subspace factorizations in the spirit of Tomasi and Kanade [146], Irani [67], or Vidal et al. [155] can be utilized to make this factorization.

Depth Estimation from Parallax

If we capture a plenoptic subset formed by three of the four spatial dimensions (known as epipolar volumes), then we can determine the depth of the scene in all the epipolar planes contained in the plenoptic subset as long the scene texture offers enough information. In regions of homogeneous color, it is of course impossible to recover information about the scene. Besides depth we can again recover all the information as described in the Section 2.3.2. As before if the local orientation of the scene texture is parallel to the epipolar plane then we cannot estimate the depth because in this case the epipolar lines and the texture line coincide. This fact which is a well-known problem in the stereo literature.

2.3.4 Four-dimensional Subspaces

Shape Estimation and Recovery of Surface Properties

In 4D, we can take a look at the 4D light field which contains information about the reflectance of a scene point as well as the depth and occlusion properties, thus allowing for computations that utilize the fact that all the dimensions constrain each other. We know for example, that all the gradients in the images should be parallel, but intensity derivatives in view direction space are scaled by the depth with respect to the intensity derivatives in view point space as described before. If we look at the different eigenvalue distributions, we see that again we have the non-informative cases of completely vanishing or non-vanishing eigenvalues corresponding to the absence of textures or the presence of too much noise or strong specularity.

Motion Stereo Estimation

Any three spatial dimensions and a temporal dimension allows us to compute the motion of the camera and the objects in the scene. Since we can compute the depth of the scene from the epipolar plane images, motion estimation and depth estimation are decoupled leading to a simpler problem. We can also solve for some motion parameters linearly in terms of plenoptic derivatives and then plug the solution into the non-linear scene-dependent motion equations. Having access to epipolar images and thus depth also simplifies the segmentation of the scene into independently moving objects since their depth is constrained by the plenoptic derivatives.

2.3.5 Five-dimensional Subspaces

Scene-independent Motion Estimation

If we have access to the complete plenoptic function, then we can compute the motion of the camera in a scene-independent manner by solving a linear system of equations in terms of the view point and view direction derivatives. Since these gradient fields only depend on the six motion parameters, it will be easy to detect independently moving objects by a simple clustering procedure using an affine motion model to describe the displacement of a scene region in the plenoptic space.

2.4 The space of images

Of special educational interest are the two-dimensional subspaces of the plenoptic function, because most continuous imaging surfaces can only capture two-dimensional images since the receptors need to lie on a two-dimensional surface in space. Two-dimensional images also enable us to utilize the correlations between different coordinate axes to infer information about the scene. In general any 2D-subset of this lightfield function constitutes an image. Recently, Yu and McMillan [161] described how all linear cameras can be described by the affine combination of three points in the light field parameterization. Some of the cameras they describe are:

- If a location (x, y) in the focal plane is fixed so that the image is of the form $L(x, y, \cdot, \cdot, t)$, then it corresponds to the image captured by a perspective camera. This image is formed by the pencil of light rays that all pass through the point (x, y) in the focal plane.

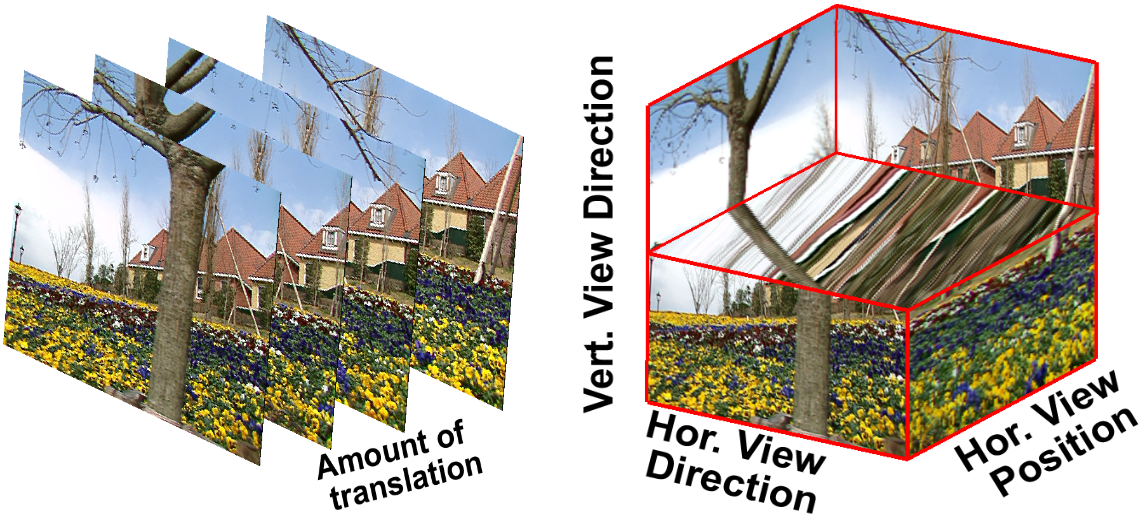
- If instead we fix the view direction (u, v), we capture an orthographic image $L(\cdot, \cdot, u, v, t)$

of the scene. In this case, all the light rays that form the image are parallel where the direction is given by the vector $(u, v, f)/\|(u, v, f)\|$.

- If we choose to fix the values on two orthogonal spatial axes, for example (x, v) or (y, u), then the subsets $L(x, \cdot, \cdot, y, t)$ and $L(\cdot, y, u, \cdot, t)$ describe linear push-broom-video sequences, where a one-dimensional perspective slit camera is moved perpendicular to the slit direction as for example in recordings made by surveillance planes. This camera is an example of the so-called crossed-slit projection camera as described by Zomet et al. [168] These images are formed by the set of rays passing through two slits lying in two planes parallel to the imaging surface.

The different kinds of images above are subsets of the plenoptic function where each light ray corresponding to an image pixel intersects the scene at a unique location. In other words this means that none of the light rays intersects another light ray that is captured in the image on the surface of an object in the scene. Given an image of the type described above, it is therefore impossible to infer something about the scene structure without prior knowledge.

In the following we will take a closer look at the subsets that correspond to light rays that intersect in space and therefore could potentially encode information about the structure. From [112, 129] we know that to form a stereo geometry, that means rays lie on an epipolar surface and intersect, the rays making up an image need to lie on a doubly-ruled quadric. The only such surfaces are planes, hyperboloids, and parabolic hyperboloids. An example for such a planar configuration are epipolar plane images. These images are formed by fixing two of the parallel axes in the two-plane parameterization, that is (x, u) or (y, v).



(a) (b)

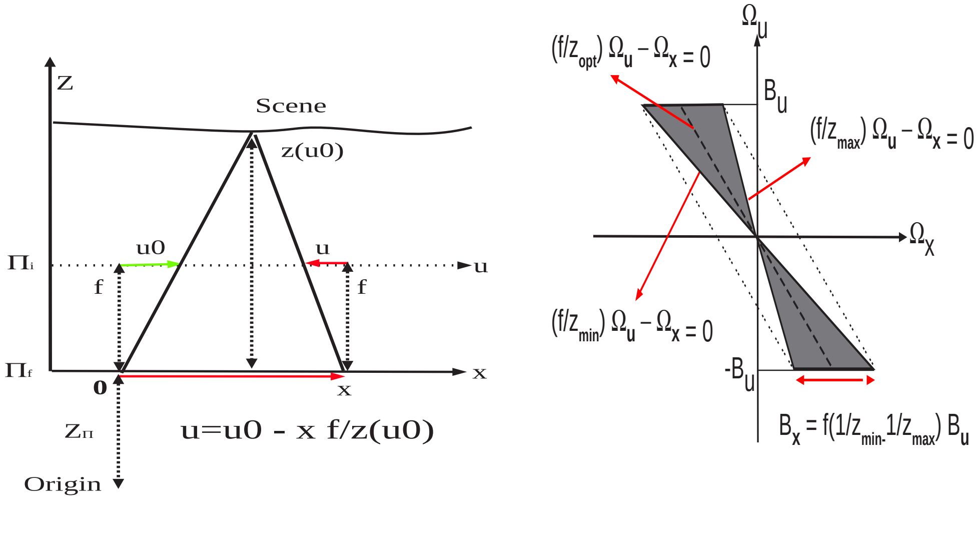
Figure 2.2: (a) Sequence of images captured by a horizontally translating camera. (b) Epipolar plane image volume formed by the image sequence where the top half of the volume has been cut away to show how a row of the image changes when the camera translates.

2.5 The Geometry of Epipolar Images

We are interested in characterizing the frequency structure of the time-varying plenoptic function captured by a moving polydioptric camera. Since visual information is in general local information, we will first study local neighborhoods of the static plenoptic function.

Most of the local structure of the space of light rays can already be described by examining two-dimensional slices, either perspective, orthographic or epipolar images. In this section we will focus on the epipolar images because in them the geometric structure and surface textures interact the strongest.

The rays in the epipolar image intersect either in free space, inside the object or on the object surface. These images exhibit a regular structure, as seen in the example in



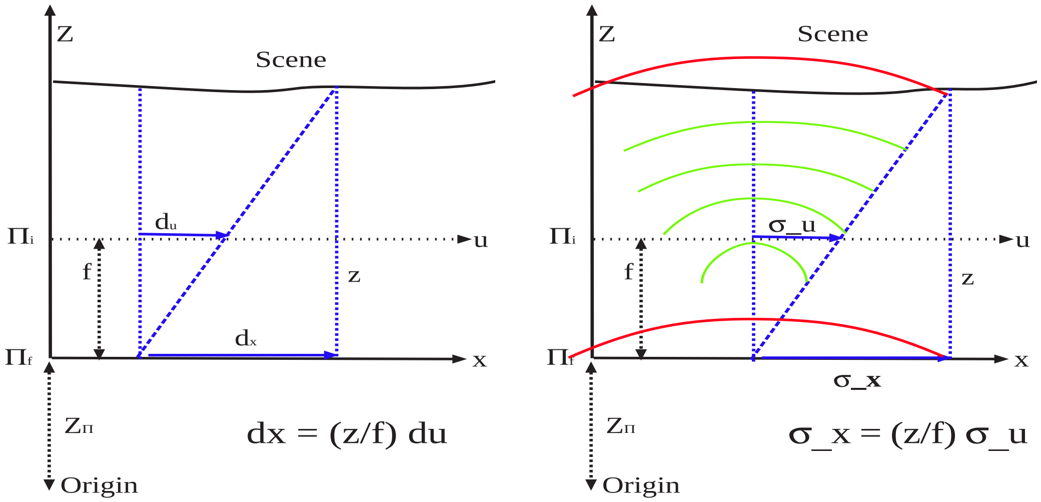
(a) (b)

Figure 2.3: (a) Light Ray Correspondence (here shown only for the light field slice spanned by axes x and u). (b) Fourier spectrum of the (x,u) light field slice with choice of "optimal" depth z_{opt} for the reconstruction filter.

Fig. 2.2b. The projection of a scene point traces out a linear line in the EPI with a slope is proportional to the depth of the scene point. Thus the differential structure of an epipolar plane image encodes the depth, as well as occlusions and the reflectance function of the surface in view.

We follow in our description [15] and analyze the motion of the image of a world point $\boldsymbol{x} = (x, z)$ in a moving line camera. If camera and world coordinate system coincide, we can apply the usual perspective projection, so that the image of the point in the image line $z = f$ is given by $u = f(x/z)$.

To describe the motion of the projected point in dependence on the camera motion for a general camera position and orientation, let $\boldsymbol{c} = (c_x, c_z)$ be the center of the camera and let the optical axis make an angle of θ with the x-axis of the coordinate system. The world coordinates of the point $\boldsymbol{x}$ are $\boldsymbol{x}_0 = (x_0, z_0) = R\boldsymbol{x} + \boldsymbol{c}$ where the rotation matrix



(a) (b)

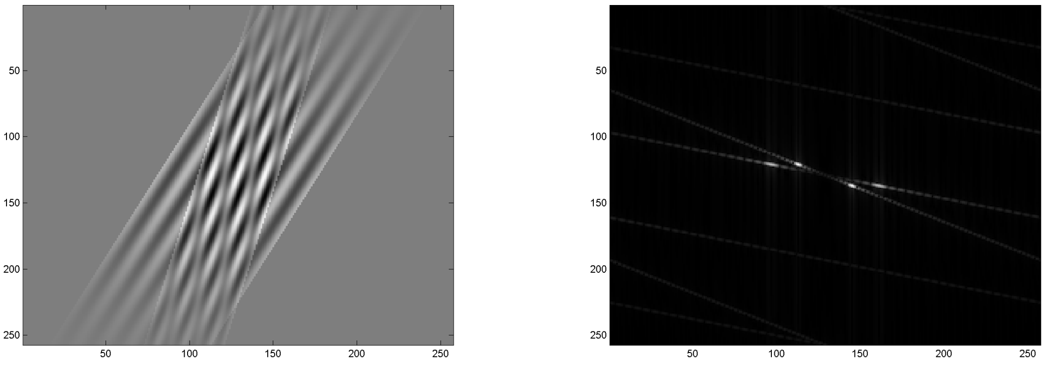
Figure 2.4: (a) The ratio between the magnitude of perspective and orthographic derivatives is linear in the depth of scene. (b) The scale at which the perspective and orthographic derivatives match depends on the depth (matching scale in red)

$R = [\cos(\theta), -\sin(\theta); \sin(\theta), \cos(\theta)]$ aligns the camera and world coordinate systems. This implies that $\boldsymbol{x} = R^{\mathrm{T}}(\boldsymbol{x}_0 - \boldsymbol{c})$, therefore the projection onto the image line in the general case is given by:

$$u = f\frac{(x_0 - c_x)\cos(\theta) + (z_0 - c_z)\sin(\theta)}{(c_x - x_0)\sin(\theta) + (z_0 - c_z)\cos(\theta)} \tag{2.13}$$

First, we will study the effect of a simple linear motion. Without loss of generality, we can assume that the camera moves with constant speed along the positive x-axis ($\boldsymbol{c} = (c_x, c_z) = (at, 0)$). This leads to

$$u = f\frac{(x_0 - at)\cos(\theta) + z_0\sin(\theta)}{(at - x_0)\sin(\theta) + z_0\cos(\theta)} \tag{2.14}$$



(a) (b)

Figure 2.5: (a) Epipolar image of a scene consisting of two translucent fronto-parallel planes. (b) The Fourier transform of (a). Notice how the energy of the signal is concentrated along two lines.

which we can expand to

$$
\begin{aligned}
0 = (a\sin(\theta))ut + (z_0\cos(\theta) - x_0\sin(\theta))u \\
+ (af\cos(\theta))t - f(x_0\cos(\theta) + z_0\sin(\theta))
\end{aligned}
\tag{2.15}
$$

which shows that the feature paths are hyperbolic curves. The asymptotes of the hyperbola are parallel to the coordinate axes as can be seen if we rewrite Eq.2.15 as:

$$
(u + f\cot(\theta))(t + (\cot(\theta) - x_0)/a) = \frac{fz_0}{a\sin(\theta)} \tag{2.16}
$$

It is to note, that it is always possible to linearize the feature paths by derotating the coordinate system, so that the viewing direction is perpendicular to the direction of motion. The transformation is given by:

$$
u' = f\frac{u\cos(\theta) - f\sin(\theta)}{f\cos(\theta) - u\sin(\theta)} \tag{2.17}
$$

If the viewing direction is perpendicular to the direction of motion ($\theta = 0$), then the feature paths degenerate to straight lines

$$z_0 u + aft - x_0 f = 0. \tag{2.18}$$

This is the canonical way to parameterize epipolar images. Each feature in the world traces out a line in the epipolar image with a slope that is proportional to the relative depth of the feature with respect to the camera.

This is also illustrated in Fig. 2.4as where we see that $dx = (z/f)du$ by the law of similar triangles. Thus, depth can be extracted from the feature paths simply by extracting the slope of a feature $z = f(dx/du)$.

2.5.1 Fourier analysis of the epipolar images

In contrast to Chai et al. [27] which parameterize the world with respect to a perspective reference image of the scene, we will follow Zhang and Chen [162] and will parameterize the 2D light field slice in terms of the planar surface light field $L_{\mathcal{S}}(\boldsymbol{x}, \theta),\ \boldsymbol{x} = [x, z]^{\mathrm{T}} \in \mathbb{R}^2, \theta \in [-\pi, \pi]$ where $\theta = 0$ corresponds to the direction opposite of the depth axis ($-\hat{\boldsymbol{z}} = [0, -1]$). The surface light field is only defined on the surface $\mathcal{S}$ of the objects which in this flat-land case is defined by the curve $\boldsymbol{x}(s)$ which we will parameterize using the arc-length parameter s.

The epipolar plane image is parameterized by its view point $\boldsymbol{x}_c = [x_c, 0]$ (we assume w.l.o.g. that the camera is at depth 0) and view direction, parameterized by the intersection u of the view ray with the image plane at distance f to focal plane. We also assume that the field of the camera is restricted to $u \in [-u_0, u_0]$.

In the following we will examine the effect of depth, slope, reflection properties and occlusions on the frequency structure of the epipolar image. Specifically, we will look at the following scenarios:

1. a single fronto-parallel plane with Lambertian surface reflection,
2. a single fronto-parallel plane with band-limited non-Lambertian surface reflection,
3. a single tilted plane,
4. and one fronto-parallel plane occluding a second one.

From our previous definition, the view ray has the implicit equation $[f, -u] \cdot (\boldsymbol{x} - \boldsymbol{x}_c) = 0$ or if multiply the equation out, we get

$$xf - x_c f - uz = 0. \tag{2.19}$$

If the normal to the surface and the viewing ray do not point along the same direction in the field of view of the camera, that is $\boldsymbol{N} = [u, f] \cdot \nabla\mathcal{D}(\boldsymbol{x}(u))^{\mathrm{T}} < 0 \ \forall u \in [-u_0, u_0]$, then we do not have to worry about self-occlusions on the surfaces. Here $\boldsymbol{x}(u) = \boldsymbol{x}_c + z(\boldsymbol{x}_c, u) * u/f$ is the intercept of the view ray at pixel u as seen from camera position $\boldsymbol{x}_c$.

2.5.2 Scenes of constant depth

In this case the equation of the object curve is given by $\boldsymbol{x}_s = [s, z_0]^{\mathrm{T}}$ and we can easily solve for the intersection of the viewing ray with the surface in terms of s which leads to the relations

$$s = u\frac{z_0}{f} + x_c \text{ and } \theta = -\tan^{-1}(u/f) \tag{2.20}$$

Thus the local light field structure of an EPI is given by the following equation,

$$L(x_c, u) = L_{\mathcal{S}}(x_c + \frac{z_0}{f}u, -\tan^{-1}(u/f)) \tag{2.21}$$

In this case the occlusion condition is satisfied for all non-horizontal viewing rays, and we do not have to worry about self-occlusions.

We can compute the depth of the scene from the image derivatives as follows, to simplify the computation, we will use the approximation $\tan^{-1}(u/f) \approx u/f$.

$$
\begin{aligned}
L(x_x, u) &= L_{\mathcal{S}}(x_c + \frac{z_0}{f}u, -\tan^{-1}(u/f)) = L_{\mathcal{S}}(x_c + \frac{z_0}{f}u, -u/f) \\
\frac{\partial}{\partial x_c} L(x_c, u) &= \frac{\partial}{\partial s} L_{\mathcal{S}}(x_c + \frac{z_0}{f}u, -u/f) \\
\frac{\partial}{\partial u} L(x_c, u) &= \frac{\partial}{\partial s} L_{\mathcal{S}}(x_c + \frac{z_0}{f}u, -u/f)\frac{z_0}{f} - \frac{\partial}{\partial \theta} L_{\mathcal{S}}(x_c + \frac{z_0}{f}u, -u/f)\frac{1}{f}
\end{aligned}
$$

If we examine the ratio between the EPI derivatives

$$
\begin{aligned}
\frac{\partial}{\partial u} L(x_c, u) / \frac{\partial}{\partial x_c} L(x_c, u) &= \frac{\frac{\partial}{\partial s} L_{\mathcal{S}}(\ldots, \ldots)\frac{z_0}{f} + \frac{\partial}{\partial \theta} L_{\mathcal{S}}(\ldots, \ldots)\frac{1}{f}}{\frac{\partial}{\partial s} L_{\mathcal{S}}(\ldots, \ldots)} \\
&= \frac{z_0}{f} + \frac{\frac{\partial}{\partial \theta} L_{\mathcal{S}}(\ldots, \ldots)\frac{1}{f}}{\frac{\partial}{\partial s} L_{\mathcal{S}}(\ldots, \ldots)} \qquad (2.22)
\end{aligned}
$$

we see that the ratio between the derivatives is proportional to the depth of the scene if the scene consists of a fronto-parallel plane and we can neglect the influence of the non-Lambertian effects (see Figure 2.4a). If the derivatives of the reflectance function are involved, then we have to separate the influence of reflection and depth on the brightness derivatives.

In the Fourier domain, we can analyze the frequency spectrum of a scene at constant depth by expressing the light field spectrum in terms of the surface light field spectrum.

$$
\begin{aligned}
\hat{L}(x_c, u) &= \iint L(x_c, u) \exp(-2\pi i(\Omega_x x_c + \Omega_u u)) dx_c du \\
&= \iint L_{\mathcal{S}}(x_c + \frac{z_0}{f}u, -\tan^{-1}(u/f)) \exp(-2\pi i(\Omega_x x_c + \Omega_u u)) dx_c du
\end{aligned}
$$

We have the identities $u = -f\tan(\theta)$ and $x_c = s + z_0 \tan(\theta)$. Here we will make the approximation $\tan(\theta) \approx \theta$ for small θ (corresponding to a small field of view camera), thus we get

$$\hat{L}(x_c, u) = \iint L_{\mathcal{S}}(s, \theta) \exp(-2\pi i(\Omega_x(s + z_0\theta) - \Omega_u f\theta) f ds d\theta$$
$$= f\hat{L}_{\mathcal{S}}(\Omega_x, \Omega_x z_0 - \Omega_u f) \tag{2.23}$$

As described in [162], if we assume that the surface light field is bandlimited by B_θ, then we have the equality $\hat{L}_{\mathcal{S}}(\Omega_s, \Omega_\theta) = \hat{L}_{\mathcal{S}}(\Omega_s, \Omega_\theta) \cdot 1_{B_\theta}(\Omega_\theta)$ where $1_{B_\theta}(\Omega_\theta)$ is an indicator function over the interval $[-B_\theta, B_\theta]$. When we have a Lambertian surface, where $B_\theta = 0$, $\hat{1}_{B_\theta}(\Omega_\theta)$ equals the Dirac functional $\delta(\Omega_\theta)$ and we get the familiar fact that the frequency spectrum of an EPI observing a Lambertian surface at constant depth is a line in frequency space with a slope proportional to the depth of the surface in the spatial domain.

2.5.3 A slanted surface in space

The computations for a slanted surface are more complicated because the depth of the scene is changing when we vary either the view direction or the view position causing a fore-shortening effect of scene texture. This causes the frequency content of the projected texture to change with position, a phenomenon known as chirping [93]. A tilted line in space can be parameterized as $(x, z) = (x_s, z_s) + s(\cos(\phi), \sin(\phi))$, thus using Eq. (2.19) we can solve for s:

$$s = \frac{(x_s - x_c)f - z_s u}{\cos(\phi)f - \sin(\phi)u}. \tag{2.24}$$

We have to make sure that the occlusion property is satisfied for the field of view of the camera $[-u_0, u_0]$. Thus we have the constraint on ϕ such that $\cos(\phi)f - \sin(\phi)u < 0\ \forall u \in [-u_0, u_0]$.

Computing the Fourier transform of the light field we get

$$\begin{aligned}\hat{L}(x_c, u) &= \iint L(x_c, u) \exp(-2\pi i(\Omega_x x_c + \Omega_u u)) dx_c du \\ &= \iint L_{\mathcal{S}}(\frac{(x_s - x_c)f - z_s u}{\cos(\phi)f - \sin(\phi)u}, -\tan^{-1}(u/f)) \exp(-2\pi i(\Omega_x x_c + \Omega_u u)) dx_c du\end{aligned}$$

Again we solve these two equations for u and x_c (while again making the approximation $\tan(\theta) \approx \theta$):

$$\begin{bmatrix} u \\ \\ x_c \end{bmatrix} = \begin{bmatrix} -f\theta \\ \\ -s\theta[\sin(\phi)] - s\cos(\phi) + \theta z_s + x_s \end{bmatrix} = \Phi(s, \theta) \tag{2.25}$$

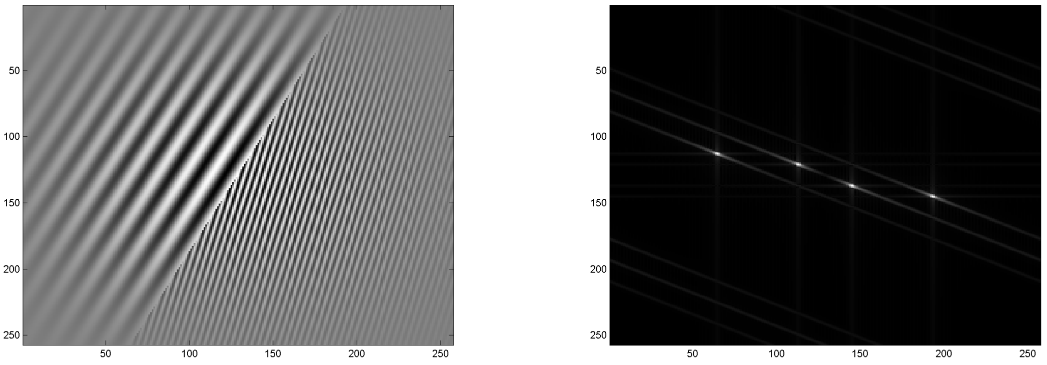
Computing the determinant of the Jacobian of the mapping $\Phi(s, \theta) = (u, x_c)$, we get $|\det(D\Phi(s, \theta))| = \cos(\phi) + \sin(\phi)\theta$ which leads to

$$\hat{L}(x_c, u) = \iint L(s, \theta) \exp(-2\pi i([\Omega_u, \Omega_x]\Phi(s, \theta) f(\cos(\phi) + \sin(\phi)\theta) ds d\theta. \tag{2.26}$$

Since s and θ appear as a product in $\Phi(s, \theta)$, in general we cannot easily additively factor Φ and express the spectrum of the epipolar image in terms of the spectrum of the surface light field. Nevertheless, if we can factor the surface light field $L_{\mathcal{S}}(s, \theta) = \rho(s)B(\theta)$ where $\rho(s)$ describes the texture on the plane, while $B(\theta)$ captures the view dependent effects, then we can simplify the equations as follows:

$$\begin{aligned}&\hat{L}(x_c, u) \qquad (2.27) \\ =&f \int \hat{\rho}(\Omega_x(\cos(\phi) + \sin(\phi)\theta)) B(\theta)(\cos(\phi) + \sin(\phi)\theta) \exp(-2\pi i(\Omega_x(\theta z_s + x_s) + \Omega_u f\theta) d\theta\end{aligned}$$

where $\hat{\rho}$ is the Fourier transform of the surface texture $\rho(s)$. By choosing a specific texture, we can then evaluate the integral and compute the spectrum of the epipolar image. We see how by varying θ, the frequency spectrum is getting spread out between the minimum and maximum depth.



(a) (b)

Figure 2.6: (a) Epipolar image of a scene where one signal occludes the other. (b) The Fourier transform of (a). Notice the ringing effects parallel to the occluding signal.

2.5.4 Occlusion

To estimate how occlusions influence the frequency structure of $L(x, u)$ we can use the approach described in [8]. They model an occlusion in a local neighborhood by splitting the light field $L(x, u)$ into two parts $L_+(x, u)$ and $L_-(x, u)$ using a binary indicator function $\Upsilon(x, u)$.

$$L(x, u) = \Upsilon(x, u)L_+(x, u) + [1 - \Upsilon(x, u)]L_-(x, u)$$

Applying the Fourier transform to $L(x, u)$ results in

$$\hat{L}(\Omega_x, \Omega_u) = \hat{\Upsilon}(\Omega_x, \Omega_u) * \hat{L}_+(\Omega_x, \Omega_u) - \hat{\Upsilon}(\Omega_x, \Omega_u)] * \hat{L}_-(\Omega_x, \Omega_u) + \hat{L}_-(\Omega_x, \Omega_u)$$

We see that due to the convolution the frequency spectra of $\hat{L}_+$ and $\hat{L}_-$ are spread by out the frequency spectrum $\hat{\Upsilon}$.

Since most of the geometrical information of a scene manifests itself only on the local scale, we can assume that locally every occlusion edge is a linear step edge. This case was studied by [8] where they looked at the Fourier space description of two moving textured planes where one plane is occluding the other. They studied the case of constant and linear motion, which basically corresponds to changing the viewpoint in the light field parameterization (constant motion case), or also changing the viewpoint along the optical axis (linear motion case). We follow their convention and model the texture of the scene as a signal that satisfies the Dirichlet conditions. This allows us to express the signal as an infinite exponential series that converges uniformly to the signal.

We define a step function Υ for one side of a fronto-parallel plane that begins at position s_+ at depth z_+ using the previous reparametrization as:

$$\Upsilon(x,u) = \Upsilon_s(x + \frac{z_+}{f}u) = \begin{cases} 1 & \text{if } x + \frac{z_+}{f}u \geq s_+ \\ 0 & \text{otherwise} \end{cases}$$

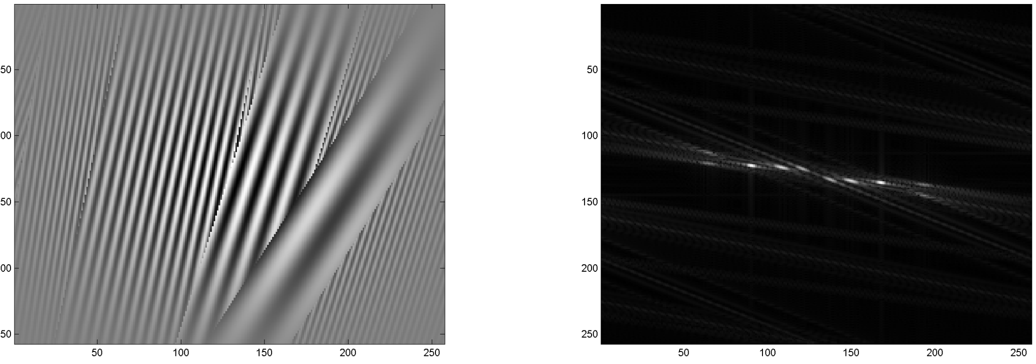
The Fourier transform of a step function is then given by:

$$\begin{aligned} \hat{\Upsilon}(\Omega_x, \Omega_u) &= \int\limits_{u=-\infty}^{\infty} \int\limits_{x=-\infty}^{\infty} \Upsilon(x,u) e^{-j(\Omega_x x + \Omega_u u)} dx du \\ &= \left(\pi\delta(\Omega_x) - \frac{i}{\Omega_x} \right) e^{-j\Omega_x x_i} \delta(\Omega_u - \Omega_x \frac{z_+}{f}) \end{aligned}$$

We see that the occlusion will cause ringing in the frequency domain that is parallel to the line $\Omega_u - \Omega_x \frac{z_i}{f} = 0$. The orientation of this line depends on the depth of the occlusion edge (Fig. 2.6).

Following [162] we can model more complicated scenes by partitioning the scene into n depth levels where each depth is assumed to have constant depth (fronto-parallel plane). The order of the depth levels be $z_1 < \ldots < z_n$. Each level extends until infinity and is only occluded by the layers in front of it closer to the camera. To model a finite extent for each depth layer, we define the mask M_i ($i = 1, \ldots, n$):

$$M_i(x,u) = \begin{cases} 1 & \text{if layer } i \text{ is blocking the path of light ray } \boldsymbol{x} \\ 0 & \text{otherwise} \end{cases}$$



(a) (b)

Figure 2.7: (a) Epipolar image of a complex scene with many occlusions.. (b) The Fourier transform of (a). There are multiple ringing effects parallel to all the occluding signals.

We will examine the epipolar images $L_i(x, u)$ of the layers separately one by one. The Fourier transform of each layer is denoted by $\hat{L}_i(\Omega_x, \Omega_u)$ and the Fourier transform of the occluded signal by $\hat{L}_i^o(\Omega_x, \Omega_u)$. The first layer is not occluded, thus its Fourier transform is identical to the unmasked Fourier transform, that is $\hat{L}_1^o = \hat{L}_1$. The second layer is occluded by the first, thus we have $L_2^o = L_2 \cdot M_1$ and its Fourier transform is $\hat{L}_2^o = \hat{L}_2 * \hat{M}_1$. We can do this now for every layer and we get $L_i^o = L_i \cdot \prod_{j=1}^{i-1} M_i$ and its Fourier transform is $\hat{L}_i^o = \hat{L}_i * \hat{M}_1 * \ldots * \hat{M}_{i-1}$. Using the results from before, we see that the Fourier transform of the occluded signal is concentrated along the line with slope z_i/f with ringing artifacts parallel to $z_1/f, \ldots, z_{i-1}/f$ (Fig. 2.7). We can also include the windowing effect due to the finite image size by choosing the first mask to be the size of the camera window which will lead to some ringing artifacts that are parallel to the coordinate axes. This also suggests that to reduce the ringing effects, we should window the local plenoptic neighborhood of interest.

If we now compute the convolution, between $\hat{\chi}_i$ and $\hat{L}_{\mathcal{S}_i}(\Omega_x)\delta(\Omega_u - \Omega_x \frac{z_i}{f})$, we get

$$\hat{\chi}_i(\Omega_x) * (\hat{L}_{\mathcal{S}_i}(\Omega_x)\delta(\Omega_u - \Omega_x \frac{z_i}{f})) \tag{2.28}$$

$$= \int e^{j(\Omega_x - \Omega_y)\frac{x_i^u + x_i^l}{2}} \frac{\sin(\frac{x_i^u - x_i^l}{2}(\Omega_x - \Omega_y))}{\Omega_x - \Omega_y} \hat{L}_{\mathcal{S}_i}(\Omega_y)\delta(\Omega_u - \Omega_y \frac{z_i}{f}) d\Omega_y \tag{2.29}$$

$$= e^{j(\Omega_x - \Omega_u \frac{f}{z_i})\frac{x_i^u + x_i^l}{2}} \frac{\sin(\frac{x_i^u - x_i^l}{2}(\Omega_x - \Omega_u \frac{f}{z_i}))}{\Omega_x - \Omega_u \frac{f}{z_i}} \hat{L}_{\mathcal{S}_i}(\Omega_u \frac{f}{z_i}) \tag{2.30}$$

and see how the occlusion again spreads out the Fourier spectrum of the texture component of the surface light.

2.6 Extension of natural image statistics to light field statistics

There exists a large amount of literature about the statistics of natural images which we can use to simulate an "average scene". The most noticeable result is that the power spectrum of a natural image falls approximately inversely proportional to the square of the spatial frequency. Dong and Atick [39] demonstrate how a similar scaling law can be derived from first principles for spatio-temporal sequences. We can use their formalism to find an expression for the power spectrum of a light field $|\hat{L}|^2$.

As we have seen in Chapter 2, the power spectrum $|\hat{L}|^2$ depends on the spatial frequencies of the textures in the scene, the orientations of the scene surfaces, as well as the depth and velocity distribution in the scene. For now we will disregard the effect of occlusions and assume that the power spectrum of a perspective image of the scene is rotationally symmetric, that means there is no special direction. For natural scenes this should be roughly satisfied, although it has been observed that horizontal and vertical orientations are often more predominant especially in man-made environments. Thus it is sufficient to find the power spectrum of a light field subspace formed by the axes u-x-t (time-varying epi-polar plane). The extension to the full 5D light field is straight forward.

For the power spectrum of the static perspective image row we can assume that it follows a power law $|\hat{L}_0(\Omega_u)|^2 = \frac{K}{\|\Omega_u\|^m}$ where $m \approx 2.3$ and K is a normalization constant [39].

As shown in Section 2.5, if we disregard occlusions the energy of the Fourier transform of an epipolar plane image (x-u-plane) observing an object at a constant depth z is concentrated along the line $\Omega_x = f\Omega_u/z$. Thus the power spectrum of the epipolar plane image of this scene is given by $|\hat{L}(\Omega_u)|^2\delta(\Omega_x - f\Omega_u/z)$. If the depth is varying, then the power spectrum will spread out to a wedge-shaped region bounded by the minimal and maximal depth (see Fig 2.3c).

For a given region in the world at depth z that moves relative to the camera with velocity $\dot{x}$, we have the brightness invariance of the form $I(u,t) = I_s(u - f\dot{x}t/z)$, thus the power spectrum of a perspective spatio-temporal image plane is concentrated along the line $\Omega_t = f\Omega_u\dot{x}/z$. For the characterization of the spectrum of an spatio-temporal image observing points following more complicated trajectories see [30]. Now we can write the power spectrum of the time-varying epipolar plane as follows:

$$|L(\Omega_x, \Omega_u, \Omega_t, z, \dot{\boldsymbol{x}})|^2 \tag{2.31}$$

$$= |L_u(\Omega_u)|^2\delta(\Omega_x - f\Omega_u/z)\delta(\Omega_t - f\Omega_u\dot{x}/z) \tag{2.32}$$

Given a probability distribution for the velocities $D_{\dot{x}}(\dot{x})$ and depths $D_z(z)$, we can express an ”average” light field power spectrum by integrating over these distributions:

$$|\hat{L}(\Omega_x, \Omega_u, \Omega_t)|^2 = \int_{-\infty}^{\infty}\int_{0}^{\infty} |L(\Omega_x, \Omega_u, \Omega_t, z, \dot{x})|^2 D_{\dot{x}}(\dot{x})D_z(z)d\dot{x}dz \tag{2.33}$$

$$= |L_u(\Omega_u)|^2 \cdot \int_{-\infty}^{\infty}\int_{0}^{\infty} \delta(\Omega_x - f\frac{\Omega_u}{z})\delta(\Omega_t - f\frac{\Omega_u\dot{x}}{z})D_z(z)D_{\dot{x}}(\dot{x})dzd\dot{x}$$

$$= \frac{K}{\|\Omega_u\|^m}D_{\dot{x}}(\frac{\Omega_t}{\Omega_x})D_z(f\frac{\Omega_u}{\Omega_x}) \tag{2.34}$$

since the integral is only non-zero when

$$z = f\Omega_u/\Omega_x \tag{2.35}$$

$$\dot{x} = (\Omega_t z)/(f\Omega_u) = \Omega_t/\Omega_x. \tag{2.36}$$

2.7 Summary

In this chapter we described the information content of the different subspaces of the plenoptic function. We saw that most of the information that relates the image information to the geometry of the scene is encoded in two-dimensional subspaces known as epipolar plane images. We then examined the frequency spectrum of these two-dimensional subspaces and saw that the energy of these images in the frequency domain is concentrated along regions in space that are determined by the depth of the scene. We will later utilize these results when we study the sampling problem for polydioptric cameras. In this thesis we will restrict our study to the dynamic plenoptic function observed by a rigidly moving observer. Since in chapter 5 we will focus specifically on this case, we will postpone the description of the details to that chapter.

Dim	Subspace	Object	Information Content
0D:		ray	-color
1D:	x,y	line of view points	○ intensity variation
	u,v	circle of view directions	○ intensity variation
	t	temporal changes	○ change detection
2D:	xy	orthographic image	○ metric measurements parallel to image plane ○ depth-independent foreshortened 2D-texture
	uv	perspective image	○ angle measurements around optical center ○ depth-dependent foreshortened texture
	xv,yu	push-broom image	○ cylindrical panorama
	xt,yt	video of parallel lines	○ motion parallel to sensor ○ temporal occlusion
	ut,vt	video of converging lines	○ radial motion ○ temporal occlusions
	xu,yv	epi-polar image	○ depth from gradients ○ occlusions from junctions

Table 2.1: Information content the axis-aligned plenoptic subspaces, Part I

Dim	Subspace	Object	Information Content
3D:	xyu,xyv	orthographic epi-polar volume	○ surface geometry
	xyu,xyv	perspective epi-polar volume	○ surface geometry
	xyt	orthographic video parallel lines	○ motion parallel to image plane
	uvt	pinhole camera video	○ motion around sphere of directions
	xvt,yut	push-broom video	○ motion parallel to line and around circle
	xut,yvt	epi-polar video	○ linear planar rigid motion estimation
4D:	xyuv	light field	○ reflection properties
	xyut,xyvt	orthographic epi-polar volume video	○ 3D motion recovery and segmentation
	xuvt,yuvt	perspective epi-polar volume video	○ 3D motion recovery and segmentation
5D:	xyuvt	light field video	○ complete scene information ○ linear metric motion estimation

Table 2.2: Information content of axis-aligned plenoptic subspaces Part II

Chapter 3

Polydioptric Image Formation: From Light Rays to Images

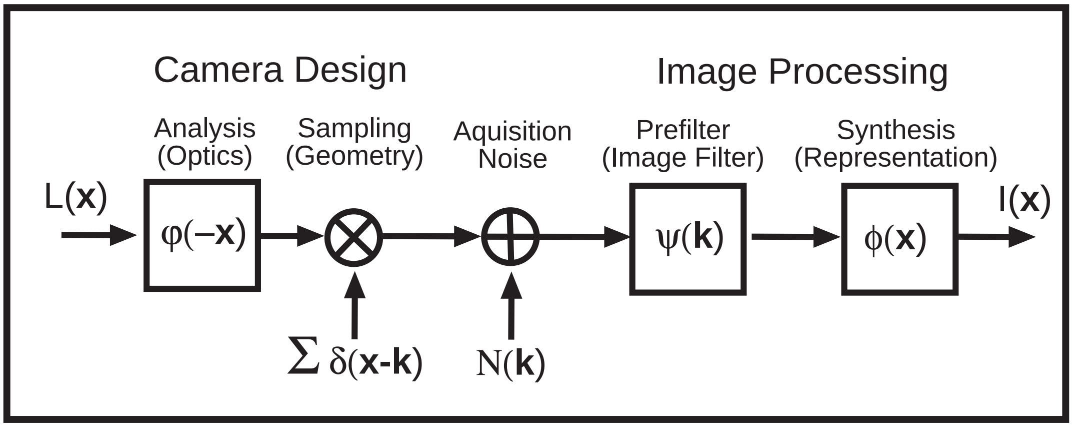


Figure 3.1: Imaging pipeline expressed in a function approximation framework

In this Chapter we will describe how the properties of the optics and the sensors affect the imaging process. Our model of the image formation pipeline for a polydioptric camera is based on the mathematical framework described in [150] and is summarized in the diagram in Fig. 3.1. The pipeline consists of the following five components:

1. **Optics of the lens system:** The radiometric and geometrical transformation from light rays in world coordinates to light rays that impinge on the sensor surface.
2. **Geometric distribution of the sensor elements:** This includes the summation of light over the sensor area and the geometric sampling pattern of the sensor element

distribution.

3. **Noise characteristics of the acquisition process:** The conversion from irradiance into an electrical signal, including shot noise due to the quantum nature of light, and noise due to the readout electronics and quantization of the measurements.

4. **Elementary processing on the image:** Filtering in the image domain to reduce the noise and prefilter the image for the reconstruction.

5. **Continuous representation of discrete measurements:** Interpolating the discrete signal to generate a continuous representation.

In the next sections we will formalize the five individual steps of the imaging process.

3.1 The Pixel Equation

The conversion of irradiance into a digital intensity value at a pixel location can be described by the *measurement equation* (also known as *pixel equation*). For a single pixel i the measured intensity I_i at time t_j is given by the following equation:

$$I_i(t_j) = \int\limits_{t \in T_j} \int\limits_{\boldsymbol{x} \in X_i} \int\limits_{\boldsymbol{r} \in A_i(x)} \mathcal{L}(G_x(\boldsymbol{x};\boldsymbol{r},t), G_r(\boldsymbol{x};\boldsymbol{r},t),t)P_i(\boldsymbol{x};t))d\boldsymbol{r}d\boldsymbol{x}dt. \tag{3.1}$$

In this equation $\boldsymbol{x}$ denotes the position vector on the imaging surface, $\boldsymbol{r}$ is the direction vector towards the lens system, and t is the time. This equation can be further generalized by including wavelength effects such as chromatic aberrations, but in this work we will not consider them further. $\mathcal{L}$ is the scene spectral radiance defined in object space, that is the usual plenoptic function. The geometric transformation of a light ray on its path through the camera system, that is the geometry of image formation, is described by the functions G_x and G_r. An example transformations would be the focusing effect of the lens. P models the behavior of the shutter and the sensor response and is dependent on position and time. The integration is done over the positions in pixel coordinates that make up the pixel area X_i, the directions that point towards the lens aperture as a function of the light ray origin $A_i(\boldsymbol{x})$, and the shutter interval T_j.

A polydioptric camera consists of a multitude of such pixels. The polydioptric sampling problem can be easier understood if we model the image formation as a sequence of convolutions on the space of light rays. We can rewrite the pixel equation by rewriting the rendering equation as a non-stationary filter operation in light ray space followed by a discrete sampling operation. The filter models the imaging acquisition process and the sampling pattern models the sensor geometry and distribution. The new form of Eq. (3.1) is then:

$$I_i(t_j) = \int\limits_{T_j}\int\limits_{X_i}\int\limits_{A_i(x)} \mathcal{L}(G_{\boldsymbol{x}}(\boldsymbol{x};\boldsymbol{r},t), G_{\boldsymbol{r}}(\boldsymbol{x};\boldsymbol{r},t),t)P_i(\boldsymbol{x};\boldsymbol{r};t))d\boldsymbol{r}d\boldsymbol{x}dt \tag{3.2}$$

$$= \int\limits_{\tilde{T}_j}\int\limits_{\tilde{X}_i}\int\limits_{\tilde{A_i}(x)} \mathcal{L}(\boldsymbol{x};\boldsymbol{r};t)P_i(G_x^{-1}(\boldsymbol{x},\boldsymbol{r},t), G_r^{-1}(\boldsymbol{x},\boldsymbol{r},t),t))d\boldsymbol{r}d\boldsymbol{x}dt \tag{3.3}$$

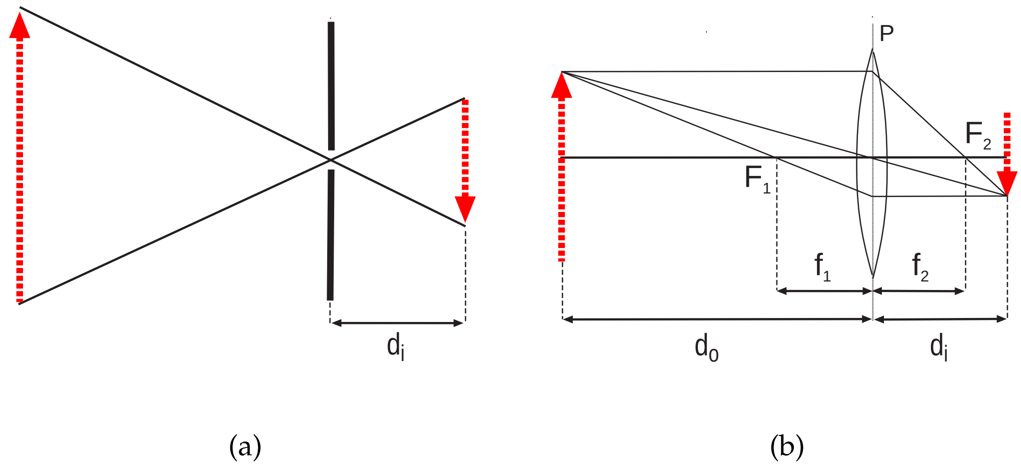
$$= \int\limits_{\mathbb{R}^3}\int\limits_{\mathbb{S}^2}\int\limits_{\mathbb{R}^+} \mathcal{L}(\boldsymbol{x};\boldsymbol{r},t)\cdot\psi(\boldsymbol{x}-\boldsymbol{x}_i,\boldsymbol{r}-\boldsymbol{r}_i,t-t_j)d\boldsymbol{r}d\boldsymbol{x}dt \tag{3.4}$$

where ψ is a function that integrates the effect of the integration limits, the geometric transformations G_x^{-1}, G_r^{-1}, as well as the optical effects captured by P.

3.2 Optics of the lens system

This section follows the exposition about how to model a correct and realistic camera model as presented in [76]. We will now describe in detail the different components of this model. We will assume that the coordinate system is located at the center of the image surface and that the axes of the coordinate system are aligned with the axes of the image. In this section we will describe two different idealizations of the lens system of a camera, the pinhole and the thin-lens model.

Pinhole Camera



(a)

(b)

Figure 3.2: (a) Pinhole and (b) Thin-Lens Camera Geometry

The simplest camera geometry is the pinhole camera model, which uses no lens at all. A point aperture is placed in front of the imaging surface, so only the rays through a single point in space pass through the aperture and fall on the imaging surface. Every point in space can be imaged in perfect focus (up to the diffraction limit) and we need only one parameter to describe this lens, that is the distance between the imaging surface and the lens center d_i, often also called the focal length f. If we choose the focal point as the origin of our fiducial coordinate system, then the projection of a point $\boldsymbol{P} \in \mathbb{R}^3$ onto the image plane results in the image point $\boldsymbol{x}$ which coordinates are given by the well-known equation

$$\boldsymbol{x} = -\frac{d_i \boldsymbol{P}}{\hat{\boldsymbol{z}} \cdot \boldsymbol{P}} = -\frac{f \boldsymbol{P}}{\hat{\boldsymbol{z}} \cdot \boldsymbol{P}} \tag{3.5}$$

where $\hat{\boldsymbol{z}}$ is the unit vector along the optical axis.

In this case the ray pixel equation becomes

$$I_i = \int_{(x,y)\in X_i} \int_{t\in T_j} L\left(\begin{bmatrix} x \\ y \\ -d_i \end{bmatrix} ; \begin{bmatrix} -x \\ -y \\ d_i \end{bmatrix}_{\mathcal{N}} ; t \right) P\left(\begin{bmatrix} x \\ y \\ -d_i \end{bmatrix} , t \right) \begin{bmatrix} dx \\ dy \\ dt \end{bmatrix} . \tag{3.6}$$

To simplify the notation we used the shorthand $[\boldsymbol{r}]_{\mathcal{N}} = \boldsymbol{r}/\|\boldsymbol{r}\|$ to describe vectors of unit length. We see that the geometric transformations of the light rays are very simple, that is $G_{\boldsymbol{x}}(\boldsymbol{x};\boldsymbol{r},t) = \boldsymbol{x}$ and $G_{\boldsymbol{r}}(\boldsymbol{x};\boldsymbol{r},t) = -[\boldsymbol{x}]_{\mathcal{N}}$, and the inverse transformations are given by $G_x^{-1}(\boldsymbol{x};\boldsymbol{r},t) = 0$ and $G_{\boldsymbol{r}}(\boldsymbol{x};\boldsymbol{r},t) = \boldsymbol{r}$.

Thin-lens Camera

In most cases the light gathering power of a simple pinhole camera is not sufficient (or it would necessitate an unreasonably long exposure time), therefore a lens is used to focus the light passing through a larger, finite aperture. Let's assume this aperture is circular and has a radius R_l.

For the thin-lens approximation, we assume that we have a lens of infinitesimally small extent along the optical axis, that means the two sides of the lens coincide in this idealization. The only parameters we need to know to describe this kind of camera are the focal length f of the lens and the distance d_i between the imaging surface and the lens center . Due to the focusing of the lens only one plane in object space at depth d_0 will be imaged in focus on the imaging surface. The relationship between d_0, d_i and f is given by the lens equation:

$$\frac{1}{d_0} + \frac{1}{d_i} = \frac{1}{f} \tag{3.7}$$

We see that the focal length of the camera is the distance from the distance at which an object at infinite depth will be imaged in focus. If the depth d of point $\boldsymbol{P}$ is at a depth different from d_0, then the point will be imaged on the imaging surface as a blurred circular patch, where the blur radius is given by the formula:

$$r_b = R_l d_i \left(\frac{1}{f} - \frac{1}{d_i} - \frac{1}{d_0} \right) \tag{3.8}$$

The blur effect due to defocus has been a popular method to recover depth in the literature(e.g., [127]). The fact that the distance between the intersection of the rays that pass through the periphery of the lens with the imaging surface and the intersection of the principal rays with the imaging surface depend on the depth of the point that is imaged is the basic idea behind the ″plenoptic cameras″ of Adelson and Wang [2] and "depth by optical differentiation″ method of Farid and Simoncelli [42].

The projection of a point in the world onto the imaging plane of a thin-lens camera is still described by Eq. (3.5), since the geometric aspects of the projection are completely defined by the principal ray. If this point is not lying on the plane of focus, then radiance leaving this point will be measured not just at a single location on the imaging surface, but over an extended area of the sensor.

Since the Eq. 3.8 is symmetric with respect to the distances of the lens to the imaging surface and the lens to the object, we can also go the other direction and use it to define what rays of the world are integrated to form the response at a single pixel. If we define a coordinate system on the lens using the coordinates u, v, and denote the set of locations on the lens by the set $A_L := \{(u, v) \in \{\|(u, v)\| \leq R_b\}$ then inside the camera we have the following expression for the pixel equation:

$$I_i = \int_{(x,y)\in X_i} \int_{(u,v)\in A_L} \int_{t \in T_j} L\left(\begin{bmatrix} x \\ y \\ -d_i \end{bmatrix} ; \begin{bmatrix} u-x \\ v-y \\ d_i \end{bmatrix}_{\mathcal{N}} ; t \right) P\left(\begin{bmatrix} x \\ y \\ -d_i \end{bmatrix}, t \right) d\boldsymbol{x} \tag{3.9}$$

Next, we have to determine what image is projected onto the lens. We know that the points at depth d_0 are imaged in perfect focus, thus the principal ray originating at $\boldsymbol{x} = [x, y, -d_i]^{\mathrm{T}}$ intersects the plane of focus at

$$\boldsymbol{x}_f = [-xd_0/d_i, -yd_0/d_i, d_0]^{\mathrm{T}} = [-\frac{xf}{d_i - f}, -\frac{yf}{d_i - f}, \frac{f}{d_i - f}]^{\mathrm{T}}. \tag{3.10}$$

Therefore the refracted direction of the ray that intersects the lens at position $\boldsymbol{x}_l = [u, v, 0]^{\mathrm{T}}$ is given by $\boldsymbol{r}_f = [\boldsymbol{x}_f - \boldsymbol{x}_l]_{\mathcal{N}}$. This leads to the following equation for the rays that are observed by a pixel:

$$I_i = \int_{(x,y)\in X_i} \int_{(u,v)\in A_L} \int_{t\in T_j} L\left(\begin{bmatrix} u \\ v \\ 0 \end{bmatrix}; \begin{bmatrix} -xf/(d_i - f) - u \\ -yf/(d_i - f) - v \\ d_i f/(d_i - f) \end{bmatrix}_{\mathcal{N}}; t\right) P\left(\begin{bmatrix} x \\ y \\ -d_i \end{bmatrix}, t\right) d\boldsymbol{x} \tag{3.11}$$

If the origin of the coordinate system is at the center of the lens, then for an imaging element at distance d_i from the lens, the plane at depth $d_o = fd_i/(f - d_i)$ will be in focus. We use this fact to rewrite the pixel equation for the thin-lens camera in the filter/sampling framework. Given the ray $(\boldsymbol{x}; \boldsymbol{r})$ with $\hat{z} \cdot \boldsymbol{x} = -d_i$ leaving the imaging element, then it will intersect the lens at location $\boldsymbol{x}_l = \boldsymbol{x} + \frac{d_i}{\hat{z}\cdot\boldsymbol{r}}\boldsymbol{r}$ and the plane of focus at $\boldsymbol{x}_f = -\frac{d_o}{d_i}\boldsymbol{x}$. Thus by setting $G_{\boldsymbol{x}}(\boldsymbol{x}; \boldsymbol{r}, t) = \boldsymbol{x}_l$ and $G_{\boldsymbol{r}}(\boldsymbol{x}; \boldsymbol{r}, t) = [\boldsymbol{x}_f - \boldsymbol{x}_l]_{\mathcal{N}}$ we can describe the geometric mapping between rays in object space and the rays leaving the sensor surface. Similar we can define the inverse functions.

3.3 Radiometry

Since many of the sources of noise are dependent on the intensity of the light falling on to the sensor, we need to analyze how the amount of light depends on the size of the sensor surface as well as the aperture of the lens. Sensor response is a function of exposure, the integral of irradiance at point $\boldsymbol{x}$ on the film plane over the time the shutter is open. If we assume constant irradiance over the shutter period (an assumption we have to relax later when using a moving sensor), then we have

$$I_s(\boldsymbol{x}; t_0) = \int_{t_0}^{t_0+T} E(\boldsymbol{x}; t) dt = E(\boldsymbol{x}; t_0)T \tag{3.12}$$

where $E(\boldsymbol{x}; t)$ is the irradiance at location $\boldsymbol{x}$ and time t, T is the exposure duration, and $H(\boldsymbol{x})$ is the exposure at $\boldsymbol{x}$ and time t_0. $E(\boldsymbol{x}; t)$ is the result of integrating the radiance at $\boldsymbol{x}$ over the solid angle subtended by the exit pupil, which is modeled as a disk. Denoting the set of locations in this disk by D we get

$$E(\boldsymbol{x}) = \int\limits_{\boldsymbol{x}' \in D} L\left(\boldsymbol{x}; \frac{\boldsymbol{x}' - \boldsymbol{x}}{\|\boldsymbol{x}' - \boldsymbol{x}\|}\right) \frac{\cos(\theta)\cos(\theta')}{\|\boldsymbol{x}' - \boldsymbol{x}\|^2} dA \tag{3.13}$$

where L is the plenoptic function , θ is the angle between the normal of the sensor surface and $\boldsymbol{x}' - \boldsymbol{x}$, θ' is the angle between the normal of the aperture stop and $\boldsymbol{x} - \boldsymbol{x}'$, and the differential area dA. If the sensor is parallel to the disk with d_i the axial distance between lens and sensor, then Eq. 3.13 can be rewritten as

$$E(\boldsymbol{x}) = \frac{1}{f^2} \int\limits_{\boldsymbol{x}' \in D} L\left(\boldsymbol{x}; \frac{\boldsymbol{x}' - \boldsymbol{x}}{\|\boldsymbol{x}' - \boldsymbol{x}\|}\right) \cos^4(\theta) dA. \tag{3.14}$$

The weighting in the irradiance integral leads to a varying irradiance across the imaging surface. If the exit pupil subtends a small angle from $\boldsymbol{x}$, we can assume that θ is constant and equal to the angle between $\boldsymbol{x}$ and the center of the disk. This leads to

$$E(\boldsymbol{x}) = L\frac{A}{f^2}\cos^4(\theta). \tag{3.15}$$

This constant multiplicative factor can easily be accounted for in the imaging process by rescaling the pixel intensity values accordingly.

3.4 Noise characteristics of a CCD sensor

In the description of the different noise processes, we follow Healey and Kondepudy [57]. Another good descriptions of all the elements involved in the imaging process can be found in [74] and [31]. CCDs convert light intensity linearly into a sensor response, but we must account for the black level, intensity scaling, vignetting, etc. in the camera system. An overview over the process can be seen in Fig. 3.3 (redrawn from [149]) that shows the image formation process.

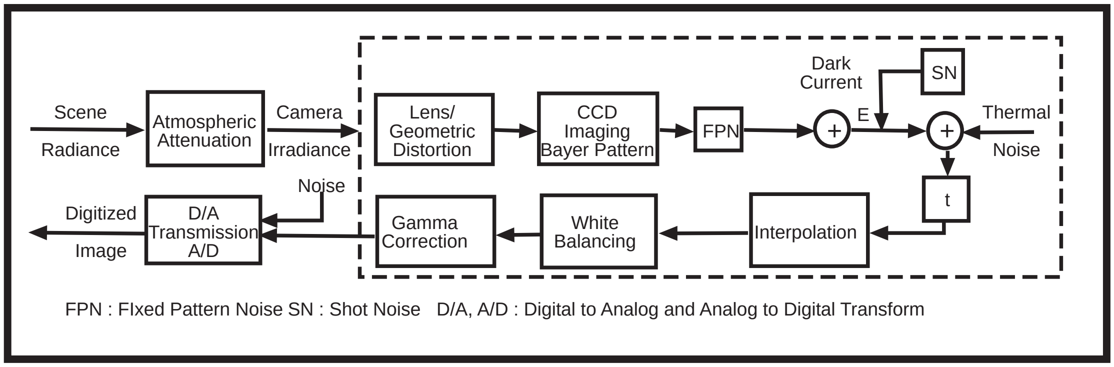


Figure 3.3: CCD Camera Imaging Pipeline (redrawn from [149])

The capture of irradiant light by a CCD sensor is analogous to measuring the amount of rainfall on a field by setting up a regular array of buckets to capture the rain per unit area during the storm and then measure it by carrying it one by one to a measuring unit. A CCD measures the amount of light falling on a thin wafer of silicon. The impact of a photon on the silicon creates an electron-hole pair (photo-electric effect). These free electrons are then collected at discretely spaced collection sites. Each collection site is formed by growing a thin layer of silicon dioxide on the silicon and depositing a conductive gate structure over the silicon oxide. If a positive electrical potential is applied to the gate, a depletion region is created that can hold the free electrons. By integrating the number of electrons stored at a collection site over a fixed time interval, the light energy is finally converted into an electronic representation.

A process called charge-coupling is used to transfer the stored charge from collection site to an adjacent collection site. The fraction of charge that can be effectively transported is known as the charge transfer efficiency of a device. An image is read out by transferring the charge packets integrated at a collection site in parallel along electron conducting channels that connect columns of collection sites. A serial output register with one element for each column receives a new row transfer after each parallel transport. In between parallel transfers, all the charge in the serial output register is transferred in sequence to an output amplifier that generates a signal proportional to the amount of charge. Once all the serial registers have been read out, the next parallel transfer happens. In analog video cameras the signal by the CCD is then transformed into an analog signal and converted into the digital domain by a frame grabber. In modern digital cameras, the signal from the CCD is directly digitized by the camera electronics and the frame grabber is not necessary anymore. In the next section, we will now examine the different sources of error during this process that one would like to calibrate for during the radiometric calibration step.

Let's start by examining the signal at a single collection site. The signal at this site is proportional to the number of electrons that arrive at the site and can be described as:

$$I = T \int\limits_{\lambda} \int\limits_{x \in X_i} E(x, y, \lambda) P(x, y) q(\lambda) dx dy d\lambda \tag{3.16}$$

Here I is again the intensity of the resulting signal, T is the integration time (assuming constant irradiance over time), $E(x, y, \lambda)$ denotes the spectral irradiance at given location of the collection site, that is the integral over all the rays that pass through the lens aperture as described in Section 3.2.

$$E(x, y, \lambda) = \int_{(u,v) \in A_l(x,y)} L(x, y, u, v, t) du dv. \tag{3.17}$$

$P(x, y)$ is the spatial response of the collection site, and $q(\lambda)$ describes the quantum efficiency of the device, that is the ratio of electron flux to incident photon flux in dependence on wavelength - which we had omitted in Eq. (3.2). We will now examine the different sources of noise in detail.

Fixed pattern noise (FPN)

Due to processing errors during CCD fabrication there exist small variations in quantum efficiency between different collection sites. Thus even if we have a completely uniform irradiance on the CCD array, we will get a non-uniform response across the array which is known as fixed pattern noise. Although this fluctuation in quantum efficiency often depends on wavelength, we will disregard this wavelength dependence for now. We model the number of electrons collected at site by KI, where I is defined as in Eq. (3.16) and K accounts for the difference between $q(\lambda)$ and $S(x, y)$ at the collection sites by scaling them appropriately. We assume that the mean of K is 1 and that the spatial variance is small and given by σ_K^2.

Blooming

Usually one assumes that the number of electrons collected at each site is independent of the number of electrons collected at neighboring sites. This can be violated if a site illuminated with enough energy to cause the potential well to overflow with charges which then contaminate the wells at other collection sites. This process is called blooming and in bad cases can affect many surrounding sites in the neighborhood of an overexposed site. Modern cameras build special walls between the sites to reduce these effects. For now we will assume appropriate illumination and thus disregard this effect.

Thermal energy

Thermal energy in silicon generates free electrons. These free electrons are known as dark current. They can also be stored at collection sites and therefore become indistinguishable from electrons due to light incidence. The expected number of dark electrons is proportional to the integration time T and is highly temperature dependent. Cooling a sensor can reduce the dark current substantially. The dark current also varies from collection site to collection site. We will denote the noise from the dark current as N_{DC}.

Photon shot noise

Shot noise is a result of the quantum nature of light and expresses the uncertainty about the actual number of electrons stored at a collection site. The number of electrons that interact with the collection site follows a Poisson distribution, so that its variance equals its mean. The probability to measure a given intensity I for the incoming radiance $\mathcal{L}$ is thus given by:

$$\mathcal{P}_{poisson}(\mathcal{I} = s) = \frac{\mathcal{L}^s}{s!} e^L \tag{3.18}$$

with a mean $\mathrm{E}[\mathcal{P}_{poisson}(\mathcal{I})] = \mathcal{L}$ and a variance $\mathrm{Var}[\mathcal{P}_{poisson}(\mathcal{I})] = \mathcal{L}$. We therefore see that the signal-to-noise-ratio is illumination and sensor size dependent and increases with the square-root of the intensity. Shot noise is a fundamental limitation and cannot be eliminated. Since the dark current increases the number of electrons stored at a site, it will increase the mean and thus the variance of the number of electrons stored. The number of electrons integrated at a collection site is given by

$$KI + N_{DC} + N_S \tag{3.19}$$

where N_S is the zero mean Poisson shot noise with a variance that depends on the number of collected photo-electrons KI and the number of dark current electrons N_{DC}.

Read-out noise

The charge transfer efficiency of a real CCD is not perfect and the noise due to charge transfer can be quantified. Modern CCDs though achieve transfer efficiencies on the order of 99.999% so that we can safely disregard this effect. During the next step, the on-chip output amplifier converts the charge collected at each site into a measurable voltage. This generates zero mean read noise N_R that is independent of the number of collected electrons. Amplifier noise will dominate shot noise at low light levels and determines the read noise floor of the CCD. The voltage signal can then be transformed into a video signal which is electronically low-pass filtered to remove high-frequency noise. The gain from both output amplifier and camera circuitry is denoted by A. For a digital camera, the conversion to video is not necessary, so A will only depend on the output amplifier. The video signal leaving the camera can then be described by

$$V = (KI + N_{DC} + N_S + N_R)A \tag{3.20}$$

Quantization error

To generate a digital image, the analog signal V needs to be converted to a digital signal. This can either be done using a frame grabber for analog video cameras or is done directly in the camera for digital cameras using an analog-to-digital (A/D) converter.

The A/D converter approximates the analog voltage V using an integer multiple of the quantization step q, so that each value of V that satisfies $(n-0.5)q < V < (n+0.5)q$ is rounded to the digital value $D = nq$ where n must satisfy $0 \leq n \leq 2^b$, where b is the number of bits used to represent D. To prevent clipping, q and b need to be chosen such that $\max(V) \leq (2^b - 0.5)q$. This quantization step can be modeled by adding a noise term N_Q that is a zero mean random variable independent of V and uniformly distributed over the range $[-0.5q, 0.5q]$ which results in a variance of $q^2/12$.

Another important influence on the sensor accuracy is the limited dynamic range of today's sensors which is a major source of error in many applications. Since only a limited range of signal magnitudes can be encoded without local processing on the chip we have to take into account the tradeoff between a high saturation threshold and a sufficient resolution at lower illumination levels. A detailed study of the benefits of high-dynamic-range imaging is beyond this thesis, and the reader is referred to the literature [92, 101].

Thus in conclusion, a comprehensive model for the image formation at a single CCD location is given by

$$D = V + N_Q = (KI + N_{DC} + N_S + N_R)A + N_Q \tag{3.21}$$

3.5 Image formation in shift-invariant spaces

Since all natural signals have finite energy, we can represent a given instance of the space of light rays using the light field parameterization as an element of $L_2(\mathbb{R}^5)$, the space of measurable, square integrable functions defined on $\mathbb{R}^5$, and phrase the light field reconstruction problem as a function approximation problem in $L_2(\mathbb{R}^5)$ using recent results in approximation theory [144, 13].

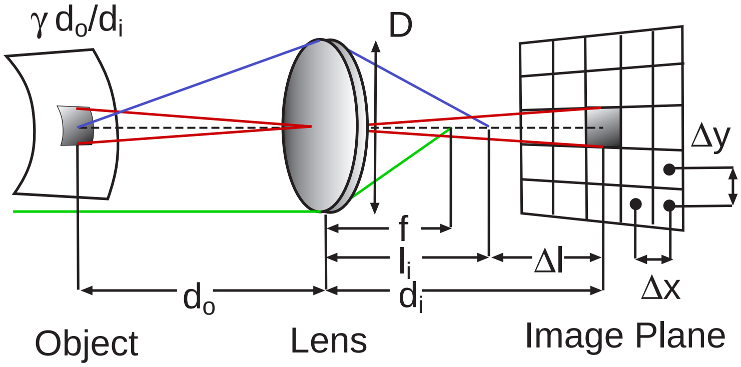


Figure 3.4: Image formation diagram.

We will restrict our analysis to regular arrays of densely spaced pinhole cameras and analyze how accurately a specified camera arrangement can reconstruct the continuous plenoptic function under a given model for the environment.

We will use the two-plane parameterization and assume that the imaging elements of the camera sample the light field on a regular lattice in the 5-D space of light rays which corresponds to a choice of camera spacing, image resolution, and frame rate. An example setup would be a set of cameras with their optical axes perpendicular to a plane containing the focal points (see Fig. 2.1). Then we can describe the 5D periodic lattice $\mathcal{A}$ using 5 vectors $[a_1, a_2, \ldots, a_5]$ which form a lattice matrix A such that $\mathcal{A} = \{A\boldsymbol{k} | \boldsymbol{k} \in \mathbb{Z}^5\}$. A unique tiling of the space of light rays can be achieved by associating with each lattice site a Voronoi cell, which contains all points that are closer to the given lattice site then to any other.

The camera output is modeled as the inner product of the light field with different translates of an analysis function ψ which was described in the previous sections:

$$\tilde{c}_\psi(\boldsymbol{k}) = \int L(\boldsymbol{x})\psi(A^{-1}\boldsymbol{x} - \boldsymbol{k})d\boldsymbol{x}; c_\psi(\boldsymbol{k}) \in l_2(\mathbb{Z}^5). \tag{3.22}$$

The function $\psi : \mathbb{R}^5 \to \mathbb{R}$ is dilated by the dilation matrix A^{-1} and sampled according to the lattice pattern A which results in Eq. (3.22). It models the effects of the Pixel Response Function (PRF) such as scattering, blurring, diffraction, flux integration across the pixel's receptive field, shutter time, and other signal degradations as explained in Section 3.1.

As described in Section 3.4 the conversion of the irradiance on the sensor surface to an electrical energy introduces noise into the measurement process which we denote by $N(\tilde{c}_\psi, \boldsymbol{k})$. The value measured by the sensor is then given by:

$$c_\psi(\boldsymbol{k}) = \tilde{c}_\psi(\boldsymbol{k}) + N(\tilde{c}_\psi, \boldsymbol{k}) \tag{3.23}$$

Since we are interested in a continuous reconstructed light field $I(\boldsymbol{x})$ we will express it as a linear combination of synthesis functions $\phi : \mathbb{R}^5 \to \mathbb{R}$ centered on the lattice points. Thus $I(\boldsymbol{x})$ is represented as

$$I(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in \mathbb{Z}^5} c_\phi(\boldsymbol{k}) \phi(A^{-1}\boldsymbol{x} - \boldsymbol{k});\ c_\phi(\boldsymbol{k}) \in l_2(\mathbb{Z}^5) \tag{3.24}$$

where l_2 is the space of square-summable sequences ($\sum_{k \in \mathbb{Z}^5} |c(\boldsymbol{k})|^2 < \infty$).

The coefficients c_ϕ are determined from the camera output c_ψ using a linear convolution filter ξ:

$$c_\phi(\boldsymbol{k}) = \sum_{i} \xi(\boldsymbol{i}) c_\psi(\boldsymbol{k} - \boldsymbol{i}). \tag{3.25}$$

The advantage of using a function approximation framework to locally describe the image formation process is that it allows us to utilize a rich assembly of tools to analyze the accuracy of the transfer of light radiance into pixel values.

3.6 Summary

In this chapter we showed how we can approximate the image formation process of a rectangular array of cameras in terms of a projection on a space formed by a shift-invariant filter. This filter accounts for the size of the individual pixels as well as the depth of field of the camera as necessary. We also described different sources of noise that are present when converting light into an electrical signal.

Chapter 4

Polydioptric Sampling Theory

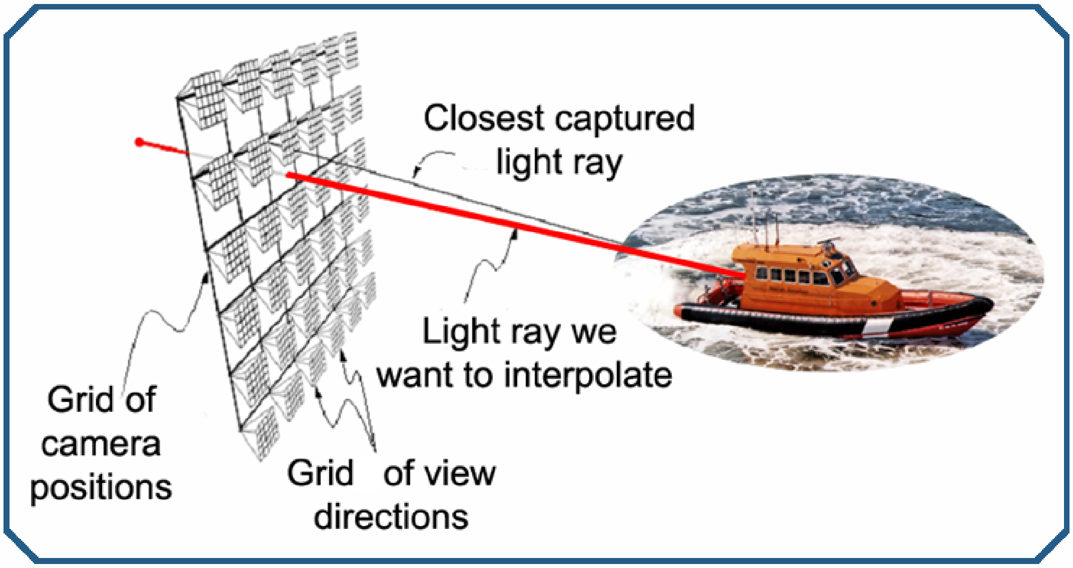


Figure 4.1: Every imaging device can only capture the plenoptic function with limited precision. By extending signal processing techniques to the space of light rays we can determine how accurately we can measure the radiance of light rays at non-sample locations.

Often we are given a camera design and we would like to know how well it will perform under a number of different conditions, or we need to design a camera that it is asked to perform well in a specific environment. In both cases we need to relate the design parameters to the error in the recovery of the necessary visual information.

Each neighborhood of imaging elements computes an approximation to the brightness structure of a localized bundle of light rays. The resolution of any camera is limited by the finite size of the imaging elements and lens aperture. This causes an uncertainty in our ability to measure the spatial position of objects, the shutter frequency and shutter time cause temporal uncertainty, and in addition the quantum nature of light measurement leads to unavoidable noise in the form of shot noise as was described in the previous chapter. An important criteria on which we will focus here is the spacing of the camera view points since this parameter is easiest to adjust by the user of multi-camera arrangements. The view point spacing can be discrete as in a light field camera or Argus eye, or it can vary more or less continuously as when a camera observes a non-focusing mirror. In this section we will show how the image formation process will be modeled in a function approximation framework which allows us to compute an accurate characterization of the measurement errors.

4.1 Plenoptic Sampling in Computer Graphics

As described in the introduction, image-based representations have become very popular in computer graphics, because they enable one to render a scene from novel view-points with minimal knowledge about the geometry and textures in the scene (for an overview see Zhang and Chen [163]) This representation makes it possible to develop display algorithms that are independent of the complexity of the scene description. The drawback is that many images need to be stored to be able to render new photo-realistic images. Due to the high data volume, it is of interest to find the minimum number of images that need to be stored to enable a perceptually valid reconstruction of all possible images in a given viewing area. The first paper that analyzed this problem was by Chai et al. [27], who studied the problem in the Fourier domain and derived a curve that related the geometrical complexity of the scene and the spectral bandwidth of the captured images to the lower bounds on the view point sampling density for non-aliased reconstruction. They modeled the image formation for an array of cameras by the following variant of the pixel equation

$$I(\boldsymbol{x}) = r(\boldsymbol{x}) * [(\mathcal{L}(\boldsymbol{x}) * p(\boldsymbol{x})) \cdot s(\boldsymbol{x})] \tag{4.1}$$

$$= r(\boldsymbol{x}) * [\mathcal{L}_p(\boldsymbol{x}) \cdot s(\boldsymbol{x})] \tag{4.2}$$

$$\mathcal{L}_p(\boldsymbol{x}) = [\mathcal{L}(\boldsymbol{x}) * p(\boldsymbol{x})] \tag{4.3}$$

which relates the reconstructed light field $I(\boldsymbol{x})$ to the ideal light field $\mathcal{L}(\boldsymbol{x})$ existing in physical space. r is the interpolation/reconstruction filter, p is the Pixel Response Function (PRF) that combines the effects of scattering, blurring, diffraction, flux integration across the pixel's receptive field, shutter time,and other signal degradations,and s is the sampling pattern defined by the camera spacing, image resolution, and frame rate, modeled as an impulse train $s(\boldsymbol{x}) = \sum\limits_{k \in \mathbb{Z}^5} \delta(\boldsymbol{x} - \Delta \boldsymbol{x} \boldsymbol{k})$. We disregard the effects of the optical elements on the view points during the image formation and model p as a combination of a low-pass filter in $\boldsymbol{x}_u$ and t, and a Dirac impulse in $\boldsymbol{x}_x$.

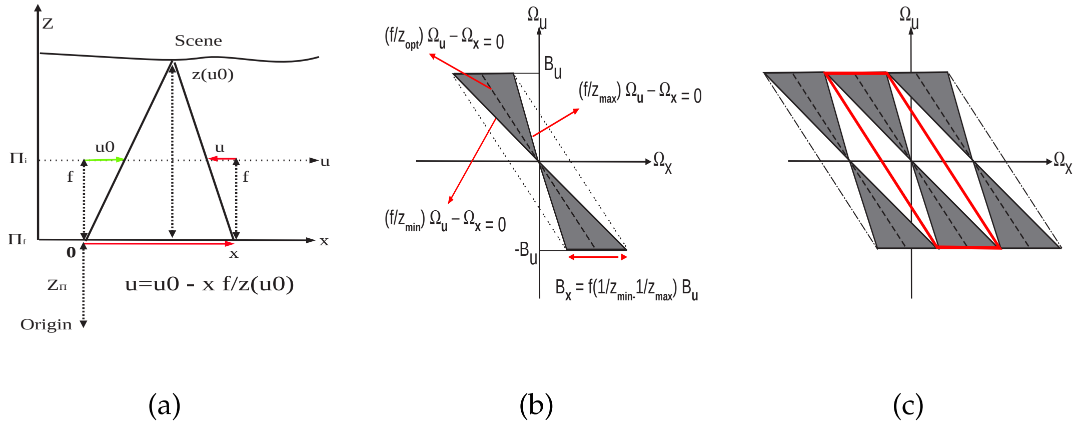
Applying the Fourier transform to Eq. (4.1), we get

$$\begin{aligned} \hat{I}(\boldsymbol{\Omega}) &= \hat{r}(\boldsymbol{\Omega}) \cdot [\hat{L}_p(\boldsymbol{\Omega}) * \hat{s}(\boldsymbol{\Omega})] \\ &= \hat{r}(\boldsymbol{\Omega}) \cdot \sum_{k \in \mathbb{Z}^5} \hat{L}_p(\boldsymbol{\Omega}_x - \frac{\boldsymbol{k}_x}{\Delta \boldsymbol{x}_x}, \boldsymbol{\Omega}_u - \frac{\boldsymbol{k}_u}{\Delta \boldsymbol{x}_u}, \Omega_t - \frac{k_t}{\Delta t}) = \hat{r}(\boldsymbol{\Omega}) \cdot \hat{L}_s \end{aligned} \tag{4.4}$$

where we use the abbreviations $\boldsymbol{\Omega}_x := [\Omega_x, \Omega_y]^{\mathrm{T}}$, $\boldsymbol{\Omega}_u := [\Omega_u, \Omega_v]^{\mathrm{T}}$, and $\boldsymbol{\Omega} := [\Omega_x, \Omega_y, \Omega_u, \Omega_v, \Omega_t]^{\mathrm{T}})$, as well as, $\Delta \boldsymbol{x}_x := [\Delta x, \Delta y]^{\mathrm{T}}$, and $\Delta \boldsymbol{x}_u := [\Delta u, \Delta v]^{\mathrm{T}}$.

We see that $\hat{L}_s$ is the sum of the copies of $\hat{L}_p$ shifted to the lattice points $[\frac{k_x}{\Delta \boldsymbol{x}_x}, \frac{k_u}{\Delta \boldsymbol{x}_u}, \frac{k_t}{\Delta t}]$. If these copies overlap we will have aliasing and will not be able to reconstruct L_p from the samples L_s exactly. We assume in the following that p pre-filters the light field sufficiently along the $\boldsymbol{x}_u$ and t dimensions, so that we can choose $\Delta u = \Delta v = \Delta t = 1$ without having any aliasing.

To understand the conditions on the sampling pattern to enable a continuous reconstruction of the plenoptic function from the image samples, we can utilize our analysis of the frequency structure of the time-varying light field that we did in Chapter 2.


Z
Scene
z(u0)
Π_l
u0
u
u
f
f
Π_f
0
x
x
Z_Π
u=u0 - x f/z(u0)
Origin
Ω_u
$(f/z_{opt})\ \Omega_u - \Omega_x = 0$
B_u
$(f/z_{max})\ \Omega_u - \Omega_x = 0$
Ω_x
$(f/z_{min})\ \Omega_u - \Omega_x = 0$
$-B_u$
$B_x = f(1/z_{min}\text{-}1/z_{max})\ B_u$
Ω_u
Ω_x

(a) (b) (c)

Figure 4.2: (a) Light Ray Correspondence (here shown only for the light field slice spanned by axes x and u). (b) Fourier spectrum of the (x,u) light field slice with choice of "optimal" depth z_{opt} for the reconstruction filter. (c) Optimal reconstruction filter in Fourier space.

There we showed that if the world consists of a number of fronto-parallel planes and we can disregard occlusion effects, then the frequency spectrum of the light field is bounded by two lines whose slopes were given by the minimal and maximal depth of the scene as shown in Fig. 4.2.

To compute a non-aliased orthographic image of the scene, one might think that it is necessary to place the cameras as closely as the size of the smallest feature in the world we would like to resolve. Fortunately, thanks to the redundancy in the light field representation, we have the following constraint on the spacing of the cameras [27]. Given the bounds on the depth $z_{\min} \leq z(\boldsymbol{x}_{u0}, t) \leq z_{\max}$ and the band limit $\boldsymbol{B}_0$ of $\hat{L}(\boldsymbol{\Omega})$, we see from Fig. 4.2 that the maximal view separation for a non-aliased image reconstruction is

$$\boldsymbol{\Delta}_x \leq \frac{1}{\boldsymbol{B}_x} = \frac{1}{f(1/z_{\min} - 1/z_{\max})\boldsymbol{B}_u} \tag{4.5}$$

If we have accurate information about the depth bounds of the scene, then we can design an interpolation filter that minimizes the error in the reconstruction of the continuous light field [27]. As can be seen in Fig. 4.2b, the copies of $\hat{L}_p(\boldsymbol{\Omega})$ are optimally compacted if we choose the principal directions of the interpolation filter to be $(f/z_{\text{opt}})\boldsymbol{\Omega}_u - \boldsymbol{\Omega}_x = 0$ and $\boldsymbol{\Omega}_u = 0$ where z_{opt} is chosen as $2/z_{\text{opt}} = 1/z_{\text{min}} + 1/z_{\text{max}}$. The ideal interpolation filter is given by the sinc interpolation filter whose pass-band region corresponds to the rectangle denoted by the dotted lines in Fig. 4.2b. As was pointed out in [27], choosing such a filter is similar to the depth corrected interpolation proposed in the original lumigraph paper [51] where we choose a local depth to improve the quadrilinear interpolation.

Assume we have cameras where the image pixels (view directions) are spaced Δu apart in the image plane. It follows that the signal we can reconstruct is band-limited by $\Omega_u \leq B_u = 1/2(\Delta u)$ due to the Nyquist theorem. To be able to compute intermediate images without aliasing artifacts, we need to apply a low-pass filter to the images with a cutoff-frequency that corresponds to the maximum signal bandwidth we can reconstruct using the view point spacing. Applying the Nyquist-theorem we know that the highest frequency we can reconstruct is $B_x = 1/(2\Delta x)$ where we denote the spacing of the different camera optical centers by Δx. We can assume using today's camera technology that $\Delta u \ll \Delta x$, therefore we can apply a low-pass filter to the images taken by the individual cameras, to reduce the aliasing at the cost of decreased resolution. We know that $B_x \in [B_u/Z_{max}, B_u/Z_{min}]$, therefore we choose the cutoff-frequency as

$$B_u \leq \min(Z_{min}/(2\Delta x), 1/(2\Delta u) \tag{4.6}$$

to avoid effects from aliasing.

4.2 Quantitative plenoptic approximation error

In the previous section we were able to establish minimal bounds on the spacing between viewpoints to achieve view interpolation free of aliasing. Unfortunately, for computer vision applications we often do not know what kind of properties a scene will have and it is very important that we do not just detect if we will have errors, but instead we would like to have an estimate about the magnitude and distribution of the errors. Since scenes and depths might vary noticeably, there does not exist a single optimal interpolation filter. Instead we need to know the relationship between the approximation error and the camera design parameters, so that this information can be used to optimize the camera design and can be included in the estimation process at the later stages of the processing pipeline. Therefore, in this section, we will derive quantitative expressions for the multi-dimensional approximation error between the original light field and the reconstructed light field which we will use to characterize the distribution of errors. Finally we will evaluate the exact average approximation error using statistics about the texture and depth distributions of the environment.

To evaluate the error of the light field reconstruction from the camera samples, we want to approximate the mean-square error $\epsilon^2 = \|L - I\|_{L^2}$ as presented in [144] using an integral of the Fourier spectrum of $\hat{L}(\mathbf{\Omega})$ and an error kernel $E(\mathbf{\Omega})$

$$\epsilon^2 \approx \mu^2(A) = \int_{-\infty}^{\infty} |\hat{L}(\mathbf{\Omega})|^2 E(A^T\mathbf{\Omega})d\mathbf{\Omega} \tag{4.7}$$

where A is the sampling lattice matrix, and the error kernel $E(\mathbf{\Omega})$ is defined as

$$\begin{aligned} E(\mathbf{\Omega}) &= |1 - \overline{\hat{\psi}(\mathbf{\Omega})}\hat{\phi}(\mathbf{\Omega})|^2 + \sum_{k\in\mathbb{Z}^n\backslash\{0\}} |\overline{\hat{\psi}(\mathbf{\Omega})}\hat{\phi}(\mathbf{\Omega}+\boldsymbol{k})|^2 \\ &= \underbrace{1 - \frac{|\hat{\tilde{\phi}}(\mathbf{\Omega})|^2}{\hat{a}_\phi(\mathbf{\Omega})}}_{E_{\text{min}}} + \underbrace{a_\phi\hat{(\mathbf{\Omega})}\left|\hat{\tilde{\psi}}(\mathbf{\Omega}) - \frac{\hat{\phi}(\mathbf{\Omega})}{a_\phi(\mathbf{\Omega})}\right|^2}_{E_{\text{res}}}. \end{aligned} \tag{4.8}$$

where $a_\phi\hat{(\mathbf{\Omega})} = \sum_{k\in\mathbb{Z}^n} |\Omega\overline{\hat{\psi}(\mathbf{\Omega})}\hat{\phi}(\mathbf{\Omega}+\boldsymbol{k})|^2$ is the sampled auto-correlation function of ϕ. Here as before ϕ is the synthesis function, and $\tilde{\psi}$ is the combination of image transfer function ψ and prefilter ξ.

This error kernel consists of two components. The first term E_{min} measures the least-squares error between the signal L and the perpendicular projection into the synthesis space spanned by the functions ϕ. The second term E_{res} then measures how orthogonal the analysis and synthesis function spaces are. We see that if ψ is the dual basis function to the synthesis function ϕ, that is $\hat{\psi} = \hat{\phi}/\hat{a}_\phi$, then the light field is orthogonally projected onto the space of the synthesis functions $\tilde{\psi} = \phi_d$ and E_{res} vanishes. In that case Eq. (4.7) reduces to the least-squares error as expected.

If L is a band-limited then we have the equality $\epsilon = \mu$ for all phase shifts of the signal with respect to the camera, otherwise $\mu(V)$ is equal to the average error over all possible phase shifts $L(A\boldsymbol{k} + \mathbf{\Delta})$ where $\mathbf{\Delta} \in \{A\boldsymbol{x}|\boldsymbol{x} \in [0,1)^5\}$. This is a very useful property because in general we are not interested in a specific position of the camera relative to the scene, but want to have an error characterization that is independent of the camera position.

Eq. (4.7) gives us a means to assess how accurately a given camera design is able to reconstruct the space of light rays in an environment, and thus how accurately we are able to estimate the quantities of interest that we need to compute to solve our task.

An important part of the design process of a computer vision system is also the image processing as a preprocessing step. Thus, the question becomes how we should choose the filter ξ to minimize the approximation error. In general, since we do not have access to the original signal L, the best we can do is to perpendicularly project the sampling space spanned by the set of analysis functions $\{\psi(\boldsymbol{x} - \cdot)\}$ on to the image space spanned by the set of synthesis functions $\{\phi(\boldsymbol{x} - \cdot)\}$. This ensures that the image $I(\boldsymbol{x})$ passes through the imaging pipeline unchanged, thus to the camera it "looks" the same as the true light field.

If the image transfer function is known (for example from measuring the point spread function of the camera optics), then we can define the cross-correlation sequence $a_{\psi\phi}(\boldsymbol{k}) = \int \overline{\phi(\boldsymbol{x})}\psi(\boldsymbol{x} - \boldsymbol{k})d\boldsymbol{x}$. If $a_{\psi\phi}(\boldsymbol{k})$ is an invertible convolution operator in l_2, then the perpendicular projection results in the synthesis coefficients $c_\phi(\boldsymbol{k}) = (a_{\psi\phi}^{-1} * c_\psi)(\boldsymbol{k})$ where the filter ξ equals $a_{\psi\phi}^{-1}$ (for more details see [150]). In the Fourier domain this filter has the expression $\hat{\xi}(\omega) = 1/(\sum_k \hat{\psi}(\omega - k)\hat{\phi}(\omega - k))$.

We can also choose the prefilter ξ such that the reconstructed signal interpolates the input signal at the lattice sites, that is $I(A\boldsymbol{k}) = L(A\boldsymbol{k})$. The coefficients of the prefilter can then be determined from the interpolation condition using filter design techniques (for an example on a hexagonal lattice see [151]).

We are interested in evaluating the shape of the error kernel for the case of a regular array of cameras. The combination of the point spread function of the camera and the integration over the pixel area determines the analysis function ψ, and can be approximated by a tensor product of B-spline of order p denoted by $\beta^p(x)$.

B-splines of order p can be defined as the $(p+1)$-fold convolution of the box-function

$$\beta^0(x) = \begin{cases} 1 & \text{if } |x| \leq 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

with itself, that is $\beta^p = \beta^{p-1} * \beta^0 = \underbrace{\beta^0 * \beta^0 * \ldots * \beta^0}_{(p+1)-\text{fold}}$.

Gaussians are often used to approximate the point spread function (PSF) of the camera lens. The relationship between a Gaussian and B-splines of order p was described in [150]) leading to the formula

$$\beta^p(x) \approx \frac{1}{\sqrt{2\pi\sigma_p}} \exp\left(-\frac{x^2}{2\sigma_p^2}\right),\ \sigma_p = \sqrt{\frac{p+1}{12}}. \tag{4.10}$$

This approximation is already very accurate for low values of p (1% error for $p = 3$), thus we can approximate the PSF of the lens by a B-spline with little loss in accuracy.

The analysis function for each pixel in the perspective image is approximated by a B-spline that is dilated according to the size of the optical and depth of field blur (we assume that the analysis is local enough, so that we can approximate the depth of field by a shift-invariant filter). The sampling due to view points that are spaced apart can be modeled by a train of delta functions $\delta(x)\delta(y)$. For the time domain we choose a simple box filter that integrates over the integration time. We will assume that the sampling grid is square resulting in A being a diagonal matrix. This allows us to write the analysis function ψ as the tensor product of B-splines and delta functions along orthogonal coordinate axes:

$$\psi(A^{-1}\boldsymbol{x} - \boldsymbol{k}) = \delta(x/a_x - k_x)\delta(y/a_y - k_y)\beta^p(u/a_u - k_u)\beta^p(v/a_v - k_v)\beta^0(t/a_t - k_t) \tag{4.11}$$

It was shown in [13] and [144] that cubic B-splines offer the best compromise between order of approximation and minimal support, thus we will choose them as our synthesis basis function ϕ. They are centered on the reconstruction grid and dilated by the same amount as the analysis function, so we have $\beta_n(A^{-1}\boldsymbol{x} - \boldsymbol{k})$.

Since the lattice matrix for the synthesis function is the same as the one for the analysis function, we can write the synthesis functions also as a tensor product of the B-spline functions along the individual coordinate axes:

$$\phi(A^{-1}\boldsymbol{x}-\boldsymbol{k}) = \beta^m(x/a_x-k_x)\beta^m(y/a_y-k_y)\beta^n(u/a_u-k_u)\beta^n(v/a_v-k_v)\beta^0(t/a_t-k_t) \tag{4.12}$$

B-splines have the following Fourier transform

$$\hat{\beta}^n(\omega) = \left[\frac{\sin(\pi\omega)}{\pi\omega}\right]^n \tag{4.13}$$

Since the analysis and synthesis functions are separable, the error kernel defined in Eq. (4.7) can be evaluated for each coordinate axis independently. If we define

$$\hat{a}_p = \sum_k \prod_i |\hat{\beta}_p(\omega_i + k)|^2 \text{ and} \tag{4.14}$$

$$\hat{b}_p = \sum_k \prod_i \hat{\beta}_p(\omega_i + k) \tag{4.15}$$

where i varies over the different dimensions x, y, u, v, t then we can write the two components of the error kernel as:

$$E_{\text{min}}(\omega) = 1 - \frac{|\hat{\phi}(\omega)|^2}{\hat{a}_\phi(\omega)} \tag{4.16}$$

$$E_{\text{res}}(\omega) = \left|\sqrt{\hat{a}_\phi(\omega)}\hat{\psi}(\omega)\hat{\xi}(\omega) - \frac{\hat{\phi}(\omega)}{\sqrt{\hat{a}_\phi(\omega)}}\right|^2 \tag{4.17}$$

In our case we have for the least-squares approximation error

$$E_{\text{min}}(\omega) = 1 - \frac{\sin(\pi\omega)^{2n}}{(\pi\omega)^{2n}\hat{a}_n(\omega)} \tag{4.18}$$

and for the residual errors we have for the view point interpolation

$$E_{\text{res}}(\omega_x) = \left|\sqrt{\hat{a}_n(\omega_x)}\hat{\xi}_x(\omega) - \frac{\sin(\pi\omega_x)^{2n}}{(\pi\omega_x)^{2n}\sqrt{\hat{a}_n(\omega_x)}}\right|^2$$

and similarly for the view direction interpolation (n is the degree of the synthesis B-spline, while p is the degree of the analysis B-spline).

$$E_{\text{res}}(\omega_u) = \left| \frac{\sqrt{\hat{a}_n(\omega_u)}\sin(\pi\omega_u)^p}{(\pi\omega_u)^p}\hat{\xi}_u(\omega_u) - \frac{\sin(\pi\omega_u)^{2n}}{(\pi\omega_u)^{2n}\sqrt{\hat{a}_n(\omega_u)}} \right|^2$$

We can evaluate the components of the error kernel by utilizing the following two identities (as described in [13]):

$$\cot(\omega) = \sum_k (\omega + \pi k)^{-1} \qquad \text{for } p \text{ even}$$

$$\sin(\omega)^{-1} = \sum_k (-1)^k (\omega + \pi n)^{-1} \qquad \text{for } p \text{ odd}$$

This leads to the definition of

$$\hat{b}_p(\omega) = -\frac{\sin(\pi\omega)^p}{(p-1)!}\frac{d^{p-1}}{d\omega^{p-1}}(\cot(\pi\omega)) \qquad \text{for } p \text{ even}$$

$$\hat{b}_p(\omega) = -\frac{\sin(\pi\omega)^p}{(p-1)!}\frac{d^{p-1}}{d\omega^{p-1}}(\sin(\pi\omega)^{-1}) \qquad \text{for } p \text{ odd}$$

and $\hat{a}_p(\omega) = \hat{b}_{2p}(\omega)$.

As explained before the optimal choice for the correction filter ξ depends on the application. The best approximation in the least-squares sense is achieved if we set $\hat{\xi}(\omega) = \hat{a}_{\psi\phi}(\omega)^{-1}$, that is the inverse of the sampled correlation function between ψ and ϕ. The cross-correlation between each component of ψ and ϕ is given as

$$a_{\psi\phi}(\boldsymbol{k}) = \int \overline{\phi(\boldsymbol{x})}\psi(\boldsymbol{x}-\boldsymbol{k})d\boldsymbol{x} = \int \overline{\beta^p(\boldsymbol{x})}\beta^n(\boldsymbol{x}-\boldsymbol{k})d\boldsymbol{x} = \beta^{p+n+1}(\boldsymbol{k}) = b^m(\boldsymbol{k}) \qquad (4.19)$$

The optimal correction filter $(b^m)^{-1}$ in this example is the so-called B-spline filter of order $m = p + n + 1$ which is stable for all m. The shape of these filters needs to be adapted to the specific properties of the light field Fourier spectrum that we determined in the previous chapters.

4.3 Evaluation of Approximation Error based Natural Image Statistics

To evaluate Eq. (4.7) we have to choose a synthesis function ϕ, determine the image transfer function ψ and compute the appropriate prefilter ξ. In addition, we need to have an idea about the power spectrum of the light field $|\hat{L}|^2$. The power spectrum depends on the scene in which the sensor operates. Depending on the task we try to solve we might have access to very specific knowledge about the scene the sensor will operate in which case we can evaluate the error for a number of specific scenes. This becomes quickly infeasible, if we want to design a visual sensor that performs well in many different environments.

Using Eq. (2.33)

$$|\hat{L}(\Omega_x, \Omega_u, \Omega_t)|^2 = \frac{K}{\|\Omega_u\|^m} D_{\dot{x}}(\frac{\Omega_t}{\Omega_x}) D_z(\frac{\Omega_u}{\Omega_x})$$

derived in Section 2.6 we can approximate the power spectrum of an average scene.

This expression allows us now evaluate Eq. (4.7) given a depth and velocity distribution. If we have a more complicated scene then we would have to integrate over the distribution of the frequency spectrum using Monte Carlo integration since in general the power spectrum of such a scene cannot be expected to admit a nice analytical expression. A detailed study is subject of future work.

To apply the framework described in Section 4.3, we need to determine the distribution of depths in the scene, and the distribution of velocities which depend on the locomotion of the sensor, as well as the spacing of the pixels on the image plane, the spacing of the view points and the frame rate of the camera (which determine the sampling lattice).

4.4 Non-uniform Sampling

In future work I would like to extend the analysis presented in this chapter to the case of non-uniformly distributed cameras. The interpolation problem for non-uniformly distributed data samples is a very active area research and can be attacked using the framework of frames. Camera systems can be interpreted as defining a frame operator on the space of light rays. By analyzing the properties of this operator such as its frame bounds we can determine how well we can reconstruct the original plenoptic function based on the light ray samples that we captured.

4.5 Summary

In this chapter we analyzed how accurately the space of light rays can be reconstructed from the image information captured by a camera. We described previous approaches in computer graphics and explained their short comings. We then described how we can derive quantitative expressions for the difference between the true and reconstructed light fields by expressing the image acquisition process in a function approximation framework. Since this expression is defined in the Fourier domain, we showed how the statistics of natural images can be used to evaluate this error expression for an average environment. This quantitative expression will be used in the following chapter to analyze the accuracy of the plenoptic motion estimation algorithm.

Chapter 5

Application: Structure from Motion

As explained in Chapter 2 we cannot infer information about the three-dimensional structure of the world without prior knowledge about the scene. If the camera is moving though, the changes in the images captured over time are related to both the motion of the camera and the structure of the world by the geometry of image formation. Thus, if we can recover the motion of the camera, we can factor its effects out and utilize the residual changes to compute the scene properties such as depth and the motion of objects in the scene.

Structure from motion is one of the fundamental and thus also most studied problems in computer vision. The study of this problem has come a long way from its early breakthroughs by Longuet-Higgins and Prazdny [84, 119, 83], Horn [20], Huang [148, 81], Maybank [95, 96], Faugeras [45], Aloimonos [134, 135] and Chellappa [18] as demonstrated by the recent arrival of comprehensive text books on the subject [55, 88]. Despite its maturity, foolproof solution still do not exist because the non-linear nature of the underlying constraints makes the error functions often non-convex. This causes algorithms to end up in local minima of the objective function and the solution is easily perturbed by noise in the input measurements which can lead to erroneous and ambiguous solutions.

For accurate results we need to employ computationally expensive global optimization techniques such as bundle adjustment [147] to compute accurate solutions.

It was quickly noticed that motion estimation benefits from the inclusion of stereo information. Usually this is done by first computing correspondence between points in different cameras and then triangulating the 3D coordinates of the points in space. In a second step then the 3D-information was used to compute the motion estimate. The advantage of this approach is that the dimensionality of the motion estimation problem is dramatically reduced since only the motion parameters are unknown. This can be done for discrete features [160, 157, 38], as well as for optical flow and dense depth maps [53, 138, 140]. With all these approaches we still have the problem, that we specifically need to determine a depth map or equivalently correspondences between the multiple views to be able to compute the motion of the camera which in itself is a difficult and often ill-posed and error-prone problem. Since finding corresponding points in different images is a notoriously hard problem, it would be of great advantage if one could estimate the motion of the camera without explicitly having to find correspondences first.

Motivated by the improvement of the camera motion estimation by the inclusion of stereo information or large field-of-view (i.e. omni-directional) cameras, it seems worthwhile to include the design of the camera when studying the motion estimation problem. Therefore, as motivated in the introduction, we will examine the problem in the space of light rays. As outlined before in the introduction to do so we need to answer the two fundamental questions of camera design:

1. How is the relevant visual information that we need to extract to solve our task encoded in the visual data that a camera can capture?

2. What is the camera design and image representation that optimally facilitates the

extraction of the relevant information?

First, we will establish the constraints on the problem in terms of the light rays captured, which leads to the description of a novel constraint, the ray identity constraint between two polydioptric images. This constraint allows a multi-viewpoint camera to compute accurate motion parameters without determining explicit correspondences or scene models by matching sets of light rays.

Then, turning to the second question, we will describe how the design of the camera used for image acquisition determines which of these constraints can be used and how accurately we can compute the motion parameters. Finally this will lead to the definition of a fitness function over the space of cameras for the problem of ego-motion estimation of a moving image sensor.

5.1 Plenoptic Video Geometry: How is 3D motion information encoded in the space of light rays?

We will interpret the images captured by a generalized camera in terms of samples of the plenoptic function. Based on these samples, we can estimate the local geometry of the plenoptic function and then utilize these features in the space of light rays to recover spatio-temporal information about the world.

We define a generalized camera in terms of a set of two-dimensional imaging surfaces $\mathcal{C}_i(u, v)$ which are indexed by pixel coordinates (u, v). Associated with each camera surface is a pair of functions that map a pixel to a ray in space. Each ray is defined by a position ($\boldsymbol{x}_i : (u, v) \in \mathbb{R}^2 \to \mathbb{R}^3$) and a direction in space ($\boldsymbol{r}_i : (u, v) \in \mathbb{R}^2 \to \mathbb{S}^2$). These functions do not need to be continuous, because adjacency in (u, v) does not necessarily imply adjacency in the space of light rays. One can for example think of a camera that

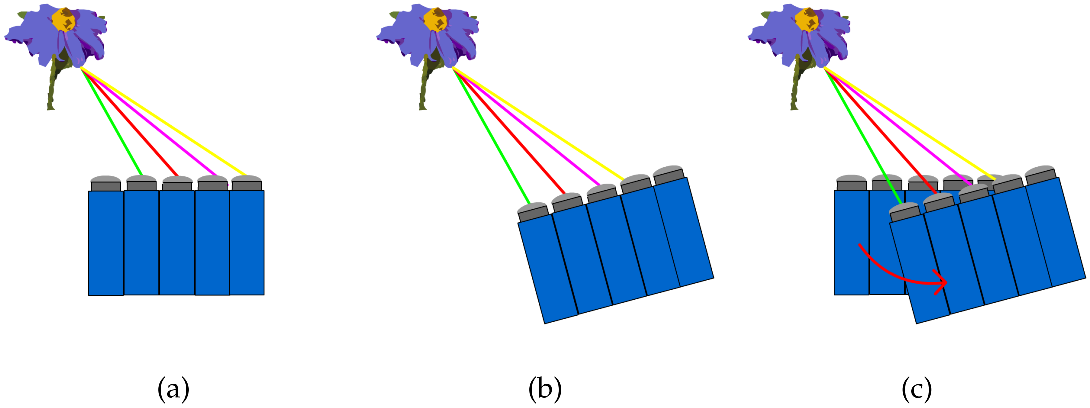


Figure 5.1: Illustration of ray incidence and ray identity: (a-b) A multi-perspective system of cameras observes an object in space while undergoing a rigid motion. Each individual camera sees a scene point on the object from a different view point which makes the correspondence of the rays depend on the scene geometry (ray incidence). (c) By computing the inter- and intra-camera correspondences of the set of light rays between the two time instants, we can recover the motion of the camera without having to estimate the scene structure since we correspond light rays and not views of scene points (ray identity).

consists of a set of mini-lenses on a CCD chip where at the boundaries between adjacent mini-lenses we will have a discontinuity in the observed ray directions and positions.

Each camera surface collects an imaging sequence $I_i(u, v, t)$. We assume that the camera is undergoing a rigid motion in space which is parameterized by a rotation matrix R and a translation vector $\boldsymbol{q}$ thus the world coordinates of a ray entering pixel (u, v) in camera $\mathcal{C}_i$ are given by $\boldsymbol{x}_i(u, v, t) = R(t)\boldsymbol{x}_i(u, v) + \boldsymbol{q}(t)$ and $\boldsymbol{r}_i(u, v, t) = R(t)\boldsymbol{r}_i(u, v)$.

Depending on the geometric properties of the camera, there are different types of features that can be computed, and we have two fundamentally different types of constraints as illustrated in Table 5.1.

If a scene, made up of diffuse reflective surfaces, is observed from multiple view points, then the views of the same surface region will look very similar. The intensity

Camera Type	Ray Incidence	Ray Identity
Single view point	Camera motion $\otimes$ depth (Structure from motion)	Camera rotation (3 d.o.f.)
Multiple view points	Stereo motion Small and large-baseline stereo	Rigid motion (6 d.o.f.) Differential Stereo

Table 5.1: Quantities that can be computed using the basic light ray constraints for different camera types. $\otimes$ denotes that two quantities cannot be independently estimated without assumptions about scene or motion.

function defined on a line bundle through a scene point is expected to have a much lower variance then a line pencil through a point in free air. This photo-consistency constraint [78] allows us to identify corresponding projections of the same scene point in the different views. This can also be extended to other features such as sets of rays through lines and planes utilizing line and texture cues [6, 86]. Given corresponding features we can then infer geometrical information such as shape and occlusions about the scene, as well as the position of the cameras. We call these constraints *ray incidence constraints*. These family of constraints have been used in stereo algorithms to find depth from correspondences, and in motion estimation algorithms to find the camera motion from correspondences. Examples are the well-known epipolar and trifocal constraints between multiple images.

If we move a polydioptric camera in space, there is a second constraint. If the camera captures overlapping sets of light rays at two time instants, then we can register the two sets of light rays and recover the motion of the camera. We term this constraint the *ray identity constraint*. The general principle is illustrated in Figure5.1.

In the following section we will show how both of these different constraints can be used to compute the motion of a camera and the shape of the scene based on measurements in the space of light rays.

5.2 Ray incidence constraint

The ray incidence constraint is defined in terms of a scene point $\boldsymbol{P}$ and set of rays $l_i := (\boldsymbol{x}_i, \boldsymbol{r}_i)$. The incidence relation between the scene point $\boldsymbol{P} \in \mathbb{R}^3$ and rays l_i, defined by their origins $\boldsymbol{x}_i \in \mathbb{R}^3$ and directions $\boldsymbol{r}_i \in \mathbb{S}^2$, can be written as

$$[\boldsymbol{r}_i]_\times \boldsymbol{x}_i = [\boldsymbol{r}_i]_\times \boldsymbol{P} \ \forall \ i \tag{5.1}$$

were $[\boldsymbol{r}]_\times$ denotes the skew-symmetric matrix so that $[\boldsymbol{r}]_\times \boldsymbol{x} = \boldsymbol{r} \times \boldsymbol{x}$. The rays that satisfy this relation form a 3D-line pencil in the space of light rays.

The geometric incidence relations for light rays lead to extensions of the familiar multi-view constraints for light rays. Depending on the design of the imaging sensor there are three types of epipolar constraints that constrain the structure and motion estimation ($\boldsymbol{q}_{j,i}, R_{j,i}$ are the translation and rotation that move l_j into the camera coordinate system of l_i in case they were measured at different times).

1. Pinhole camera: If we have a conventional single-view point camera, then each scene point is only observed once for each frame and we have the usual single view point epipolar constraint that constrains the camera motion up to scale and has the form

$$\boldsymbol{r}_i^{\mathrm{T}} [\boldsymbol{q}_{i,j}]_\times R_{j,i} \boldsymbol{r}_j = 0.$$

2. Non-central "Argus Eye" camera: If we observe the world using multiple single viewpoint cameras with non-overlapping field of view, then we can utilize an

epipolar constraint on the motion between frames including the scale of the reconstruction which has the form

$$\boldsymbol{r}_i^{\mathrm{T}}[\boldsymbol{q}_{i,j} + (\boldsymbol{x}_i - R_{j,i}\boldsymbol{x}_j)]_\times R_{j,i}\boldsymbol{r}_j = 0.$$

3. Polydioptric or stereo camera: If we have a multi-perspective camera and the scene point $\boldsymbol{P}_k$ projects into multiple locations at the same time, then in addition to the previous two types of epipolar constraints, we can also utilize a stereo constraint between two feature locations l_i and l_j that is independent of the motion between frames and is of the form

$$\boldsymbol{r}_i^{\mathrm{T}}[(\boldsymbol{x}_i - \boldsymbol{x}_j)]_\times \boldsymbol{r}_j = 0.$$

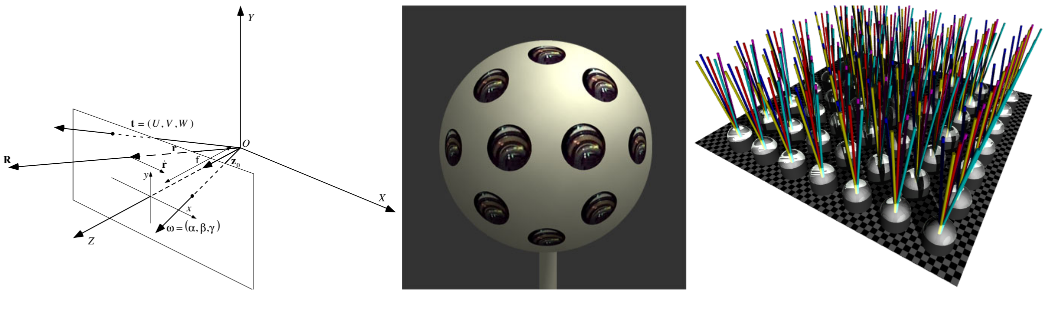


Figure 5.2: Examples of the three different camera types: (a) Pinhole camera (b) Spherical Argus eye and (c) Polydioptric lenslet camera.

Possible implementations of these three camera types can be seen in Fig. 5.2

To solve for the motions and 3D positions of points, we need to optimize over the manifold of rotations and translations. The description of this optimization is beyond the scope of this thesis, but its experimental validation in this context is planned for the future. The interested reader is referred to the papers [87] for the two view case and [89] for the multiple view case, as well as the general description of how to optimize functions on spaces with orthogonality constraints [40]. The books [44, 55, 88] also include a large number of algorithms to find depth and camera motion from ray incidence constraints.

5.3 Ray Identity Constraint

In a static world where the albedo of the scene points is not changing over time, the brightness structure of the space of light rays is time-invariant. Therefore, as was shown in Chapter 2, if a camera moves rigidly and captures two overlapping sets of light rays at two different time instants, then a subset of these rays should match exactly and would allow us to recover the rigid motion from the light ray correspondences. Note that this is a true brightness constancy constraint because we compare each light ray to itself. This is in contrast to the usual assumption of brightness constancy, where we have to assume a notion of view invariance since we compare two views of the same scene point. This is illustrated in Fig. 5.1.

5.3.1 Discrete plenoptic motion equations

As we had shown in previous work [104, 103], this brightness constancy constraint can be utilized to estimate the motion of the camera in a scene-independent way. The set of imaging elements that make up a camera each capture the radiance at a given position $\boldsymbol{x} \in \mathbb{R}^3$ coming from a given direction $\boldsymbol{r} \in \mathbb{S}^2$. If the camera undergoes a rigid motion and we choose the camera coordinate system as our fiducial coordinate system, then we can describe this motion by an opposite rigid coordinate transformation of the ambient space of light rays in the camera coordinate system. This rigid transformation, parameterized by the rotation matrix $R(t)$ and a translation vector $\boldsymbol{q}(t)$, results in the following *exact* equality which is called the *discrete plenoptic motion constraint*

$$\mathcal{L}(R(t)\boldsymbol{x} + \boldsymbol{q}(t); R(t)\boldsymbol{r}; t) = \mathcal{L}(\boldsymbol{x}; \boldsymbol{r}; 0) \tag{5.2}$$

since the rigid motion maps the time-invariant space of light rays upon itself. Thus, if a sensor is able to capture a continuous, non-degenerate subset of the plenoptic function, then the problem of estimating the rigid motion of this sensor has become an image registration problem that is *independent of the scene*. Therefore the only free parameters are the six degrees of freedom of the rigid motion. This global parameterization leads to a highly constrained estimation problem that can be solved with any multi-dimensional image registration criterion.

5.3.2 Differential plenoptic motion equations

If in the neighborhood of the intersection point $\boldsymbol{y} \in \mathbb{R}^3$ of the ray ϕ ($\phi(\lambda) = \boldsymbol{x} + \lambda\boldsymbol{r}$) with the scene surface the albedo is continuously varying and no occlusion boundaries are present, then the plenoptic function $\mathcal{L}$ changes smoothly and we can develop the plenoptic function $\mathcal{L}$ in the neighborhood of $(\boldsymbol{x}; \boldsymbol{r}; t)$ into a Taylor series (we use $\mathcal{L}_t$ as an abbreviation for $\partial\mathcal{L}/\partial t$):

$$\begin{gathered}\mathcal{L}(\boldsymbol{x} + d\boldsymbol{x}; \boldsymbol{r} + d\boldsymbol{r}; t + dt) = \mathcal{L}(\boldsymbol{x}; \boldsymbol{r}; t) \qquad (5.3)\\ +\mathcal{L}_t dt + \nabla_{\boldsymbol{x}}\mathcal{L}^{\mathrm{T}} d\boldsymbol{x} + \nabla_{\boldsymbol{r}}\mathcal{L}^{\mathrm{T}} d\boldsymbol{r} + \mathcal{O}(\|d\boldsymbol{r}, d\boldsymbol{x}, dt\|^2).\end{gathered}$$

where $\nabla_{\boldsymbol{x}}\mathcal{L}$ and $\nabla_{\boldsymbol{r}}\mathcal{L}$ are the partial derivatives of $\mathcal{L}$ with respect to $\boldsymbol{x}$ and $\boldsymbol{r}$. This expression now relates a local change in view ray position and direction to the first-order differential brightness structure of the plenoptic function.

We define the *plenoptic ray flow* $(d\boldsymbol{x}/dt, d\boldsymbol{r}/dt)$ as the difference in position and orientation between the two rays that are captured by the same imaging element at two consecutive time instants. This allows us to use the spatio-temporal brightness derivatives of the light rays captured by an imaging device to constrain the plenoptic ray flow. This generalizes the well-known *Image Brightness Constancy Constraint* to the *Plenoptic Brightness Constancy Constraint*:

$$\frac{d}{dt}\mathcal{L}(\boldsymbol{r};\boldsymbol{x};t) = \mathcal{L}_t + \nabla_{\boldsymbol{r}}\mathcal{L}^{\mathrm{T}}\frac{d\boldsymbol{r}}{dt} + \nabla_{\boldsymbol{x}}\mathcal{L}^{\mathrm{T}}\frac{d\boldsymbol{x}}{dt} = 0. \tag{5.4}$$

We assume that the imaging sensor can capture images at a rate that allows us to use the instantaneous approximation of the rotation matrix $R \approx I + [\boldsymbol{\omega}]_\times$ where $[\boldsymbol{\omega}]_\times$ is a skew-symmetric matrix parameterized by the axis of the instantaneous rotation $\boldsymbol{\omega}$. Now we can define the plenoptic ray flow for the ray captured by the imaging element located at location $\boldsymbol{x}$ and looking in direction $\boldsymbol{r}$ as

$$\frac{d\boldsymbol{r}}{dt} = \boldsymbol{\omega} \times \boldsymbol{r} \text{ and } \frac{d\boldsymbol{x}}{dt} = \boldsymbol{\omega} \times \boldsymbol{x} + \dot{\boldsymbol{q}} \tag{5.5}$$

where $\dot{\boldsymbol{q}} = d\boldsymbol{q}/dt$ is the instantaneous translation. As before in the discrete case (Eq.(5.2)), the plenoptic ray flow is completely specified by the six rigid motion parameters. This regular global structure of the rigid plenoptic ray flow makes the estimation of the differential rigid motion parameters very well-posed.

Combining Eqs. 5.4 and 5.5 leads to the *differential plenoptic motion constraint*

$$\begin{aligned} -\mathcal{L}_t &= \nabla_{\boldsymbol{x}}\mathcal{L} \cdot (\boldsymbol{\omega} \times \boldsymbol{x} + \dot{\boldsymbol{q}}) + \nabla_{\boldsymbol{r}}\mathcal{L} \cdot (\boldsymbol{\omega} \times \boldsymbol{r}) \qquad (5.6) \\ &= \nabla_{\boldsymbol{x}}\mathcal{L} \cdot \dot{\boldsymbol{q}} + (\boldsymbol{x} \times \nabla_{\boldsymbol{x}}\mathcal{L} + \boldsymbol{r} \times \nabla_{\boldsymbol{r}}\mathcal{L}) \cdot \boldsymbol{\omega} \end{aligned}$$

which is a linear, scene-independent constraint in the motion parameters and the plenoptic partial derivatives.

5.3.3 Differential Light Field Motion Equations

Using the light field parameterization we can rewrite the plenoptic motion equation (Eq. 5.6) by setting $\boldsymbol{x} = [x, y, Z_\Pi]^\mathrm{T}$ and $\boldsymbol{r} = \frac{[u,v,f]^\mathrm{T}}{\|[u,v,f]^\mathrm{T}\|}$. We plug these expressions into Eq.5.6, and convert the spatial partial derivatives of the light field $L_x = \partial L/\partial x, \ldots, L_v = \partial L/\partial v$ into the three-dimensional plenoptic derivatives $\nabla_{\boldsymbol{x}}\mathcal{L}$ and $\nabla_{\boldsymbol{r}}\mathcal{L}$. To do so, we consider the projections of $\nabla_{\boldsymbol{x}}\mathcal{L}$ and $\nabla_{\boldsymbol{r}}\mathcal{L}$ on three directions $\boldsymbol{c}_x$, $\boldsymbol{c}_r$, and $\boldsymbol{r}$ to obtain the following linear system which we solve for $\nabla_{\boldsymbol{x}}\mathcal{L}$ and $\nabla_{\boldsymbol{r}}\mathcal{L}$ ($\boldsymbol{c}_x = [1, 0, 0]^\mathrm{T}$, $\boldsymbol{c}_y = [0, 1, 0]^\mathrm{T}$, and $n_{\boldsymbol{r}} = \|[u, v, f]^\mathrm{T}\|$):

$$\begin{pmatrix} \boldsymbol{c}_x^\mathrm{T} \\ \boldsymbol{c}_y^\mathrm{T} \\ \boldsymbol{r}^\mathrm{T} \end{pmatrix} [\nabla_{\boldsymbol{x}}\mathcal{L}, \nabla_{\boldsymbol{r}}\mathcal{L}] = \begin{pmatrix} L_x & n_{\boldsymbol{r}} L_u \\ L_y & n_{\boldsymbol{r}} L_v \\ 0 & 0 \end{pmatrix}.$$

This results in the following expressions for the plenoptic derivatives

$$\nabla_{\boldsymbol{x}}\mathcal{L} = [L_x, L_y, -\frac{u}{f}L_x - \frac{v}{f}L_y]^\mathrm{T}$$
$$\nabla_{\boldsymbol{r}}\mathcal{L} = n_{\boldsymbol{r}}[L_u, L_v, -\frac{u}{f}L_u - \frac{v}{f}L_v]^\mathrm{T}.$$

Using these expressions, we define the plenoptic motion constraint for the ray indexed by (x, y, u, v, t) as ($[\cdot;\cdot]$ denotes the vertical stacking of vectors):

$$\begin{aligned} -L_t = -\mathcal{L}_t &= \nabla_{\boldsymbol{x}}\mathcal{L} \cdot \dot{\boldsymbol{q}} + (\boldsymbol{x} \times \nabla_{\boldsymbol{x}}\mathcal{L} + \boldsymbol{r} \times \nabla_{\boldsymbol{r}}\mathcal{L}) \cdot \boldsymbol{\omega} \\ &= [L_x, L_y, -\frac{u}{f}L_x - \frac{v}{f}L_y]\dot{\boldsymbol{q}} \\ &- [L_x, L_y, -\frac{u}{f}L_x - \frac{v}{f}L_y]([x, y, Z_\Pi]^\mathrm{T} \times \boldsymbol{\omega}) \\ &- [L_u, L_v, -\frac{u}{f}L_u - \frac{v}{f}L_v]([u, v, f]^\mathrm{T} \times \boldsymbol{\omega}) \\ &= [L_x, L_y, L_u, L_v][M_t, M_\omega][\dot{\boldsymbol{q}}; \boldsymbol{\omega}] \end{aligned} \tag{5.7}$$

where

$$M_t = \begin{pmatrix} 1 & 0 & -\frac{u}{f} \\ 0 & 1 & -\frac{v}{f} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \text{and} \quad M_\omega = \begin{pmatrix} -\frac{uy}{f} & \frac{ux}{f} + Z_\Pi & -y \\ -(\frac{vy}{f} + Z_\Pi) & \frac{vx}{f} & x \\ -\frac{uv}{f} & \frac{u^2}{f} + f & -v \\ -(\frac{v^2}{f} + f) & \frac{vu}{f} & u \end{pmatrix} \tag{5.8}$$

By combining the constraints across the light field, we can form a highly overdetermined linear system and solve for the rigid motion parameters.

The light field derivatives $L_x, \ldots, L_t$ can be obtained directly from the image information captured by a polydioptric camera. For example, to convert the image information captured by a collection of pinhole cameras into a light field, for each camera we simply have to intersect the rays from its optical center through each pixel with the two planes Π_f and Π_i and set the corresponding light field value to the pixel intensity. Since our measurements are only at discrete locations, we have to use appropriate interpolation schemes as described in more detail in Chapter 4. The light field derivatives can then easily be computed by applying standard image derivative operators to the continuous light field. The plenoptic motion constraint is extended to the other faces of the nested cube by pre-multiplying $\dot{\boldsymbol{q}}$ and $\boldsymbol{\omega}$ with the appropriate rotation matrices to rotate the motion vectors into local light field coordinates.

5.3.4 Derivation of light field motion equations

In this section we will derive the discrete and differential motion equation for a polydioptric camera in the light field parameterization. The plenoptic parameterization is defined in world space. Each ray is parameterized by a point on the ray $\boldsymbol{x} \in \mathbb{R}^3$ and a direction vector $\boldsymbol{r} \in \mathbb{S}^2$. As described in Section 2.2.3 we can also parameterize a bundle of rays within a cone of directions using the intersection of the rays with two planes Π_f and Π_i. We will assume as before that these planes are perpendicular to the z-axis of the local camera coordinate and are at depths Z_Π and $Z_\Pi + f$. The camera coordinate system of camera C_i is related to the world coordinate system by a rotation matrix R_i and a translation vector $\boldsymbol{T}_i$. Thus, the ray defined by the coordinate vector (x, y, u, v) corresponding to the ray (we use the abbreviations $\boldsymbol{x}_{lf} = [x, y, Z_\Pi]^{\mathrm{T}}$ and $\boldsymbol{r}_{lf} = [u, v, f]^{\mathrm{T}}$):

$$\boldsymbol{l} = \left\{\boldsymbol{p} \in \mathbb{R}^3 | \boldsymbol{p} = \boldsymbol{x}_{lf} + \lambda \boldsymbol{r}_{lf}, \lambda \in \mathbb{R}\right\}$$

in camera coordinates corresponds to the following ray coordinates in the plenoptic parameterization:

$$\boldsymbol{l} = (\boldsymbol{x}, \boldsymbol{r}) \text{ with } \boldsymbol{x} = R_i \boldsymbol{x}_{lf} + \boldsymbol{T}_i \text{ and } \boldsymbol{r} = R_i \boldsymbol{r}_{lf} / \left\|\boldsymbol{r}_{lf}\right\| \tag{5.9}$$

If the camera is now displaced by a rotation matrix $R(t)$ and a translation $\boldsymbol{q}(t)$, then the ray $\boldsymbol{l}$ has the following equation in camera coordinates ($\boldsymbol{q}(t) = [q_x, q_y, q_z]^{\mathrm{T}}$):

$$\boldsymbol{l} = \left\{\boldsymbol{p} \in \mathbb{R}^3 | \boldsymbol{p} = R(t)^{\mathrm{T}} \left(\boldsymbol{x}_{lf} - \boldsymbol{q}\right) + \lambda R(t)^{\mathrm{T}} \boldsymbol{r}_{lf}, \lambda \in \mathbb{R}\right\} \tag{5.10}$$

To find the coordinates of this ray in the light field parameterization, we have to intersect this ray with the plane Z_Π to find the (x, y) coordinates and rescale the direction vector so that its z-coordinate equals f. Denoting the columns of $R(t)$ by $R_1(t), R_2(t), R_3(t)$, we can easily find the value of λ where l intersects the plane Z_Π resulting in:

$$\lambda = \left[Z_\pi - R_3(t)^{\mathrm{T}} \left(\boldsymbol{x}_{lf} - \boldsymbol{q}(\boldsymbol{t})\right)\right] / R_3(t)^{\mathrm{T}} \boldsymbol{r}_{lf} \tag{5.11}$$

Thus, the new equations for the light field coordinates of the ray $\boldsymbol{l}$ are:

$$\begin{aligned} x' &= R_1(t)^{\mathrm{T}} \left(\boldsymbol{x}_{lf} - \boldsymbol{q}(t)\right) + \left[Z_\pi - R_3(t)^{\mathrm{T}} \left(\boldsymbol{x}_{lf} - \boldsymbol{q}(t)\right)\right] \frac{R_1(t)^{\mathrm{T}}\boldsymbol{r}_{lf}}{R_3(t)^{\mathrm{T}}\boldsymbol{r}_{lf}}, \\ y' &= R_2(t)^{\mathrm{T}} \left(\boldsymbol{x}_{lf} - \boldsymbol{q}(t)\right) + \left[Z_\pi - R_3(t)^{\mathrm{T}} \left(\boldsymbol{x}_{lf} - \boldsymbol{q}(t)\right)\right] \frac{R_2(t)^{\mathrm{T}}\boldsymbol{r}_{lf}}{R_3(t)^{\mathrm{T}}\boldsymbol{r}_{lf}}, \\ u' &= f\frac{R_1(t)^{\mathrm{T}}\boldsymbol{r}_{lf}}{R_3(t)^{\mathrm{T}}\boldsymbol{r}_{lf}}, \text{ and } v' = f\frac{R_2(t)^{\mathrm{T}}\boldsymbol{r}_{lf}}{R_3(t)^{\mathrm{T}}\boldsymbol{r}_{lf}} \end{aligned} \tag{5.12}$$

which we can summarize as $l' = \mathcal{T}(l, \boldsymbol{q}(t), R(t))$. Using the Rodrigues formula we can parameterize the rotation matrix $R(t)$ by the axis of rotation $\boldsymbol{\omega}(t) = [\omega_1(t), \omega_2(t), \omega_3(t)]$. Then $\|\boldsymbol{\omega}(t)\|$ is the angle of rotation, and $\hat{\boldsymbol{\omega}}(t) = \boldsymbol{\omega}(t)/\|\boldsymbol{\omega}(t)\|$ is the direction of the axis of rotation.

$$R(t) = I_3 + \sin(\|\boldsymbol{\omega}(t)\|)[\boldsymbol{\omega}(t)/\|\boldsymbol{\omega}(t)\|]_x + (1 - \cos(\|\boldsymbol{\omega}(t)\|))[\boldsymbol{\omega}(t)/\|\boldsymbol{\omega}(t)\|]_x^2 \tag{5.13}$$

If the angle of rotation is small we can make the approximation $R(t) \approx I_3 + [\boldsymbol{\omega}(t)]_x$ which we will use to derive the instantaneous ray motion equations in the light field parameterization. We have then (dropping the time index for now)

$$\begin{aligned} u' &= f\frac{R_1^{\mathrm{T}}\boldsymbol{r}_{lf}}{R_3^{\mathrm{T}}\boldsymbol{r}_{lf}} \approx f\frac{u - \omega_3 v + \omega_2 f}{f - \omega_2 u + \omega_1 v} = \frac{u - \omega_3 v + \omega_2 f}{1 - (\omega_2 \frac{u}{f} - \omega_1 \frac{v}{f})} \\ &= (u - \omega_3 v + \omega_2 f)(1 + (\omega_2 \frac{u}{f} - \omega_1 \frac{v}{f}) + (\omega_2 \frac{u}{f} - \omega_1 \frac{v}{f})^2 + (\ldots)^3 + \ldots) \\ &= u - \frac{uv}{f}\omega_1 + (f + \frac{u^2}{f})\omega_2 - v\omega_3 + \mathcal{O}(\|\boldsymbol{\omega}\|^2) \end{aligned}$$

where we used the Neumann series expansion $1/(1-x) = \sum_{i=0}^{\infty} x^i$ and omit all second-order terms in the rotation parameters.

A similar derivation can be done for the change of view point. Using the instantaneous approximation for the rotation we can write:

$$
\begin{aligned}
x' &= R_1^{\mathrm{T}}\left(\boldsymbol{x}_{lf} - \boldsymbol{q}\right) + \left[Z_\pi - R_3^{\mathrm{T}}\left(\boldsymbol{x}_{lf} - \boldsymbol{q}\right)\right] \frac{R_1^{\mathrm{T}}\boldsymbol{r}_{lf}}{R_3^{\mathrm{T}}\boldsymbol{r}_{lf}} \\
&\approx (x - q_x) + (y - q_y)\omega_3 - (Z_\Pi - q_z)\omega_2 \\
&+ \frac{Z_\Pi}{f}(u - \frac{uv}{f}\omega_1 + (f + \frac{u^2}{f})\omega_2 - v\omega_3) \\
&+ \frac{1}{f}((y - q_y)\omega_1 - (x - q_x)\omega_2 - Z_\Pi + q_z)(u - \frac{uv}{f}\omega_1 + (f + \frac{u^2}{f})\omega_2 - v\omega_3) \\
&= x - q_x + \frac{u}{f}q_z + \frac{yu}{f}\omega_1 - \left(Z_\Pi + \frac{xu}{f}\right)\omega_2 + y\omega_3 + \mathcal{O}(\|\boldsymbol{\omega}, \boldsymbol{q}\|^2)
\end{aligned}
$$

The derivations for v' and y' are similar. This results in the differential motion equations in the light field parameterization given by Eq. (5.7).

5.4 Variational Formulation of the Plenoptic Structure from Motion

We would like to describe the structure from motion estimation using the plenoptic function in a variational framework. To find the rigid motion parameters, we need to minimize the discrete plenoptic motion constraint as defined in Eq. (5.2). Φ will denote the error function which could be the L_2-Norm or some robust error measure that compares two sets of captured light rays. D describes the sensor surface, and $\mathcal{T}$ is a transformation of the ray $\boldsymbol{l}$ in the space of light rays parameterized by set of parameters Θ (e.g., rotation R and translation $\boldsymbol{q}$ of the camera) that we want to recover . It is now our goal to find the set of parameters $\hat{\Theta}$ so that $\mathcal{L}(\mathcal{T}(\boldsymbol{l}, \Theta), t + \Delta t) = \mathcal{L}(\boldsymbol{l}, t)$ and thus minimizes the matching functional

$$\hat{\Theta} = \arg\min_{\Theta} \int\limits_{\boldsymbol{l} \in D} \Phi(\mathcal{L}(\mathcal{T}(\boldsymbol{l}, \Theta), t + \Delta t) - \mathcal{L}(\boldsymbol{l}, t))d\boldsymbol{l} \tag{5.14}$$

To solve for $\hat{\Theta}$, we form the Lagrange equations of Eq. (5.14) and set them to zero.

This leads to a system of non-linear equations for the optimal value $\hat{\Theta}$:

$$\int_{l \in D} \Phi'(\mathcal{L}(\mathcal{T}(\boldsymbol{l}, \hat{\Theta}), t + \Delta t), \mathcal{L}(\boldsymbol{l}, t)) \nabla \mathcal{L}(\mathcal{T}(\boldsymbol{l}, \hat{\Theta}), t + \Delta t) J_{\mathcal{T}|\Theta}(\boldsymbol{l}, \hat{\Theta}) d\boldsymbol{l} = \mathbf{0} \tag{5.15}$$

where $J_{\mathcal{T}|\Theta}$ is the Jacobian of the transformation $\mathcal{T}$ with respect to the parameters Θ. For the case of rigid motion estimation using the light field parameterization it is given by the motion matrices in Eq. (5.7).

We can minimize this functional using a gradient descent minimization. Since the error function is in general not convex due to the non-convexity of the image function and thus the minimization can get stuck in a local minima. Therefore, use a scale-space focusing strategy by embedding the estimation in a coarse-to-fine framework. The estimation consists of two nested fix-point iterations. The outer loop consists of solving the motion estimation equation starting using a smoothed image $\mathcal{L}^k = (G_{\sigma_k} * \mathcal{L})$ where G_{σ_k} is a smoothing filter in the space of light rays, and then using the solution Θ_k at the coarse scale as an initialization for the solution at the finer scale. We will choose the coarse scale such that there will be only subpixel differences between the view point images. This can be achieved by utilizing our sampling analysis from Chapter 4.

Since the resulting equations given in the form of Eq. (5.15) are still non-linear we will follow a numerical optimization scheme similar to the one described in [19]. We will update the motion parameters using small update steps along the tangent plane of $SE(3)$ given by the instantaneous motion vectors $\boldsymbol{\omega}$ and $\dot{\boldsymbol{q}}$. Using the linearization of the plenoptic function around our current motion estimate, we can then incrementally update the camera motion with the solution of the linear system defined by the local constraint equations in the form of the plenoptic motion constraint in Eq. (5.6) evaluated at the current value of the rigid motion parameters $\hat{R}$ and $\hat{q}$.

Thus, we linearize the difference between the warped and the original light field as follows:

$$\begin{aligned}&\mathcal{L}(\mathcal{T}(\boldsymbol{l},\Theta^{i+1}),t+\Delta t)-\mathcal{L}(\boldsymbol{l},t)\\=&\mathcal{L}(\mathcal{T}(\boldsymbol{l},\Theta^{i}),t+\Delta t)-\mathcal{L}(\boldsymbol{l},t)+\nabla\mathcal{L}(\mathcal{T}(\boldsymbol{l},\Theta^{i}),t+\Delta t)J_{\mathcal{T}|\Theta}d\boldsymbol{\Theta^{i}}\end{aligned}$$

This will now lead to a new set of equations in the update parameters $d\boldsymbol{\Theta^i}$. Since $J_{\mathcal{T}|\Theta}$ is independent of $d\boldsymbol{\Theta^i}$, the only remaining non-linearity is possibly due to the function Φ, unless the function Φ is a simple function such as the squared-error $\Phi(x) = x^2$. Therefore, we will use another inner fix-point iteration where we will use the previous estimate $d\boldsymbol{\Theta^i}$ to compute the value for $\Phi(l, \Theta^i)$ and keep it fixed while solving the linear outer equation for $d\boldsymbol{\Theta^{i+1}}$. Thus, we use a form of iterated weighted least-squares minimization to implement the robust error function Φ. Example functions would be non-linear M-estimators such as the Huber function or modified L_1 minimization (e.g. $\Phi(x) = \sqrt{x^2 + \epsilon^2} - \epsilon$) as described by Zhang [166] or Meer [97].

Here we are using a similarity measure that depends on a dense sampling of the plenoptic function because the evaluation points for the moving image are not necessarily the same as the original sampling points. As explained in Chapter 2, the plenoptic parameterization is not optimal for signal processing. The problem is that we cannot express the distance between two lines in terms of the distance between the line coordinates interpreted as points in a Euclidean space. Since this is possible for the light field parameterization and many camera configurations consist of clusters of axis-aligned cameras the light field parameterization is our representation of choice. In this case, each cluster can be parameterized using the light field parameterization by rectifying the images to the plane that is the best linear fit through the optical centers of the camera focal points without introducing any major distortions. To evaluate the matching function, we need to solve an interpolation problem in the light field parameterization.

5.5 Feature computation in the space of light rays

To utilize the constraints described above we need to define the notion of correspondence in mathematical terms. In the case of the ray identity constraint we have to evaluate if two sets of light rays are identical. If the scene is static we can use the difference between the sets of light rays that are aligned according to the current motion estimate as our matching criterion. This criterion is integrated over all rays and we expect that at the correct solution, we will have a minimum. As with all registration algorithms, we need signals that contain enough information for the matching. As we analyzed in Section 2.3, if the surface texture is only a homogeneous color or consists only of gradients in a limited number of directions, then we might not be able to compute the motion of scene points uniquely. This is known as the aperture problem, which expresses the fact that we cannot detect any information about changes of position along iso-brightness contours. The amount of information which corresponds to the amount of texture in the perspective images is often measured in terms of the eigenvalues of the structure tensor, that is the outer product of the local intensity gradients integrated over a local neighborhood [131, 68]. This criterion can be extended to the space of light rays by examining the structure tensor of the plenoptic function. The plenoptic structure tensor $\nabla\mathcal{L}\nabla\mathcal{L}^{\mathrm{T}}$ can be computed from the intensity gradients of the plenoptic function with respect to view point $\nabla_x\mathcal{L}$ and view direction $\nabla_r\mathcal{L}$. By examining subspaces of the plenoptic structure tensor, we can determine what kind of features can be reliably computed. The structure tensor of the plenoptic function has a simple structure because we have the relationship $\mathcal{Z}(\boldsymbol{x},\boldsymbol{r},t)\nabla_x\mathcal{L}(\boldsymbol{x},\boldsymbol{r},t) = \nabla_r\mathcal{L}(\boldsymbol{x},\boldsymbol{r},t)$ where $\mathcal{Z}(\boldsymbol{x},\boldsymbol{r},t)$ is again the depth to the scene from location $\boldsymbol{x}$ in direction $\boldsymbol{r}$ at time t.

The amount of texture of the scene can thus be measured by examining the sub-structure tensors $\nabla_x \mathcal{L} \nabla_x \mathcal{L}^{\mathrm{T}}$ and $\nabla_r \mathcal{L} \nabla_r \mathcal{L}^{\mathrm{T}}$ which correspond to the structure tensors of the perspective and orthographic images of the scene. As suggested in the literature before, we can use the inverse of the intensity Hessian as a measure for the variance of the estimated feature positions [131].

If we analyze the portion of the tensor formed by the image derivatives $\nabla_{\mathrm{EPI}} \mathcal{L}$ in the epipolar plane images (EPI) [14], we can analyze how much information is available to compute the depth of the scene. EPI images are two-dimensional subspaces of the plenoptic function where both view point and view direction are varying. We have $\nabla_{\mathrm{EPI}} \mathcal{L} = [\boldsymbol{m}^{\mathrm{T}} \nabla_x \mathcal{L}; \boldsymbol{m}^{\mathrm{T}} \nabla_r \mathcal{L}]$ where $\|\boldsymbol{m}\| = 1$ and $\boldsymbol{m}^{\mathrm{T}} \boldsymbol{r} = 0$. The structure tensor will have a single non-zero eigenvalue for fronto-parallel planes (lines in the EPI), and two non-zero eigenvalues for depth discontinuities. To estimate the depth and the local shape we need to make differential measurements of the plenoptic function. The accuracy of the depth estimates depend on how accurately we can reconstruct the light field based on the samples captured by a multi-perspective camera. The main advantage of such a polydioptric camera is that geometric properties of the scene such as depth, surface slope, and occlusions can be easily computed from the intensity structure of the polydioptric images.

5.6 How much do we need to see of the world?

In the previous section we saw how we can compute the motion parameters by utilizing the polydioptric image derivatives. If we choose a set of the spatial dimensions x, y, u, v together with the time dimension t, then we get spatio-temporal images where the gradients in the image describe the relative motion between the imaging sensor and points in the scene. The gradients due to a rigid motion of a camera are given by the equation for the plenoptic motion constraint. Depending on the type of sensor we are restricted in our choice of constraints on the motion parameters as collected in Table 5.2.

Subspace Axes	Example Motion Constraint Equation (zero columns imply that no constraint exists for that parameter)	Parameters to estimate
2D: xt,yt **ut**,vt	$-L_t = L_u \left(\frac{f}{z}\ 0\ -\frac{u}{z}\ 0\ \frac{u^2}{f}+f\ 0 \right) \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{pmatrix}$	2+N
3D: xyt,xvt yut,**uvt**	$-L_t = \begin{pmatrix} L_u \\ L_v \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \frac{f}{z} & 0 & -\frac{u}{z} & -\frac{uv}{f} & \frac{u^2}{f}+f & -v \\ 0 & \frac{f}{z} & -\frac{v}{z} & -(\frac{v^2}{f}+f) & \frac{uv}{f} & u \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{pmatrix}$	5+N
3D: **xut**,yvt	$-L_t = \begin{pmatrix} L_x \\ L_u \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} 1 & 0 & -\frac{u}{f} & 0 & \frac{ux}{f}+Z_\Pi & 0 \\ 0 & 0 & 0 & 0 & \frac{u^2}{f}+f & 0 \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{pmatrix}$	3
4D: xyut,xyvt **xuvt**,yuvt	$-L_t = \begin{pmatrix} L_x \\ L_u \\ L_v \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} 1 & 0 & -\frac{u}{f} & 0 & \frac{ux}{f}+Z_\Pi & 0 \\ 0 & 0 & 0 & -\frac{uv}{f} & \frac{u^2}{f}+f & -v \\ 0 & \frac{f}{z} & -\frac{v}{z} & -(\frac{v^2}{f}+f+\frac{f}{z}Z_\Pi) & \frac{vu}{f}+\frac{vx}{z} & u+\frac{f}{z}x \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{pmatrix}$	6 + N
5D: **xyuvt**	$-L_t = \begin{pmatrix} L_x \\ L_y \\ L_u \\ L_v \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} 1 & 0 & -\frac{u}{f} & -\frac{uy}{f} & \frac{ux}{f}+Z_\Pi & -y \\ 0 & 1 & -\frac{v}{f} & -(\frac{vy}{f}+Z_\Pi) & \frac{vx}{f} & x \\ 0 & 0 & 0 & -\frac{uv}{f} & \frac{u^2}{f}+f & -v \\ 0 & 0 & 0 & -(\frac{v^2}{f}+f) & \frac{vu}{f} & u \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{pmatrix}$	6

Table 5.2: Rigid motion constraint equations for plenoptic subspaces of different dimensions. In each row the corresponding subspace for each motion constraint equation is in bold letters.

We see that depending on the subspace that the camera captures an algorithm can compute the ego-motion estimate with or without having to estimate the scene structure at the same time. For a single viewpoint camera, we can either make use of an image registration algorithms that finds a parametric mapping between two images (in case of pure rotation or planar scene) or we can use correspondences to solve a global bundle adjustment problem. In general for a single view camera the estimation of motion and structure are coupled and both have to be estimated simultaneously. In the case of multiple views we have two options. Each measurement that a camera captures corresponds to a bundle of light rays in space, for any scene and motion we can find a rigid motion that maps one set of light rays to another. This is equivalent to the case of estimating the rotation of a pinhole camera which can be done independently of the scene that is observed. Given the multi-view image information it is of course also possible to compute approximate 3D information to improve the search for corresponding points and the chance to presegment the scene into different depth layers.

5.7 Influence of the field of view on the motion estimation

Another important criteria for the sensitivity of the motion estimation problem is the size of the field of view (FOV) of the camera system. The basic understanding of these difficulties has attracted a number of investigators over the years [33, 34, 70, 77, 96, 111]. These difficulties are based on the geometry of the problem and they exist in the cases of small and large baselines between the views, that is for the case of continuous motion as well as for the case of discrete displacements of the cameras.

If we increase the field of view of a sensor to 360° proofs in the literature show that we should be able to accurately recover 3D motion and subsequently shape [47]. Catadioptric sensors could provide the field of view but they have poor resolution, making it difficult to recover shape models. Instead assemblies of cameras, such as the Argus eye [4], a construction consisting of six cameras pointing outwards, offer an alternative to catadioptric systems. When this structure is moved arbitrarily in space, then data from all six cameras can be used to very accurately recover 3D motion, which can then be used in conjunction with the individual videos to recover shape.

Since the six cameras do not have the same center of projection, the motion estimation for this camera is more elaborate than for a spherical one, but because the geometric configuration between the cameras is known from the camera calibration, one can obtain all three translational parameters.

For every direction of translation one finds the corresponding best rotation which minimizes deviation from a brightness-based constraint. Fig. 5.3 shows (on the sphere of possible translations) the residuals of the epipolar error color coded for each individual camera computed for a real image sequence that we captured with a multi-perspective camera setup, the ”Argus Eye”. Noting that the red areas are all the points within a small percentage of the minimum, we can see the valleys which clearly demonstrates the ambiguity theoretically shown in the proofs in the literature. In contrast, we see in Fig. 5.4 a well-defined minimum (in red) when estimating the motion globally over all the Argus

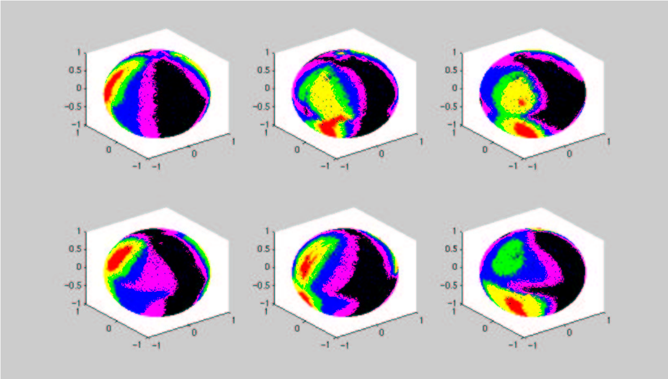


Figure 5.3: Deviation from the epipolar constraints for motion estimation from individual cameras (from [5])

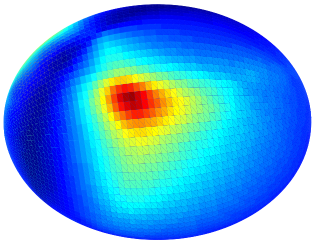


Figure 5.4: Combination of error residuals for a six camera Argus eye (from [5])

cameras, indicating that the direction of the translation obtained is not ambiguous when using information from a full field of view.

5.8 Sensitivity of motion and depth estimation using perturbation analysis

To assess the performance of different camera designs we have to make sure that the algorithms we use to estimate the motion and shape are comparable.

If the camera moves differentially between frames, then the ray intersection constraint leads to the optical flow equations for a rigidly moving camera. If we observe the same scene point $\boldsymbol{P}$ from two different locations $\boldsymbol{x}$ and $\boldsymbol{x} + \Delta\boldsymbol{x}$, then we have for a Lambertian surface

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{r}, t) = \mathcal{L}(\boldsymbol{x}, (\boldsymbol{p} - \boldsymbol{x})/\|\boldsymbol{p} - \boldsymbol{x}\|, t) = \mathcal{L}(\boldsymbol{x} + \Delta\boldsymbol{x}, \frac{\boldsymbol{p} - \boldsymbol{x} - \Delta\boldsymbol{x}}{\|\boldsymbol{p} - \boldsymbol{x} - \Delta\boldsymbol{x}\|}, t) \qquad (5.16)$$

We can write (where $\lambda = \mathcal{D}(\boldsymbol{x}, \boldsymbol{r}, t)$ is the distance between $\boldsymbol{x}$ and $\boldsymbol{P}$).

$$\frac{\boldsymbol{p} - \boldsymbol{x} - \Delta\boldsymbol{x}}{\|\boldsymbol{p} - \boldsymbol{x} - \Delta\boldsymbol{x}\|} = \frac{\lambda\boldsymbol{r} - \Delta\boldsymbol{x}}{\|\lambda\boldsymbol{r} - \Delta\boldsymbol{x}\|} = \boldsymbol{r} - \frac{(I_3 - \boldsymbol{r}\boldsymbol{r}^{\mathrm{T}})\Delta\boldsymbol{x}}{\lambda} + \mathcal{O}(\left\|\frac{\Delta\boldsymbol{x}}{\lambda}\right\|^2)$$

which is the well-known expression for translational motion flow on a spherical imaging surface. Assuming brightness constancy, we have

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{r}, t) = \mathcal{L}(\boldsymbol{x} + \Delta\boldsymbol{x}, \boldsymbol{r} - \mathcal{P}(\boldsymbol{r})\Delta\boldsymbol{x}/\lambda, t) = \mathcal{L}(\boldsymbol{x} + \Delta\boldsymbol{x}, \boldsymbol{r} + \Delta\boldsymbol{r}, t). \qquad (5.17)$$

Thus, we can express a displacement in view point space by an equivalent parallel displacement in view direction space which only differs by a scale factor that is proportional to the depth. This relationship allows us compare algorithms based on ray incidence constraints (scene point correspondences) with algorithms based on ray identity constraints (light ray correspondences).

We choose to compare a number of standard algorithms for ego-motion estimation as described in [145] to solving a linear system based on the plenoptic motion flow equation Eq. (5.5). This linear system relates the plenoptic motion flow to the rigid motion parameters, that is the translation $\boldsymbol{q}$ and axis of rotation $\boldsymbol{\omega}$. Since multi-view information can be used to infer depth information, we will project the motion flow onto a spherical retina and account for the multi-view information by scaling the flow by an approximate inverse depth value. Since we compute the depth using the polydioptric derivatives, the distribution of the depth errors can be computed using the error expression presented in Chapter 4. Then the flow for the ray $l_i = (\boldsymbol{x}_i, \boldsymbol{r}_i, t)$ is given by (we drop the argument and write $Z_i = \mathcal{Z}(\boldsymbol{x}_i, \boldsymbol{r}_i, t)$):

$$-\frac{1}{Z_i}[\boldsymbol{r}_i]_x^2\boldsymbol{q} - \left(\frac{1}{Z_i}[\boldsymbol{r}_i]_x^2[\boldsymbol{x}_i]_x + [\boldsymbol{r}_{\boldsymbol{i}}]_x\right)\boldsymbol{\omega} = \dot{\boldsymbol{r}}_i \tag{5.18}$$

Given a multi-perspective camera system we can utilize the multi-view information to generate an approximate local depth map. The accuracy of these 3D measurements depends on the noise in our image measurements, the accuracy of the calibration of the camera, and the baseline between the views. In this section we will apply the ideas of stochastic perturbation theory [139] to analyze the influence of depth errors on the accuracy of instantaneous motion estimation. We will denote the inverse depth at a given point $\boldsymbol{P}_i$ by $D_i = 1/Z_i$. Since each measurement is scaled by the depth individually, we can combine the individual depth measurements to form the diagonal matrices $D = \text{diag}(D_1, \ldots, D_n)$ and $Z = \text{diag}(Z_1, \ldots, Z_n)$. If we know the inverse depths in D and the spherical flow $\dot{\boldsymbol{r}}$ in the images, then we can form a linear system of the form ($\boldsymbol{m} = [\boldsymbol{q}; \boldsymbol{\omega}]$):

$$A\boldsymbol{m} = \boldsymbol{b} \text{ that is } DA_z[\boldsymbol{q}; \boldsymbol{\omega}] + A_\omega\boldsymbol{\omega} = [\dot{\boldsymbol{r}}_1; \dot{\boldsymbol{r}}_2; \dot{\boldsymbol{r}}_3; ...] \tag{5.19}$$

where $\boldsymbol{b}$ contains the flow between frames. $A_z = [A_q, A_c]$ is formed by stacking the 3×3

matrices $A_{qi} := [\boldsymbol{r}_i]_x^2$ and $A_{ci} := [\boldsymbol{r}_i]_x^2[\boldsymbol{x}_i]_x$ that contain the terms in Eq. (5.18) that are scaled by the depth, and (A_ω) is constructed by stacking the terms $[\boldsymbol{r_i}]_x$ that do not get scaled by the depth.

There are two sources for error in this system: the error in the inverse depth D_ϵ and the error in the computed optical flows $\dot{\boldsymbol{r}}_i$ which we stack to form the vector $\boldsymbol{b}$. We write the estimated inverse depth as $\tilde{D} = D + D_\epsilon$ and estimated spherical flow as $\tilde{\boldsymbol{b}} = \boldsymbol{b} + \boldsymbol{b}_\epsilon$. Then we can write the linear system including the error terms as:

$$(A + E)\boldsymbol{m} = (D + D_\epsilon)A_z\boldsymbol{m} + A_\omega\boldsymbol{\omega} = \boldsymbol{b} + \boldsymbol{b}_\epsilon = \tilde{\boldsymbol{b}} \tag{5.20}$$

We can characterize the error matrix in a probabilistic sense by writing it as a stochastic matrix [139] of the form

$$E := S_c H S_r = \hat{D}_\epsilon H A_z \tag{5.21}$$

where H is a diagonal stochastic matrix where the entries have zero mean and unit variance. The entries in H are multiplied from the left by the inverse depth errors in the diagonal matrix $\hat{D}_\epsilon$ to scale the errors to the correct size and multiplied from the right by A_z, the entries of the system matrix A that get scaled by the depth, to account for the correlations between the depth values and the motion parameters.

If we write $C = A^{\mathrm{T}}A$, then the first order change in the solution is given by [139]:

$$\hat{\boldsymbol{m}} = \boldsymbol{m} - A^{+}E\boldsymbol{m} + C^{-1}E^{\mathrm{T}}\boldsymbol{b}_\epsilon \tag{5.22}$$

The stochastic norm $\|A\|_s$ of a matrix A is defined as the expected Frobenius norm of the matrix that is $\|A\|_s = E[\|A\|_F] = E[\sqrt{\text{trace}(A^{\mathrm{T}}A)}]$. We can express the difference between the true and estimated motion parameters in terms of stochastic norms as:

$$\|\hat{\boldsymbol{m}} - \boldsymbol{m}\|_S = \sqrt{\|A^{+}S_c\|_F^2\|S_r\boldsymbol{m}\|^2 + \|S_c\boldsymbol{b}_\epsilon\|^2\|S_rC^{-1}\|_F^2} \tag{5.23}$$

or for each component separately

$$\|\hat{\boldsymbol{m}}_i - \boldsymbol{m}_i\|_S = \sqrt{\|S_c A_i^+\|^2 \|S_r \boldsymbol{m}\|^2 + \|S_c \boldsymbol{b}_\epsilon\|^2 \|S_r C_i^{-1}\|^2} \tag{5.24}$$

where A_i^+ and C_i^{-1} are the i-th rows of the matrices A^+ and C^{-1}. Assuming that the errors in the stereo correspondences and the errors in the optical flow are identically independently distributed with variances σ_D and σ_b, then we can simplify the expressions for the error in the motion parameters. In this case we have that $S_c = \hat{D}_\epsilon = \sigma_D I$ and $S_r \boldsymbol{m} = A_z \boldsymbol{m}$ is the magnitude of the depth dependent flow for a scene of unit depth. The term $\|S_c \boldsymbol{b}_\epsilon\|^2 = \|\sigma_D I \boldsymbol{b}_\epsilon\|^2$ reduces to $\sigma_D^2 \sigma_b^2$. Then we can rewrite the expected error in the motion estimate as

$$\|\hat{\boldsymbol{m}} - \boldsymbol{m}\|_S = \sqrt{\sigma_D^2 \|C^{-1}(DA_z + A_\omega)^{\mathrm{T}}\|_F^2 \|A_z \boldsymbol{m}\|^2 + \sigma_D^2 \sigma_b^2 \|A_z C^{-1}\|_F^2} \tag{5.25}$$

We can see that the amplification of the noise, and thus the sensitivity of the motion estimation to errors in the depth and in the flow, depends on the matrix $C = (A^{\mathrm{T}} A)$. How much C^{-1} inflates the errors depends on the eigenvalue distribution of the matrix C, which in turn is determined by the field of view of the camera. The larger the field of view, the smaller will be the condition number of C. If A is well-conditioned, i.e. the ratio between its largest and smallest singular values is close to 1, then the solution $\boldsymbol{m}$ will not be sensitive to small perturbations D_ϵ and b_ϵ. If instead A is badly conditioned, i.e. A is close to singular and therefore some eigenvalues of C^{-1} are very large, then small errors in the measurements can cause large errors in the motion estimate. Therefore, the effect of the field of view on the sensitivity of the motion estimation can be analyzed by examining how the singular values of the matrix A depend on the field of view.

We performed some synthetic experiments, where we simulated four different spatial arrangements of polydioptric cameras (see Fig. 5.5) similar to possible polydioptric camera configurations. For each individual camera we defined the imaging surface to be the set of rays that made an angle of less than α degrees with the optical axis, where α was varied to simulate different fields of view. The sum of the measurements for the whole system of cameras was kept constant to make the results for the different setups comparable. The same experiments also tell us about the influence of the field of view in the case of single-viewpoint cameras with known depth. For a forward-looking camera (Fig. 5.5 a) and forward and backward looking cameras (Fig. 5.5 b), two of the singular values vanish for a small field of view, implying that the estimation of the motion parameters is ill-posed. If we increase the field of view the linear system becomes better and better conditioned. When we arrange the cameras so that they face in perpendicular directions (Figs. 5.5 c, d), we see that the conditioning of the linear system is nearly independent of the fields of view of the individual cameras, suggesting that motion estimation using a configuration of conventional planar cameras pointing in orthogonal direction is as robust as when using an ideal spherical eye.

To give some geometric intuition, we write the motion constraint on the sphere as

$$-L_t = \nabla_r L \cdot \frac{\boldsymbol{t}}{|\boldsymbol{R}|} + (\boldsymbol{r} \times \nabla_r L) \cdot \boldsymbol{\omega} \tag{5.26}$$

Since $\nabla_r L$ is perpendicular to $(\boldsymbol{r} \times \nabla_r L)$, for a small field of view ($\boldsymbol{r}$ varies very little) and little variation in depth, a translational error $\boldsymbol{t}_\epsilon$ can be compensated by a rotational error $\boldsymbol{\omega}_\epsilon$ without violating the constraint in Eq. 5.26 as long as the errors have the following relationship:

$$\frac{1}{|\boldsymbol{R}|}\boldsymbol{r} \times \boldsymbol{t}_\epsilon = -\boldsymbol{r} \times (\boldsymbol{r} \times \boldsymbol{\omega}_\epsilon). \tag{5.27}$$

That is, the projections of the translational and rotational errors on the tangent plane to the sphere at $\boldsymbol{r}$ need to be perpendicular. This is known as the orthogonality constraint on the plane [46]. If we now increase the field of view, the constraint on the errors in Eq. 5.27 cannot be satisfied for all $\boldsymbol{r}$, thus the confusion disappears.

There is another ambiguity. Looking at the first term in Eq.5.26, that is $\nabla_r L \cdot \boldsymbol{t}/|\boldsymbol{R}|$, we see that the component of $\boldsymbol{t}$ parallel to $\boldsymbol{r}$ does not factor into the equation (since $\nabla_r L \cdot \boldsymbol{r} = 0$) and therefore cannot be recovered from the projection onto the gradients for a small field of view. We call this the line constraint on the plane, because the projections of the actual $\boldsymbol{t}$ (FOE) and the estimated $\tilde{\boldsymbol{t}} = \boldsymbol{t} + \lambda \boldsymbol{r}, \lambda \in \mathbb{R}$ onto the image plane lie on a line through the image center. Again an increase in the field of view will eliminate this ambiguity, since then measurements at other image locations enable us to estimate the component of $\boldsymbol{t}$ parallel to $\boldsymbol{r}$.

5.9 Stability Analysis using the Cramer-Rao lower bound

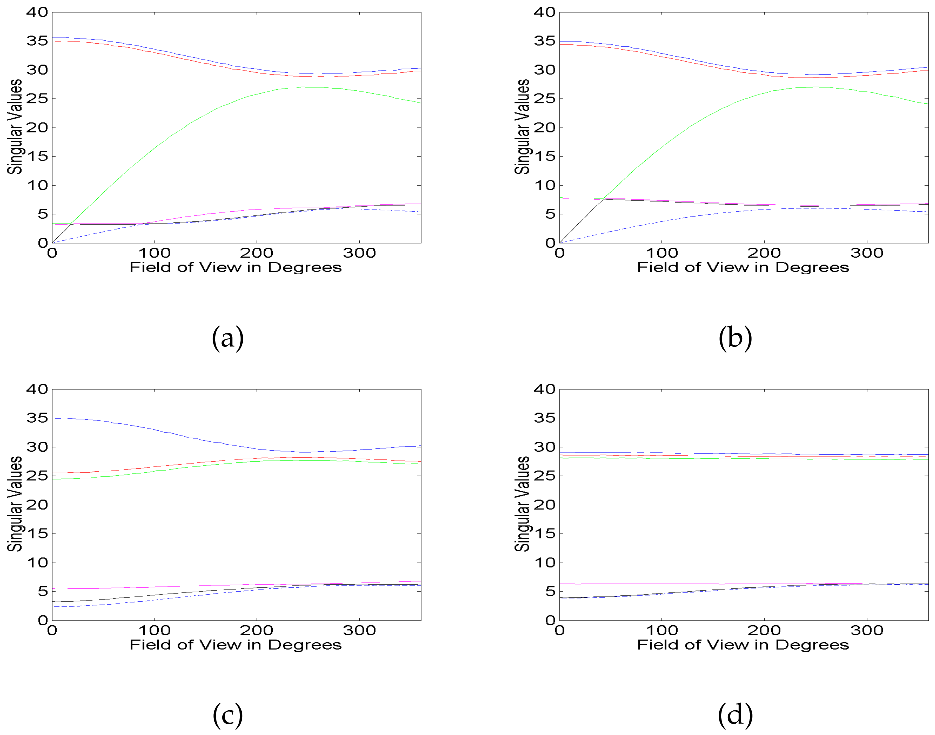
The sensitivity of an estimation problem that was geometrically motivated in the previous section can be more accurately characterized by the Fisher-Information matrix of the system which is defined as

$$F = E\left[\frac{\partial \ln(\boldsymbol{h}|\boldsymbol{p})}{\partial \boldsymbol{p}}^{\mathrm{T}} \frac{\partial \ln(\boldsymbol{h}|\boldsymbol{p})}{\partial \boldsymbol{p}}\right] \tag{5.28}$$

where $\boldsymbol{h}$ are the measurements of the system (the optic flow $\dot{\boldsymbol{r}}$ or the plenoptic flow $[\dot{\boldsymbol{r}}, \dot{\boldsymbol{x}}]$, and $\boldsymbol{p} = [\dot{\boldsymbol{q}}; \boldsymbol{\omega}]$ are the parameters. Thus it measures how much the probability of a given measurement changes for a change in the model parameters. The Fisher-information matrix is used in Cramer-Rao inequality that states that for an unbiased estimator the covariance matrix is bounded from below by the inverse of the Fisher information matrix:

$$E\left[(\boldsymbol{p} - \hat{\boldsymbol{p}})^{\mathrm{T}}(\boldsymbol{p} - \hat{\boldsymbol{p}})\right] \geq F^{-1} \tag{5.29}$$

If the error in the measurements follows a normal distribution with zero mean and



(a)

(b)

(c)

(d)

Figure 5.5: Singular values of matrix A for different camera setups in dependence on the field of view of the individual cameras:

(a) single camera, (b) two cameras facing forward and backward, (c) two cameras facing forward and sideways, and (d) three cameras facing forward, sideways, and upward

.

all the measurements are independent, then the Fisher information matrix has the simple expression

$$F = \frac{1}{\sigma^2} \sum_i \frac{\partial h_i^{\mathrm{T}}}{\partial p} \frac{\partial h_i}{\partial p} \tag{5.30}$$

Thus, if there is an easy relationship between the optical flow in the image and the motion parameters, we can numerically integrate over the distribution of scene parameters and camera parameters to estimate the covariance of the estimation. A similar analysis has been done for the case of planar scene structure in [34] and for the case of generalized cameras using Plücker coordinates in [116]. We will use it to characterize the uncertainty in terms of the field of view and the camera spacing.

Since our parameters $\boldsymbol{\omega}$ and $\boldsymbol{t}$ are linearly related to the spherical motion flow, the Jacobian $\partial \boldsymbol{h}_i / \partial \boldsymbol{p}$ in the Fisher information matrix is simply given by the motion matrices M_i in Table 5.3. Thus the Fisher information matrix can be written as the sum over all the cameras and measurements

$$F_{t,\omega} = \sum_{c_i} \sum_{p_i} M^{\mathrm{T}}_{(c_i,p_i)} M_{(c_i,p_i)} \tag{5.31}$$

where c_i is the camera index and p_i are the point indices.

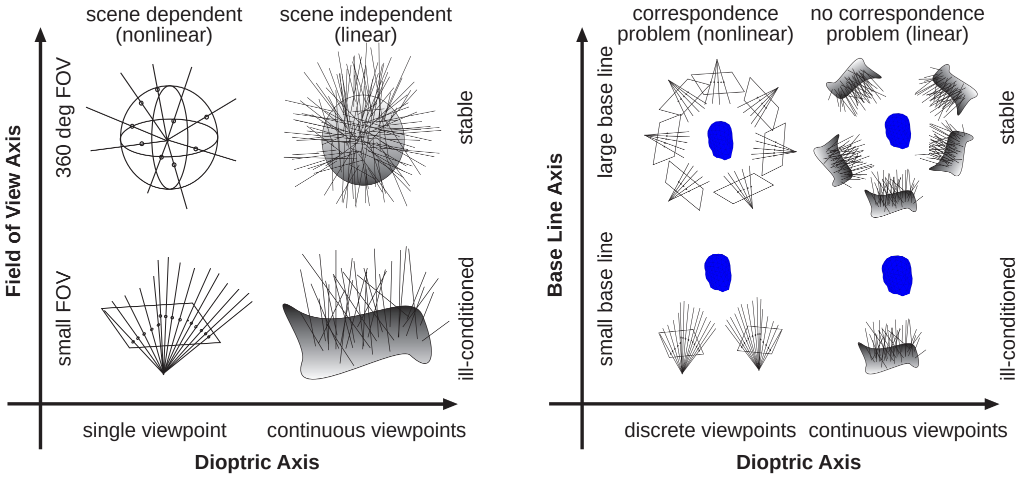
It is still very difficult to evaluate this expression analytically, since it involves the scene structure, but if we assume a certain distribution of values, similar to [116] we can numerically estimate the covariance matrix over the samples.

We collected all the expressions for the Fisher information matrix using different camera parameterizations in table 5.5.

This framework is based on a systematic study of the relationship between the shape of an imaging sensor and the task performance of the entity using this sensor.

Combining the two criteria, the field of view and the subset of the space of light rays that a sensor captures, we can rank different eye design in a hierarchy as shown in Fig. 5.6 which expresses a qualitative measure of how hard the task of motion estimation is to solve for a given sensor design [105].

One can see in the figure that the conventional pinhole camera is at the bottom of the hierarchy because the small field of view makes the motion estimation ill-posed and it is necessary to estimate depth and motion simultaneously. Although the estimation of the 3D motion for a single-viewpoint spherical camera is stable and robust, it is still scene-dependent, and the algorithms which give the most accurate results are search techniques, and thus rather elaborate. One can conclude that a spherical polydioptric



(a) (b)

Figure 5.6: (a) Hierarchy of Cameras for 3D Motion Estimation and (b) 3D Shape estimation. The different camera models are classified according to the field of view (FOV) and the number and proximity of the different viewpoints that are captured (Dioptric Axis). The camera models are clockwise from the lower left: small FOV pinhole camera, spherical pinhole camera, spherical polydioptric camera, and small FOV polydioptric camera.

camera is the camera of choice to solve the 3D motion estimation problem since it combines the stability of full field of view motion estimation with the linearity and scene independence of the polydioptric motion estimation.

5.10 Experimental validation of plenoptic motion estimation

It is difficult to experimentally validate the power of the plenoptic motion constraint under real world conditions because camera technology is not yet available at a scale that allows for a real-world implementation of insect compound eye-inspired camera systems. Tp experimentally validate different camera designs we need to build and modify different hardware camera configurations. Since this was infeasible in the course of this thesis, we will utilize the following three types of experiments to assess the performance of the plenoptic motion estimation framework.

1. Use computer graphics to generate polydioptric image sequences for different camera configurations and motions and evaluate against ground truth,

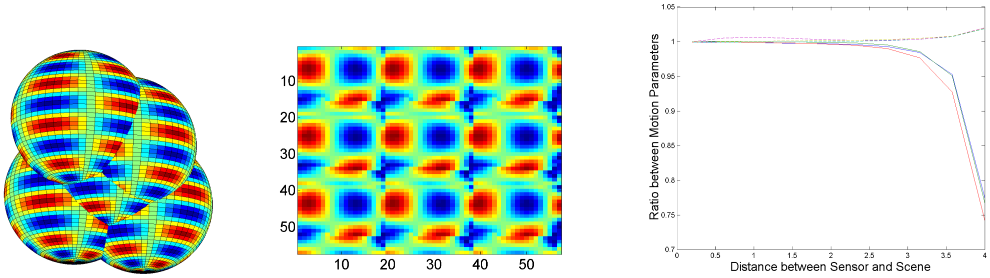
2. Capture an epipolar image volume of a scene and generate new polydioptric image sequences by resampling this volume for different camera configurations and motions and evaluate against ground truth,

3. build a polydioptric camera and capture a few polydioptric sequences and evaluate motion accuracy based on reprojection error of features.

5.10.1 Polydioptric sequences generated using computer graphics

To examine the performance of an algorithm using the plenoptic motion constraint, we did experiments with synthetic data. We distributed spheres, textured with a smoothly varying pattern, randomly in the scene according to natural scene statistics [64] so that they filled the horizon of the camera (see Fig. 5.7a). We then computed the light fields for all the faces of the nested cube surrounding the camera through ray tracing, computed the derivatives, stacked the linear equations (Eq. 5.6) to form a linear system, and solved for the motion parameters. Even using derivatives only at one scale, we find that the motion is recovered very accurately as seen in Fig. 5.7c. As long as the relative scales of the derivatives are similar enough (scene not too far away) the error in the motion parameters varies between 1% and 3%.

We used the scene description language Renderman to describe artificial scenes consisting of planes at different distances surrounding the camera from all sides. The camera was moved with a known motion to generate short polydioptric image sequences. The resolutions in the perspective images ranged from 16 by 16 pixels to 256 by 256 pixels, and the number of images ranged from 5 by 5 arrays to 64 by 64. Different fields of view were generated by having identical cameras face in different directions. All the images were generated in floating point format to avoid problems with dynamic range. An example scene is shown in Figure 5.7a.



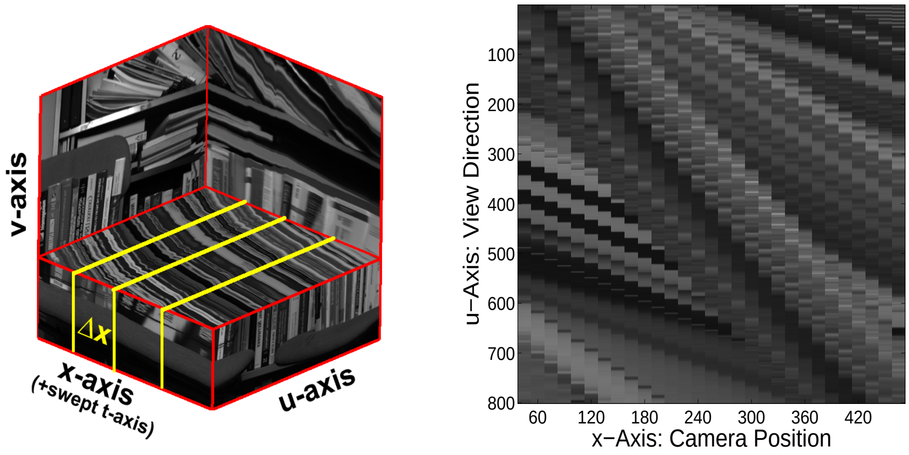
(a) (b) (c)

Figure 5.7: (a) Subset of an Example Scene, (b) the corresponding light field (c) Accuracy of Plenoptic Motion Estimation. The plot shows the ratio of the true and estimated motion parameters (vertical axis) in dependence of the distance between the sensor surface and the scene (horizontal axis) for $f = 60$ and spheres of unit radius.

5.10.2 Polydioptric sequences generated by resampling epipolar volumes

Another approach to simulating and evaluating camera designs without actually building them is based on resampling previously recorded subsets of the space of light rays (see Fig. 5.8).

We captured a number of epipolar volumes [14] of different scenes which varied in depth and texture complexity. Given such a continuous subset of the plenoptic function



(a) (b)

Figure 5.8: Evaluation of camera designs by resampling epipolar volumes. (a) By sampling brightness values in the epipolar volume at different spacings along the view dimensions we can simulate a variety of camera arrangements. (b) Example: 29 camera system with noticeable aliasing between the views.

we are able to generate new line-camera image sequences by resampling this set of light rays. These generated image sequences are essentially identical to the image sequences that a true physical line camera would capture as long as the camera motion is chosen such that all pixels of the camera can be interpolated from the voxels of the epipolar plane volume. By varying the spatio-temporal sampling pattern we can simulate a wide range of camera motions as well as camera designs. We generated a large number of image sequences for various camera motions and distances between the camera centers. For each frame of a sequence we formed the plenoptic motion constraint equations (consisting in this case of the rows of Eq. (5.7) corresponding to a planar motion) and solved for the planar motion parameters using the plenoptic derivatives. As an example result, we show in Fig. 5.9 how the accuracy of the rotation and translation estimates improve when the field of view increases, and how the accuracy decreases when the spacing between

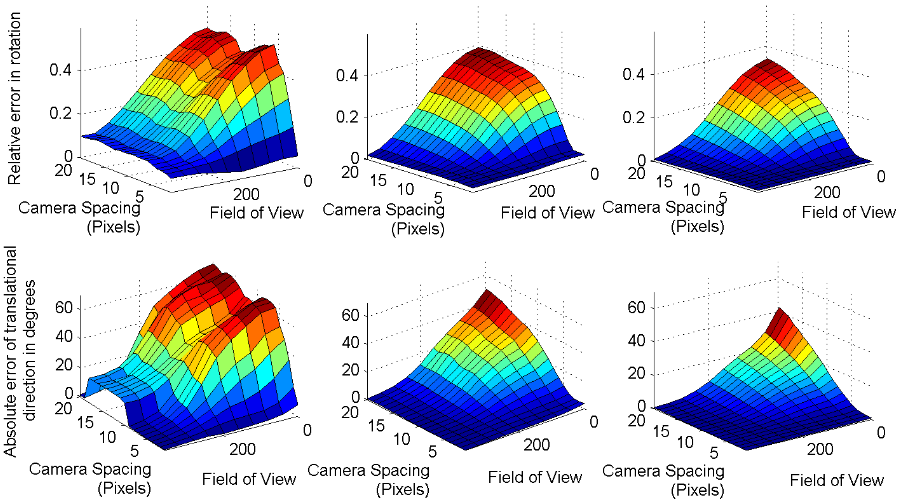


Figure 5.9: Relationship between camera spacing, image smoothing and accuracy of motion estimation based on integrating over many image sequences generated from an epipolar volume. The standard deviation of the Gaussian smoothing filter increases from left to right from 1 over 5 to 11 pixels.

the cameras increases. We also see how thanks to the inherent redundancy of the data, the accuracy and robustness of the estimation increases noticeably when we increase the amount of smoothing in the perspective images. We can use the error analysis to derive the best interpolation and smoothing filters based on the depth that we recovered.

5.10.3 Polydioptric sequences captured by a multi-camera rig

We used the multi-perspective camera concepts described in this paper to recover shape from real world sequences. We built a polydioptric camera consisting of two linear arrays of cameras looking at perpendicular directions. This camera configuration was moved in a planar motion in front of an office scene (Fig. 5.10a). Using the variational motion estimation described in Section 5.4, we estimated the motion based on the plenoptic derivatives and were able to compute accurately rectified epipolar image volumes (Fig. 5.10b). Finally, we used the recovered motion to segment the scene according to depth and recovered the position of the main objects in the scene (Fig. 5.10c).

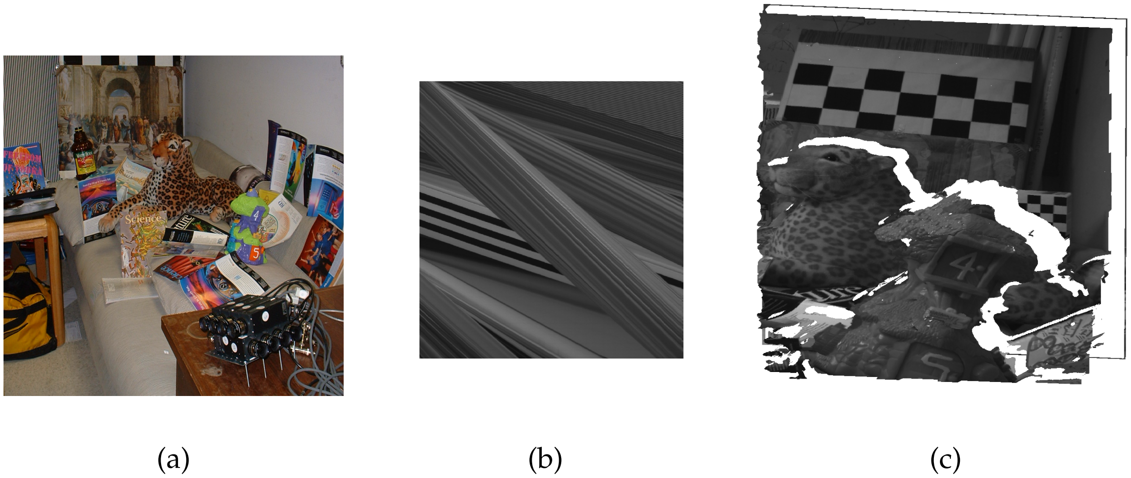

(a) (b) (c)

Figure 5.10: (a) Example indoor scene for motion estimation. The camera can be seen on the table. It was moved in a planar motion on the table. (b) The epipolar image that was recovered after compensating for varying translation and rotation of the camera. The straight lines indicate that the motion has been accurately recovered. (c) Recovered depth model.

We also build a camera rack and captured polydioptric image sequences outdoors (Fig. 5.11) where the camera rack consisted of two camera clusters of five and four cameras each which were facing in orthogonal directions. Since the individual clusters do not share a common view we needed to calibrate this polydioptric camera using the Keck laboratory. The image sequences were radiometrically calibrated to enable the computation of accurate derivatives. We then computed the plenoptic derivatives. The view direction derivatives were computed for each image independently and the view point derivatives were computed by fitting the best linear function to the brightness values of the pixels in the individual cameras facing in the same direction. Unfortunately, we were not able to record ground truth information about the camera motion, therefore the evaluation of the accuracy of the recovered motion was done by comparing the path of the camera with the path of features in the images such as the light pole to right.

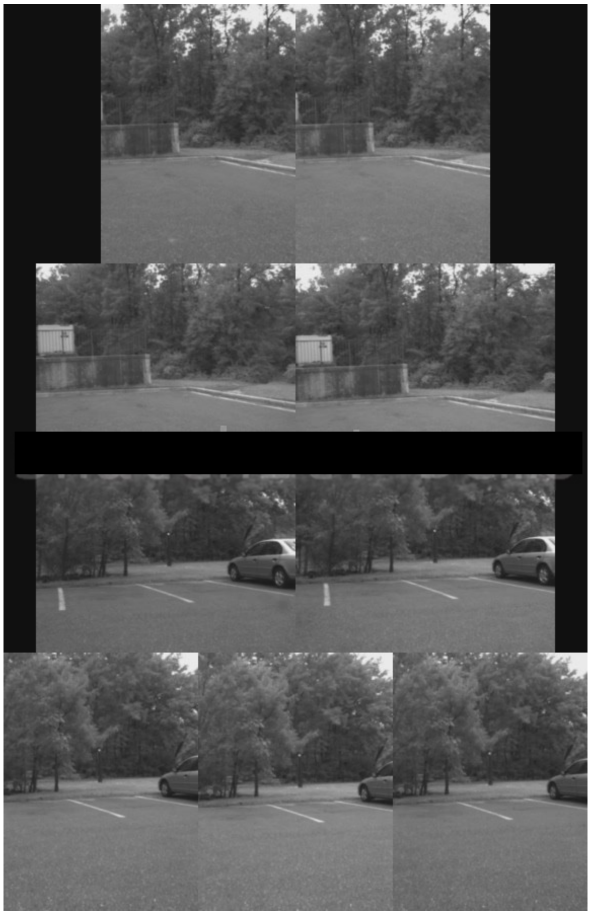


Figure 5.11: Example outdoor scene. The camera was moved in a straight line while being turned.

5.10.4 Comparison of single view and polydioptric cameras

To assess the performance of different camera models with regard to motion estimation, we compare a number of standard algorithms for ego-motion estimation as described in [145] against a multi-camera stereo system. We used Jepson and Heeger's linear sub-

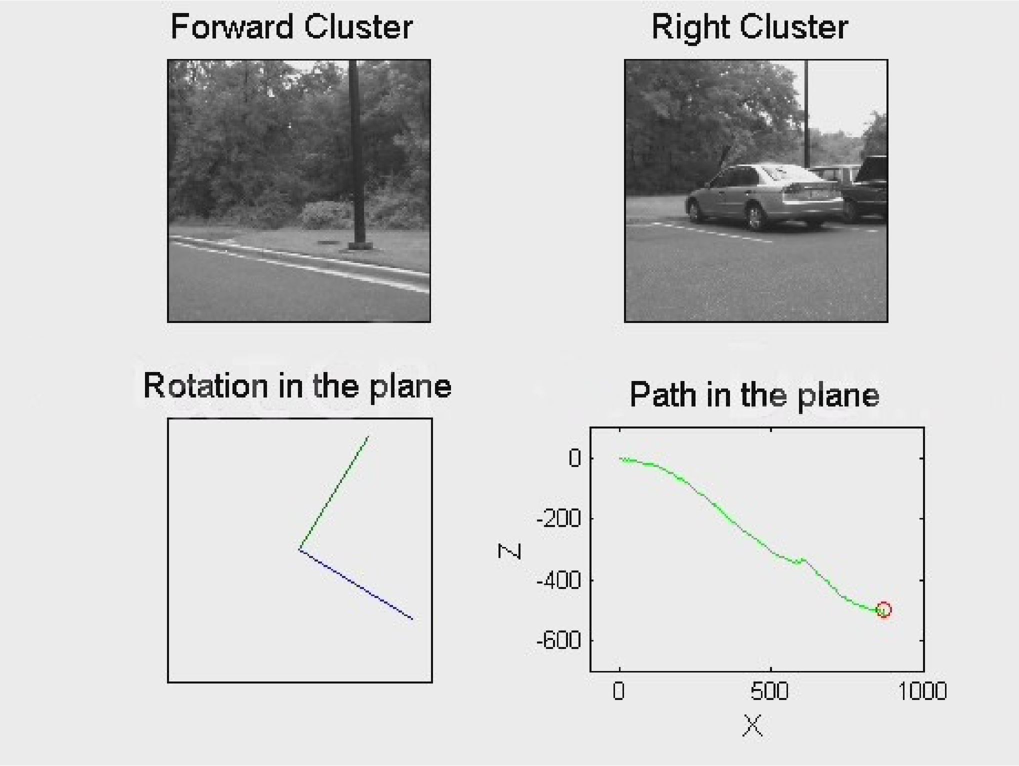


Figure 5.12: Motion estimation results for example outdoor scene. The comparison of the parking lot geometry with the recovered path indicates that the motion was accurately recovered.

space algorithm and Kanatani's normalized minimization of the epipolar constraint. We assume similar error distributions for the optical flow and disparity distributions, both varying from 0 to 0.04 radians (0-2 degrees) in angular error. This corresponds to about a pixel error of 0 to 4.5 pixels in a 256x256 image. We ran each algorithm 100 times on different randomly generated flow vectors and point clouds, and measured the angular deviation from the true translation and rotation. The results in Figure 5.13 demonstrate that for a similar distribution of errors in the disparities and the optical flow, solving a linear system with approximate depth knowledge outperforms the algorithms that algebraically eliminate the depth from the equation noticeably.

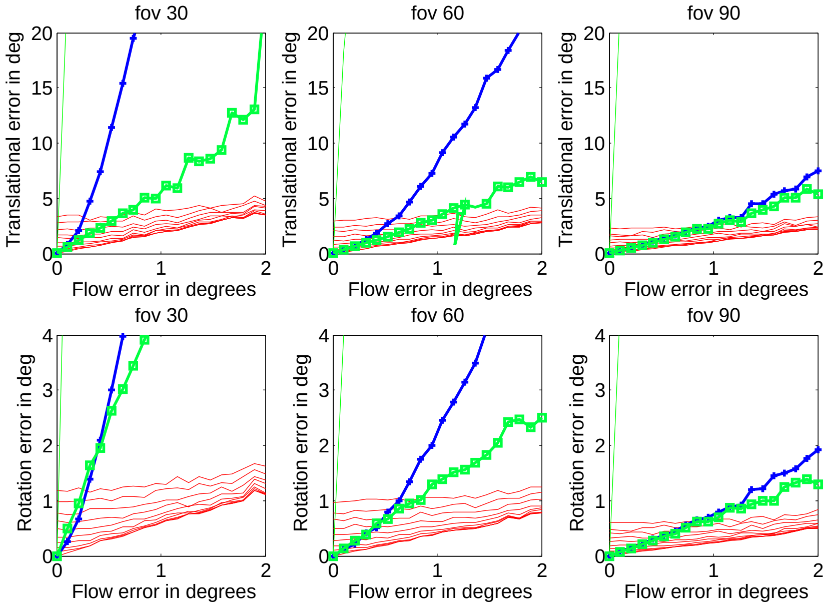


Figure 5.13: Comparison of motion estimation using single and multi-perspective cameras. The errors in the correspondences varied between 0 and 0.04 radians (0-2 degrees), and we computed the mean squared error in the translation and rotation direction for 3 different camera field of views (30,60, and 90 degrees). The blue line (+) and green line (squares) are Jepson-Heeger's subspace algorithm and Kanatani's normalized minimization of the instantaneous differential constraint as implemented by [145]. The red lines denote the performance of motion estimation using Eq. (5.19) where the errors in the disparities are normally distributed with a standard deviation that varies between 0 and 5 degrees.

The effect of the camera on depth estimation, can similarly analyzed. To estimate the depth from a motion estimate, we can invert Eq. (5.19) pointwise to get:

$$Z = \frac{A_q \boldsymbol{q} + A_c \boldsymbol{\omega}}{\boldsymbol{b} - A_\omega \boldsymbol{\omega}} \tag{5.32}$$

For the case of stereo, we have $\boldsymbol{\omega} = 0$, and the equation reduces to $Z = \frac{\boldsymbol{r}_\perp^\top [\boldsymbol{r}]_\times^2 \boldsymbol{q}}{\boldsymbol{r}_\perp^\top \boldsymbol{b}}$. It has been observed before that if we have errors in the motion estimation, then the reconstructed shape will be distorted [7]. This necessitates the use of Kalman filters or sophisticated fusion algorithms. By using polydioptric cameras, we can solve the correspondence problem easily due to the small baseline, and at the same time, since we know the calibrated imaging geometry, we can estimate the local depth models with greater accuracy. This improves the motion estimation and allows for easier correspondence over larger baselines. Finally, stochastic fusion algorithms such as described in [29] can be used to integrate the local depth estimates.

5.11 Summary

In this chapter we applied the framework of polydioptric motion estimation to the problem of 3D camera motion estimation. We used our previous analysis of the geometry of the plenoptic function to derive the constraints that relate the motion of a camera to the measurements in the images. Besides the well-known constraints relating different views of the same scene point, we also described the novel constraint relating different views of the same light ray. This lead to discrete and differential plenoptic motion constraints. The power of these constraints was experimentally demonstrated using synthetic and real images sequences. Finally, we showed how we can use the sampling analysis from Chapter 4 to devise smoothing filters for the perspective images that increase the accuracy of the motion estimation noticeably.

<table>
<tr><th>Camera Model</th><th>Motion Flow Equation</th></tr>
<tr><td>Spherical Pinhole
$\boldsymbol{r}$: ray direction
$\boldsymbol{x}$: ray origin
Z: depth along ray</td><td>Discrete:
$$r(t) = \frac{\boldsymbol{q}(t)+R(t)\boldsymbol{x}_0+ZR(t)\boldsymbol{r}_0}{\hat{\boldsymbol{z}}^{\mathrm{T}}(\boldsymbol{q}(t)+R(t)\boldsymbol{x}_0+ZR(t)\boldsymbol{r}_0)}$$
Differential:
$$\dot{\boldsymbol{r}} = \underbrace{\left[\begin{array}{c|c} -\frac{[\boldsymbol{r}]_\times^2}{Z} & [\boldsymbol{r}]_\times + \frac{[\boldsymbol{r}]_\times^2}{Z}[\boldsymbol{x}]_\times \end{array}\right]}_{M_{\text{pinhole}}} \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{bmatrix}$$</td></tr>
<tr><td>Plücker Lines
$\boldsymbol{r}$: ray direction
$\boldsymbol{m}$: moment vector</td><td>Discrete:
$$\begin{bmatrix} \boldsymbol{r}(t) \\ \boldsymbol{m}(t) \end{bmatrix} = \left[\begin{array}{c|c} R(t) & 0_3 \\ -\left[\boldsymbol{q}(t)\right]_\times R(t) & R(t) \end{array}\right] \begin{bmatrix} \boldsymbol{r}_0 \\ \boldsymbol{m}_0 \end{bmatrix}$$
Differential:
$$\begin{bmatrix} \dot{\boldsymbol{r}} \\ \dot{\boldsymbol{m}} \end{bmatrix} = \underbrace{\left[\begin{array}{c|c} 0 & [\boldsymbol{r}]_\times \\ [\boldsymbol{r}]_\times & [\boldsymbol{m}]_\times \end{array}\right]}_{M_{\text{Plücker}}} \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{bmatrix}$$</td></tr>
<tr><td>Polydioptric Parameterization
$\boldsymbol{r}$: ray direction
$\boldsymbol{x}$: ray origin
$\boldsymbol{n}$: normal to
projection plane</td><td>Discrete:
$$\begin{bmatrix} \boldsymbol{r}(t) \\ \boldsymbol{x}(t) \end{bmatrix} = \begin{bmatrix} R(t)\boldsymbol{r}_0 \\ -\left[\boldsymbol{n}\right]_\times^2 R(t)\boldsymbol{x}_0 \end{bmatrix} + \begin{bmatrix} 0 \\ -\left[\boldsymbol{n}\right]_\times^2 \boldsymbol{q}(t) \end{bmatrix}$$
Differential:
$$\begin{bmatrix} \dot{\boldsymbol{r}} \\ \dot{\boldsymbol{x}} \end{bmatrix} = \underbrace{\left[\begin{array}{c|c} 0 & [\boldsymbol{r}]_\times \\ -[\boldsymbol{n}]_\times^2 & [\boldsymbol{n}]_\times^2 [\boldsymbol{x}]_\times \end{array}\right]}_{M_{\text{polydioptric}}} \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\omega} \end{bmatrix}$$</td></tr>
</table>

Table 5.3: Motion Flow Constraint Equations for Rigid Motion Estimation

Camera Model	Motion Constraint Equation
Spherical Pinhole $\boldsymbol{r}$: ray direction $\boldsymbol{x}$: ray origin	$-\mathcal{L}_t = \nabla_r\mathcal{L}\left[\begin{array}{c\|c} -\frac{I_3}{\\|\boldsymbol{R}\\|} & [\boldsymbol{r}]_\times + \frac{[\boldsymbol{x}]_\times}{\\|\boldsymbol{R}\\|}\end{array}\right]\begin{bmatrix}\dot{\boldsymbol{q}}\\ \boldsymbol{\omega}\end{bmatrix}$
Plücker Lines $\boldsymbol{r}$: ray direction $\boldsymbol{m}$: moment vector	$-L_t = \begin{bmatrix}\nabla_r L\\ \nabla_m L\end{bmatrix}\left[\begin{array}{c\|c} 0 & [\boldsymbol{r}]_\times\\ [\boldsymbol{r}]_\times & [\boldsymbol{m}]_\times\end{array}\right]\begin{bmatrix}\dot{\boldsymbol{q}}\\ \boldsymbol{\omega}\end{bmatrix}$
Polydioptric Camera $\boldsymbol{r}$: ray direction $\boldsymbol{x}$: ray origin	$-\mathcal{L}_t = \begin{bmatrix}\nabla_r\mathcal{L}\\ \nabla_x\mathcal{L}\end{bmatrix}\left[\begin{array}{c\|c} 0 & [\boldsymbol{r}]_\times\\ -[\boldsymbol{n}]^2_\times & [\boldsymbol{n}]^2_\times[\boldsymbol{x}]_\times\end{array}\right]\begin{bmatrix}\dot{\boldsymbol{q}}\\ \boldsymbol{\omega}\end{bmatrix}$

Table 5.4: Brightness constancy constraint equations for rigid motion estimation. Since we have the brightness invariance along the ray that can be expressed as $\nabla_r\mathcal{L}^{\mathrm{T}}\boldsymbol{r} = \nabla_x\mathcal{L}^{\mathrm{T}}\boldsymbol{r} = 0$ and $\nabla_r L^{\mathrm{T}}\boldsymbol{r} = \nabla_m L^{\mathrm{T}}\boldsymbol{r} = 0$, we can omit the projection operators in the constraint equations.

Camera Model	Fisher Information Matrix $\left[\begin{smallmatrix} A & B \\ B^{\mathrm{T}} & C \end{smallmatrix}\right]$
Spherical Pinhole $\boldsymbol{r}$: ray direction $\boldsymbol{X}$: ray origin	$A = [\boldsymbol{r}]_\times^2/\|\boldsymbol{R}\|^2$ $B = [\boldsymbol{r}]_\times^2[\boldsymbol{X}]_x/\|\boldsymbol{R}\|^2 + [\boldsymbol{r}]_\times/\|\boldsymbol{R}\|$ $C = [\boldsymbol{X}]_\times[\boldsymbol{r}]_\times^2[\boldsymbol{X}]_\times/\|\boldsymbol{R}\|^2$ $+([\boldsymbol{X}]_\times[\boldsymbol{r}]_\times + [\boldsymbol{r}]_\times[\boldsymbol{X}]_\times)/\|\boldsymbol{R}\| + [\boldsymbol{r}]_\times^2$
Plücker Lines $\boldsymbol{r}$: ray direction $\boldsymbol{m}$: moment vector	$A = [\boldsymbol{r}]_\times^2$ $B = -[\boldsymbol{r}]_\times[\boldsymbol{m}]_x$ $C = [\boldsymbol{r}]_\times^2 + [\boldsymbol{m}]_\times^2$
Polydioptric Camera $\boldsymbol{r}$: ray direction $\boldsymbol{X}$: ray origin $\boldsymbol{X}^\perp$: normal to camera surface	$A = [\boldsymbol{r}]_\times^2$ $B = -[\boldsymbol{X}^\perp]_\times^2[\boldsymbol{X}]_\times$ $C = [\boldsymbol{r}]_\times^2 - [\boldsymbol{X}]_\times[\boldsymbol{X}^\perp]_\times^2[\boldsymbol{X}]_\times$

Table 5.5: Single term $M_{c_i,p_i}^{\mathrm{T}} M_{c_i,p_i}$ of the Fisher Information Matrix for Rigid Motion Estimation

Chapter 6

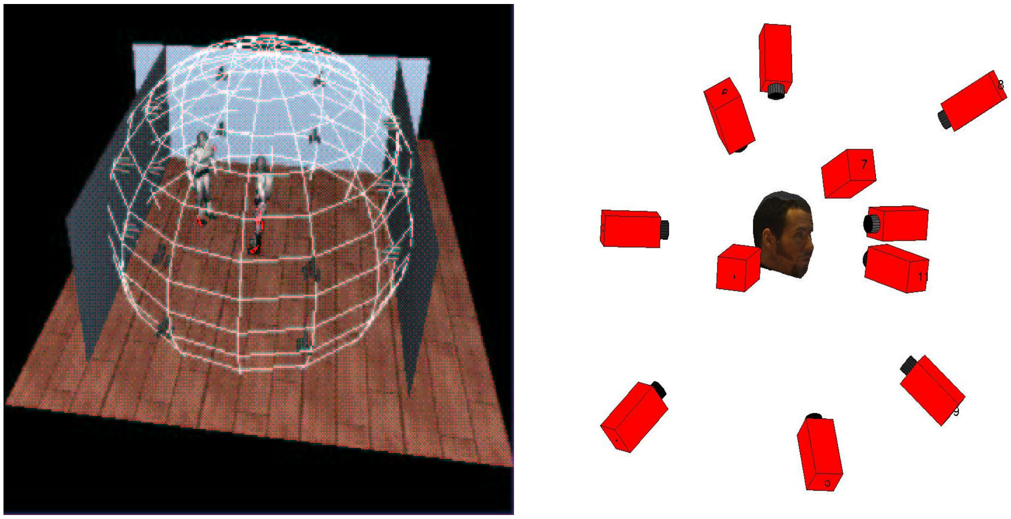
Application: Spatio-temporal Reconstruction from Mulitple Video Sequences

The concept of three-dimensional(3D) photography and imaging has always been of great interest to humans. Early attempts to record and recreate images with depth were the stereoscopic drawings of Giovanni Battista della Porta around 1600, and the stereoscopic viewers devised by Wheatstone and Brewster in the 19th century. As described in [107], in the 1860's Francois Villème invented a process known as photo sculpture, which used 24 cameras, to capture the notion of a three-dimensional scene. Later a three-dimensional photography and imaging technique was invented by G. Lippmann in 1908 under the name of integral photography where the object was observed by a large number of small lenses arranged on a photographic sheet resulting in many views of the object from different directions [110]. Today modern electronic display techniques enable the observer to view objects from arbitrary view points and explore virtual worlds freely. These worlds need to be populated with realistic renderings of real life objects to give the observer the feel of truly spatial immersion. This need fuels the demand for accurate ways to recover the 3D shape and motion of real world objects. In general, the approaches to recover the structure of an object are either based on active or passive vision sensors, i.e. sensors that interact with their environment or sensors that just observe without interference. The main examples for the former are laser range scanners [125] and structured light based stereo configurations where a pattern is projected onto the scene and the sensor uses the image of the projection on the structure to recover depth from triangulation [9, 36, 164]. For a recent overview over different approaches to active range sensing and available commercial systems see [10]. The category of passive approaches consists of stereo algorithms based on visual correspondence where the cameras are separated by a large baseline [126] and structure from motion algorithms on which we will concentrate [54, 88]. Since correspondence is a hard problem for widely separated views, we believe that the structure from motion paradigm offers the best approach to 3D photography [117, 48], because it interferes the least with the scene being imaged and the recovery of the sensor motion enables us to integrate depth information from far apart views for greater accuracy while taking advantage of the easier correspondence due to dense video. In addition, the estimation of the motion of the camera is a fundamental component of most image-based rendering algorithms.

Having recovered the camera positions using the polydioptric motion estimation algorithm, we can use this calibration information to extract spatio-temporal representations of the scene observed. A spatio-temporal representation that captures all the essential shape and motion information is necessary to understand, simulate and copy an activity of a person or an object in the scene. This view-independent representation is made up by the 3D spatial structure as well as the 3D motion field on the object surface (also called range flow [136] or scene flow [152]) describing the velocity distribution on the object surface as described in Chapter 2. Such a representation of the shape and motion of an object is very useful in a large number of applications. Even for skilled artists it is very hard to animate computer graphics models of living non-rigid objects in a natural way, thus a technique that enables one to recover the accurate 3D displacements without the need for manual input, would reduce costs in animation production dramatically and add another dimension to the realism of the animation. Instead of having to simulate the natural motion dynamics by extensive parameter tweaking of complicated bone and muscle models, one will be able to extract the necessary information directly from video sequences.

Accurate 3D motion estimation is also of great interest to the fields of medicine and kinesiology where it can give rise to new diagnostic methods when the motion of athletes or patients is analyzed. Given access to a complete and accurate 3-dimensional model of a moving human, the performance of athletes can be examined with unparalleled ease and accuracy. In medicine the recovery of 3D motion fields could be used to diagnose strokes, by detecting differences in the motion patterns before and after the incident. The recovery of dense 3D motion fields will also help us to understand the human representation of action by comparing the information content of 3D and 2D motion fields for object recognition and task understanding.

We present a new algorithm that given calibrated image sequences computes accurately the 3D structure and motion fields of an object solely from image data utilizing silhouette and spatio-temporal image information without any manual intervention. There are many methods in the literature that recover three-dimensional structure information from image sequences captured by multiple cameras. Some example techniques are based on voxel carving (Kutulakos and Seitz [78], Seitz and Dyer [130], DeBonet and Viola [16], etc.), silhouette intersection (Matusik et al. [94, 133]) or multi-base line stereo (Vedula et al. [154], Fua [48], Faugeras and Keriven, etc. [43]), sometimes followed by a second stage to compute the 3D flow (Vedula et al. [152]) from optical flow information. Unfortunately, these approaches usually do not try to compute the motion and structure of the object simultaneously, thereby not exploiting the synergistic relationship between structure and motion constraints. Approaches that recover the motion and structure of an object simultaneously, most often depend on the availability of prior information such as an articulated human model (Kakadiaris and Metaxas [73], Plänkers and Fua [115]) or an animation mask for the face (Essa and Pentland [41]). Some of the few examples of combined structure and motion estimation that is not using a domain specific prior model are Zhang and Khambhamettu [165] as well as Malassiotis and Strintzis [90]. But in contrast to our approach the scene is still parameterized with respect to a base view resulting in a moving 2.5D surface, whereas we use an object space parameterization that represents true view-independent 3D information. In other related work, Vedula et al. [153] used a motion and voxel carving approach to recover both motion and structure, but due to the computational and memory demands of the 6D-voxel grid, their models and motion were of only low resolution. Our subdivision surface representation enables us to adapt the representation locally to the complexity of shape and motion. Borrowing terminology from signal processing, we can say that we optimize the different "frequency bands" of the mesh separately similar to multi-grid methods in numerical analysis. We note though that the term "frequency" is not well defined for meshes and should only be seen as an analogy. Recently, Carceroni and Kutulakos [25] presented an algorithm that computes motion and shape of dynamic surface elements to represent the spatio-temporal structure of a scene observed by multiple cameras under known lighting conditions. The adaptive subdivision hierarchy of our algorithm enables us to avoid the preset subdivision of the space into surface elements and to integrate measurements over neighborhoods that



(a) (b)

Figure 6.1: (a) Sketch of multi camera dome in the Keck Lab to capture a large environment and (b) calibrated camera setup for detailed capture of small objects

adapt to the complexity of the shape and motion.

The rest of this chapter is organized in 4 different sections. First, in section 6.1 we present a general description of the camera setup and the scene, justifying our assumptions that lead to the error criterion used later. Then, in the following section we explain the details of the multi-resolution subdivision framework consisting of the representation for the moving shape as well as the operators that transform a spatio-temporal structure into a multi-resolution representation and vice versa. Section 6.7 contains the initialization and refinement steps of the shape and motion estimation algorithm, before we conclude with the results.

6.1 Preliminaries

We will use the formalism described in section 2.1 to describe the shape and motion of the object in the scene. Our goal it is now to estimate the spatio-temporal surface $\mathcal{D}(\boldsymbol{x};t)$

as well as the motion vector field $\mathcal{F}(\boldsymbol{x};t)$ defined on its surface from image data alone without having to make any prior assumptions about shape and motion.

The algorithms presented here can be applied to large environments where the cameras are widely spread (see Figure 6.1a that displays a schematic description of our large baseline lab) and the aim is to reconstruct the interaction of people and objects, as well as to small environments where it is more important to capture subtle details of shape and motion as shown in Figure 6.1b. The experiments presented here were done in the small environment.

The camera configuration is parameterized by the camera projection matrices M_k that relate the world coordinate system to the image coordinate system of camera k. The calibration is done using a calibration frame. In the following we assume that the images have already been corrected for radial and tangential distortion. Therefore, the geometric mapping between points in space and the image is given by the conventional pinhole camera model. The surface point $\boldsymbol{P} = [x, y, z]^{\mathrm{T}}, \boldsymbol{P} \in \mathcal{S}$ in world coordinates projects to image point $\boldsymbol{p}_k$ in camera k which image coordinates are given by

$$\boldsymbol{p}_k = f\frac{M_k[\boldsymbol{P};1]}{M_k^3[\boldsymbol{P};1]} \tag{6.1}$$

where f is the focal length, M_k^3 is the third row of M_k, and $[\boldsymbol{P};1]$ is the homogeneous representation of $\boldsymbol{P}$.

The object surface is assumed to have Lambertian reflectance properties, thus following equation 2.4, the brightness intensity of a pixel $\boldsymbol{p}_k$ in camera k is given by (β_k is the constant that describes the brightness gain for each camera)

$$I(\boldsymbol{p}_k;t) = -\beta_k \cdot \rho(\boldsymbol{P}) \cdot [\boldsymbol{n}(\boldsymbol{P};t) \cdot \boldsymbol{s}(\boldsymbol{P};t)]. \tag{6.2}$$

To be able to identify unique points on the object surface, we assume that the albedo of a surface point $\rho(\boldsymbol{P})$ is constant over time ($d\rho/dt = 0$). In addition, since we record the video sequences with 60 frames per second, the illumination and orientation of the surface will change very little from frame to frame, thus we can assume total derivative of the irradiance on a pixel vanishes ($d/dt[\boldsymbol{n}(\boldsymbol{P};t) \cdot \boldsymbol{s}(\boldsymbol{P};t)] = 0$) which leads to the image brightness constancy constraint

$$-\frac{\partial I(\boldsymbol{p}_k)}{\partial t} = \nabla I(\boldsymbol{p}_k)^\top \cdot \frac{d\boldsymbol{p}_k}{dt}. \tag{6.3}$$

6.2 Multi-Resolution Subdivision Surfaces

There are many different ways to represent surfaces in space. Some examples are B-spline surfaces, deformable models, iso-surfaces or level-sets, and polygonal meshes (for an overview see [65]). Subdivision surfaces combine the ease of manipulating polygonal meshes with the implicit smoothness properties of B-spline surfaces. They are defined by a control mesh that determines the topology and shape of the object and a subdivision operator that determines how the mesh is refined. Repeated refinement will ultimately lead to a smooth limit surface. If we replace each static control point with a point trajectory that for example could be described by a subdivision curve we end up with a hierarchical multi-resolution representation for the spatio-temporal surface.

6.3 Subdivision

In our notation we follow Zorin et al. [169], who describe a system that enables the user to edit complex polygon meshes by manipulating different levels of resolution independently. Our goal is similar except that the user is not editing the shape, but the shape and motion are changed to optimize an error criterion. At each time instant the object surface $\mathcal{S}$ will be represented by a subdivision hierarchy of triangle meshes. On each level i of the hierarchy, there is a triangle mesh T^i that consists of a set of vertices V^i. Starting from an initial mesh T^0 with vertices V^0 that determines the topology of the shape at each time instant, we build a hierarchy of meshes $T^0, T^1, \ldots, T^i$ by successively refining each triangle into 4 sub-triangles. The vertex sets are nested, that is $V^j \subset V^i$ if $j < i$. We define *odd* vertices on level i as $O^i = V^{i+1} \setminus V^i$, thus V^{i+1} consists of two disjoint sets, the *even* vertices V^i and the *odd* vertices O^i. With each set of vertices V^i we can associate a map that relates vertices $v \in V^i$ to control point trajectories $\boldsymbol{c}^i(\cdot, t) : V^i \times \mathbb{R}_+ \rightarrow \mathbb{R}^3$ in the world. Thus for each vertex v, level i and time instant t, we get a 3D point $\boldsymbol{c}^i(v,t) \in \mathbb{R}^3$. The set $\boldsymbol{c}^i(t)$ contains all points at level i and time t and describes the shape of the mesh. The changes of $\boldsymbol{c}^i(t)$ over time determine the motion of the surface. We will drop the time dependence of $\boldsymbol{c}^i$ for the rest of this section, implicitly knowing that $\boldsymbol{c}^i$ is different at different times.

A subdivision scheme defines now a linear operator C^i that takes the points sets from a level i to a finer level $i+1 : \boldsymbol{c}^{i+1} = C^i \boldsymbol{c}^i$. If the subdivision scheme converges, we can define a limit surface $\boldsymbol{s} = \boldsymbol{c}^\infty = \prod\limits_{k=0}^{\infty} C^k \boldsymbol{c}^0$ where $\boldsymbol{s}(v,t)$ is the trajectory of vertex v on the surface of the shape. Examples for different subdivision schemes can be found in [128].

We choose the Loop-subdivision scheme (see [85]) for our purposes, because as an approximating subdivision scheme it smoothes the final surface and thereby regularizes the estimation process. In addition, the scheme is easy to implement because the refined position of a vertex depends only on the positions of its immediate neighbors and its limit surface can be analytically evaluated for arbitrary points on the surface.

We denote the K immediate neighbors (the 1-ring) of a vertex $v \in V^i$ by $v_k \in$

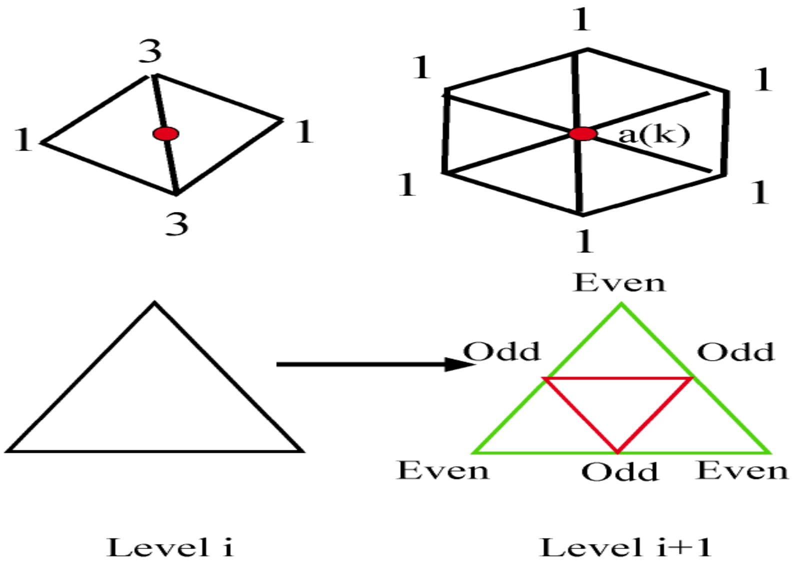


Figure 6.2: Stencils for Loop Refinement and 1-4 Triangle Split

$V^i, 1 \leq k \leq K$. We now define the new point $\boldsymbol{c}^{i+1}(v)$ as (see Figure 6.2):

$$\boldsymbol{c}^{i+1}(v) = \frac{a(K)\boldsymbol{c}^i(v) + \sum_{k=1}^{K} \boldsymbol{c}^i(v_k)}{a(K) + K} \tag{6.4}$$

$$a(K) = \frac{K(1 - \alpha(K)}{\alpha(k)} \tag{6.5}$$

$$\alpha(K) = \frac{5}{8} - \frac{(3 + 2\cos(\frac{2\pi}{K}))^2}{64} \tag{6.6}$$

For the odd vertices that are introduced at the midpoints, we use the stencil as described in Figure 6.2.

Since the refinement operator C^i is a linear operator, we can write this refinement process as matrix-vector multiplication (where the matrix is very sparse). Thus given the triangulation on level i, we split the control points $\boldsymbol{c}^i$ into the set of even $\boldsymbol{c}^i_e$ and odd c^i_o vertices and write the refinement equation as

$$\boldsymbol{c}^{i+1} = \begin{bmatrix} \boldsymbol{c}_e^i \\ \\ \boldsymbol{c}_o^i \end{bmatrix} = \underbrace{\begin{bmatrix} C_e^i \\ \\ C_o^i \end{bmatrix}}_{C_i} \boldsymbol{c}_i \tag{6.7}$$

The Loop scheme is a generalization of quartic box spline patches. This enables us to evaluate analytically the 3D position of each surface point in dependence of its control points similar to a parametric surface as was shown by Stam [137]. Given a triangle $r = (v_a, v_b, v_c) \in T^i$ corresponding to a patch $\boldsymbol{s}(r)$ on the object surface, and the surrounding control points $\boldsymbol{c}^i(r)$ defining the shape of the patch $\boldsymbol{s}(r)$ (the union of the 1-rings of v_a, v_b, and v_c) , then we can express the position of any mesh surface point $\boldsymbol{s}(u_1, u_2, u_3)$ that is defined by a set of barycentric coordinates $u_1, u_2, u_3 \in [0,1], u_1 + u_2 + u_3 = 1$ with respect to the triangle patch as a linear combination of the subdivision control points surrounding the patch

$$\boldsymbol{s}(u_1, u_2, u_3) = \boldsymbol{b}(u_1, u_2, u_3)^{\mathrm{T}} \boldsymbol{c}_i(r). \tag{6.8}$$

See Figure 6.3 for an illustration. The positions of points on the surface patch (indicated by the vertices of the tessellation in the center triangle) depend only on the positions of the vertices of the surrounding large triangles. Concatenating all the linear equations defining all the points over all the patches, we can form the limit matrix L^i, which enables us to write

$$\boldsymbol{s} = L^i \boldsymbol{c}^i \tag{6.9}$$

for an arbitrary tessellation of the smooth limit surface. Since the motion of the mesh is described by the trajectories $\boldsymbol{c}^i(t)$, the motion vector field on the mesh surface is given by

$$\frac{\partial \boldsymbol{s}(t)}{\partial t} = L^i \frac{\partial \boldsymbol{c}^i(t)}{\partial t}, \tag{6.10}$$

which is smoothly varying across surface.

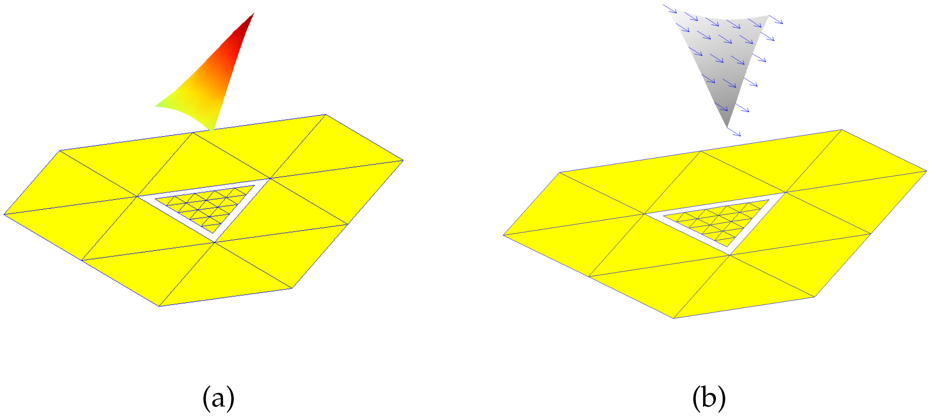


Figure 6.3: Neighbouring patches that influence the (a) 3D limit shape and (b) 3D limit motion field of the spatio-temporal surface

6.4 Smoothing and Detail Computation

After having defined the refinement operations that transform coarse meshes into finer, smoother versions of themselves, we now describe how we can construct the other direction of the multi-resolution transformation. To be able to analyze a mesh on different levels of resolution, we first need to have a linear operator that takes a fine mesh at level i and builds a smoothed, coarse version of it at level $i-1$

$$\boldsymbol{c}^{i-1} = H\boldsymbol{c}^{i}. \tag{6.11}$$

Many operators are possible, for example we could solve a linear system to find the best coarse mesh in the least-squares sense

$$\min_{\boldsymbol{c}^{i-1}} \|\boldsymbol{c}^{i} - C^{i-1}\boldsymbol{c}^{i-1}\|. \tag{6.12}$$

Unfortunately, this leads to a smoothing operator that is not sparse and local anymore and, therefore, is expensive to compute (inversion of a large non-sparse matrix). Similar problems arise if we solve for the coarse surface in the framework of a global variational problem (springs, thin-plate minimization). In this case we decided on the non-shrinking smoothing filter described by Taubin in [143] because of its computational simplicity. Since we are not dependent on an orthogonal decomposition of the surface we do not need to choose the dual to the Loop subdivision operator as our smoothing operator. Given a vertex $v \in V^i$ and its K neighbors $v_k \in V^i$ we can define the discrete Laplacian as

$$\mathcal{L} = \boldsymbol{s} - \frac{\sum\limits_{k=1}^{K} \boldsymbol{c}^i(v_k)}{K}. \tag{6.13}$$

Following Taubin, we can now define a smoothing operator H that acts similar to Gaussian smoother but does not exhibit shrinkage of the mesh by setting

$$H := (I - \mu\mathcal{L})(I - \lambda\mathcal{L}) \tag{6.14}$$

where μ and λ are constants that determine the properties of the filter. We chose the standard values of $\mu = -0.6364$ and $\lambda = 0.6324$ as suggested by Taubin in [143]. Combining the refinement and the smoothing operator, we can now define the analysis component of the multi-resolution transform as follows:

$$\boldsymbol{c}^{i-1} = H^i \boldsymbol{c}^i \tag{6.15}$$

$$\boldsymbol{d}^i = (\boldsymbol{c}^i - C^{i-1}\boldsymbol{c}^{i-1}) = (I - C^{i-1}H^i)\boldsymbol{c}^i \tag{6.16}$$

We sample the detail vectors $\boldsymbol{d}^i$ on the finer level i to avoid aliasing and thus end up with an over-complete representation of the mesh similar to the Laplacian pyramid described by Burt and Adelson [21].

6.5 Synthesis

This decomposition now leads to the following linear synthesis algorithm. As described in the section 6.3, on each level i, we can express the position of the control points $\boldsymbol{c}^i$ as a

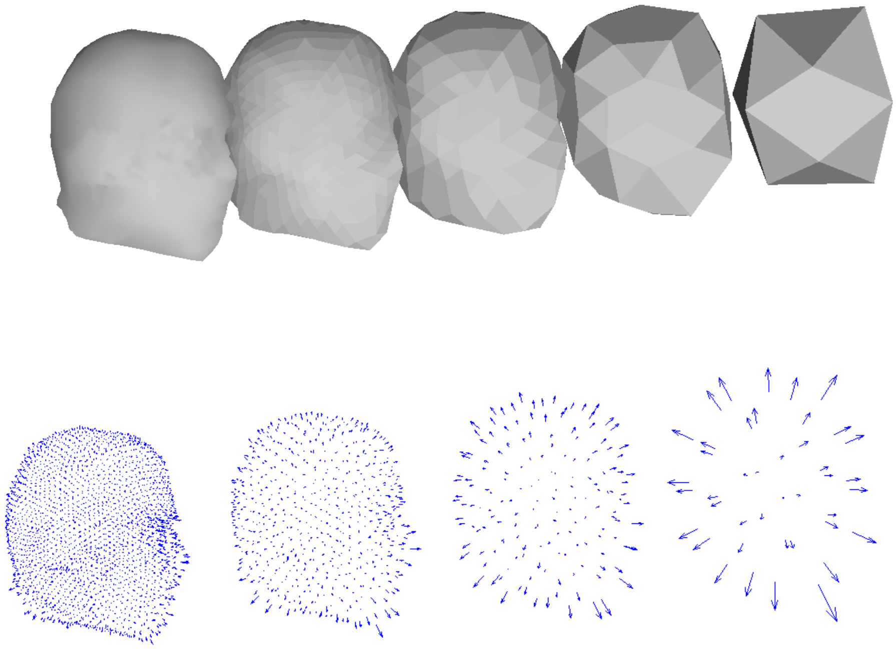


Figure 6.4: Control Meshes at different Resolutions and their detail differences

linear combination of the control points on a coarser level $\boldsymbol{c}^{i-1}$ and the detail coefficients $\boldsymbol{d^{i-1}}$ that express the additional degrees of freedom of the vertices in the refined level i. Formally, we can write

$$\boldsymbol{c}^{i} = [C^{i-1}D^{i}] \begin{bmatrix} \boldsymbol{c}^{i-1} \\ \boldsymbol{d}^{i} \end{bmatrix}. \tag{6.17}$$

D^{i} can be defined in numerous ways, e.g. as the identity matrix as in Equation 6.16 or defining a local coordinate system as in section 6.6.

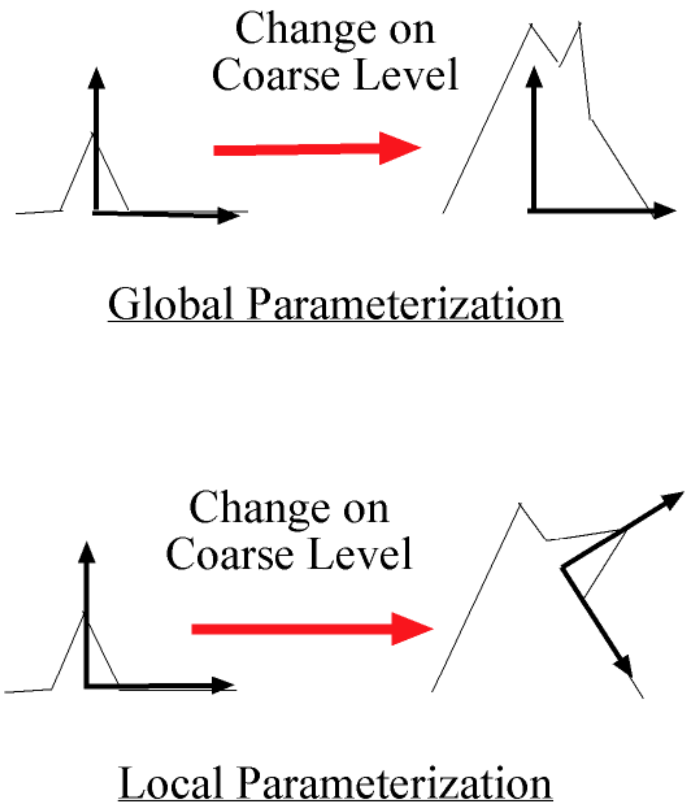


Figure 6.5: Detail Encoding in Global and Local Coordinate Systems

The decomposition can now be iterated (similar to [91])

$$\begin{aligned} \boldsymbol{c}^i &= [C^{i-1} D^i] \begin{bmatrix} \boldsymbol{c}^{i-1} \\ \boldsymbol{d}^i \end{bmatrix} \\ &= [C^{i-1} D^i] \begin{bmatrix} C^{i-2} & D^{i-1} & 0 \\ 0 & 0 & I^i \end{bmatrix} \begin{bmatrix} \boldsymbol{c}^{i-2} \\ \boldsymbol{d}^{i-1} \\ \boldsymbol{d}^i \end{bmatrix} . \end{aligned} \tag{6.18}$$

and we end up with the following expression that relates the limit surface linearly to the control points $\boldsymbol{c}^0$ of the coarse initial mesh and the detail coefficients $\boldsymbol{d}^j$ on all levels

$1 \leq j \leq i$:

$$\boldsymbol{s} = L^i \widehat{C^i} \boldsymbol{c}_*^i \tag{6.19}$$

$$\widehat{C^i} = \left[\prod_{j=0}^{i-1} C^j, \prod_{j=1}^{i-1} C^j D^1, \prod_{j=2}^{i-1} C^j D^2, \ldots, C_{i-1} D^{i-1}, D^i \right]$$

$$\boldsymbol{c}_*^i = \left[\boldsymbol{c}^0, \boldsymbol{d}^1, \boldsymbol{d}^2, \ldots \boldsymbol{d}^i\right]$$

Although we have a nice and simple decomposition that relates the motion and shape of the object linearly to the values of the control and detail coefficients, there is a problem with the approach so far. To achieve a linear representation of the surface in terms of its control vectors, we need to encode the detail level in a global coordinate system. Unfortunately, this prevents us from optimizing the different levels of resolution independently, because changes on a coarse level will cause unintended changes of the global object shape as can be seen in Figure 6.5. Most of the difference in shape between successive levels can be represented by a displacement along the normal to the surface as indicated by the histogram of the magnitudes of the detail coefficients in Figure 6.6. This is quite obvious since the discrete Laplacian is an approximation of the local normal direction, thus most of the smoothing in the analysis step occurs parallel to the surface normal. In summary, most of the characteristic shape information is encoded along the local normal directions and during the synthesis step the local detail should be added back relative to the local coordinate frame to take advantage of this and to decorrelate the different levels of resolution. If we were just interested in shape estimation, this would also suggest to take advantage of local encoding to reduce the dimensionality of our estimation problem by restricting the detail vectors to vary only along the normal directions. Unfortunately, since we are also interested in motion estimation we need the tangential component of the local encoding to be able to represent the part of the motion

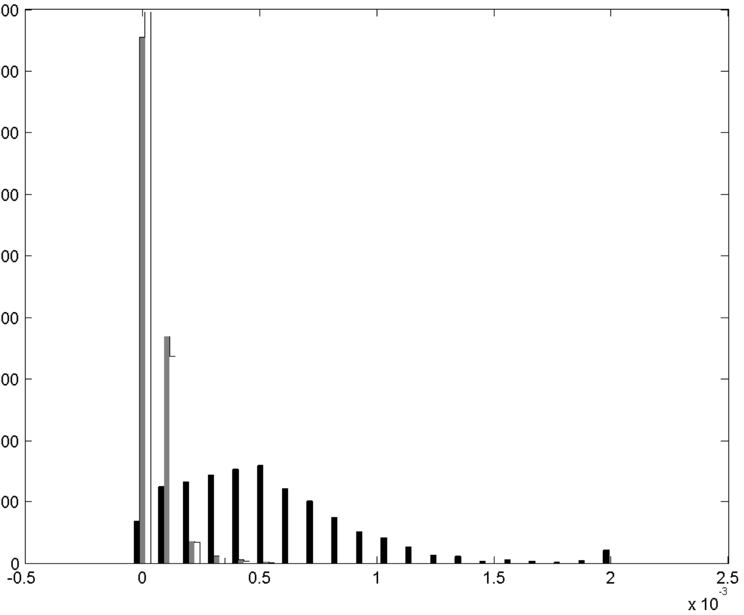


Figure 6.6: Histogram encoding the magnitude of the normal (black) and tangential (gray,white) components of the detail vectors in the multi-resolution encoding of the mesh

field parallel to the tangent plane of the surface.

A local encoding of the detail vectors is especially important in the case of motion estimation, because if we have a motion that is well described by the trajectories of the control vectors on a coarser level, then the trajectories of the locally encoded detail coefficients will not need to be adjusted. An example would be the encoding of a moving arm including a detailed hand that it is not moving relative to the arm. If we encode the fingers in a global coordinate system, we would get large changes in the detail coefficients to be able describe the motion of the fingers. If we have a local encoding, we just need to add the local shape details to the refined control mesh on the coarser level that was moved by the global motion of the arm and we the new shape is accurately synthesized

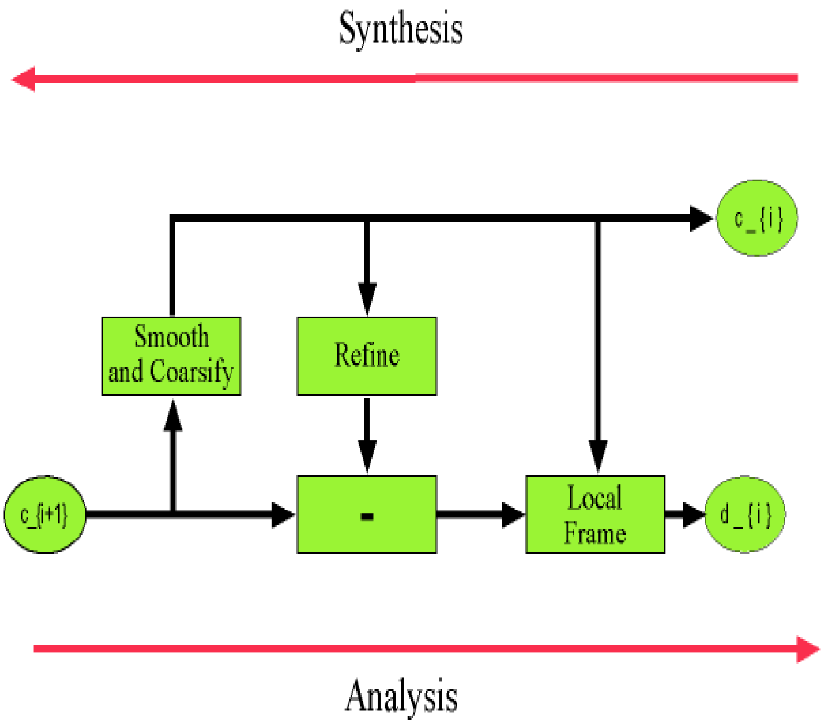


Figure 6.7: Flow chart describing the multi-resolution analysis and synthesis

since the local frames moved with the global motion.

6.6 Encoding of Detail in Local Frames

To define the detail coefficients with respect to a local frame, we apply two linear operators R and Q to the control mesh $\boldsymbol{c}^i$ that result in two linearly independent vectors $\boldsymbol{r}^i(v) = (R\boldsymbol{c}^i)(v)$ and $\boldsymbol{q}^i(v) = (Q\boldsymbol{c}^i)(v)$ in the tangent plane to $\boldsymbol{s}(v)$. We can use them to define a local orthonormal frame at v $F^i(v) = (\boldsymbol{n}^i(v), \boldsymbol{r}^i(v), \boldsymbol{q}^i(r))$ where $\boldsymbol{n}^i(v) = \boldsymbol{r}^i(v) \times \boldsymbol{q}^i(r)$. Details about how to define these operators in the case of the Loop subdivision scheme can be found in Hoppe [60].

Including the local encoding in our multi-resolution framework, we summarize the steps of the transform (for a graphical flow chart see Figure 6.7)

```
ANALYSIS:

for i = n:-1:1
        $\boldsymbol{d}^i = (F^i)^{\mathrm{T}}(I - SH)\boldsymbol{c}^i$
        $\boldsymbol{c}^{i-1} = H\boldsymbol{c}^i$
end

SYNTHESIS:

for i = 1:n
        $\boldsymbol{c}^i = S\boldsymbol{c}^{i-1} + (F^i)\boldsymbol{d}^i$
end
        $\boldsymbol{s} = L^n\boldsymbol{c}^n$
```

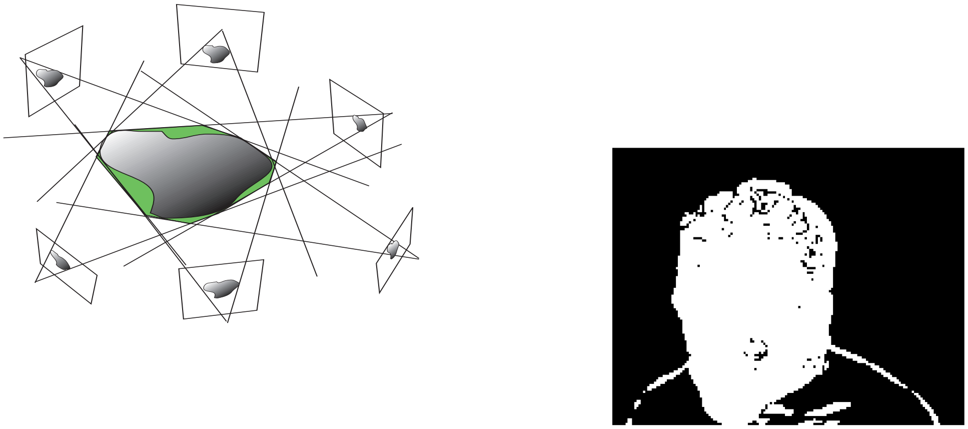
This forms the basis of our multi-resolution algorithm. Given a decomposition of our shape estimate into the different levels of resolution, the detail coefficients are modified one level at a time. If we do not have a shape estimate yet, we start from a coarse mesh c^0 with few vertices and set all the detail coefficients to zero. The estimation of the spatio-temporal structure is always done top-down, from coarser decomposition levels to finder decomposition levels. When optimizing detail coefficients on level i, we compute the value of $\boldsymbol{c}^{i-1}$ using the coarse base mesh $\boldsymbol{c}^0$ and the detail levels $\boldsymbol{d}^j$ $1 \leq j \leq i - 1$. This also determines the local frame F^{i-1}. The changes in the detail coefficients are then propagated to the mesh surface by continuing the synthesis procedure, and then we can evaluate the error measure. After the optimization converges or a fixed number of iterations has been applied, we precompute $\boldsymbol{c}^i$ using our estimate of $\boldsymbol{d}^i$ and continue to optimize over d^{i+1}, thereby increasing the degrees of freedom of the mesh. This is continued until we reach the maximum refinement depth. The refinement can be locally controlled by the magnitude of the detail coefficients. If they are small in a region of the mesh, there is no need for further subdivision. This enables the algorithm to adapt the mesh resolution according to the complexity of the surface geometry.

6.7 Multi-Camera Shape and Motion Estimation

The shape and motion estimation can be subdivided into two different parts. Initially, we do not know anything about our object except that it is located somewhere inside the volume observed by the cameras (which may see only parts of the object at any given time) and that the object is moving (if it is not moving then we need background images to distinguish the object from the background). Thus, the first step of the algorithm tries to locate the object of interest in the working volume, and estimate an approximation to its shape as well as motion. Having an initial estimate of the spatio-temporal structure of the object, we then apply our multi-resolution optimization framework to the structure using stereo constraints.

6.8 Shape Initialization

The shape initialization is based up on a subdivision of the working volume into voxels, which are then projected into all the images. We then accumulate the evidence for the voxel being inside or outside the object. The evidence is defined in Eq. 6.20 as the ratio between the temporal image gradient and the local spatial gradient (λ is a small positive



(a) (b)

Figure 6.8: (a) Multi Camera Silhouette Intersection from (b) Image Silhouettes

constant to ensure a well-defined measure everywhere):

$$\theta(\boldsymbol{P}) = \sum_{k \in Cameras} \theta_k(\boldsymbol{P}) = \frac{\frac{\partial I(\boldsymbol{p}_k)}{\partial t}}{\lambda + \|\nabla I(\boldsymbol{p}_k)\|^2} \qquad (6.20)$$

For each image the ratios $\theta_i(\boldsymbol{P})$ are formed by integrating over the footprint of the voxel corresponding to the 3D location $\boldsymbol{P}$ (see Fig. 6.8b for two examples of the thresholded images of ratios). If the assumption of a moving object is violated, we can also incorporate the difference between the image sequence and previously recorded background images into the algorithm. The voxel volume is smoothed and thresholded using 3D morphological filters, before an iso-surface extraction algorithm determines the initial shape and topology of the mesh. Given an iso-surface, we then convert it into a base mesh with subdivision connectivity using the following algorithm:

1. Simplify the triangle mesh representation down to a coarse base resolution

2. Displace the vertices of the subdivision mesh along their surface normal direction

until they lie on the iso-surface

3. Refine the mesh as described in Section 6.3

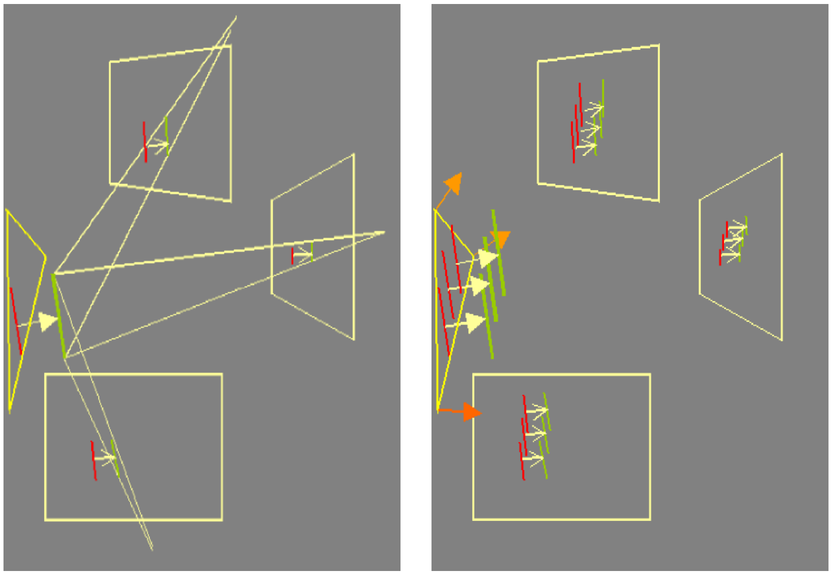
4. Repeat this process until the subdivision mesh approximates the iso-surface well.

Using a voxel algorithm based on the intersection of silhouettes in the images only allows us to find the visual hull [79] of the object in view (Fig. 6.8). For some applications this might be good enough [94], but we would like to capture all the shape and motion detail of the object.

We apply this algorithm to all the frames of the sequence which results in an approximation to the shape of the object at each time instance. Unfortunately, from these shapes we can only extract the component of the 3D motion that is normal to the object surface, but not the tangential component. Thus to compute the trajectories of the mesh vertices over time, we need to include image derivative information.

6.9 Motion Initialization

To find the correspondence between vertices over time and to compute an initial estimate for the 3D motion field, we use the spatio-temporal gradients in the images and relate them to the motion vectors on the object surface. From equation 6.3 it follows that each normal flow measurement, that is the component of the image flow that is perpendicular to the local brightness gradient in an image, constrains the projection of the 3D motion flow to lie along a line parallel to the iso-brightness contour in the image, the normal flow constraint line. Thus the 3D motion flow vector has to lie on the plane defined by the normal flow constraint line and the optical center of the camera as shown in Figure 6.9a. Using the approximation to the shape computed before, we intersect the planes



(a) (b)

Figure 6.9: (a) 3D Normal Flow Constraint. The planes formed by corresponding image edges and the optical centers intersect in a line in space. (b) By integrating over a surface patch we can estimate the full 3D Flow

of corresponding measurements in space. The intersection should ideally be a single line in space, the 3D normal flow constraint line that is parallel to the iso-brightness contour on the object. The component of the 3D motion along the iso-brightness contour is not recoverable. This is the aperture problem revisited in 3D. Since each control motion vector is constrained by many samples (see the parameterization of each sample in a patch in Figure 6.3), and we expect the gradient directions to vary on the object surface, we can expect the estimation of these motion vectors to be nevertheless well-defined (see 6.9b for an illustration).

Since the shape we use to correspond the measurements is only an approximation, we can expect to have errors when corresponding measurements across cameras. To detect bad correspondences, we compute a measure for the collinearity of the intersections between the normal flow constraint planes and use it to prune bad correspondences.

Taking the derivative of Equation 6.1 with respect to time, and substituting it into equation 6.3, we can define the following linear constraint at each sample point $\boldsymbol{P} = \boldsymbol{s}(u_1, u_2, u_3)$ on the object surface patch r:

$$\begin{aligned}
-\frac{\partial I(\boldsymbol{p}_k)}{\partial t} &= \nabla I(\boldsymbol{p}_k)^{\mathrm{T}} \cdot \frac{d\boldsymbol{p}_k}{dt} \\
&= \nabla I(\boldsymbol{p}_k)^{\top} \cdot \frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{p}} \frac{\partial \boldsymbol{p}}{\partial t} \\
&= \nabla I(\boldsymbol{p}_k)^{\top} \cdot \frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{P}} \frac{\partial \boldsymbol{s}(u_1, u_2, u_3)}{\partial t} \\
&= \nabla I(\boldsymbol{p}_k)^{\top} \cdot \frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{P}} L^n(\boldsymbol{P}) \widehat{C^n}(\boldsymbol{P}) \frac{\partial \boldsymbol{c}_*^n(\boldsymbol{P})}{\partial t}
\end{aligned} \tag{6.21}$$

where $L^n(\boldsymbol{P})\widehat{C^n}(\boldsymbol{P})\boldsymbol{c}_*^n(\boldsymbol{P})$ are the components of equation 6.19 corresponding to surface point $\boldsymbol{P}$. Choosing the detail refinement matrices D^i to be the local frame matrices F^i corresponding to the current control vector values c^i $1 \leq i \leq n$, we can define a large linear system by stacking all the equations 6.21. This linear system can now be solved for the derivative of the base control and detail vectors with respect to time. These estimated temporal derivatives together with the our mesh approximations form an estimate for the trajectories of the control and detail vectors. We use a preconditioned conjugate gradient algorithm to solve this overdetermined linear system which converges always in a few iterations.

6.10 Shape and Motion Refinement through Spatio-Temporal Stereo

To refine our estimate of the spatio-temporal surface that describes the object, we adapt the vertex trajectories of the mesh, such that a multi-view stereo criterion is optimized.

We assume as expressed in equation 6.2 that the brightness of the projection of $\boldsymbol{P}$, that is the pixel value, is similar across cameras up to a linear transformation and that is only changing slowly over time. Combining both constancy constraints, we have that the similarity between the spatio-temporal volumes that each patch r on the object surface traces out in the spatio-temporal image space of each camera can be used as a measure for the correctness of our shape and motion estimation.

To evaluate the error measure, we choose a regular sampling pattern inside each surface patch, where the sampling density is adjusted per patch in such a way that we have approximately one sampling point per pixel in the highest resolution image the patch is visible in. The visibility of each patch is determined by a z-buffer algorithm and we denote the set of camera pairs that mutually see a patch on the object surface by $\mathcal{V}(r)$. Using the subdivision framework presented in section 6.2, we synthesize the shape at a number of consecutive time instances based on the current values of the base control and detail vectors and evaluate the following matching functional in space-time based on normalized correlation

$$\begin{aligned}
\mathcal{E}(r,t) &= \sum_{(i,j)\in\mathcal{V}(r)} \mathcal{W}(i,j) \int\limits_{t-\Delta t}^{t+\Delta t} \frac{\langle I_i(s), I_j(s)\rangle}{|I_i(s)|\cdot|I_j(s)|} ds \qquad (6.22)\\
&+ \sum_i \int\limits_{t-\Delta t}^{t+\Delta t} \frac{\langle I_i(t), I_i(s)\rangle}{|I_i(t)|\cdot|I_i(s)|} ds\\
\langle I_i(t), I_j(t)\rangle &= \int\limits_{\boldsymbol{P}\in r} I_i(\tilde{\boldsymbol{p}_i}(\boldsymbol{P},t))\cdot I_j(\tilde{\boldsymbol{p}_j}(\boldsymbol{P},t)) d\boldsymbol{P}\\
I_i(\tilde{\boldsymbol{p}_i}(\boldsymbol{P},t)) &= I_i(\boldsymbol{p}_i(\boldsymbol{P},t)) - \overline{I_i(t)}\\
\overline{I_i(t)} &= \int\limits_{\boldsymbol{P}\in r} I_i(\boldsymbol{p}_i(\boldsymbol{P},t)) d\boldsymbol{P}\\
|I_i(t)|^2 &= \langle I_i(t), I_i(t)\rangle
\end{aligned}$$

to compare corresponding spatio-temporal image volumes between pairs of cameras. For each sampling point on the mesh surface, we determine the intensity values by bilinear interpolation from the images.

We combine the correlation scores from all the camera pairs by taking a weighted average with the weights $\mathcal{W}(i, j)$ depending on the angles between optical axes of cameras i and j and the surface normal at point $\boldsymbol{P}$. Notice that each vertex $v \in V^i$ influences the shape only in those patches that involve vertices that are part of v's 2-ring (vertices that are at most 2 edges away). We will denote this set of patches by $\mathcal{T}_2(v)$. Thus, when we want to compute the derivatives of the error function with respect to the control points, we only need to evaluate the changes of the error measure on $\mathcal{T}_2(v)$. Due to the non-linearity that was introduced by the local encoding of the detail, we cannot the express the change of the surface directly as a linear function of the change in the detail coefficients, but have to synthesize the change on the surface as described in Figure 6.7. Since we only have to evaluate the changes on small patches of the object surface at a time, this can be done efficiently though. The final derivative can then be computed easily using the chain rule from the derivative of the projection equation (Eq. 6.1) and the spatio-temporal image derivatives.

We use the BFGS-quasi newton method in MATLAB^{TM} Optimization Toolbox to optimize over the control point positions. The upper bounds for their displacement is given by the boundaries of the voxel volume, which we include as inequality constraints in the optimization.

So far we have only applied our algorithm on objects consisting only of one component, but if the initial shape approximation consists of several disconnected parts, we could use the distance between the disconnected parts to merge or separate the recovered surfaces. Then we can apply the refinement algorithm to every object in turn, while updating the visibility globally.

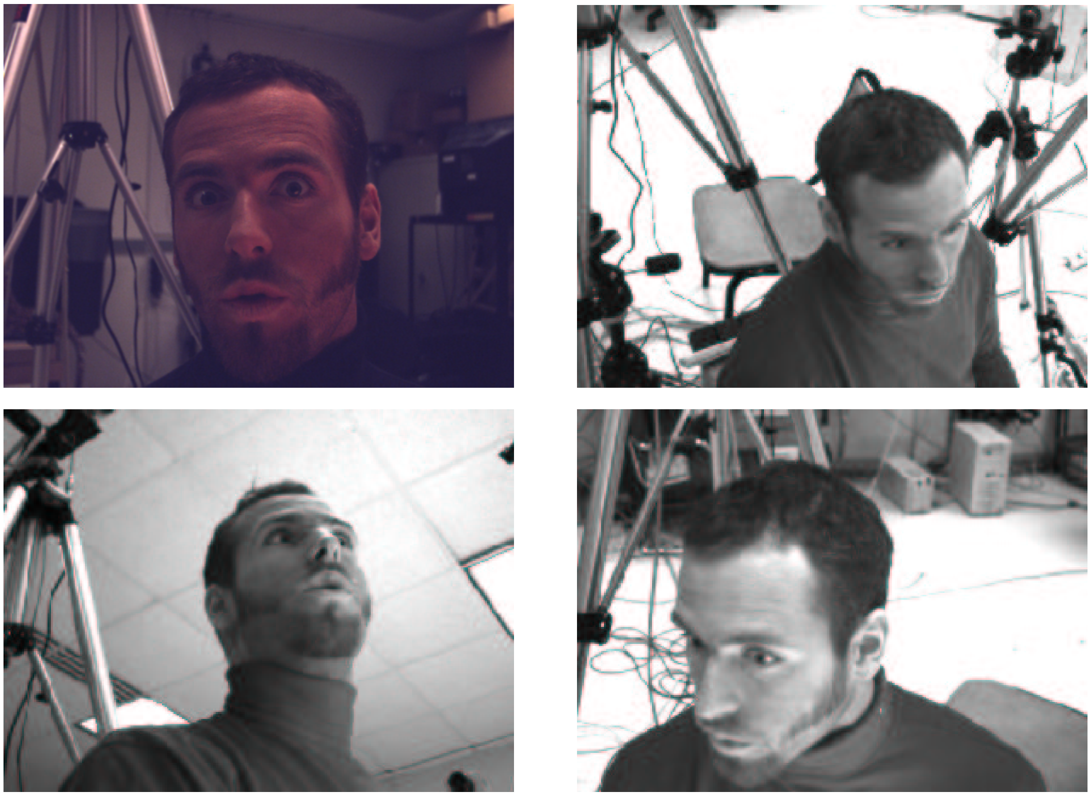


Figure 6.10: Four Example Input Views

6.11 Results

We have established in our laboratory a multi-camera network consisting of sixty-four cameras, Kodak ES-310, providing images at a rate of up to eighty-five frames per second; the video is collected directly on disk. The cameras are connected by a high-speed network consisting of sixteen dual processor Pentium 450s with 1 GB of RAM each which process the data.

For our experiments we used eleven cameras, 9 gray scale and 2 color, placed in a dome-like arrangement around the head of a person, who was opening his mouth to express surprise (example images in Figure 6.10) and blinking his eyes while turning his head and moving it forward.

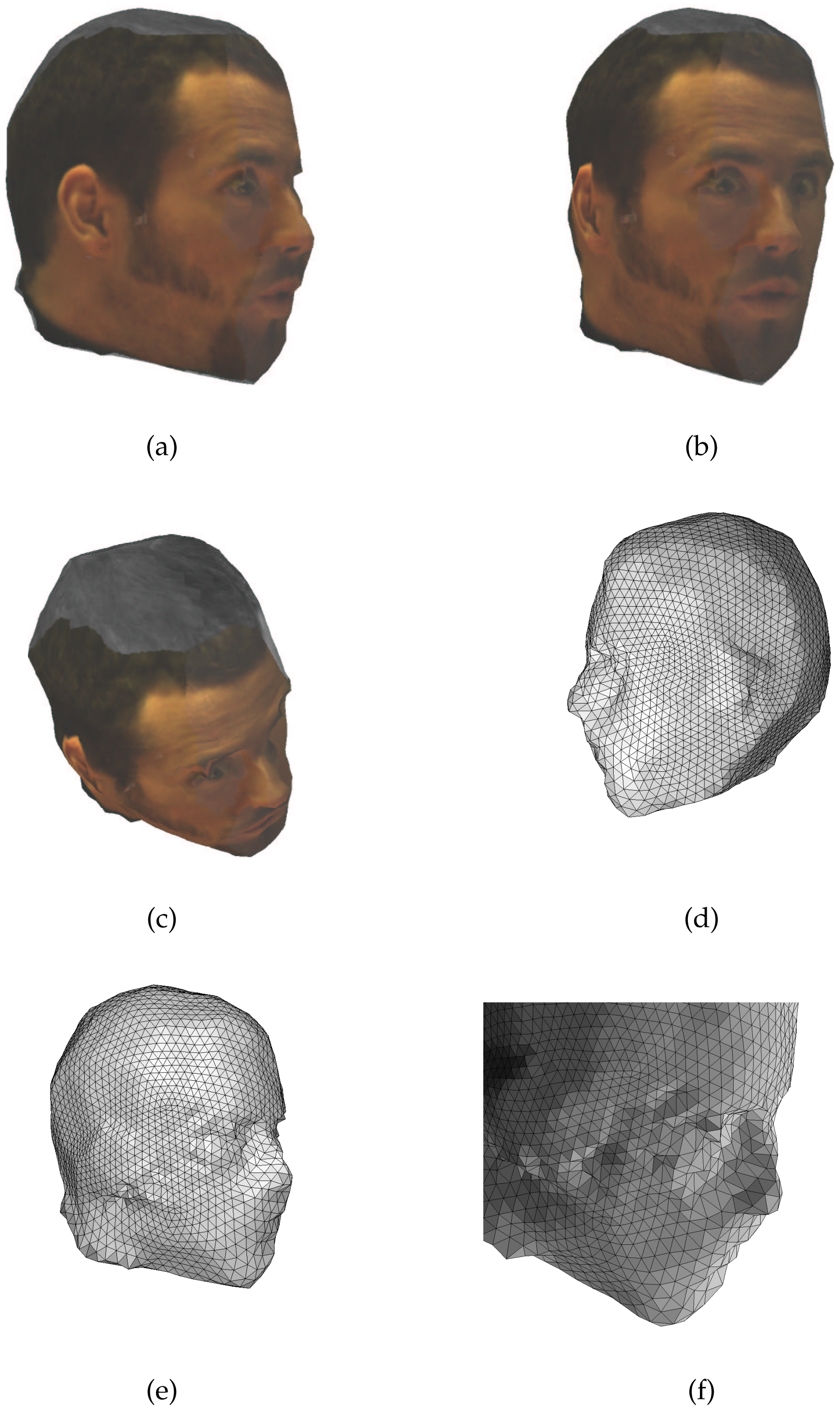
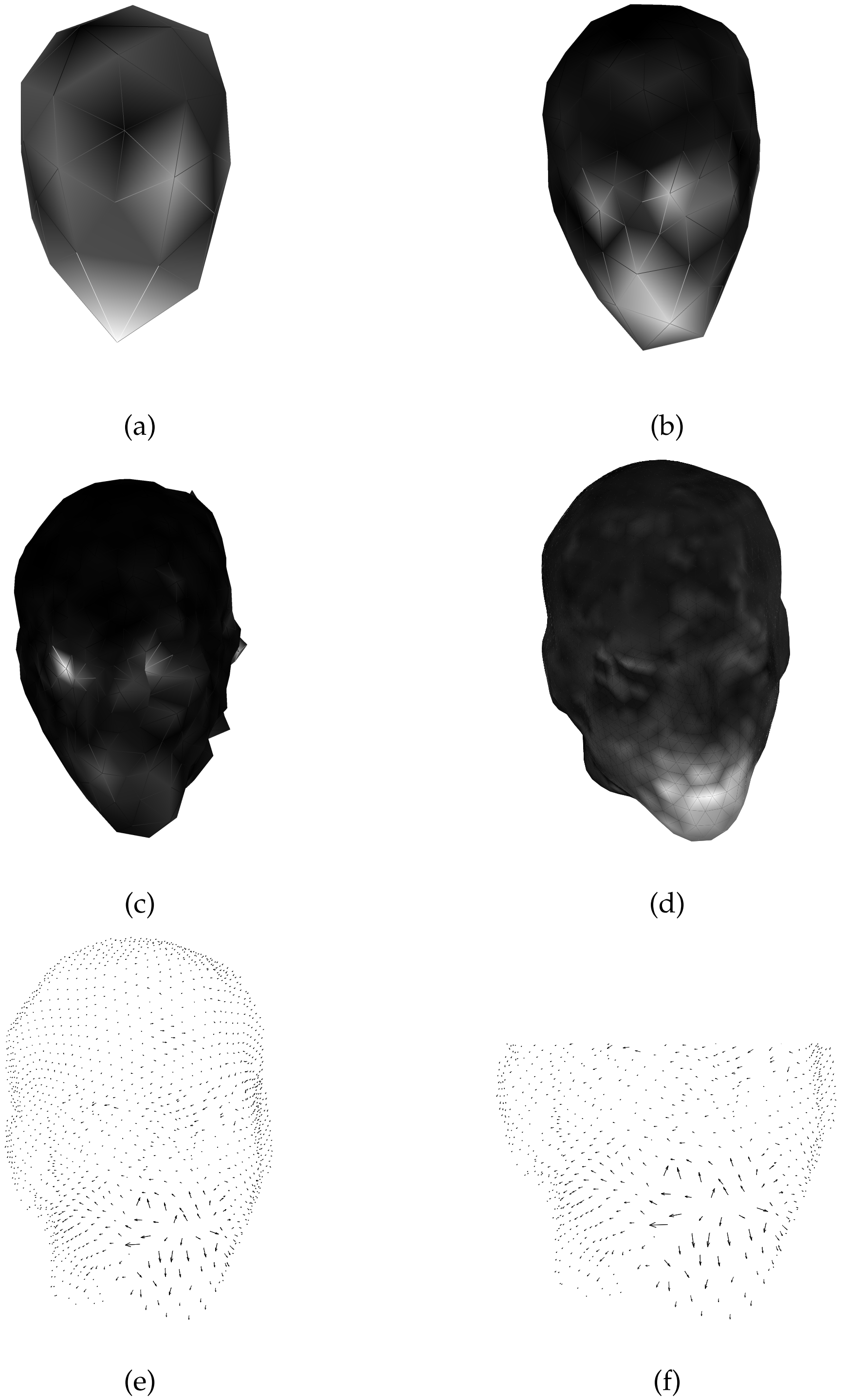


Figure 6.11: Results of 3D Structure and Motion Flow Estimation: Structure.(a-c) Three Novel Views from the Spatio-Temporal Model (d) Left View of Control Mesh (e) Right Side of Final Control Mesh (f) Close Up of Face



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

Figure 6.12: Results: Motion Flow. (a-c) Magnitude of Motion Vectors at Different Levels of Resolution (d) Magnitude of Non-Rigid 3D Flow Summed over the Sequence (e) Non-Rigid 3D Motion Flow (f) Non-Rigid Flow Close Up of Mouth

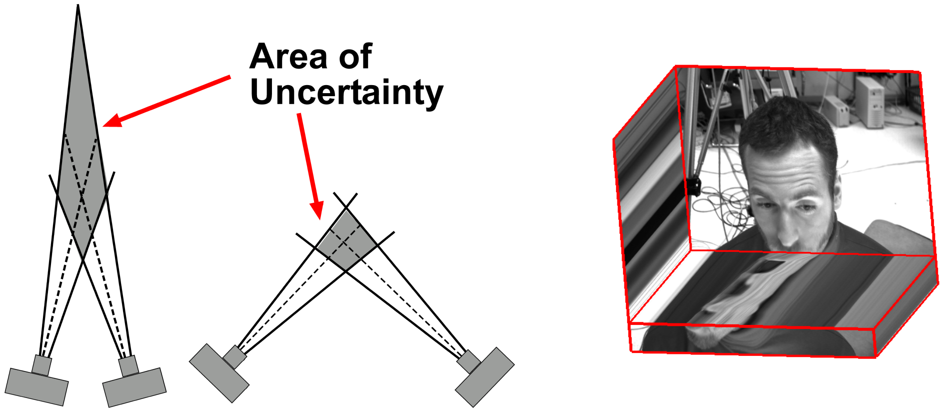
The recovered spatio-temporal structure enables us to synthesize texture-mapped views of the head from arbitrary viewing directions (Figures 6.11a-6.11c). The textures, coming always from the least oblique camera with respect to a given mesh triangle, were not blended together to illustrate the good agreement between adjacent texture region boundaries (note the agreement between the grey-value structures in Figure 6.11c despite absolute grey-value differences). Unfortunately, we did not have access to a laser range scan to generate ground-truth for the shape, but we believe the recovered control meshes in Figures 6.11d-6.11f show that despite some artifacts near the eyes the spatial structure of the head was recovered well. For a full view of the reconstruction, please see the accompanying videos at the web site *www.videogeometry.com*. Since only two cameras were color, we were only able to texture map parts of the head in color.

Examining the 3D motion fields at different resolutions we see that the multi-resolution representation is seperating the motion field into different components. In Figures 6.12a-6.12d we encoded the magnitude of the motion vectors on the object surface as brighntess values that vary from bright for large displacements to dark for small displacements. The brightness values are increasing proportionally with the magnitude of the motion energy. Figure 6.12a shows that at the coarsest level the magnitudes of the 3 flow vectors vary little across the object, which is to be expected since the trajectories on the coarsest level should encode the rigid motion of the object. At next finer level (Figure 6.12b) we see that most of the motion energy is concentrated in the eye and motion region which corresponds well to the activity in the scene. Further increasing the scale, we notice that the magnitude of the motion vectors concentrates more and more at only a few places (Figure6.12c), in this example the fast blinking motion of the eye is the prominent motion.

To separate the non-rigid 3D motion flow of the mouth gesture from the motion field caused by the turning of the head, we fit a rigid flow field to the trajectories of the coarsest mesh level $\boldsymbol{c}^0$ by parameterizing the 3D motion flow vectors by the instantaneous rigid motion $\partial \boldsymbol{c}^0/\partial t = \mathbf{v} + \boldsymbol{\omega} \times \boldsymbol{c}^0$, where $\mathbf{v}$ and $\boldsymbol{\omega}$ are the instantaneous translation, and rotation ([61]). By subtracting the rigid motion flow from the full flow, we extract the non-rigid flow. It can be seen that the motion energy (integrated magnitude of the flow over the whole sequence) is concentrated in the non-rigid regions of the face such as the mouth and the jaw as indicated by the higher brightness values in Figure 6.12d. In contrast, the motion energy on the rigid part of the head (e.g., forehead, nose and ears) is significantly smaller. In the close up view of the mouth region (Figure 6.12f) we can easily see, how the mouth opens, and the jaw moves down. Although, it is obviously hard to visualize dynamic movement by static imagery, the vector field and motion energy plots (Figure 6.12) illustrate that the dominant motion – the opening of the mouth – has been correctly estimated.

6.12 Hierarchies of cameras for 3D photography

For 3D photography we need to reconstruct the scene structure from multiple views. How well this can be done depends mainly on our abilities to compute correspondences between the views and then how accurately we can triangulate the correct position of the scene points. If we have a polydioptric camera we can compute local shape estimates from the multiple small-baseline stereo systems which allows one to use shape invariants in addition to intensity invariants to find correspondences between different views, while a single view point camera has to rely completely on intensity information. The accuracy of the triangulation depends on the base line between the cameras, the larger the baseline the more robust the estimation as can be seen in Fig. 6.13. Based on these two criteria, we can also define a hierarchy of cameras for the 3D shape estimation problem.



(a) (b)

Figure 6.13: (a) Triangulation-Correspondence Tradeoff in dependence on baseline between cameras. For a small baseline system, we can solve the correspondence problem easily, but have a high uncertainty in the depth structure. For a large baseline system this uncertainty is much reduced, but the correspondence problem is harder. (b) Motion and Shape can be reconstructed directly from feature traces in spatio-temporal and epipolar image volumes.

6.13 Conclusion and Future Work

To conclude, we presented a method that is able to recover an accurate 3D spatio-temporal description of an object by combining the structure and motion estimation in a unified framework. The technique can incorporate any number of cameras and the achievable depth and motion resolution depends only on the available imaging hardware.

In the future, we plan to explore other surface representations where we are able to adapt not just the geometry, but also the connectivity (see [75]) according the some optimization criterion. It is also interesting to study the connection between multi-scale mesh representation and multi-scale structure of image sequences that observe them to increase the robustness of the algorithm even further by improving the stopping criteria for the mesh refinement and the optimization. I would also like to explore the use of implicit representations for the spatio-temporal surface because if we want to reconstruct more complex scenes consisting of many objects in the scene, we need to have a representation that can change its topology during its evolution. For parameterized surfaces as presented in this work, this is a non-trivial problem, while implicit representations do not have this problem at all.

Another important issue for the 3D structure and motion estimation problem is the absence of good benchmark sequences including ground truth data. Due to the technical difficulties involved in the capture and calibration of these sequences, there are only very few sequences available for processing right now. We hope that in the future a benchmark data collection will evolve, so that we will be able to compare our algorithms on a standard data set against other researcher's algorithms as it is customary for the stereo or optical flow problem.

Chapter 7

Physical Implementation of a Polydioptric Camera

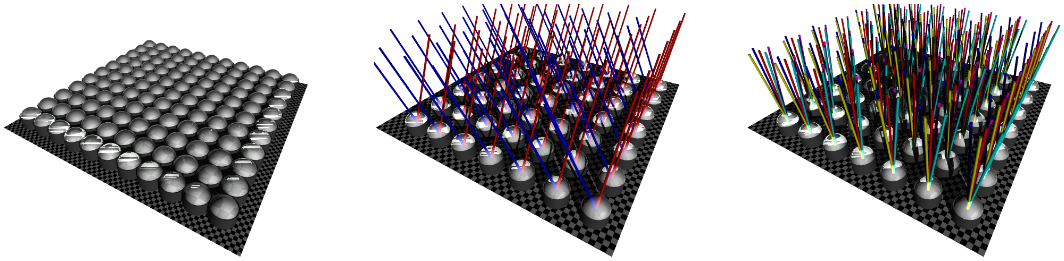


Figure 7.1: Design of a Polydioptric Camera (a) capturing Parallel Rays (b) and simultaneously capturing a Pencil of Rays (c).

A “plenoptic camera” has been proposed in [2], but since no physical device can capture the true time-varying plenoptic function, we prefer the term polydioptric to emphasize the difference between the theoretical concept and the implementation. With a polydioptric camera we observe points in the scene in view from many different viewpoints (theoretically, from every point on S) and thus we capture many rays emanating from that point. The components of the word polydioptric are based on the Greek words *polys* which means many and dioptric (from Greek *dioptrikos*)that according to Webster’s Dictionary means “assisting vision by refracting and focalizing light”. Thus a polydioptric camera is an imaging sensor that captures light that has been refracted and focussed in a multitude of ways.

A polydioptric camera can be implemented by arranging ordinary cameras very close to each other (Figs. 7.1b and 7.1c). This camera has an additional property arising from the proximity of the individual cameras: it can form a very large number of orthographic images, in addition to the perspective ones.

Indeed, consider a direction $\boldsymbol{r}$ in space and then consider in each individual camera the captured ray parallel to $\boldsymbol{r}$. All these rays together, one from each camera, form an image with rays that are parallel. For different directions $\boldsymbol{r}$ different orthographic image are formed. For example, Fig. 7.1b shows that we can select one appropriate pixel in each camera to form an orthographic image that looks to one side (blue rays pointing to the left) or another (red rays pointing to the right). Fig. 7.1c shows all the captured rays, thus illustrating that each individual camera collects conventional pinhole images.

7.1 The Physical Implementation of Argus and Polydioptric Eyes

We are currently working on developing a highly integrated tennis-ball-sized Argus eye with embedded DSP power, integrated spherical image frame memory and high-speed interface to a PC. A possible appearance of such an Argus eye is shown in Fig. 7.2. A simplified block diagram is shown in Fig. 7.3. A number of CCD or CMOS image sensor chip sets are interfaced to their own DSP chips. Normally, DSP chips have fast ports for communicating with other DSP chips, forming a parallel processor. Also, modern DSPs provide a host port through which a host computer can address and control the DSP as well as access its memory space. We envision using a P1394 serial bus (Firewire). Through six wires this serial bus can recreate a complete parallel bus (e.g., PCI bus) at a remote location away from the host PC. In our case this remote location is inside the

Argus eye. In effect, Firewire brings the PCI bus inside the Argus eye, allowing complete addressability, programmability and control from a single PC. Ultimately, the Argus eye will integrate our motion estimation algorithms within on-board DSP chips.

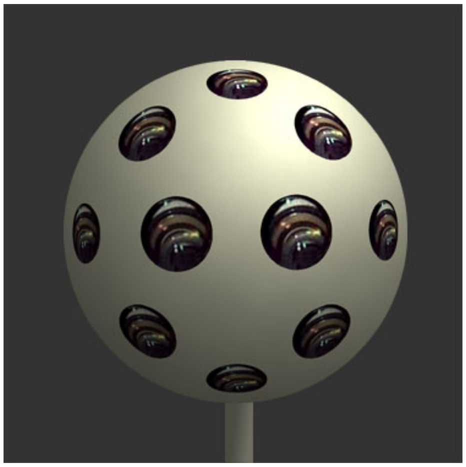


Figure 7.2: A highly integrated tennis-ball-sized Argus eye.

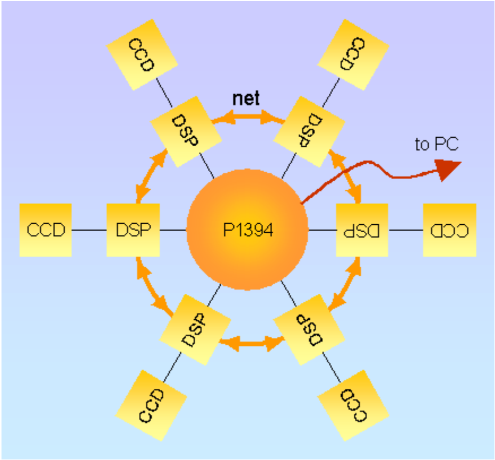


Figure 7.3: Block diagram of an Argus eye.

Of course one could think of numerous alternative implementations of Argus and polydioptric eyes. Ideas involving fish-eye lenses and catadioptric mirrors [100] to project wide-angle panoramas onto a single sensor are first to come to mind. In principle, such cameras can be used for experimentation. Nonetheless, any optical approach that warps the panoramic optical field to project it onto one or two planar image sensors will suffer reduced spatial resolution, because too wide an angle is squeezed onto a limited number of pixels. Given the low price of common resolution image sensors[1] as well as inexpensive plastic optics, there is no reason not to build Argus eye as suggested above by pointing many cameras to look all around. In fact, the Argus eye is not only a panoramic spherical camera; it is a compound eye with many overlapping fields of view. A simple

[1] A 1/4" quality color CCD chipset with about 640 × 480 pixels can be purchased for about $100. It is estimated that in 10 years such cameras will cost only a few dollars.

Argus eye could also be built by combining mirrors with conventional cameras, using the mirrors to split the field of view in such a way that we capture many directions in one image (e.g., one forward and one sideways).

By using special pixel readout from an array of tightly spaced cameras we can obtain a polydioptric camera. Perhaps the biggest challenge in making a polydioptric camera is to make sure that neighboring cameras are at a distance that allows estimation of the "orthographic" derivatives of rays, i.e., the change in ray intensity when the ray is moved parallel to itself. For scenes that are not too close to the cameras it is not necessary to have the individual cameras very tightly packed; therefore, miniature cameras may be sufficient. The idea of lenticular sheets has appeared in the literature [2, 110] for 3D imaging. These sheets employ cylindrical lenses and are not very appealing because of the blurring they create. There are, however, similar micro-image formation ideas that would fully support the mathematical advantages of polydioptric eyes suggested in the previous section. One such idea is shown in Fig. 7.4. A micro-lens array is mounted on the surface of an image sensor directly, emulating an insect compound eye. Fig. 7.5 shows the imaging geometry. As an alternative to micro-lens arrays one could also consider coherent optical fiber bundles as image guides.

In this example, micro-lenses are focused at infinity. Advances in MEMS and micro-machining has resulted in the wide availability of micro-optics and lens arrays. They can be as small as tens of microns in diameter and can be arranged in a rectangular or hexagonal grid. The image sensor detects a plurality of optical images. These images are very small, perhaps 16–32 pixels on a side. They may not be useful for extensive imaging; however, they directly support computation that unleashes the power of polydioptric eyes. Fig. 7.6 depicts how these small images might look; they appear more like textures than like detailed images of the scene. However, these textures are sufficient for obtaining the rotational and translational derivatives of the light rays as 3D motion of the camera (or scene) occurs (i.e., $\nabla_r L$ and $\nabla_x L$). Fig. 7.6 shows that the derivatives of ray rotation (blue to red ray change in Fig. 7.5) would be computed by estimating the texture motion in individual sub-retinas over time. On the other hand, the derivatives of ray translation would be computed by estimating the time-evolving disparity of texture motions across pairs of sub-retinas. This is only one idea for building a miniature polydioptric eye. With a conventional image sensor we could have as many as 60×60 micro-lenses over a single image sensor. A plurality of such polydioptric eyes could be arranged to yield new camera topologies for vision computation.

Some form of acceptance optics would be needed to resize the field of view to meaningful sizes. These are commonly used as image tapers to resize the field of view in scientific cameras such as digital X-ray cameras. Also, image guides are manufactured for use in medical instruments for laparoscopy. A plurality of image guides would be used. The acceptance faces of the image guides would be tightly arranged in an imaging surface; the exit faces could then be spaced apart and coupled to image sensors. A bundle

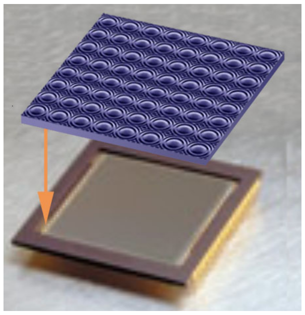


Figure 7.4: Forming a kind of polydioptric eye.

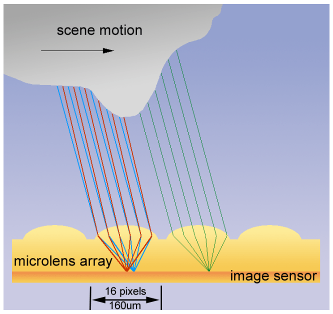


Figure 7.5: Plenoptic projection geometry for micro-lenses.

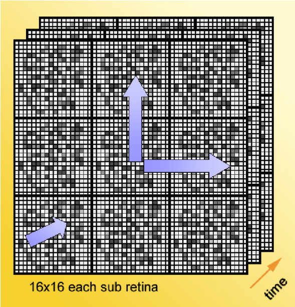


Figure 7.6: Plenoptic image processing.

of thousands of such fibers, appropriately calibrated, may constitute the ultimate design. We are currently experimenting with small versions and investigating the possibility of different optical materials. Using image guides is a more expensive proposition, but it is an attractive alternative, as this approach builds a superb compound eye with each "ommatidium" having a complete perspective image of the scene. Similar methods could be used to build small polydioptric domes: the acceptance faces would surround the dome volume, and at some distance behind them, the image sensors can be placed.

Finally, we should emphasize that not all images collected by an Argus eye need to be kept in memory. For example, a polydioptric eye with a few thousand optical fibers can create the same number of perspective images plus several thousand more affine images for different directions. As we only need some calculations of raw derivatives in order to derive a scene model, we can be selective regarding the images we wish to keep. In fact, our dedicated DSP engines can sift through their individual image data and preserve only higher-level entities, such as derivatives, needed for later computation. Perhaps one spherical image would also be computed and stored locally, but not all the raw images from the eye.

7.2 The plenoptic camera by Adelson and Wang

This is the "original" plenoptic camera as described in [2]. The term *plenoptic* is derived from *plenus*, full, complete, and *optic*, view. They place a lenticular array over the CCD sensor to simulate a collection of small pinhole cameras which allows them to capture light from many viewpoints where each lens formed a so-called 5x5 macro-pixel. The problems the authors note with this approach are aliasing and the alignment of the lenticular sheet with the CCD-array. To reduce aliasing they also place a diffuser in front of the lenticular array. To compute the depth from the plenoptic images they choose a least-squares estimator that uses an directional derivative weighted ratio between the directional and positional derivatives:

$$d(x) = \frac{\sum_P I_x(x) I_v(x)}{\sum_P [I_x(x)]^2} \tag{7.1}$$

7.3 The optically differentiating sensor by Farid and Simoncelli

In [42] Farid and Simoncelli describe an interesting idea to compute the view-point derivatives. The main idea is that a lens in comparison to a pin-hole camera captures a continuum of rays and focuses them on one sensor element (if the rays originate from a point in focus). Thus if one is able to compute the viewpoint derivative in front of the lens, one could use the whole continuum of rays to compute the derivative, not just the summed projection at the sensor surface. If a point moves out of focus (either closer or farther than the plane of focus), then the rays will not converge on a single point on the sensor plane, but they will intersect the sensor plane in an area that is dependent on the depth of the ray origin. The blurring that results is again dependent on the depth and thus can be used to infer depth. If now a mask is placed in front of the lens and a point light source is imaged, then the image of the mask and the mask derivative will be

$$I(x) = \frac{1}{\alpha} M\left(\frac{x}{\alpha}\right) \text{ and } I_v(x) = \frac{1}{\alpha} M'\left(\frac{x}{\alpha}\right) \tag{7.2}$$

where α is related to the depth as follows :

$$\alpha = 1 - \frac{d_s}{f} + \frac{d_s}{Z} \tag{7.3}$$

Thus by taking the ratio between

$$I_x(x) = \frac{\partial}{\partial x}\left[\frac{1}{\alpha} M\left(\frac{x}{\alpha}\right)\right] = \frac{1}{\alpha^2} M'\left(\frac{x}{\alpha}\right) = \frac{1}{\alpha} I_v(x) \tag{7.4}$$

and I_v we can compute the depth: $\alpha = \frac{I_v(x)}{I_x(x)}$.

If a sensor captures light from several points then the formulation above can be extended by integrating over all the visible points, that is :

$$I(x) = \int \frac{1}{\alpha_p} M\left(\frac{x - x_p}{\alpha_p}\right) L\left(\frac{x_p}{\alpha_p}\right) dx_p \tag{7.5}$$

$$I_v(x) = \int \frac{1}{\alpha_p} M'\left(\frac{x - x_p}{\alpha}\right) L\left(\frac{x_p}{\alpha_p}\right) dx_p \tag{7.6}$$

where x_p is the image of the point p on the sensor surface through the center of the lens and L is the light intensity coming from p that is assumed to be constant across the lens. As before if we differentiate $I(x)$ we get

$$I_x(x) = \int \frac{1}{\alpha_p^2} M'\left(\frac{x - x_p}{\alpha}\right) L\left(\frac{x_p}{\alpha_p}\right) dx_p \tag{7.7}$$

and the two derivatives only differ by a scale factor. Unfortunately, α_p is changing with x_p, thus unless we have a fronto-parallel plane it is difficult to solve for the parameter directly.

7.4 Compound Eyes of Insects

A subject of future work is to determine if the compound eyes of insects can act as polydioptric cameras. Given the anatomical data is possible to compute the view direction and view position derivatives by comparing brightness values inside and between neighboring ommatidia. A neuronal architecture similar to correlators described by Richardt [121, 122, 123]. Here we include a conceptual description of an analog VLSI implementation based on the plenoptic motion estimation framework is first described in Neumann et al. [106].

Starting from the plenoptic motion constraint (Eq. 5.6) for the ray indexed by (x, y, u, v, t)

as ($[\cdot;\cdot]$ denotes the vertical stacking of vectors):

$$\begin{aligned}
-\mathcal{L}_t &= \nabla_{\boldsymbol{x}}\mathcal{L} \cdot \dot{\boldsymbol{q}} + (\boldsymbol{x} \times \nabla_{\boldsymbol{x}}\mathcal{L} + \boldsymbol{r} \times \nabla_{\boldsymbol{r}}\mathcal{L}) \cdot \boldsymbol{\omega} \\
&= [L_x, L_y, -\frac{u}{f}L_x - \frac{v}{f}L_y]\dot{\boldsymbol{q}} \\
&- [L_x, L_y, -\frac{u}{f}L_x - \frac{v}{f}L_y]([x, y, Z_\Pi]^{\mathrm{T}} \times \boldsymbol{\omega}) \\
&- [L_u, L_v, -\frac{u}{f}L_u - \frac{v}{f}L_v]([u, v, f]^{\mathrm{T}} \times \boldsymbol{\omega}) \\
-L_t &= [L_x, L_y, L_u, L_v][M_q, M_\omega][\dot{\boldsymbol{q}}; \boldsymbol{\omega}] \qquad (7.8)
\end{aligned}$$

$$M_q = \begin{pmatrix} 1 & 0 & -\frac{u}{f} \\ 0 & 1 & -\frac{v}{f} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \; M_\omega = \begin{pmatrix} -\frac{uy}{f} & \frac{ux}{f}+Z_\Pi & -y \\ -(\frac{vy}{f}+Z_\Pi) & \frac{vx}{f} & x \\ -\frac{uv}{f} & \frac{u^2}{f}+f & -v \\ -(\frac{v^2}{f}+f) & \frac{vu}{f} & u \end{pmatrix} \qquad (7.9)$$

As described in Chapter 5 by combining the constraints across the sensor surface, we can form a highly over-determined linear system and solve for the rigid motion parameters robustly and sufficiently.

The plenoptic motion constraint is extended to the other elemental images of the sensor surface by pre-multiplying $\dot{\boldsymbol{q}}$ and $\boldsymbol{\omega}$ with the appropriate rotation matrices to rotate the motion vectors into the local coordinates. For example, if the local coordinate system of an elemental image is related to the global sensor coordinate system by the rotation matrix R_i, then the motion matrices in Eq. (7.9) are post-multiplied with R_i to yield the local motion matrices $M_{qi} = M_q R_i$ and $M_{\omega i} = M_\omega R_i$.

To actually compute the plenoptic derivatives $\nabla_{\boldsymbol{x}}\mathcal{L}$ and $\nabla_{\boldsymbol{x}}\mathcal{L}$ we have to express them in terms of the derivatives between the imaging elements on the surface of the actual sensor. These derivatives can be computed as shown in Fig. 7.7a. The spherical imaging surface will be tessellated into elemental images (Fig. 7.7c). In each elemental image the individual sensor imaging elements will be arranged next to each other (Fig. 7.7b). As described by Horn [63], the gradient descent evolution that solves the linear system as

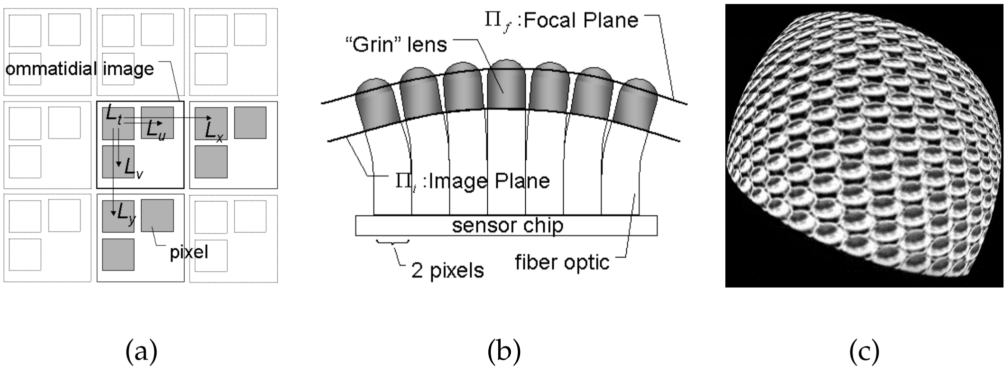


Figure 7.7: (a) The inter-ommatidial derivatives L_u and L_v are computed within each ommatidial image. Intra-ommatidial derivatives L_x and L_y are computed across ommatidial images. The constant matrix M_i, which in general will be different for each pixel, encapsulates for all geometric relationships of how individual ommatidial images are formed. (b) A curved compound eye design for ego motion sensor. Gradient Index (Grin) lenses are arranged along a curved surface Π_f forming optical images on the curved surface Π_i. The two planes are used to parameterize the Light Field. The elemental images from each Grin lens are brought to a planar sensor chip using coherent fiber optic bundles. (c) A two dimensional array of Grin lenses for the compound eye 3D ego motion sensor.

given in Eq. (7.8) can be efficiently computed using an analog VLSI hardware implementation. Let $\mathcal{S}_i$ denote the set of imaging elements $\boldsymbol{s} = [x, y, u, v]$ of the elemental image i for which we can form the constraint equation Eq. (7.8). In the extreme case where each elemental image only consists of a 2x2 array of pixels, $\mathcal{S}_i$ would only contain a single imaging element since we can only compute a single set of spatial derivative measurements L_u and L_v per elemental image (Fig. 7.7a). Based on Eq. (7.8), for each imaging element in $\mathcal{S}_i$ we want to minimize the error

$$E_i(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) = L_t(\boldsymbol{s}) + \nabla L(\boldsymbol{s})^{\mathrm{T}} M_i(\boldsymbol{s})[\dot{\boldsymbol{q}}; \boldsymbol{\omega}] \tag{7.10}$$

where $\nabla L(\boldsymbol{s}) = [L_x(\boldsymbol{s}), L_y(\boldsymbol{s}), L_u(\boldsymbol{s}), L_v(\boldsymbol{s})]^{\mathrm{T}}$ and $M_i(\boldsymbol{s}) = [M_{qi}(\boldsymbol{s}), M_{\omega i}(\boldsymbol{s})]$.

Summing over all imaging elements and elemental images we define the global error function $J(\dot{\boldsymbol{q}}, \boldsymbol{\omega})$ as

$$J(\dot{\boldsymbol{q}}, \boldsymbol{\omega}) = \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} |E_i(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega})|^2 \tag{7.11}$$

The gradient of this error function with respect to the motion parameters is then given by

$$\nabla J(\dot{\boldsymbol{q}}, \boldsymbol{\omega}) = \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} 2E_i(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) \nabla E_i(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) \tag{7.12}$$

If we split Eq. (7.12) into its components, we end up with the following six continuous update equations for the motion parameters

(we denote the columns of the local motion matrices M_i by $[M_{i1}, \ldots, M_{i6}]$ and scale the update by a factor λ):

$$\begin{aligned}
\frac{\partial \dot{q}_1}{\partial t} &= -\lambda \frac{\partial J}{\partial \dot{q}_1} = -\lambda \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} M_{i1}^{\mathrm{T}} \nabla L(\boldsymbol{s}) E(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) \\
\frac{\partial \dot{q}_2}{\partial t} &= -\lambda \frac{\partial J}{\partial \dot{q}_2} = -\lambda \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} M_{i2}^{\mathrm{T}} \nabla L(\boldsymbol{s}) E(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) \\
\frac{\partial \dot{q}_3}{\partial t} &= -\lambda \frac{\partial J}{\partial \dot{q}_3} = -\lambda \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} M_{i3}^{\mathrm{T}} \nabla L(\boldsymbol{s}) E(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) \\
\frac{\partial \omega_1}{\partial t} &= -\lambda \frac{\partial J}{\partial \omega_1} = -\lambda \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} M_{i4}^{\mathrm{T}} \nabla L(\boldsymbol{s}) E(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) \\
\frac{\partial \omega_2}{\partial t} &= -\lambda \frac{\partial J}{\partial \omega_2} = -\lambda \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} M_{i5}^{\mathrm{T}} \nabla L(\boldsymbol{s}) E(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega}) \\
\frac{\partial \omega_3}{\partial t} &= -\lambda \frac{\partial J}{\partial \omega_3} = -\lambda \sum_i \sum_{\boldsymbol{s} \in \mathcal{S}_i} M_{i6}^{\mathrm{T}} \nabla L(\boldsymbol{s}) E(\boldsymbol{s}; \dot{\boldsymbol{q}}, \boldsymbol{\omega})
\end{aligned}$$

We see that each update equation consists of the sum over all imaging elements and elemental images of the derivatives multiplied with a fixed set of weights that depends on the sensor geometry and is precomputed during the design phase of the sensor. Due to the implementation of the algorithm in analog electronic circuitry the motion parameters are updated continuously thus allowing us to update the motion parameters at a temporal refresh rate that is much higher then for conventional motion sensors.

Referring to Fig. 7.8, for each pixel spatial derivatives are computed using differential amplifiers, while the temporal derivatives are computed using a differentiator. These derivative measurements are then multiplied with the pixel positional weights defined by the motion matrices M_i. These positional weights can be supplied to each cell using resistive chains connected to fixed potentials. Based on these computations at each local pixel, six currents are generated that are summed up in their corresponding wire on the 6-wire analog bus. These 6 wires transverse all pixel locations. The summation of the currents emulates the summations in Eq. (7.12).

The total injected current in each wire updates the voltage on the wire. Once the minimum of Eq. (7.11) is reached, the gradients in Eq. (7.12) becomes zero and no further change on the voltage on the six wires is observed. Effectively, the steady state voltages on the six wires represent the sensor's 6DOF motion measurement. It is interesting to note that these instantaneous voltages are also good initial "guesses" for future gradient descent updates as the motion of the sensor changes over time. In essence, the dynamics of minimizing Eq. (7.12) can be made much faster than the changes of motion expected in majority of robotics applications.

When computing spatial derivatives it is easy to calculate L_u and L_v, because these are the first neighbor differences in each elemental image (each ommatidium) of the compound eye. Calculating L_x and L_y requires wiring between pixels with the first neighbor ommatidium. This is not major problem, since we envision that each ommatidium has very small pixel arrays, probably only 2 by 2 pixels or so.

In all, our proposed 6DOF sensor is both a mathematical generalization of the classical constant optical flow equation [61] and a consequent hardware generalization of Tanner's constant optical flow sensor [142]. It is important to note, however, that due to the compound eye arrangement and more general view of the environment, our sensor is expected to provide stable motion measurements independent of the structure of the scene.

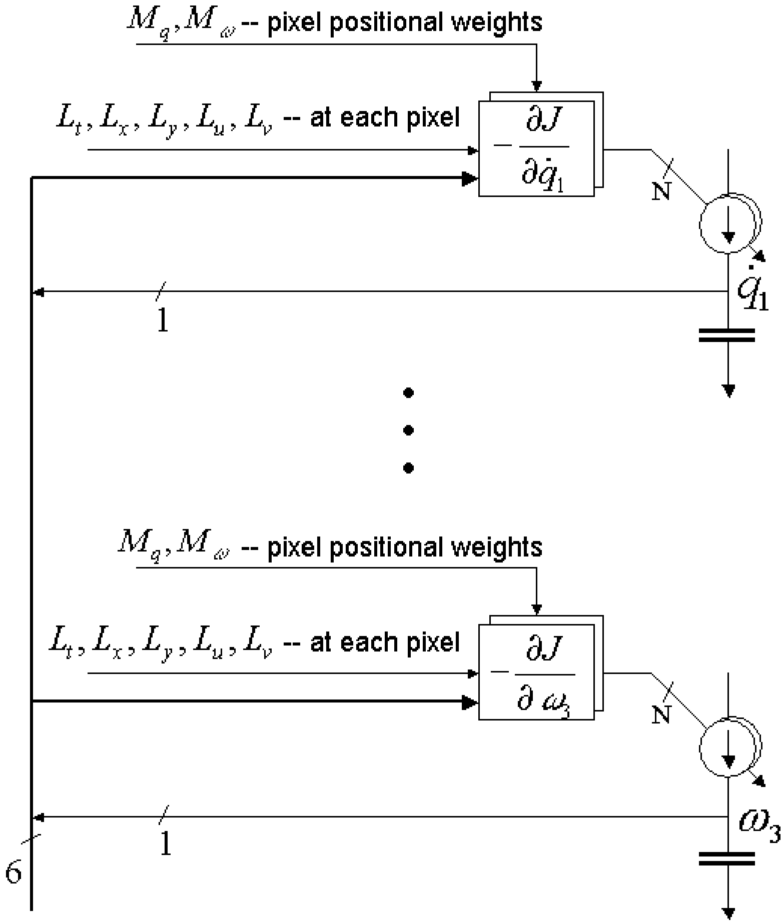

M_q, M_ω -- pixel positional weights
L_t, L_x, L_y, L_u, L_v -- at each pixel
$-\frac{\partial J}{\partial \dot{q}_1}$
N
$\dot{q}_1$
1
M_q, M_ω -- pixel positional weights
L_t, L_x, L_y, L_u, L_v -- at each pixel
$-\frac{\partial J}{\partial \omega_3}$
N
ω_3
1
6

Figure 7.8: The gradient update is computed at each pixel and proportional bits of current are summed into a global wire. One wire is used for each of the six motion parameters $(\dot{q}, \omega)$. Once the solution for the motion parameters are reached, based on minimizing Eq. (7.11), the gradients (I.e., the total current into each of six wires) are zero thus the steady state voltage solution on the capacitors is reached. The voltage from the six wires is continuously readout from the sensor representing the instantaneous 6DOF motion measurements. (The capacitors are explicitly drawn in this figure, although in actual implementation the parasitic capacitance of the wires is sufficient.)

Chapter 8

Conclusion

According to ancient Greek mythology Argus, the hundred-eyed guardian of Hera, the goddess of Olympus, alone defeated a whole army of Cyclopes, one-eyed giants. Inspired by the mythological power of many eyes I proposed in this thesis that computer vision researchers should study the design of sensors and algorithm in the space of light rays. This allows for unified treatment of both sensors and algorithms and allows us to optimize both components simultaneously. Based on a mathematical analysis of the differential structure of the space of light rays, I presented a framework to systematically study the relationship between the properties of an imaging sensor and the task performance of the entity using this sensor.

In this thesis I focused on the relation between the local differential structure of the time-varying plenoptic function and the ego-motion estimation of an imaging sensor. This led to a new motion estimation algorithm based on matching subsets of light rays in a scene independent manner. Since cameras cannot sample the space of light rays with arbitrary precision, we analyze the plenoptic sampling problem in a function approximation framework leading to a quantitative expression for the difference between the true and reconstructed light rays. This allows us to determine the best tradeoff between camera spacing and image resolution for polydioptric cameras.

Although we only focused on the study of camera motion estimation, I believe this framework can be extended to many more tasks. For example, as described in Chapter 2, we can utilize the differential structure of the plenoptic function to find the depth of the scene and detect occlusions. This would be very useful to improve the shape estimation for distributed camera systems, because the local depth and shape information of each polydioptric camera can be used to improve the matching process between views that are further apart. The widely separated camera clusters in turn then can estimate the depth of the scene with higher accuracy due to the larger baseline.

Differential plenoptic constraints are also very useful for segmenting objects out of a scene. Separating different objects in an image based on texture cues is difficult, because discontinuities in texture space can caused by transitions between regions on the same surface or by occlusions. In contrast if we utilize depth information the segmentation task becomes much easier, because it is very likely that image regions that lie at very different depths belong to different objects. Similarly moving objects lie on low dimensional image manifolds since the multi-view information constrain the possible depth values of the object observed. This reduces the ambiguities and allows us to use depth-based regularization schemes for independent motion estimation and detection.

An interesting object for further study is the task of recognition. In recognition we use a discriminant function to assign different classes to regions in the image. Although textural information is a very strong cue it was shown that the inclusion of depth knowledge increases the accuracy of the recognition task [167, 11]. In this case one should analyze how the image formation affects the shape of the discriminant function.

An important aspect of the imaging process which I did not address is the limited dynamic range of image sensors. It is often the largest source error in algorithms, and an inclusion in this camera design framework would be of great utility. Novel techniques for automatic gain control that actively change the sensing process by using polarized masks [101] or micro-mirror arrays [102] before or after the lens have been recently described in the literature.

Cameras nowadays become smaller and more affordable by the day, thus soon it will be in anyone's reach to use polydioptric assemblies of cameras for the tasks they try to solve. Since these assemblies can be reconfigured easily, the design of novel polydioptric eyes will be possible with little effort. The main challenges that remain for the implementation of polydioptric cameras are the current size of the individual imaging elements that make up the polydioptric sensor. Since the surface of the camera is restricted to be two-dimensional and we are sampling a four-dimensional function, the spacing between the imaging elements will always be discrete along some dimensions. Thus, it is paramount to develop small lens-sensor systems (MEMS, nano-technology) that address the optical imaging as well as the sensing problem. The smaller the imaging elements become, the more noise our measurements will contain due to the quantum nature of light. This necessitates intelligent sensor that adaptively combine information from neighboring measurements. We believe that the analysis of camera designs based on the structure of the space of light rays has great potential especially with advent of optical nano-technology around the corner which will offer new opportunities to design cameras that sample the space of light rays in ways unimaginable to us today.

Acknowledgements

The quote by Leonardo Da Vinci is taken from [2] and the help of Vladimir Brajovic who supplied Figures 7.2-7.8 is gratefully acknowledged.

Appendix A

Mathematical Tools

A.1 Preliminaries

A.1.1 Fourier Transform

The Fourier transform of a function f can be defined in 1D as

$$\mathcal{F}[f](\Omega) = \hat{f}(\Omega) = \int\limits_{-\infty}^{\infty} f(x)\exp(-2\pi j\Omega x)dx \tag{A.1}$$

and its inverse Fourier transform as

$$\mathcal{F}^{-1}[\hat{f}](x) = f(x) = \int\limits_{-\infty}^{\infty} \hat{f}(\Omega)\exp(2\pi j\Omega x)d\Omega. \tag{A.2}$$

If we are given a multi-dimensional signal, we can apply the Fourier transform to each dimension separately, which leads to the definition of the multi-dimensional Fourier transform (here for d dimensions):

$$\mathcal{F}[f](\boldsymbol{\Omega}) = \hat{f}(\boldsymbol{\Omega}) = \int\limits_{-\infty}^{\infty} \dots \int\limits_{-\infty}^{\infty} f(\boldsymbol{x})\exp(-2\pi j\boldsymbol{\Omega}^{\mathrm{T}}\boldsymbol{x})d\boldsymbol{x} \tag{A.3}$$

and

$$\mathcal{F}^{-1}[\hat{f}](\boldsymbol{x}) = f(\boldsymbol{x}) = \int\limits_{-\infty}^{\infty} \dots \int\limits_{-\infty}^{\infty} \hat{f}(\boldsymbol{\Omega})\exp(2\pi j\boldsymbol{\Omega}^{\mathrm{T}}\boldsymbol{x})d\boldsymbol{x} \tag{A.4}$$

Parseval Relations: If a function f is absolutely and square integrable ($f \in L^1 \cap L^2$), then we have the following equations between the energy in the signal f and the energy of its Fourier Coefficients:

$$\frac{1}{T}\int\limits_0^T |f(x)|^2 dx = \sum_{k=-\infty}^{\infty} |\hat{f}(k)|^2 \tag{A.5}$$

For the Fourier transform pairs f and $\hat{f}$, we also have the relationship

$$\int\limits_{\mathbb{R}} |f(x)|^2 dx = \int\limits_{\mathbb{R}} |\hat{f}(\omega)|^2 d\omega \tag{A.6}$$

Similarly, there exists a theorem that relates the magnitude of the product of functions to the product of their Fourier series coefficients:

$$\frac{1}{T}\int\limits_0^T f(x)\overline{g(x)} dx = \sum_{k=-\infty}^{\infty} \hat{f}(k)\overline{\hat{g}(k)} \tag{A.7}$$

Some Fourier transforms for elementary functions, that we use in this work are for one dimension:

$$\begin{aligned}
\mathcal{F}[1](\Omega) &= \delta(\Omega) \\
\mathcal{F}[\exp(2\pi\Omega_0 x](\Omega) &= \delta(\Omega - \Omega_0) \\
\mathcal{F}[\delta(x - x_0)](\Omega) &= \exp(2\pi x_0 \Omega) \\
\mathcal{F}[\Upsilon(x)](\Omega) &= \int\limits_0^{\infty} e^{-2\pi j \Omega x} dx = \frac{1}{2}\left[\delta(\Omega) - \frac{i}{\pi\Omega}\right] \\
\mathcal{F}[\chi(x)](\Omega) &= \int\limits_{-1/2}^{1/2} e^{-2\pi j \Omega x} dx = \frac{\sin(\pi\Omega)}{\pi\Omega}
\end{aligned}$$

Lemma A.1. *If a signal varies only along one dimension, that means the intrinsic dimensionality of a signal is only one dimensional $f : \mathbb{R}^d \to \mathbb{R}$ with $f(\boldsymbol{x}) = f_{\boldsymbol{n}}(\boldsymbol{n}^{\mathrm{T}}\boldsymbol{x})$, then we get the following expression for its Fourier transform:*

$$\hat{f}(\Omega) = \hat{f}_{\boldsymbol{n}}(\Omega^{\mathrm{T}}\boldsymbol{n})\delta(\Omega^{\mathrm{T}}\boldsymbol{n}_2^{\perp}) \cdot \ldots \cdot \delta(\Omega^{\mathrm{T}}\boldsymbol{n}_d^{\perp}) \tag{A.8}$$

Proof.

$$\begin{aligned}
\hat{f}(\Omega) &= \int f(\boldsymbol{x}) e^{-2\pi i \Omega^{\mathrm{T}} \boldsymbol{x}} d\boldsymbol{x} \\
&= \int f_{\boldsymbol{n}}(\boldsymbol{n}^{\mathrm{T}} \boldsymbol{x}) e^{-2\pi i \Omega^{\mathrm{T}} \boldsymbol{x}} d\boldsymbol{x} \\
& \text{change coordinates } \boldsymbol{x} = N \boldsymbol{x}_{\boldsymbol{n}} = [\boldsymbol{n}, \boldsymbol{n}_2^{\perp}, \ldots, \boldsymbol{n}_d^{\perp}][x_{\boldsymbol{n}}, x_{\boldsymbol{n}_2^{\perp}}, \ldots, x_{\boldsymbol{n}_d^{\perp}}]^{\mathrm{T}} \\
& \text{where } \boldsymbol{n}^{\mathrm{T}} \boldsymbol{n}_i^{\perp} = 0 \forall i,\ det(N) = 1 \\
&= \int f_{\boldsymbol{n}}(\boldsymbol{n}^{\mathrm{T}} N \boldsymbol{x}_{\boldsymbol{n}}) e^{-2\pi i \Omega^{\mathrm{T}} N \boldsymbol{x}_{\boldsymbol{n}}} d\boldsymbol{x}_{\boldsymbol{n}} \\
&= \int \left(f_{\boldsymbol{n}}(x_{\boldsymbol{n}}) e^{-2\pi i \Omega^{\mathrm{T}} \boldsymbol{n} x_{\boldsymbol{n}}} \right) e^{-2\pi i \Omega^{\mathrm{T}} [\boldsymbol{n}_2^{\perp}, \ldots, \boldsymbol{n}_d^{\perp}][x_{\boldsymbol{n}_2^{\perp}}, \ldots, x_{\boldsymbol{n}_d^{\perp}}]^{\mathrm{T}}} d\boldsymbol{x}_{\boldsymbol{n}} \\
&= \hat{f}_{\boldsymbol{n}}(\Omega^{\mathrm{T}} \boldsymbol{n}) \delta(\Omega^{\mathrm{T}} \boldsymbol{n}_2^{\perp}) \cdot \ldots \cdot \delta(\Omega^{\mathrm{T}} \boldsymbol{n}_d^{\perp}) \qquad \text{(A.9)}
\end{aligned}$$

□

This result can now be applied to the case of a linear step edge function of arbitrary orientation in the 2D plane which is defined as:

$$\Upsilon(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{n}^{\mathrm{T}} \boldsymbol{x} \geq 0 \\ 0 & \text{otherwise} \end{cases} \qquad \text{(A.10)}$$

Since the Fourier transform of a 1-dimensional step function is given by $\hat{\Upsilon}(\Omega) = \frac{1}{2}(\delta(\Omega) - \frac{i}{\pi\Omega})$, we get using the result from Eq.A.9:

$$\hat{\Upsilon}(\boldsymbol{\Omega}) = \frac{1}{2}(\delta(\boldsymbol{\Omega}^{\mathrm{T}} \boldsymbol{n}) - \frac{i}{\pi \boldsymbol{\Omega}^{\mathrm{T}} \boldsymbol{n}}) \delta(\boldsymbol{\Omega}^{\mathrm{T}} \boldsymbol{n}^{\perp}) = \frac{1}{2}(\delta(\boldsymbol{\Omega}) - \frac{i\delta(\boldsymbol{\Omega}^{\mathrm{T}} \boldsymbol{n}^{\perp})}{\pi \boldsymbol{\Omega}^{\mathrm{T}} \boldsymbol{n}}) \qquad \text{(A.11)}$$

We can generalize this to describe the Fourier transform of any degenerate signal. Given the intensity function $f : \mathbb{R}^m \to \mathbb{R}$ and the matrix $J \in \mathbb{R}^{m \times n}$, then the Fourier transform of the function $f(J\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^n$, is given by

$$\hat{f}(\boldsymbol{\Omega}) = \int f(J\boldsymbol{x}) e^{-\boldsymbol{\Omega}^{\mathrm{T}} \boldsymbol{x}} d\boldsymbol{x} \qquad \text{(A.12)}$$

Let's assume that the singular value decomposition of J is given by

$$J = VSU^{\mathrm{T}} = V[S_{\|} S_{\perp}] \begin{bmatrix} U_{\|}^{\mathrm{T}} \\ U_{\perp}^{\mathrm{T}} \end{bmatrix} \tag{A.13}$$

where $V \in \mathbb{R}^{m \times m}$ and $U \in \mathbb{R}^{n \times n}$ are unitary matrices, S is split in a diagonal matrix $S_{\|}$ that contains the nonzero and $S_{\perp}$ that contains the zero singular values of J. $U_{\|}$ and $U_{\perp}$ contain the corresponding singular vectors. Thus when we do the variable substitution $\tilde{\boldsymbol{x}} = J\boldsymbol{x}$, we can express $\boldsymbol{x}$ as

$$\boldsymbol{x} = U_{\|} S_{\|}^{-1} V^{\mathrm{T}} \tilde{\boldsymbol{x}} + U_{\perp} \boldsymbol{x}_{\perp} \tag{A.14}$$

where $\boldsymbol{x}_{\perp}$ are parameters that can be chosen arbitrary because the vectors $U_{\perp}\boldsymbol{x}_{\perp}$ are lying in the null space of J. Thus, if we plug these values into the Eq. (A.12), we get

$$\hat{f}(\boldsymbol{\Omega}) = \frac{1}{\det(S)} \int f(\tilde{\boldsymbol{x}}) e^{-\boldsymbol{\Omega}^{\mathrm{T}} U_{\|} S_{\|}^{-1} V^{\mathrm{T}} \tilde{\boldsymbol{x}}} e^{-\boldsymbol{\Omega}^{\mathrm{T}} U_{\perp} \boldsymbol{x}_{\perp}} d\tilde{\boldsymbol{x}} d\boldsymbol{x}_{\perp} \tag{A.15}$$

$$= \frac{1}{\det(S)} \hat{f}(VS^{-1} U_{\|}^{\mathrm{T}} \boldsymbol{\Omega}) \delta(U_{\perp}^{\mathrm{T}} \boldsymbol{\Omega}) \tag{A.16}$$

We see that Eq. (A.11) is a special case of the above, where we set $S_{\|} = 1$, $U_{\|} = \boldsymbol{n}$, and $U_{\perp} = [x_{\boldsymbol{n}_2^{\perp}}, \ldots, x_{\boldsymbol{n}_d^{\perp}}]$.

A.1.2 Poisson Summation Formula

The following relation is known as the Poisson Summation Formula. Here we will give the n-dimensional version for the lattice $\mathcal{A}$ defined by the lattice matrix A.

$$\sum_{\boldsymbol{k} \in \mathbb{Z}^n} f(\boldsymbol{x} + A\boldsymbol{k}) = \frac{1}{|\det(A)|} \sum_{\boldsymbol{m} \in \mathbb{Z}^n} \hat{f}(A^{-\mathrm{T}} \boldsymbol{m}) \exp(-2\pi i (A^{-\mathrm{T}} \boldsymbol{m})^{\mathrm{T}} \boldsymbol{x}) \tag{A.17}$$

which can be proven quite easily for nice functions, for example if $f \in \mathbf{L}_1(\mathbb{R}^n)$ and smooth so that $\|(1 + \boldsymbol{x}^2)^N f(x)\| < \infty \; \forall N$.

We start by defining $F(\boldsymbol{x}) = \sum_{k\in\mathbb{Z}^n} f(\boldsymbol{x} + A\boldsymbol{k})$. Since $f \in \mathbf{L}_1(\mathbb{R}^n)$, this sum converges absolutely and uniformly and the resulting function is smooth and is periodic over the lattice $\mathcal{A}$. It follows that F has a Fourier expansion $F(\boldsymbol{x}) = \sum_{m\in\mathbb{Z}^n} a_m \exp(-2\pi i\boldsymbol{m}^{\mathrm{T}}A^{-1}\boldsymbol{x})$ where the coefficients are given by (V_A is the Voronoi cell corresponding to the lattice matrix A)

$$a_m = \frac{1}{|\det(A)|}\int\limits_{V_A} F(\boldsymbol{x})\exp(2\pi i\boldsymbol{m}^{\mathrm{T}}A^{-1}\boldsymbol{x})d\boldsymbol{x}$$

$$= \frac{1}{|\det(A)|}\int\limits_{V_A}\sum_{\boldsymbol{k}\in\mathbb{Z}^n} f(\boldsymbol{x}+A\boldsymbol{k})\exp(2\pi i\boldsymbol{m}^{\mathrm{T}}A^{-1}\boldsymbol{x})d\boldsymbol{x}.$$

since $\exp(2\pi i\boldsymbol{m}^{\mathrm{T}}A^{-1}\boldsymbol{x}) = \exp(2\pi i\boldsymbol{m}^{\mathrm{T}}A^{-1}(\boldsymbol{x}+A\boldsymbol{k}))$ we have

$$a_m = \frac{1}{|\det(A)|}\int\limits_{\mathbb{R}^n} f(\boldsymbol{x})\exp(2\pi i\boldsymbol{m}^{\mathrm{T}}A^{-1}\boldsymbol{x}) = \frac{1}{|\det(A)|}\hat{f}(A^{-\mathrm{T}}\boldsymbol{m})$$

and we see that

$$F(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^n} f(\boldsymbol{x}+A\boldsymbol{k}) = \frac{1}{|\det(A)|}\sum_{\boldsymbol{m}\in\mathbb{Z}^n}\hat{f}(A^{-\mathrm{T}}\boldsymbol{m})\exp(-2\pi i(A^{-\mathrm{T}}\boldsymbol{m})^{\mathrm{T}}\boldsymbol{x})$$

If we set $n = A = 1$, then we have the well-known one-dimensional case, $\sum_n \hat{f}(n)e^{2i\pi nx} = \sum_n f(x+n)$.

A.1.3 Cauchy-Schwartz Inequality

Let $a, b \in \mathbb{R}^n$, then the Cauchy-Schwartz Inequality states that

$$\left|\sum_{k=1}^{n} a_k b_k\right| \leq \sqrt{\sum_{k=1}^{n} a_k^2}\sqrt{\sum_{k=1}^{n} b_k^2}$$

A.1.4 Minkowski Inequality

If $p > 1$, then the Minkowski inequality states that

$$\left(\int\limits_a^b |f(x)+g(x)|^p dx\right)^{\frac{1}{p}} \leq \left(\int\limits_a^b |f(x)|^p dx\right)^{\frac{1}{p}} + \left(\int\limits_a^b |g(x)|^p dx\right)^{\frac{1}{p}}$$

Similarly, if $p > 1$, and $a_k, b_k > 0$, then the Minkowski inequality is also valid for sums of sequences and we have

$$\left(\sum_{k=1}^{n}(a_k + b_k)^p\right)^{\frac{1}{p}} \leq \left(\sum_{k=1}^{n}(a_k)^p\right)^{\frac{1}{p}} \left(\sum_{k=1}^{n}(b_k)^p\right)^{\frac{1}{p}}$$

A.1.5 Sobolev Space

Let r be a positive real number and $\Omega \subset \mathbb{R}^n$. The Sobolev space $\mathbf{W}_p^r$ is defined as

$$\mathbf{W}_p^r(\Omega) := \{f \in L^p(\Omega) | \partial^\alpha f \in L^p(\Omega) \forall \text{ multi-index } \alpha, |\alpha| < r\} \tag{A.18}$$

That is the collection of functions satisfying $\int (1 + v^2)^r |\hat{f}(v)|^2 dv < \infty$. This is equivalent to saying that f and its first r derivatives are square-integrable. This definition extends to n dimensions. Let f be a real-valued function on $\mathbb{R}^n$, then we can define

$$\|f\|_r = \left(\sum_{i=0}^{r} \int |D^i f(\boldsymbol{x})|^2 d\boldsymbol{x}\right)^{\frac{1}{2}} = \left(\|2\pi \boldsymbol{v}\|^{2r} |\hat{f}(\boldsymbol{v})|^r d\boldsymbol{v}\right)^{\frac{1}{2}}.$$

The space of functions for which this quantity is finite is the Sobolev space $\mathbf{W}_2^r$. Here, $|D^i f|^2$ denotes the sum of the squares of all partial derivatives of f of order i. Thus, the Sobolev space $\mathbf{W}_2^r$ is the space of functions whose partial derivatives up to order r are square integrable. Similar spaces can be defined for vector valued functions by taking a sum of contributions from the separate components in the integral. It is also possible to define Sobolev spaces on any Riemannian manifold, using covariant derivatives. In the case of natural signals, we can assume that all natural signals after they passed through an optical system have a sufficiently high degree of smoothness.

A.2 Derivation of Quantitative Error Bound

A.2.1 Definitions for Sampling and Synthesis Functions

In contrast to the work by Blu et al. [12], whose argument we will follow closely, we are interested in a function space where the spacing of the samples and the scale of the synthesis function are not necessarily coupled. This means, instead of a function space of the form $\{\phi(A^{-1} \cdot -\boldsymbol{k})\}$, we will consider a function space of the form $\{\phi(A^{-1}(\cdot - B\boldsymbol{k}))\}$, which will lead to synthesis functions whose sampling lattice is defined by a lattice matrix B and a scaling matrix A. We do this, because we are interacting with a physical space, the environment, of a fixed physical dimension. Thus, there is a difference between the scaling of the sampling function, integration domain of a pixel, and the distance between sampling points which is given by the geometry of the imaging surface.

Thus, we define the sampling operator $\mathcal{S}_{(A,B)}$ as

$$\mathcal{S}_{(A,B)} :\to \left\{ \frac{1}{|\det(A)|} \int f(\boldsymbol{\tau})\psi_k(A^{-1}(\boldsymbol{\tau} - B\boldsymbol{k}))d\boldsymbol{\tau} \right\}_{\boldsymbol{k}\in\mathbb{Z}^n} \tag{A.19}$$

that defines the coefficients of the linear decomposition of the approximation functional

$$\mathcal{D}_{(A,B)}f(\boldsymbol{x}) = \frac{1}{|\det(A)|} \sum_k \int f(\boldsymbol{\tau})\psi(A^{-1}(\boldsymbol{\tau} - B\boldsymbol{k}))\phi(A^{-1}(\boldsymbol{x} - B\boldsymbol{k}))d\boldsymbol{\tau} \tag{A.20}$$

Definition A.2. Definition of Order The set of synthesis functions ϕ is of order L iff there exists L real sequences $\{\lambda_n^{(s)}\}_{n\in\mathbb{Z}}$ such that, in the sense of distributions,

$$x^s = \sum_n \lambda_n^{(s)} \phi_n(x) \text{ for } s = 0, \ldots, L-1 \tag{A.21}$$

This implies that the synthesis functions can reproduce polynomials up to order L.

Definition A.3. Quasi-orthonormality of Order L A set of sampling distributions $\{\psi_n\}_{n\in\mathbb{Z}}$

and synthesis functions $\{\phi_n\}_{n\mathbb{Z}}$ constitute a quasi-orthonormal set of order L iff the following two conditions are met:

- the functions ϕ_n are of order L
- the distributions ψ_n satisfy the moment conditions

$$\int x^s \psi_n(x)dx = \lambda_n^{(s)} \text{ for } n \in \mathbb{Z} \text{ and } s = 0, \ldots, L-1 \tag{A.22}$$

  where the sequence $\lambda_n^{(s)}$ satisfies (A.21).

This condition ensures that polynomial functions up to order L can be recovered exactly by this pair of sampling and synthesis spaces.

A.2.2 Hypotheses

For the following theorems to hold, we have to make the following hypotheses

- on the synthesis functions ϕ_k:

  1. $\phi_k \in \mathbf{L}_2(\mathbb{R}^n)$ and it satisfies the Riesz condition

  $$C_1 \sum_k |c_k|^2 \leq \| \sum_k c_k \phi_k \|_{L_2}^2 \leq C_2 \sum_k |c_k|^2 \; \forall \{c_k\} \in l^2(\mathbb{Z}^n)$$

     where $0 < C_1 \leq C_2 < \infty$. This implies that the synthesis functions are linearly independent whenever l^2-sequences are considered.

  2. The vector space M of all possible coefficients λ_n such that $\sum \lambda_n \phi_n = 0$ in the sense of distributions has a finite dimension.

  3. If ϕ_n are of order L, then $\int |x|^r |\phi_k(x)| dx$ is finite for $r = 0, \ldots, L$. This implies that the derivatives of the Fourier transform are bounded and L-times continuously differentiable.

- on the sampling functions: we only assume that ψ has a bounded Fourier transform,
- on the functions to approximate: f needs to be chosen such that $\{\mathcal{S}_{(A,B)}(f)_k\} \in l^2(\mathbb{Z}^n)$. The convergence of the coefficients is ensured if f is in $\mathbf{W}_2^r$ for $r > \frac{1}{2}$. This implies the requirement of Hölder continuity with exponent $r - \frac{1}{2}$.

It is of interest how well a given synthesis function is able to approximate a given function. We will measure this using the $\mathbf{L}^2$-norm of the difference between original function and approximation given by

$$\epsilon_f = \|f - \mathcal{D}_{(A,B)}f\|_{\mathbf{L}^2} \tag{A.23}$$

This error term ϵ_f can be split into two terms, one dominating main term that expresses the idea of an average error over all possible phase positions of a signal and a perturbation. The main term is computed by integrating $|\hat{f}(\boldsymbol{v})|^2$, where $\hat{f}$ is the Fourier transform of f, against an error kernel of the form

$$E(\boldsymbol{v}) = |1 - \Omega\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v})|^2 + \sum_{k \in \mathbb{Z}^n\{0\}} |\Omega\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})|^2 \tag{A.24}$$

where $\Omega = |\det(A)|/|\det(B)|$. The additional correction term $e(f, A, B)$ depends on the Sobolev regularity exponent of the function to be approximated.

In this chapter we will outline the steps to derive the quantization of the error with respect to camera spacing and frequency of the signal. This is based on the descriptions in the papers [13, 12]. They describe the 1D case and we will extend it to the n-D case where the sampling pattern is given by a general scaling matrix A and sampling matrix B.

Theorem A.4. *(Adapted from [12]) For all $f \in \mathbf{W}_2^r$ with $r > 1/2$ the approximation error averaged over all possible phase shifts of the function f with regard to the sampling operator*

$\mathcal{D}_{(A,B)}$ *is given by*

$$\epsilon_f = \langle \|f - \mathcal{D}_{(A,B)} f\|_{\mathbf{L}^2} \rangle = \left[\int |\hat{f}(\boldsymbol{v})|^2 E(A^{\mathrm{T}}\boldsymbol{v}) dv \right]^{1/2} \tag{A.25}$$

The proof consists of three parts, which we will detail in the next subsections.

A.2.3 l^2 Convergence of the Samples

First we will define the following $B^{-\mathrm{T}}$-periodic functions

$$U(\boldsymbol{v}) = \frac{1}{\det(B)} \sum_{m} \overline{\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{m})} \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{m}) \tag{A.26}$$

$$V(\boldsymbol{v}) = \frac{1}{\det(B)} \sum_{m} \overline{\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{m})} \hat{\psi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{m}) \tag{A.27}$$

and prove that these functions are well defined and belong to $\mathbf{L}_2(V_{B^{-\mathrm{T}}})$ where $V_{B^{-\mathrm{T}}}$ is the Voronoi cell defined by the lattice matrix $B^{-\mathrm{T}}$, that is $V_{B^{-\mathrm{T}}} := \{B^{-\mathrm{T}}\boldsymbol{x} | \boldsymbol{x} \in [-1/2, 1/2]^n\}$.

If this is the case, then we can develop these $B^{-\mathrm{T}}$-periodic functions into a Fourier series

$$U(\boldsymbol{v}) = \sum_{k \in \mathbb{Z}^n} a_k \exp(2\pi i \boldsymbol{v}^{\mathrm{T}} B\boldsymbol{k}) \tag{A.28}$$

$$a_k = \int\limits_{V_{B^{-\mathrm{T}}}} \frac{|\det(B)|}{|\det(B)|} \sum_{k} \overline{\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})} \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k}) \exp(2\pi i \boldsymbol{v}^{\mathrm{T}} B\boldsymbol{k}) d\boldsymbol{v} \tag{A.29}$$

$$= \int\limits_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})} \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v}) \exp(2\pi i \boldsymbol{v}^{\mathrm{T}} B\boldsymbol{m}) d\boldsymbol{v} \tag{A.30}$$

Using the fact that the inverse Fourier transform of the product of one function and the complex conjugate of another in the Fourier domain equals the correlation of the two functions in the signal domain, that is

$$\int\limits_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})} \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v}) \exp(-2\pi i \boldsymbol{v}^{\mathrm{T}}(-B\boldsymbol{k})) d\boldsymbol{v} = \mathcal{F}^{-1}\left\{ \overline{\hat{f}(\boldsymbol{v})} \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v}) \right\} (-B\boldsymbol{k})$$

$$= \frac{1}{|\det(A)|} (f \circ \phi_{A^{-1}})(-B\boldsymbol{k}) = \frac{1}{|\det(A)|} \int\limits_{\mathbb{R}^n} f(\tau) \phi(A^{-1}(\boldsymbol{\tau} - B\boldsymbol{k})) d\boldsymbol{\tau}$$

we can write down an alternative definition for U and V given by

$$U(\boldsymbol{v}) = \sum_k \left[\frac{1}{|\det(A)|}\int f(\boldsymbol{\tau})\phi(A^{-1}(\boldsymbol{\tau} - B\boldsymbol{k}))d\boldsymbol{\tau}\right]\exp(2\pi i \boldsymbol{v}^{\mathrm{T}}B\boldsymbol{k})$$
$$V(\boldsymbol{v}) = \sum_k \left[\frac{1}{|\det(A)|}\int f(\boldsymbol{\tau})\psi(A^{-1}(\boldsymbol{\tau} - B\boldsymbol{k}))d\boldsymbol{\tau}\right]\exp(2\pi i \boldsymbol{v}^{\mathrm{T}}B\boldsymbol{k}) \tag{A.31}$$

Remark A.5. The following section has not been checked yet!

We will prove the well-posedness of the function U as follows (the procedure for V is exactly the same). First we will define the functional sequence

$$U_K(\boldsymbol{v}) = \frac{1}{|\det(B)|}\sum_{|\boldsymbol{k}|\le K}\overline{\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})$$

for $K \in \mathbb{N}_+^n$. We would like to prove that U_K is a Cauchy sequence, that means we need to show that

$$\lim_{K\to\infty}\sup_{K'>K}\|U_{K'} - U_K\|_{\mathbf{L}_2(I)} = 0$$

By the Fisher-Riesz theorem this will automatically prove the convergence of U_K towards an $\mathbf{L}_2(\mathbb{R}^n)$ function U.

We choose $K' > K$, and assume that $f \in \mathbf{W}_2^r(\mathbb{R}^n)$ with $r > \frac{1}{2}$ and that $\|\hat{\phi}\|_\infty \le C < \infty$. Then we define the set of bandpass functions f_m which are defined as

$$\hat{f}_m = \begin{cases} \hat{f}(\boldsymbol{v}) & \text{if } \frac{m}{2|\det(B)|} \le \|\boldsymbol{v}\| < \frac{m+1}{2|\det(B)|} \\ \\ 0 & \text{elsewhere} \end{cases} \tag{A.32}$$

which allows us to write

$$U_{K'}(\boldsymbol{v}) - U_K(\boldsymbol{v}) = \frac{1}{|\det(B)|}\sum_{m\ge 0}\sum_{K<|k|\le K'}\overline{\hat{f}_m(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})$$

Not all f_m contribute to the sum on the right-hand because of their limited support. We choose only the $m > 2K$, then by applying the Minkowski Inequality and the bound on $\hat{\phi}$ assumed before we find that

$$\|U_{K'}(\boldsymbol{v}) - U_K(\boldsymbol{v})\|_{\mathbf{L}_2(I)} \leq \frac{C}{|\det(A)|} \sum_{m>2K} \|f_m\|_{\mathbf{L}_2}$$

Looking at the definition of f_m in (A.32), we see that on the support of f_m we have $\|2\pi\boldsymbol{v}\|^{-r} \leq (|\det(B)|/(m\pi))^r$, so that we can bound the norm of f_m by the norm of its rth derivative $\|f_m\|_{\mathbf{L}_2} \leq \left(\frac{|\det(B)|}{m\pi}\right)^r \|f_m^{(r)}\|_{\mathbf{L}_2}$

Using the Cauchy-Schwartz Inequality for discrete sequences, we find that

$$\sum_{m>2K} \|f_m\|_{\mathbf{L}_2} \leq \left(\frac{\det(B)}{\pi}\right)^r \sqrt{\sum_{m>2K} m^{-2r}} \|f^{(r)}\|_{\mathbf{L}_2}$$

This expression tends to zero as K goes to infinity, therefore U_K is a functional Cauchy sequence. This makes sure that the sum of the squared samples converges and proves that $U \in \mathbf{L}_2(I)$.

A.2.4 Expression of ϵ_f in Fourier Variables

In this section, we will expand the $\mathbf{L}_2$-error ϵ_f into three terms

$$\epsilon_f^2 = \|f\|_{\mathbf{L}_2}^2 - 2\langle f, \mathcal{D}_{(A,B)} f\rangle + \|\mathcal{D}_{(A,B)} f\|_{\mathbf{L}_2}^2$$

and examine each term. We will start with Eq.(A.20)

$$\mathcal{D}_{(A,B)} f(\boldsymbol{x}) = \frac{1}{|\det(A)|} \sum_{k} \int f(\boldsymbol{\tau}) \psi(A^{-1}(\boldsymbol{\tau} - B\boldsymbol{k})) \phi(A^{-1}(\boldsymbol{x} - B\boldsymbol{k})) d\boldsymbol{\tau}$$

We see that the part of the integral looks a lot like the expression for U in Eq. (A.31). We now express $\phi(A^{-1}(\boldsymbol{x} - B\boldsymbol{k}))$ in terms of its Fourier transform

$$\phi(A^{-1}(\boldsymbol{x} - B\boldsymbol{k})) = |\det(A)| \int \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v}) \exp(2\pi i \boldsymbol{v}^{\mathrm{T}} B\boldsymbol{k}) \exp(-2\pi i \boldsymbol{x}^{\mathrm{T}} \boldsymbol{v}) d\boldsymbol{v}$$

then we can manipulate Eq. (A.2.4)

$$\begin{aligned}\mathcal{D}_{(A,B)}f(\boldsymbol{x}) = \int \left[\sum_{k} \int f(\tau)\psi(A^{-1}(\boldsymbol{\tau} - B\boldsymbol{k}))d\boldsymbol{\tau}\exp(2\pi i\boldsymbol{v}^{\mathrm{T}}B\boldsymbol{k})\right] \cdot \\ \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})\exp(-2\pi i\boldsymbol{x}^{\mathrm{T}}\boldsymbol{v})d\boldsymbol{v} \\ = |\det(A)| \int \overline{V(\boldsymbol{v})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})\exp(-2\pi i v^{\mathrm{T}}\boldsymbol{v})d\boldsymbol{v}\end{aligned}$$

We can see that $\mathcal{D}_{(A,B)}f(\boldsymbol{x})$ and $|\det(A)| \int \overline{V(\boldsymbol{v})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{x})$ are Fourier transforms of each other and thus have the same $\mathbf{L}_2$-norm. Remembering that $V(\boldsymbol{v})$ is $B^{-\mathrm{T}}$-periodic, we can write

$$\begin{aligned}&\|\mathcal{D}_{(A,B)}f(\boldsymbol{x})\|_{\mathbf{L}_2} && \text{(A.33)}\\ =&|\det(A)|^2 \int \overline{V(\boldsymbol{v})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})\overline{\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})}V(\boldsymbol{v})d\boldsymbol{v}\\ =&|\det(A)|^2 \sum_{k} \int\limits_{V_{B^{-\mathrm{T}}}} \overline{V(\boldsymbol{v})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})}V(\boldsymbol{v})d\boldsymbol{v}\\ =&|\det(A)|^2 \int\limits_{V_{B^{-\mathrm{T}}}} \overline{V(\boldsymbol{v})}\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v})V(\boldsymbol{v})d\boldsymbol{v} && \text{(A.34)}\end{aligned}$$

where

$$\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v}) = \left[\sum_{k} \hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})}\right] \tag{A.35}$$

The same trick (splitting or reassembling the infinite integral by using the periodicity) can now be applied to write the product of the two infinite sums in $V(\boldsymbol{v})$ as a combination of a sum and an infinite integral. Starting from

$$
\begin{aligned}
&\|\mathcal{D}_{(A,B)}f(\boldsymbol{t})\|^2_{\mathbf{L}_2}\\
=&|\det(A)|^2 \int\limits_{V_{B^{-\mathrm{T}}}} \overline{V(\boldsymbol{v})}\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v})V(\boldsymbol{v})d\boldsymbol{v}\\
=&\frac{|\det(A)|^2}{|\det(B)|^2}\int\limits_{V_{B^{-\mathrm{T}}}} \left[\sum_k \hat{f}(\boldsymbol{v}+B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v}+A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})}\right]\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v})\\
&\left[\sum_k \overline{\hat{f}(\boldsymbol{v}+B^{-\mathrm{T}}\boldsymbol{k})}\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v}+A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})\right]d\boldsymbol{v}\\
=&\ \frac{|\det(A)|^2}{|\det(B)|^2}\sum_k\int\limits_{\mathbb{R}^n}\overline{\hat{f}(\boldsymbol{v})}\hat{f}(\boldsymbol{v}+B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v})}\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v})\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v}+A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})d\boldsymbol{v} \qquad \text{(A.36)}
\end{aligned}
$$

A similar approach can now be done for $\langle f, \mathcal{D}_{(A,B)}f\rangle$. We have

$$
\begin{aligned}
&\langle f, \mathcal{D}_{(A,B)}f\rangle\\
=&\int\limits_{\mathbb{R}^n}\overline{f(\boldsymbol{x})}\left[\sum_{k\in\mathbb{Z}^n}\int\limits_{\mathbb{R}^n} f(\boldsymbol{\tau})\hat{\psi}(A^{-1}(\boldsymbol{\tau}-B\boldsymbol{k}))\hat{\phi}(A^{-1}(\boldsymbol{x}-B\boldsymbol{k}))d\frac{\boldsymbol{\tau}}{|\det(A)|}\right]d\boldsymbol{x}\\
=&\left[\int\limits_{\mathbb{R}^n}\overline{f(\boldsymbol{x})}\hat{\phi}(A^{-1}(\boldsymbol{x}-B\boldsymbol{k}))d\boldsymbol{x}\right]\left[\sum_k\int\limits_{\mathbb{R}^n} f(\tau)\hat{\psi}(A^{-1}(\boldsymbol{\tau}-B\boldsymbol{k})d\frac{\boldsymbol{\tau}}{|\det(A)|}\right]
\end{aligned}
$$

We now use the previous derivation to write

$$
\begin{aligned}
\langle f, \mathcal{D}_{(A,B)}f\rangle &= \int\limits_{\mathbb{R}^n}\overline{f(\boldsymbol{x})}\left[|\det(A)|\int\limits_{\mathbb{R}^n}V(\boldsymbol{v})\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})\exp(2\pi i\boldsymbol{x}^{\mathrm{T}}\boldsymbol{v})d\boldsymbol{v}\right]d\boldsymbol{x}\\
&= |\det(A)|\int\limits_{\mathbb{R}^n}\left[\int\limits_{\mathbb{R}^n}\overline{f(\boldsymbol{x})\exp(2\pi i\boldsymbol{x}^{\mathrm{T}}\boldsymbol{v})}d\boldsymbol{x}\right]V(\boldsymbol{v})\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})d\boldsymbol{v}\\
&= |\det(A)|\int\limits_{\mathbb{R}^n}\overline{f(\hat{\boldsymbol{v}})}V(\boldsymbol{v})\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})d\boldsymbol{v}
\end{aligned}
$$

This can now further simplified by

$$
\begin{aligned}
\langle f, \mathcal{D}_{(A,B)} f\rangle &= |\det(A)| \int\limits_{V_{B^{-\mathrm{T}}}} \sum_k \overline{f(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})V(\boldsymbol{v})d\boldsymbol{v} \\
&= |\det(A)||\det(B)| \int\limits_{V_{B^{-\mathrm{T}}}} \overline{U(\boldsymbol{v})}V(\boldsymbol{v})d\boldsymbol{v}
\end{aligned}
$$

Again, we write U and V as sums

$$
\begin{aligned}
&= \frac{|\det(A)|}{|\det(B)|} \int\limits_{V_B^{-\mathrm{T}}} \left[\sum_k \overline{\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{
\right. \\
&\cdot \left[\sum_k \hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})}\right] d\boldsymbol{v} \\
&= \frac{|\det(A)|}{|\det(B)|} \sum_n \int\limits_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})}d\boldsymbol{v}
\end{aligned}
$$

Thus we can conclude that the final error measure consists of the following terms (we write $\Omega = |\det(A)|/|\det(B)|$):

$$
\begin{aligned}
\epsilon_f &= \|f\|^2_{\mathbf{L}_2} - 2\langle f, \mathcal{D}_T f\rangle + \|\mathcal{D}_T f\|^2_{\mathbf{L}_2} \\
&= \int\limits_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})}\hat{f}(\boldsymbol{v})d\boldsymbol{v} \\
&- 2\Omega \sum_k \int\limits_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})}\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{n})\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})d\boldsymbol{v} \\
&+ \Omega^2 \sum_k \int\limits_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})}\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{n})\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})}\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v})\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v})d\boldsymbol{v}
\end{aligned}
$$

We will now collect the terms for $\boldsymbol{k} = 0$ and $\boldsymbol{k} \neq 0$:

$$
\begin{aligned}
\epsilon_f &= \int\limits_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})}\hat{f}(\boldsymbol{v}) \left[1 - 2\cdot\Omega\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v})}\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v}) + \Omega^2\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v})}\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v})\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v})\right] d\boldsymbol{v} && \text{(A.37)} \\
&+ \sum_{k\neq 0} \int_{\mathbb{R}^n} \overline{\hat{f}(\boldsymbol{v})}\hat{f}(\boldsymbol{v} + B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})} \left[\Omega^2\mathcal{A}(A^{\mathrm{T}}\boldsymbol{v})\hat{\psi}(A^{\mathrm{T}}\boldsymbol{v}) - 2\Omega\hat{\phi}(A^{\mathrm{T}}\boldsymbol{v})\right] d\boldsymbol{v} \\
&= \epsilon_1^2 + \epsilon_2^2 && \text{(A.38)}
\end{aligned}
$$

Then we can rewrite ϵ_1 as the integration of $|\hat{f}(\boldsymbol{v})\|^2$ with the error Kernel $E(A^{\mathrm{T}}\boldsymbol{v})$ where the error kernel becomes:

$$\begin{aligned}
E(v) &= \left[1 - 2\Omega \cdot \overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v}) + \Omega^2 \overline{\hat{\psi}(\boldsymbol{v})}\mathcal{A}(\boldsymbol{v})\hat{\psi}(\boldsymbol{v})\right] \\
&= 1 - 2\Omega\mathcal{R}\{\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v})\} + \Omega^2|\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v})|^2 + \Omega^2 \sum_{k\neq 0} \left|\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})\right|^2 \\
&= \left|1 - \Omega\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v})\right|^2 + \sum_{k\neq 0}\left|\Omega\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})\right|^2
\end{aligned}$$

If we use orthonormalized basis function, we get a simpler expression for the error kernel. That is we normalize ϕ and ψ by premultiplying with $1/\sqrt{\mathcal{A}(\boldsymbol{v})}$ where

$$\mathcal{A}(\boldsymbol{v}) = \sum_k \hat{\phi}(\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})\overline{\hat{\phi}(\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})} = \sum_n |\hat{\phi}(\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})|^2 \tag{A.39}$$

is the Fourier transform of the sampled auto-correlation sequence $\hat{\mathcal{A}}(\phi)$ of ϕ,

$$\hat{\mathcal{A}}(\phi) = \sum_k \phi(\boldsymbol{x})\phi(\boldsymbol{x} - A^{-1}B\boldsymbol{k})$$

to compute the ortho-normal basis functions $\tilde{\phi}$ and $\tilde{\psi}$.

$$\begin{aligned}
E(\boldsymbol{v}) &= 1 - 2\Omega\mathcal{R}\{\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v})\} + |\hat{\psi}(v)|^2 \sum_k |\Omega\hat{\phi}(\boldsymbol{v} + A^{\mathrm{T}}B^{-\mathrm{T}}\boldsymbol{k})|^2 \\
&= 1 - 2\Omega\mathcal{R}\{\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v})\} + |\Omega\hat{\psi}(\boldsymbol{v})|^2\mathcal{A}(\boldsymbol{v}) \\
&= 1 - \frac{|\hat{\phi}(\boldsymbol{v})|^2}{A(\boldsymbol{v})} + A(\boldsymbol{v})\left(\frac{|\hat{\phi}(\boldsymbol{v})|^2}{A(\boldsymbol{v})^2} - 2\Omega\frac{\mathcal{R}\{\overline{\hat{\psi}(\boldsymbol{v})}\hat{\phi}(\boldsymbol{v})\}}{A(\boldsymbol{v})} + |\Omega\hat{\psi}(\boldsymbol{v})|^2\right) \\
&= 1 - \frac{|\hat{\phi}(\boldsymbol{v})|^2}{A(v)} + A(\boldsymbol{v})\left|\Omega\hat{\psi}(\boldsymbol{v}) - \frac{|\hat{\phi}(\boldsymbol{v})|}{A(\boldsymbol{v})}\right|^2
\end{aligned}$$

A.2.5 Average Approximation Error

The approximation is very powerful, because it does not just express an upper bound on the error, but it is actually the exact expression for the average error over all the possible phase shifts of the function f [12], that means we have the identity

$$\int\limits_{V_B} \|f_u - \mathcal{D}_T f\|_{\mathbf{L}^2} du = \left[\int |\hat{f}(\boldsymbol{v})|^2 E(A^{\mathrm{T}}\boldsymbol{v}) dv\right]^{1/2} \tag{A.40}$$

where $f_u := f(\boldsymbol{x} - \boldsymbol{u})$.

Proof. The shift by $\boldsymbol{u}$ corresponds to a multiplication of $\hat{f}(\boldsymbol{v})$ by $\exp(2\pi i \boldsymbol{u}^{\mathrm{T}} \boldsymbol{v})$. The first term of the error kernel ϵ_1 is unchanged since $|\hat{f}(\boldsymbol{v})|^2$ is not affected by this modulation. The effect on the second term of the error kernel ϵ_2 is a premultiplied with a term of the form $\exp(2\pi i \boldsymbol{u}^{\mathrm{T}} B^{\mathrm{T}} \boldsymbol{k})$ which is independent of $\boldsymbol{v}$ and thus we can move it in front of the integral. The integral $\int_{V_B} \exp(2\pi i \boldsymbol{u}^{\mathrm{T}} B^{\mathrm{T}} \boldsymbol{k}) d\boldsymbol{u}$ equals 0 for all $k \neq 0$ and thus ϵ_2 vanishes. □

BIBLIOGRAPHY

[1] E. H. Adelson and J. R. Bergen. The plenoptic function and the elements of early vision. In M. Landy and J. A. Movshon, editors, *Computational Models of Visual Processing*, pages 3–20. MIT Press, Cambridge, MA, 1991.

[2] E. H. Adelson and J. Y. A. Wang. Single lens stereo with a plenoptic camera. *IEEE Trans. PAMI*, 14:99–106, 1992.

[3] G. Adiv. Inherent ambiguities in recovering 3D motion and structure from a noisy flow field. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 70–77, 1985.

[4] P. Baker, R. Pless, C. Fermuller, and Y. Aloimonos. Camera networks for building shape models from video. In *Workshop on 3D Structure from Multiple Images of Large-scale Environments (SMILE 2000)*, 2000.

[5] P. Baker, R. Pless, C. Fermuller, and Y. Aloimonos. A spherical eye from multiple cameras (makes better models of the world). In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[6] Patrick Baker and Yiannis Aloimonos. Translational camera constraints on parallel lines. In *Proc. Europ. Conf. Computer Vision*, page in press, 2004.

[7] G. Baratoff and Y. Aloimonos. Changes in surface convexity and topology caused by distortions of stereoscopic visual space. In *Proc. European Conference on Computer Vision*, volume 2, pages 226–240, 1998.

[8] S.S. Beauchemin and J.L. Barron. On the fourier properties of discontinuous motion. *Journal of Mathematical Imaging and Vision (JMIV)*, 13:155–172, 2000.

[9] P.J. Besl. Active optical range imaging sensors. *Machine Vision Appl.*, 1(2):127–152, 1988.

[10] F. Blais. A review of 20 years of ranges sensor development. In *Videometrics VII Proceedings of SPIE-IST Electronic Imaging*, volume 5013, pages 62–76, 2003.

[11] V. Blanz and T Vetter. Face recognition based on fitting a 3d morphable model. *IEEE Trans. PAMI*, 25(9), 2003.

[12] T. Blu and M. Unser. Approximation error for quasi-interpolators and (multi-) wavelet expansions. *Applied and Computational Harmonic Analysis*, 6(2):219–251, March 1999.

[13] T. Blu and M. Unser. Quantitative Fourier analysis of approximation techniques: Part I—Interpolators and projectors. *IEEE Transactions on Signal Processing*, 47(10):2783–2795, October 1999.

[14] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1:7–55, 1987.

[15] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1:7–55, 1987.

[16] J. S. De Bonet and P. Viola. Roxels: Responsibility weighted 3d volume reconstruction. In *Proceedings of ICCV*, September 1999.

[17] O. Bottema and B. Roth. *Theoretical Kinematics*. North-Holland, 1979.

[18] T.J. Broida, S. Chandrashekhar, and R. Chellappa. Recursive 3-D motion estimation from a monocular image sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26(4):639–656, 1990.

[19] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estiation based on a theory for warping. In *Proc. European Conference on Computer Vision*, volume 4, pages 25–36, 2004.

[20] A. Bruss and B. K. P. Horn. Passive navigation. *CVGIP: Image Understanding*, 21:3–20, 1983.

[21] P. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communication*, 31, 1983.

[22] E. Camahort and D. Fussell. A geometric study of light field representations. Technical Report TR99-35, Dept. of Computer Sciences, The University of Texas at Austin, 1999.

[23] E. Camahort, A. Lerios, and D. Fussell. Uniformly sampled light fields. In *Eurographics Workshop on Rendering*, pages 117–130, 1997.

[24] C. Capurro, F. Panerai, and G. Sandini. Vergence and tracking fusing log-polar images. In *Proc. International Conference on Pattern Recognition*, 1996.

[25] R. L. Carceroni and K. Kutulakos. Multi-view scene capture by surfel sampling: From video streams to non-rigid 3d motion, shape, and reflectance. In *Proc. International Conference on Computer Vision*, June 2001.

[26] J. Chai and H. Shum. Parallel projections for stereo reconstruction. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 493 –500, 2000.

[27] J. Chai, X. Tong, and H. Shum. Plenoptic sampling. In *Proc. of ACM SIGGRAPH*, pages 307–318, 2000.

[28] W. Chen, J. Bouguet, M. Chu, and R. Grzeszczuk. Light field mapping: efficient representation and hardware rendering of surface light fields. *ACM Transactions on Graphics (TOG)*, 21(3):447–456, 2002.

[29] Amit Roy Chowdhury and Rama Chellappa. Stochastic approximation and rate-distortion analysis for robust structure and motion estimation. *International Journal of Computer Vision*, 55(1):27–53, October 2003.

[30] G.M. Cortelazzo and M. Balanza. Frequency domain analysis of translations with piecewise cubic trajectories. *PAMI*, 15(4):411–416, April 1993.

[31] R. Costantini and S. Süsstrunk. Virtual sensor design. In *Proc. IS&T/SPIE Electronic Imaging 2004: Sensors and Camera Systems for Scientific, Industrial, and Digital Photography Applications V*, volume 5301, pages 408–419, 2004.

[32] S. Crossley, N.A. Thacker, and N.L. Seed. Benchmarking of bootstrap temporal stereo using statistical and physical scene modelling. In *British Machine Vision Conference*, pages 346–355, 1998.

[33] K. Daniilidis. *Zur Fehlerempfindlichkeit in der Ermittlung von Objektbeschreibungen und relativen Bewegungen aus monokularen Bildfolgen*. PhD thesis, Fakultät für Informatik, Universität Karlsruhe (TH), 1992.

[34] K. Daniilidis and M. Spetsakis. Understanding noise sensitivity in structure from motion. In *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*, chapter 4, pages 61–88. Lawrence Erlbaum Associates, Hillsdale, NJ, 1997.

[35] K. Daniilidis and M. E. Spetsakis. Understanding noise sensitivity in structure from motion. In Y. Aloimonos, editor, *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*, Advances in Computer Vision, chapter 4. Lawrence Erlbaum Associates, Mahwah, NJ, 1997.

[36] J. Davis, R. Ramamoorthi, and S. Rusinkiewicz. Spacetime stereo: A unifying framework for depth from triangulation. In *2003 Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, pages 359–366, June 2003.

[37] R. Dawkins. *Climbing Mount Improbable*. Norton, New York, 1996.

[38] D. Demirdjian and T. Darrell. Motion estimation from disparity images. In *Proc. International Conference on Computer Vision*, volume 1, pages 213–218, July 2001.

[39] D.W. Dong and J.J. Atick. Statistics of natural time-varying images. *Network: Computation in Neural Systems*, 6(3):345–358, 1995.

[40] A. Edelman, T. Arian, and S. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 1998.

[41] I.A. Essa and A.P. Pentland. Coding, analysis, interpretation, and recognition of facial expressions. *IEEE Trans. PAMI*, 19:757–763, 1997.

[42] H. Farid and E. Simoncelli. Range estimation by optical differentiation. *Journal of the Optical Society of America*, 15(7):1777–1786, 1998.

[43] O. Faugeras and R. Keriven. Complete dense stereovision using level set methods. In *Proc. European Conference on Computer Vision*, pages 379–393, Freiburg, Germany, 1998.

[44] O. D. Faugeras. *Three-Dimensional Computer Vision*. MIT Press, Cambridge, MA, 1992.

[45] O.D. Faugeras, F. Lustman, and G. Toscani. Motion and structure from motion from point and line matches. In *Proc. Int. Conf. Computer Vision*, pages 25–34, 1987.

[46] C. Fermüller and Y. Aloimonos. Ambiguity in structure from motion: Sphere versus plane. *International Journal of Computer Vision*, 28(2):137–154, 1998.

[47] C. Fermüller and Y. Aloimonos. Observability of 3D motion. *International Journal of Computer Vision*, 37:43–63, 2000.

[48] P. Fua. Regularized bundle-adjustment to model heads from image sequences without calibration data. *International Journal of Computer Vision*, 38:153–171, 2000.

[49] A. Gershun. The light field. *Journal of Mathematics and Physics*, 12:51–151, 1939.

[50] C. Geyer and K. Daniilidis. Catadioptric projective geometry. *International Journal of Computer Vision*, 43:223–243, 2001.

[51] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. The lumigraph. In *Proceedings of ACM SIGGRAPH 96*, Computer Graphics (Annual Conference Series), pages 43–54, New York, 1996. ACM, ACM Press.

[52] M. D. Grossberg and S. K. Nayar. A general imaging model and a method for finding its parameters. In *Proc. International Conference on Computer Vision*, pages 108–115, 2001.

[53] K.J. Hanna and N.E. Okamoto. Combining stereo and motion analysis for direct estimation of scenestructure. In *Proc. International Conference on Computer Vision*, pages 357–365, 1993.

[54] R. Hartley and A. Zisserman. *Multiple View Geometry*. Cambridge University Press, 2000.

[55] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, UK, 2000.

[56] H. Haussecker and B. Jähne. A tensor approach for precise computation of dense displacement vector fields. In *DAGM Symposium*, pages 199–208, September 1997.

[57] G. Healey and R. Kondepudy. Radiometric ccd camera calibration and noise estimation. *IEEE Trans. PAMI*, 16(3):267–276, 1994.

[58] Andrew Hicks. arXiv preprint cs.CV/0303024.

[59] R. Andrew Hicks and Ronald K. Perline. Geometric distributions for catadioptric sensor design. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 584–589, 2001.

[60] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In *Proc. of ACM SIGGRAPH*, pages 295–302, 1994.

[61] B. K. P. Horn. *Robot Vision*. McGraw Hill, New York, 1986.

[62] B. K. P. Horn. *Robot Vision*. McGraw Hill, New York, 1986.

[63] Berthold K.P. Horn. Parallel networks for machine vision. Technical report, MIT A.I. Lab, 1988.

[64] J. Huang, A. Lee, and D. Mumford. Statistics of range images. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[65] A. Hubeli and M. Gross. A survey of surface representations for geometric modeling. Technical Report 335, ETH Zürich, Institute of Scientific Computing, March 2000.

[66] F. Huck, C. Fales, and Z. Rahman. *Visual Communication*. Kluwer, Boston, 1997.

[67] M. Irani. Multi-frame optical flow estimation using subspace constraints. In *Proc. International Conference on Computer Vision*, Corfu, Greece, 1999.

[68] B. Jähne, J. Haussecker, and P. Geissler, editors. *Handbook on Computer Vision and Applications*. Academic Press, Boston, 1999.

[69] H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan. A practical model for subsurface light transport. In *Proc. of ACM SIGGRAPH*, 2001.

[70] A. D. Jepson and D. J. Heeger. Subspace methods for recovering rigid motion II: Theory. Technical Report RBCV-TR-90-36, University of Toronto, 1990.

[71] H. Jin, S. Soatto, and A. J. Yezzi. Multi-view stereo beyond lambert. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 171–178, June 2003.

[72] J. T. Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150. ACM Press, 1986.

[73] I.A. Kakadiaris and D. Metaxas. Three-dimensional human body model acquisition from multiple views. *International Journal of Computer Vision*, 30:191–218, 1998.

[74] G. Kamberova. Understanding the systematic and random errors in video sensor data.

[75] L. Kobbelt, T. Bareuther, and H.-P. Seidel. Multiresolution shape deformations for meshes with dynamic vertex connectivity. In *Computer Graphics Forum 19 (2000), Eurographics '00 issue*, pages 249–260, 2000.

[76] C. Kolb, D. Mitchell, and P. Hanrahan. A realistic camera model for computer graphics. In *Proc. of ACM SIGGRAPH*, pages 317–324, 1995.

[77] J. Kosecka, Y. Ma, and S. Sastry. Optimization criteria, sensitivity and robustness of motion and structure estimation. In *Vision Algorithms Workshop, ICCV*, 1999.

[78] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38:199–218, 2000.

[79] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. PAMI*, 16:150–162, 1994.

[80] M. Levoy and P. Hanrahan. Light field rendering. In *Proceedings of ACM SIGGRAPH 96*, Computer Graphics (Annual Conference Series), pages 161–170, New York, 1996. ACM, ACM Press.

[81] Y. Liu and T. S. Huang. Estimation of rigid body motion using straight line correspondences. *CVGIP: Image Understanding*, 43(1):37–52, 1988.

[82] B. London, J. Upton, K. Kobre, and B. Brill. *Photography*. Prentice Hall Inc., Upper Saddle River, NJ, 7 edition, 2002.

[83] H. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[84] H.C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. *Proc. R. Soc. London B*, 208:385–397, 1980.

[85] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, 1987.

[86] Y. Ma, K. Huang, R. Vidal, J. Kosecka, and S. Sastry. Rank condition on the multiple view matrix. *International Journal of Computer Vision*, 59(2), 2004.

[87] Y. Ma, J. Kosecka, and S. Sastry. Optimization criteria and geometric algorithms for motion and structure estimation. *International Journal of Computer Vision*, 44(3):219–249, 2001.

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

[89] Y. Ma, R. Vidal, S. Hsu, and S. Sastry. Optimal motion from multiple views by normalized epipolar constraints. *Communications in Information and Systems*, 1(1), 2001.

[90] S. Malassiotis and M.G. Strintzis. Model-based joint motion and structure estimation from stereo images. *Computer Vision and Image Understanding*, 65:79–94, 1997.

[91] C. Mandal, H. Qin, and B.C. Vemuri. Physics-based shape modeling and shape recovery using multiresolution subdivision surfaces. In *Proc. of ACM SIGGRAPH*, 1999.

[92] S. Mann. Pencigraphy with acg: joint parameter estimation in both domain and range of functions in same orbit of projective-wyckoff group. Technical Report 384, MIT Media Lab, December 1994. also appears in: Proceedings of the IEEE International Conference on Image Processing (ICIP–96), Lausanne, Switzerland, September 16–19, 1996, pages 193–196.

[93] S. Mann and S. Haykin. The chirplet transform: Physical considerations. *IEEE Trans. Signal Processing*, 43:2745–2761, November 1995.

[94] W. Matusik, C. Buehler, S. J. Gortler, R. Raskar, and L. McMillan. Image based visual hulls. In *Proc. of ACM SIGGRAPH*, 2000.

[95] S. J. Maybank. The angular velocity associated with the optical flowfield arising from motion through a rigid environment. *Proc. R. Soc. London A*, 401:317–326, 1985.

[96] S. J. Maybank. Algorithm for analysing optical flow based on the least-squares method. *Image and Vision Computing*, 4:38–42, 1986.

[97] P. Meer. Robust techniques for computer vision. In G. Medioni and S. B. Kang, editors, *Emerging Topics in Computer Vision*. Prentice Hall, 2004.

[98] P. Moon and D.E. Spencer. *The Photic Field*. MIT Press, Cambridge, 1981.

[99] T. Naemura, T. Yoshida, and H. Harashima. 3-d computer graphics based on integral photography. *Optics Express*, 10:255–262, 2001.

[100] S. Nayar. Catadioptric omnidirectional camera. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 482–488, 1997.

[101] S. K. Nayar and V. Branzoi. Adaptive dynamic range imaging: Optical control of pixel exposures over space and time. In *Proc. International Conference on Computer Vision*, Nice, France, 2003.

[102] S. K. Nayar, V. Branzoi, and T. Boult. Programmable imaging using a digital micromirror array. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[103] J. Neumann and C. Fermüller. Plenoptic video geometry. *Visual Computer*, 19(6):395–404, 2003.

[104] J. Neumann, C. Fermüller, and Y. Aloimonos. Eyes from eyes: New cameras for structure from motion. In *IEEE Workshop on Omnidirectional Vision 2002*, pages 19–26, 2002.

[105] J. Neumann, C. Fermüller, and Y. Aloimonos. Polydioptric camera design and 3d motion estimation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume II, pages 294–301, 2003.

[106] J. Neumann, C. Fermüller, Y. Aloimonos, and V. Brajovic. Compound eye sensor for 3d ego motion estimation. In *accepted for presentation at the IEEE International Conference on Robotics and Automation*, 2004.

[107] B. Newhall. Photosculpture. *Image*, 7(5):100–105, 1958.

[108] F.E. Nicodemus, J.C. Richmond, J.J. Hsia, L.W. Ginsberg, and T. Limperis. *Geometric Considerations and Nomenclature for Reflectance*. National Bureau of Standards (US), 1977.

[109] Editors of Time Life, editor. *Light and Film*. Time Inc., 1970.

[110] T. Okoshi. *Three-dimensional Imaging Techniques*. Academic Press, 1976.

[111] J. Oliensis. The error surface for structure from motion. Neci tr, NEC, 2001.

[112] T. Pajdla. Stereo with oblique cameras. *International Journal of Computer Vision*, 47(1/2/3):161–170, 2002.

[113] S. Peleg, B. Rousso, A. Rav-Acha, and A. Zomet. Mosaicing on adaptive manifolds. *IEEE Trans. on PAMI*, pages 1144–1154, October 2000.

[114] M. Peternell and H. Pottmann. Interpolating functions on lines in 3-space. In Ch. Rabut A. Cohen and L.L. Schumaker, editors, *Curve and Surface Fitting: Saint Malo 1999*, pages 351–358. Vanderbilt Univ. Press, Nashville, TN, 2000.

[115] R. Plänkers and P. Fua. Tracking and modeling people in video sequences. *International Journal of Computer Vision*, 81:285–302, 2001.

[116] R. Pless. Using many cameras as one. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 587–593, 2003.

[117] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *ICCV*, pages 90–95, 1998.

[118] H. Pottmann and J. Wallner. *Computational Line Geometry*. Springer Verlag, Berlin, 2001.

[119] K. Prazdny. Egomotion and relative depth map from optical flow. *Biological Cybernetics*, 36:87–102, 1980.

[120] P. Rademacher and G. Bishop. Multiple-center-of-projection images. In *Proceedings of ACM SIGGRAPH 98*, Computer Graphics (Annual Conference Series), pages 199–206, New York, NY, 1998. ACM, ACM Press.

[121] W. Reichardt. Movement perception in insects. In Werner Reichardt, editor, *Processing of Optical Information by Organisms and Machines*, pages 465–493. Academic Press, 1969.

[122] W. Reichardt. Evaluation of optical motion information by movement detectors. *Journal of Comparative Physiology A*, 161:533–547, 1987.

[123] W. Reichardt and R. W. Schlögl. A two dimensional field theory for motion computation. *Biological Cybernetics*, 60:23–35, 1988.

[124] J. P. Richter, editor. *The Notebooks of Leonardo da Vinci*, volume 1, p.39. Dover, New York, 1970.

[125] M Rioux. Laser range finder based on synchronized scanners. *Applied Optics*, 23(21):3837–3844, 1984.

[126] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1/2/3):17–42, 2002.

[127] Yoav Y. Schechner and Nahum Kiryati. Depth from defocus vs. stereo: How different really are they? *International Journal of Computer Vision*, 89:141–162, 2000.

[128] P. Schröder and D. Zorin. Subdivision for modeling and animation. Siggraph 2000 Course Notes, 2000.

[129] S. Seitz. The space of all stereo images. In *Proc. International Conference on Computer Vision*, pages 307–314, 2001.

[130] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 25, November 1999.

[131] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 593 – 600, 1994.

[132] H.Y. Shum, A. Kalai, and S. M. Seitz. Omnivergent stereo. In *Proc. International Conference on Computer Vision*, pages 22–29, 1999.

[133] D. Snow, P. Viola, and R. Zabih. Exact voxel occupancy with graph cuts. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, 2000.

[134] M. E. Spetsakis and Y. Aloimonos. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4:171–183, 1990.

[135] M. E. Spetsakis and Y. Aloimonos. A multi-frame approach to visual motion perception. *International Journal of Computer Vision*, 6(3):245–255, 1991.

[136] H. Spies, B. Jähne, and J.L. Barron. Regularised range flow. In *Proc. European Conference on Computer Vision*, June 2000.

[137] J. Stam. Evaluation of loop subdivision surfaces. SIGGRAPH'99 Course Notes, 1999.

[138] G.P. Stein and A. Shashua. Model-based brightness constraints: on direct estimation of structure and motion. *IEEE Trans. PAMI*, 22(9):992–1015, 2000.

[139] G.W. Stewart. Stochastic perturbation theory. *SIAM Review*, 32:576–610, 1990.

[140] C. Strecha and L. Van Gool. Motion-stereo integration for depth estimation. In *ECCV*, volume 2, pages 170–185, 2002.

[141] R. Swaminathan, M. D. Grossberg, and S. K. Nayar. Framework for designing catadioptric imaging and projection systems. In *International Workshop on Projector-Camera Systems, ICCV 2003*, 2003.

[142] J. Tanner and C. Mead. An integrated analog optical motion sensor. In R.W. Brodersen and H.S. Moscovitz, editors, *VLSI Signal Processing*, volume 2, pages 59–87. IEEE, New York, 1988.

[143] G. Taubin. A signal processing approach to fair surface design. In *Proc. of ACM SIGGRAPH*, 1995.

[144] P. Thévenaz, T. Blu, and M. Unser. Interpolation revisited. *IEEE Transactions on Medical Imaging*, 19(7):739–758, July 2000.

[145] T. Tian, C. Tomasi, and D. Heeger. Comparison of approaches to egomotion computation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 315–320, June 1996.

[146] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[147] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment — a modern synthesis. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 298–373, Corfu, Greece, September 1999. Springer-Verlag.

[148] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Trans. PAMI*, 6(1):13–27, 1984.

[149] Y. Tsin, V. Ramesh, and T. Kanade. Statistical calibration of ccd imaging process. In *Proc. International Conference on Computer Vision*, volume 1, pages 480–487, 2001.

[150] M. Unser and A. Aldroubi. A general sampling theory for nonideal acquisition devices. *IEEE Transactions on Signal Processing*, 42(11):2915–2925, November 1994.

[151] D. Van De Ville, T. Blu, and M. Unser. Recursive filtering for splines on hexagonal lattices. In *Proceedings of the Twenty-Eighth IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03)*, volume III, pages 301–304, 2003.

[152] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *Proc. International Conference on Computer Vision*, Corfu,Greece, September 1999.

[153] S. Vedula, S. Baker, S. Seitz, and T. Kanade. Shape and motion carving in 6d. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Head Island, South Carolina, USA, June 2000.

[154] S. Vedula, P. Rander, H. Saito, and T. Kanade. Modeling, combining, and rendering dynamic real-world events from image sequences. In *Proc. of Int. Conf. on Virtual Systems and Multimedia*, November 1998.

[155] R. Vidal and S. Sastry. Segmentation of dynamic scenes from image intensities. In *IEEE Workshop on Vision and Motion Computing*, pages 44–49, 2002.

[156] G. Ward. Measuring and modeling anisotropic reflection. In *Proc. of ACM SIGGRAPH*, volume 26, pages 265–272, 1992.

[157] J. Weng, T.S. Huang, and N. Ahuja. *Motion and Structure from Image Sequences*. Springer-Verlag, 1992.

[158] B. Wilburn, M. Smulski, H.-K. Lee, and M. Horowitz. The light field video camera. In *Proceedings of Media Processors*. SPIE Electronic Imaging, 2002.

[159] D. N. Wood, A. Finkelstein, J. F. Hughes, C. E. Thayer, and D. H. Salesin. Multiperspective panoramas for cel animation. *Proc. of ACM SIGGRAPH*, pages 243–250, 1997.

[160] G. Young and R. Chellappa. 3-d motion estimation using a sequence of noisy stereo images: Models, estimation, and uniqueness results. *IEEE Trans. PAMI*, 12(8):735–759, 1990.

[161] Jingyi Yu and Leonard McMillan. General linear cameras. In *8th European Conference on Computer Vision, ECCV 2004*, Prague, Czech Republic, 2004.

[162] C. Zhang and T. Chen. Spectral analysis for sampling image-based rendering data. *IEEE Trans. on Circuits and Systems for Video Technology: Special Issue on Image-based Modeling, Rendering and Animation*, 13:1038–1050, Nov 2003.

[163] C. Zhang and T. Chen. A survey on image-based rendering - representation, sampling and compression. *EURASIP Signal Processing: Image Communication*, 19(1):1–28, 2004.

[164] L. Zhang, B. Curless, and S. M. Seitz. Spacetime stereo: Shape recovery for dynamic scenes. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 367–374, 2003.

[165] Y. Zhang and C. Kambhamettu. Integrated 3d scene flow and structure recovery from multiview image sequences. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages II:674–681, Hilton Head, 2000.

[166] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *International Journal of Image and Vision Computing*, 15(1):59–76, 1997.

[167] W. Zhao, R. Chellappa, J. Phillips, and A. Rosenfeld. Face recognition in still and video images: A literature surrey. *ACM Computing Surveys*, pages 399–458, 2003.

[168] A. Zomet, D. Feldman, S. Peleg, and D. Weinshall. Mosaicing new views: The crossed-slits projection. *IEEE Trans. PAMI*, pages 741–754, 2003.

[169] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. In *Proc. of ACM SIGGRAPH*, 1997.