UNIVERSITY OF CALIFORNIA SAN DIEGO

Epipolar Constrained Multi-camera Arrays for Embedded 3D Vision

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Computer Science (Computer Engineering)

by

Dominique Ernest Meyer

Committee in charge:

Professor Falko Kuester, Chair
Professor Henrik Christensen, Co-Chair
Professor Neal Driscoll
Professor Ryan Kastner
Professor Hao Su

2021

The dissertation of Dominique Ernest Meyer is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

DEDICATION

This dissertation is dedicated to my parents, Hans-Rudolf Meyer and Christine Meyer-Merian for their infinite love and support.

EPIGRAPH

Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough.

—Richard P. Feynman

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

My dear parents, **Christine and Hans-Rudolf Meyer**. Throughout my whole education you were behind me, supporting me and accepting the career track I have come to follow. It is certain that without your continued words of encouragement and dedication to keep me disciplined that I would not have come this far. I owe you my greatest gratitude, forever.

To my love, **Danielle Mercure**, you have been by my side through times of success and failure. Sharing my weight has enabled me to strive. Thank you for always believing in me.

To my research colleagues, and great friends, **Dr. Tom Wypych and Dr. James Strawson**. If it wasnt for you, I would not be writing this right now. You have enlightened me through graduate school, taught me engineering, backed me up in life, and have been the hidden mentors everyone could dream of. My heartfelt thank you.

To my advisor, **Prof. Falko Kuester**. Through my time at UC San Diego, you have had my back, have opened doors, and have guided me to who I am today. It is admirable that you have given me endless privileges in your research lab as a student, and I want to thank you for having trusted me and always believed in me.

To my advisor, **Prof. Henrik Christensen**. Since sitting in your first robotics class, I owe you my appreciation for merging hardware and software into systems, and the unavoidable numerical methods that enabled me to do so. You have always been supportive and guided me through decomposing my convoluted thoughts into simple and meaningful terms, thank you.

To my advisor, **Prof. Ryan Kastner**. Few students have travelled through jungles together and skied down mountains with their professor, all while being taught one of the most fascinating computing fields of modern computers- embedded systems. It is thanks to you that I have discovered and come to love the process of taking your average software, and making it work on the smallest devices that will be used throughout the world. You have always made time for me, and have given me the benefit of the doubt to enter the field of computer science- thank you!

To **Eric Lo**. We have shared expeditions, travel, resources and a passion of everything

UAV and camera related. You have tolerated my years of incompetence in engineering, but have trusted me to get certain things done. I look up to you for your hard work and willingness to dive into as many fascinating and exciting research things you can come across. Keep it up!

To the whole **Dronelab/CHEI Team**. everyone has been an absolute pleasure to work with. A special few words of appreciation and encouragement to **Pranav Verma, Jonathan Klingspon, Akhil Birlangi, Danylo Dorhobytsky and Joseph Philips**. You have all shown what young, passionate, driven students are capable of achieving. Also a big thank you everyone who welcomed me into the group: **Dr. Michael Hess, Vid Petrovic, Christopher McFarland, Aliya Hoff and the many others**.

To my colleague and advisor **Dr. Dominique Rissolo**. You have provided me with the opportunity to pair archaeology and engineering- a rare blend of fields that has made my path fun. Over the many years, you gave me the delightful opportunity to explore some of the hidden wonders of the world, all while innovating on technology that is revolutionizing the digital age of archaeology. Thank you for always having my back, advocating for me and driving a passion in everyone's work. Ni'bo'olal!

To **Prof. Benjamin Ochoa** and **Dr. Harlyn Baker**. Computer vision is a big field, but grown from a small group of experts. I would not have found my appreciation for it if it wasn't for your teaching, fascinating discussions and endless expertise on it. Thank you for answering my endless questions, and for sharing many tricks of the trade.

To the whole **Calit2 Team- Dr. Tom DeFanti, Joseph Keefe, Jennifer Hollis, Crystal Liu, Frank Cardone, Alex Grant, Joel Polizzi, Yuki Marsden and the many others**- this place has become my home for many years, and it is undoubtedly the rich environment of intellectual diversity that has given fruit to the many unique research successes. I thank everyone who has made my time here memorable.

Finally, to everyone else, family, friends and colleagues that have spoken words of encouragement. The combined support is what has made this possible.

Chapter 1, 2 and 3, in part, are a reprint of the material as it appears, titled "A Survey of Dense 3D Reconstruction Methods for Multi-Sensor Embedded Vision" in UC San Diego Research Exam 2019. Meyer, Dominique. The dissertation author was the primary investigator and author of this material.

Chapter 3, in part, is a reprint of the material as it appears, titled "StarCAM - A 16K stereo panoramic video camera with a novel parallel interleaved arrangement of sensors" in Electronic Imaging 2019. Meyer, Dominique Ernest; Lo, Eric; McFarland, Chris; Dawe, Gregory; Dai, Ji; Nguyen, Truong; DeFanti, Thomas; Wang, Haoyu; Sandin, Dan; Brown, Maxine; Kuester, Falko. The dissertation author was the primary investigator and author of this material.

Chapter 3, in part, is a reprint of the material as it appears, titled "Omniscopic Vision for Robotic Control" in IEEE Aerospace 2019. Meyer, Dominique Ernest; Lo, Eric; McFarland, Chris; Strawson, James; Drohobytsky, Danylo; Dawe, Gregory; Dai, Ji; Nguyen, Truong; DeFanti, Thomas; Wang, Haoyu; Sandin, Dan; Brown, Maxine; Kuester, Falko. The dissertation author was the primary researcher and author of this material.

Chapter 3, in part, is currently being prepared for submission for publication of the material, titled "3D Multi-Camera Calibration with a Fractal Encoded Multi-Shape Target" in Electronic Imaging 2021. Meyer, Dominique Ernest; Verma, Pranav; Kuester, Falko. The dissertation author was the primary researcher and author of this material.

Chapter 4 and 7, are currently being prepared for submission for publication of the material, titled "Stereo Panoramic Sceneflow: A Framework for Sensing and Perception of Dynamic Environments " as it may appear in IEEE Transactions on Pattern Analysis and Machine Intelligence 2021. Meyer, Dominique Ernest; Verma, Pranav; Niradi, Shreyas; Christensen, Henrik; Kuester, Falko. The dissertation author is the primary researcher and author of this material.

Chapter 5, in part, is a reprint of the material as it appears, titled "Splatty- A Unified Image De-Mosaicing and Rectification Method" as it may appear in Winter Applications of

Computer Vision (WACV) 2021. Verma, Pranav; Meyer, Dominique Ernest; Xu, Hanyan; Kuester, Falko. The dissertation author proposed the algorithmic approach to the work and oversaw the implementation in software and hardware.

Chapter 6, in part, is currently being prepared for submission for publication of the material, titled "DevCAM: An Open-Source Multi-Camera Development System for Embedded Vision" in Electronic Imaging 2021. Meyer, Dominique Ernest; Birlangi, Meher Akhil; Kuester, Falko. The dissertation author was the primary researcher and author of this material.

VITA

2017	B. S. in Physics, University of California San Diego
2019	M. S. in Computer Science (Computer Engineering), University of California San Diego
2021	Ph. D. in Computer Science (Computer Engineering), University of California San Diego

ABSTRACT OF THE DISSERTATION

Epipolar Constrained Multi-camera Arrays for Embedded 3D Vision

by

Dominique Ernest Meyer

Doctor of Philosophy in Computer Science (Computer Engineering)

University of California San Diego, 2021

Professor Falko Kuester, Chair
Professor Henrik Christensen, Co-Chair

High-resolution and real-time 3D sensing from cameras is a long standing challenge in the robotics and computer vision community. This thesis proposes a system wide optimization strategy for embedded vision systems by providing a software-hardware design approach that jointly improves multi-camera arrays and reconstruction algorithms for improved 3D reconstruction.

The proposed approach builds on state of the art research efforts of multi-camera systems and multi-view methods to simultaneously estimate system pose and 3D scene. Existing works are reviewed and summarized to provide a theoretical baseline of the field. To improve the robustness and efficiency of the estimation, geometric and epipolar principles of cameras are

leveraged to formulate configuration decisions, on which the proposed algorithmic methods are based. These configuration implications are leveraged to formulate highly parallelizable and scalable reconstruction methods. The first goal of the estimation methods involves sparse 3D solutions for localization and map initialization that in turn serves as a prior for the dense reconstruction. A set of semi-dense and dense reconstructions algorithms are introduced that leverage the sensory configuration and ego-motion estimation to improve on state of the art stereo and partial light-field depth estimation methods, both in terms of estimation performance, and computational efficiency.

Real-world evaluation of the proposed system design is accomplished through the development of an open-source multi-camera sensing system, DevCAM. A heterogeneous FPGA, DSP, CPU and GPU architecture is used as the basis for an experimentation ground that accommodates 18 high resolution cameras in a trinocular panoramic configuration, an Inertial Nagivation System with differential GPS and high-performance networking capabilities. On this system, hardware optimized pre-processing is demonstrated, alongside early results for 3D mapping.

Chapter 1

Introduction

1.1 Motivation

3D computer vision involves the inverse estimation of scene geometry from a set of 2D images. This core task is the foundation of two of the most rapidly growing research fields in modern computer vision: Photogrammetry and Simultaneous Localization and Mapping (SLAM). The applications for these techniques are widespread: robotics, autonomous vehicles, 3D mapping, virtual and augmented reality, all of which revolve around a common theme of understanding where a camera or set of cameras are located in the environment, and what the scene geometry is which they observe. Many of these applications require this estimation of ego-motion and scene reconstruction to be completed on a mobile device, and in real-time. This dissertation examines the system design approach to localize and estimate a 3D environment on devices that are limited in size, power and computational resources, more broadly known as embedded systems.

The scope of this thesis spans the full-stack system development: from low-level algorithmic research of computer vision methods, through physical array designs, hardware architecture and embedded system development. What may appear more like a set of tasks for different engineering teams in an established company, is actually the result of a targeted system develop-

ment philosophy. Computational development has up to now often seen a divided approach to bringing algorithms to hardware: begin by building great algorithms that prioritize quality and accelerate them on specific hardware; or build hardware architectures that tackle one or a set of algorithms. Much of this work idealizes a joint development approach- lets build a system that simultaneously leverages physical constraints to facilitate both the hardware and software development, to together improve the system performance, capabilities and reduce engineering difficulty.

The development of embedded 3D multi-view cameras is motivated by the need of dense, accurate and scalable scene reconstructions in real-time, results that are unmet between photogrammetry and SLAM. Photogrammetry leads the geometric accuracy and model density at the cost of computational time, and SLAM often produces real-time results, but far from the qualitative results of dense photogrammetric reconstruction as portrayed in Figure 1.1. A set of applications which will be discussed in the introduction and throughout this thesis motivate the need ts gap, providing a solution that achieves processing of higher resolution data, on embedded systems to yield denser and more accurate representations of the world.

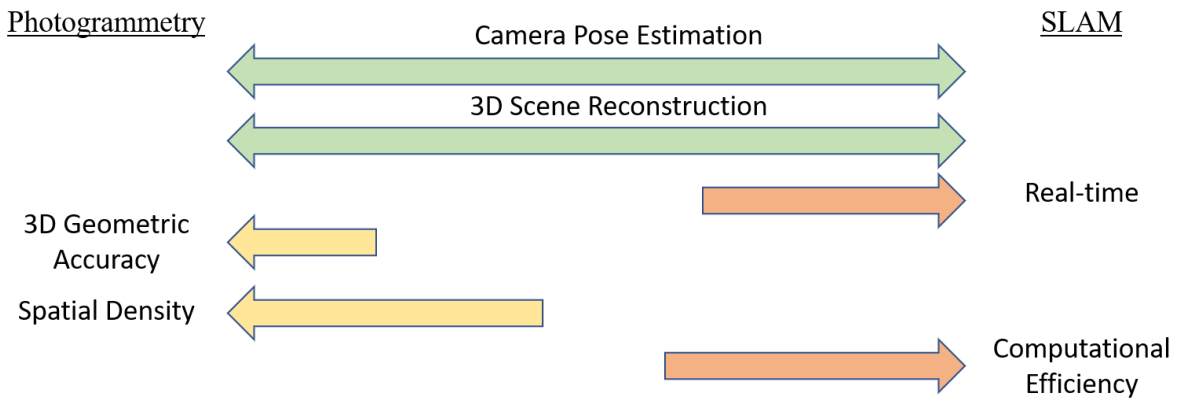

Figure 1.1: Comparison of Photogrammetry and SLAM Advantages

1.2 Problem Formulation

A camera consists of a sensor-lens pair, which when combined with a set of processors can be considered an embedded vision systems [WBI+19]. Commonly the sensors are accompanied with a set of Digital Signal Processors (DSPs), which in the imaging field are known as Image Signal Processors (ISPs) [DD19]. Higher-level processors such as Central Processing Units (CPUs) and Graphics Processing Units (GPUs) accompany camera systems for a variety of tasks such as additional processing, interfacing and system control [NJ19]. When images are acquired by a set of sensors S_k, we receive discreet 2D matrices of pixels $I(x,y)_t$ representing the accumulated light flux over an exposure time at time t. These pixel values can be referred to as 2D projections of light rays reflected from the points in the 3D world [GW02]. For all image points x_i there exists a 3D world point X_i which lies on the line in space though that point in the image plane and the camera centre C_k [HZ03]. From a single viewpoint, it is therefore not possible to determine the exact location of the world point X_i, but it is possible to constrain the world point to being somewhere on a line defined by the image point, the camera center and the camera projection matrix P. When two or more cameras see the same world point, it is possible to recover the location, assuming that the lines defined by each image point and camera centers converge to a point in 3D space. More commonly, we aim to find the point in 3D space which minimizes the overall re-projection error of the estimated point for all cameras [HZ03].

A combination of optical and analog sensor performance affects the Signal to Noise Ratio (SNR) as well as the efficiency by which light rays are converted into optical signals. These factors all contribute to different types of image noise which is extensively described in [VA13] which factor into mis-estimations of image point locations affecting 3D reconstruction accuracy [HJ11]. Recent sensor developments have enable for various image processing tasks to take place directly in the silicon of the sensor instead of requiring them to be computed with other resources. This is normally for pre-processing tasks such as down-sampling, filtering and rectification.

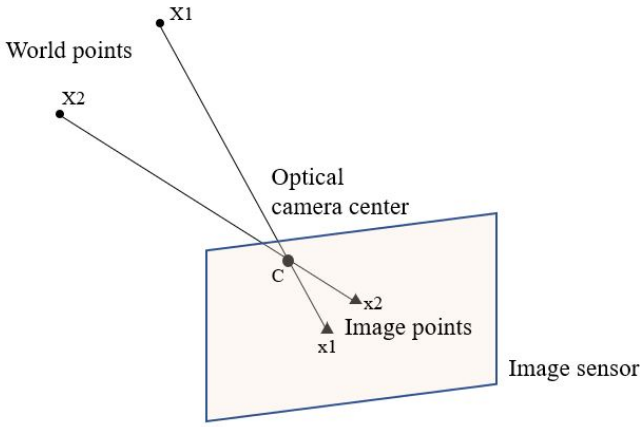

Figure 1.2: Single camera geometry assuming a Pinhole camera model

In multi-sensor systems, cross-view correlations are required to compute scene geometry based on projective geometry. Cognitive matching of areas across multiple viewpoints is commonly formulated as a statistical selection problem [Hir05]. We aim to find a point to point match across images that agrees in a local neighborhood as well as globally across the image. Sparse reconstructions only address points of particular uniqueness while dense methods commonly evaluate points for every pixel [Jes09]. In this work, we will discuss both sparse, semi-dense and dense methods for scene reconstruction. Sparse approaches will mainly be discussed as a means to estimating relative poses and motion of camera systems, as well as scene initialization. Semi-dense approaches highlight features in density that vary across the image, often coming down to contours The computational cost of evaluating every pixel with every other pixel in a set of images scales quadratically with the number of pixels and has a general complexity of $O(N^2)$. This has fostered the development of 1) algorithms that approach linear complexity with number of pixels $O(N)$ and 2) hardware acceleration methods that enable the processing to be highly parallelized [SHW+14].

In embedded camera systems, the system requirements are driven by the application and as such it is possible to optimize for specific performance characteristics. While an optimization

in all objectives is often not feasible, we formulate a generalized goal for embedded 3D vision systems as a minimization problem of the following properties.

- Scene reconstruction error- the Sum of Absolute Differences (SAD) or Sum of Squared Differences (SSD) (Equation 1.1) of estimated world points $\hat{X}$ to world points X. This is a measure of correctness of geometric estimation of world points. When considering a vision system, we typically consider errors in both image and world points and as such minimize a weighted sum of both errors as described in Equation 1.2 from [HZ03]. The d_{Mah} represents the Mahalanobis distance with respect to the error covariance matrices for the measurements x_i and X_i.

$$SSD = \sum_i d(X_i, \hat{X}_i)^2 \tag{1.1}$$

$$\min_{x_i, \hat{X}_i, P} \sum_i d_{Mah}(x_i, P\hat{X}_i)^2 + d_{Mah}(X_i, \hat{X}_i)^2 \tag{1.2}$$

- Reconstruction latency- the duration between when the frame set from the sensors is captured and when the computed depthmap or world points are estimated.

- Reconstruction throughput- how many world points that are estimated per second. In the case of depthmap data, this value is in pixel disparity estimates per second.

- Hardware processing- the power efficiency, cost and size of the compute modules accompanying the optical sensors that estimate scene geometry from raw input data. These more generally also apply to memory requirements, I/O performance specifications and digital operator performance.

- Engineering development time- while not a system specific objective, being able to rapidly iterate

System design requires a selection of design parameters which in the case of passive vision systems is the hardware configuration, reconstruction algorithm, and choice of hardware acceleration units for each part of the algorithms. Formulating an objective for each design choice allows for improved overall performance. This section will introduce the problems related to the individual parts of the system design choices with common objectives.

The first part of every 3D reconstruction pipeline in physical systems consist of a set of pre-processing tasks that prepare a raw image for image point matching and triangulation across multiple sensors. These pre-processing tasks are critical to multi-view reconstruction performance due to their direct effect on image point accuracy, matching performance and reconstruction robustness [ACM04]. Debayering algorithms estimate a per pixel RGB value from a sensor where individual pixels are filtered to different light wavelengths. Due to the interpolating nature of the algorithms over 3x3 and 5x5 windows, this acts as a low-pass filter [LGS+14] adding noise to the image point locations and as such reducing reconstruction accuracy. Further processes such as flat-field corrections, de-noising and rectification all affect pixel intensities $I(x,y)$ and need to be chosen and built to minimize reconstruction error. While these techniques affect geometric accuracy, they to a large extent also affect hardware acceleration requirements and performance. Depending on the algorithms and operation precision it is possible to impact overall throughput, latency and drive hardware specification such as on-chip memory and operations per second [QMS+19].

To reconstruct a scene there needs to be knowledge of the camera projection matrix P which maps a world point to an image point in a camera. This is commonly assumed to be known as a calibrated prior in the system. The camera matrix P is composed of intrinsic K and extrinsic matrices $[R|t]$ which describe the internal distortion and camera pose respectively $P = K \times [R|t]$. During the reconstruction, we want to choose the P matrix which minimizes the geometric error as in equation 1.3 as shown in [HZ03]. The calibration of the system should therefore the geometric accuracy of the system when completed in advance of system use, or during auto-calibration and

dynamic re-calibration.

$$\min_{P} \sum_{i} d(x_i, PX_i)^2 \tag{1.3}$$

The work [HZ03] summarizes the method of triangulating two images points to one world point. We can intuitively break down the problem of improving the world point estimate from two or more views to improving the image point location estimate in each view, and choosing the correct match between the views. Image point estimates, whether in sparse or dense methods are therefore targeted at being solved to sub-pixel coordinates in an image $I(x_i, y_i)$ which is the most accurate point projection of the world point X_i. In the case of Epipolar imaging such as in the work by [BBM87, BB89], this extends to sub-pixel accurate 2D line definitions that describe the Laplacian zero-crossings. The second problem of matching between views assumes a binary result- it is either a correct or incorrect image point match. The point matches are determined from heuristic decision of finding the most likely match from a set of potential image points. Using more sophisticated feature descriptor improves matching performance but also impacts computation time and complexity negatively [Bau00]. Every incorrect match which is not rejected by an outlier rejection method, negatively contributes to the reconstruction accuracy. It is therefore important that the algorithmic choice of image point identification and matching methods be carefully choices to align with the overall system goals of reducing geometric error and computational requirements.

1.2.1 Hardware Acceleration Design

A majority of embedded computer vision pipelines leverage functional decomposition to distribute computational tasks across a set of processing resources. The most common set of available processing architectures are grouped into general-purpose architectures (CPU, GPU), dedicated architectures (ASIC, DSP) and configurable architectures (FPGA) [ZSC+17]. Figure

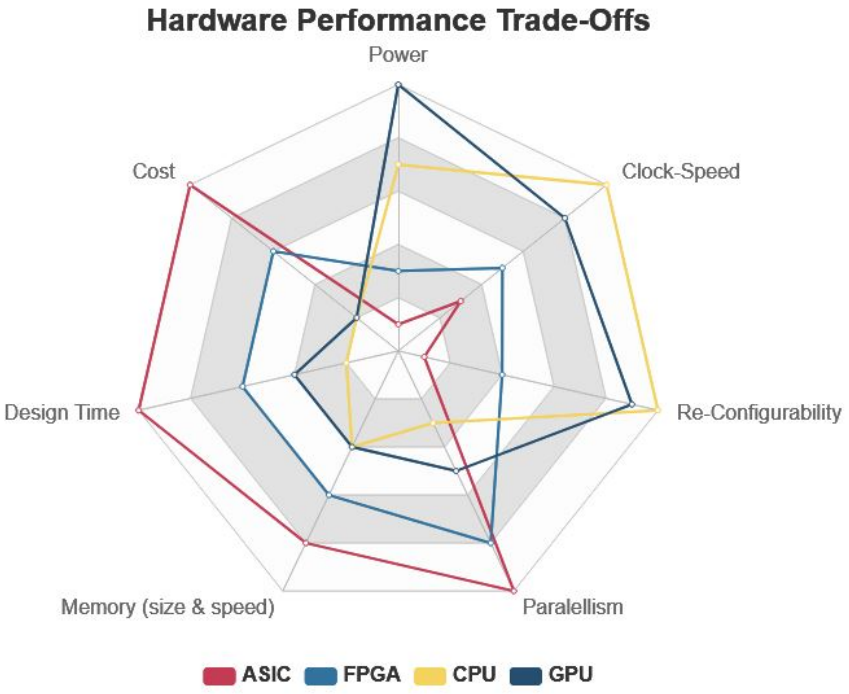

Figure 1.3: Hardware design trade-offs for ASIC, FPGA, CPU and GPU architectures based on evaluations from [ZSC+17]

1.3 provides an overview of the hardware architecture trade-offs from a set of commonly available embedded architectures based on summaries provided by [ZSC+17]. A key factor which plays into every architecture is the ability for data streaming to be buffered into memory elements and the input/output capabilities. These frequently bottleneck system throughput and are mostly architecture manufacturer dependent. We can generalize the goal of the hardware-software co-design choices as the design space optimization $\mathcal{D} = \mathcal{H} \times \mathcal{A} \times I \times \mathcal{P}$ where $\mathcal{H}, \mathcal{A}, I, \mathcal{P}$ are hardware, algorithmic, parameter implementation choices, to achieve minimal power consumption, smallest form-factor, lowest latency, and highest throughput.

1.3 Hypothesis

Multi-directional epipolar geometry can be used to improve embedded multi-camera scene reconstruction accuracy and efficiency.

This thesis will focus on the joint formulation of the design of an embedded depth estimation system from acquisition platform through reconstruction algorithms and implementation implications. To validate the hypothesis, quantitative and qualitative comparisons will be done on a novel multi-directional sensor configuration. The hypothesis will be tested in the following ways:

1. Comparing the reconstruction accuracy of a trinocular camera system and accompanying algorithm to that of traditional stereo reconstructions. If the estimated depth values are statistically more accurate across observations in a scene, the reconstruction improvement is deemed validated.

2. Comparing the pose estimation of a trinocular panoramic system to a stereo panoramic system. If the pose error is less for the trinocular system, the scene reconstruction accuracy improvement is also deemed validated.

3. Finally, the proposed geometric camera configuration and it's associated semi-dense implementation are compared in efficiency through quantitative spatial resolution per unit time. If higher resolutions or throughputs are demonstrated, the efficiency is validated to have been improved.

The scientific impact of validating this hypothesis will positively impact the capabilities of mobile multi-camera systems in their use of estimating 3D scenes. The improved reconstruction accuracy will enable systems to achieve greater ranging distances for a given baseline and will improve the spatial resolution of panoramic systems. Added processing efficiency will reduce both power and computational resource requirements, making the adoption of such systems as

imaging and sensing modules easier. These key enabling points would greatly impact how passive vision sensors fit into the larger 3D sensing ecosystem, capturing more sensing value across more application domains.

1.4 Applications in Autonomous Vehicles

Figure 1.4: StarCAM Camera Array on the UCSD AVL Autonomous Vehicle Platform

One application domain that is a prime example highlighting the need of scalable and robust embedded multi-camera systems are autonomous vehicles. The industry at large has faced a long standing development challenge of 3D sensors with the necessary ranging, resolution and cost to meet the requirements of widespread adoption in consumer vehicles, rideshare and delivery systems. Current state of the art vehicles capable of L4 autonomy commonly employ a suite of sensors including LiDAR, Cameras, Radar and Inertial Navigation Systems (INS) to combine strengths of the different sensing modalities, while also serving as information redundancy. This combination of sensors sums to costs, an order of magnitude too high for commercialization, driving the need for systems that are capable of similar sensing and perception at drastically more favorable pricing. Due to the affordability and availability of CMOS sensors (largely driven by mobile telecommunication market), leveraging arrays of these enables us to create sensing systems favorable in capabilities to high end LiDARs but at a fraction of the cost. Table 1.1 compares the technical sensing capabilities of LiDAR sensors and a research camera array that

Table 1.1: A comparison of LiDAR and a Proposed Multi-Camera System for 3D sensing

	Velodyne-LIDARs				DevCAM
	HDL-32	HDL-64E	Puck	Alpha Puck	Trinocular Panoramic
Range (m)	100	120	100	300	80
Range Accuracy (cm)	2	2	3	3	5-200
Horizontal FOV (deg)	360	360	360	360	360
Vertical FOV (deg)	41.3	26.9	30	40	86
Refresh Rate (Hz)	5-20	5-20	5-20	5-20	20
Points per Second (Millions)	0.695	1.3	0.3	2.4	1990
Spatial Resolution (points/m^2@100m)	2.94	8.65	1.78	10.49	914.29
Power Consumption (W)	12	60	8	30	60
Output Speed (Gb/s)	0.1	0.1	0.1	1	3
Price ($)	$70,000	$85,000	$7,999	$100,000+	$20,000

will be introduced in Chapter 6 and is depicted in Figure 1.4.

One of the technical challenges faced in existing Autonomous Vehicle systems is the robust discernment and tracking of 3D objects in a scene, at distances sufficient to meet stopping distances and guarantee safe navigation. Assuming a vehicle operates in an urban environments at speeds of 100 km/h with an overall system reaction time of 1 second, it's detection distance for pedestrians, bikers and road construction needs to exceed 100m to reach a full stop. While these numbers are solely to convey a general sense of requirements, we can now consider the sensor data requirements. A Velodyne HDL-32 LiDAR would only return a handful of points per square meter at 100m, a vision system can easily return in the hundreds to thousands of spatial samples for the same spatial swath. They key to note is that due to the density in sensory samples, we are able to greatly improve our spatial detection and tracking of objects by leveraging image sensors. In Chapter 5.1 a review of ranging accuracy will be discussed. While LiDAR sensors have superior absolute and constant ranging accuracies, multi-camera arrays will be shown to demonstrate reasonable depth accuracies despite the inverse relationship of the accuracy to the distance of the detection.

1.5 Applications in Cultural Heritage

Another application field of 3D sensors is Cultural Heritage documentation. Across the world, historic sites are subject to environmental and man-incurred degradation, and it is a matter of time before certain history will forever be lost. Cultural heritage digital documentation involves the detailed mapping of such sites using non-destructive methods, often in 3D. Different archaeological and historical context is required for different sites, changing the demand of spatial resolution and geometric accuracy. Figure 1.8 illustrates a high level workflow for data acquisition, processing and analysis across various landscape scale, building scale and artifact scale documentations.

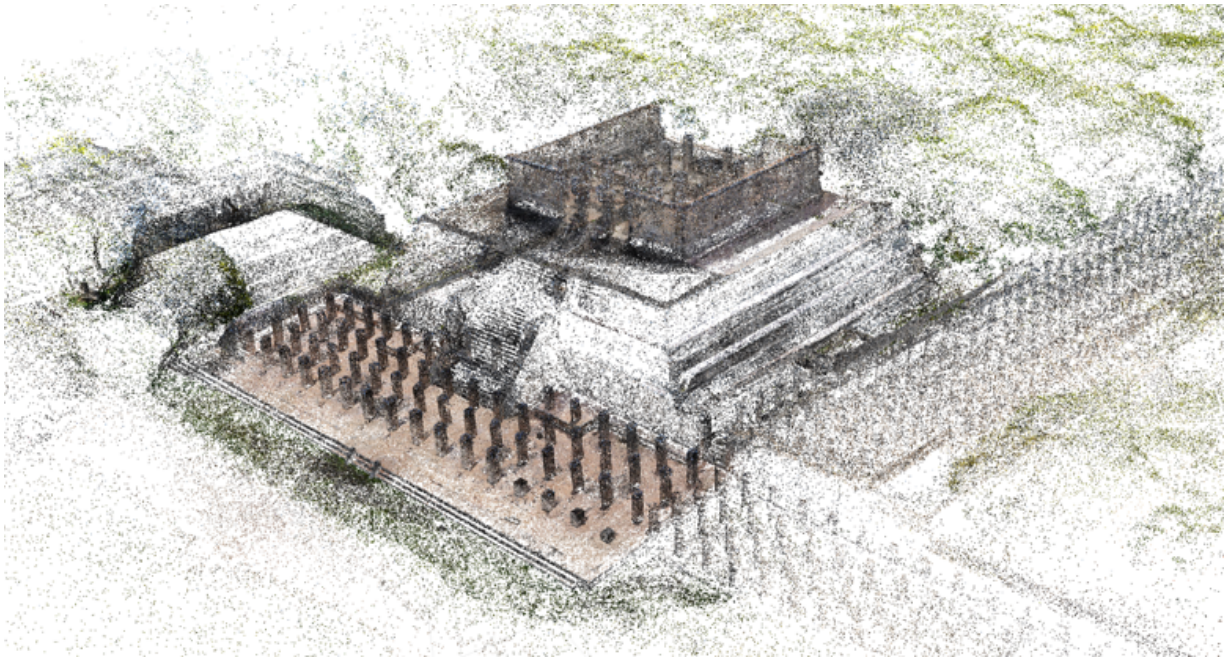

Figure 1.5: A sparse photogrammetric 3D reconstruction of the Templo de Los Guerreros at Chichen Itza using ground and aerial photography.

While various sensors provide different modalities across the geometric and photometric (light color and intensity) reconstructions, one of the key challenges throughout is to complete data acquisition in a thorough and efficient manner at resolutions satisfactory to long-term analysis needs. Certain remote sites may only allow for a few hours of data acquisition because of weather

conditions, and others in war zones are subject to days before being permanently destroyed. Rapid data capture with real-time feedback and coverage is therefore desired. Let us consider a case study of an iconic Maya site, Chichen Itza, located in Quintana Roo, Mexico. The site covers a surface area of approximately 5 square kilometers, with hundreds of structures ranging in size between large pyramids, temples to small single-room sized buildings. Iconography covers certain walls and ceilings of the structures, with geometric details as small as millimeters. Archaeologists have studied the site for over a century, investigating the structures, artifacts and iconography, all to decipher a better understanding of the Maya civilization. Due to strong regulatory access control and political implications, few field experts get the opportunity to access many parts of the site in person, and those that do, sometimes only get a handful of hours. This motivates the digital documentation to complete archaeological research digitally, rather than in person. The only problem being- how to digitize a site of this size to resolutions required for meaningful study? Figure 1.5 shows a sparse photogrammetric model of one small section of the site, derived from ground and aerial imagery. A majority of the site was captured using 120,000 high-resolution images over the course of a week using a 42MP Sony mirror-less camera and a set of DJI UAVs. While the data collection was carefully planned, there remained areas that were missed, and others were captured with insufficient resolution for thorough documentation. The reconstructions alongside camera pose estimates were calculated months later leveraging clusters of GPU workstations and state of the art photogrammetry packages to derive results as can be seen in Figure 1.6.

The reality of many reconstructions that happen months after data collection, is that you realize that acquisition mistakes were made, data is missing, incomplete, and that there isn't always a way to go back. Much of this thesis has been motivated by the mistakes faced in the field. The below points summarize some of the desired imaging system characteristics that would facilitate the efficient 3D mapping of archaeological sites.

1. The system should be able to localize itself within the environment that it is capturing,

Figure 1.6: A dense photogrammetric 3D reconstruction of the Templo of the Grupo Inicial at Chichen Itza using ground and aerial photography.

providing feedback of areas that have already been capture, and those that have not.

2. The system would ideally reconstruct the scene geometry on device, providing real-time feedback on reconstruction accuracy.

3. The system should be compatible with the logistical acquisition implications of mapping a large-scale site. Power consumption should be minimal, the system should be compact, the imaging system should support the available lighting conditions, and capture both detail and context.

The requirements for a real-time mapping system, as defined, are ambitious, but it is with these goals in mind, that the set of systems derived within this work are developed.

1.6 Organization of Material

This thesis follows a set of co-dependent research areas (Figure 1.7), focused on building an overall multi-view reconstruction infrastructure. In the introductory Chapter 1, the research focus was introduced with a general formulation of the embedded multi-view design problem. It was motivated through two application domains: Autonomous Vehicles and Cultural Heritage Documentation.

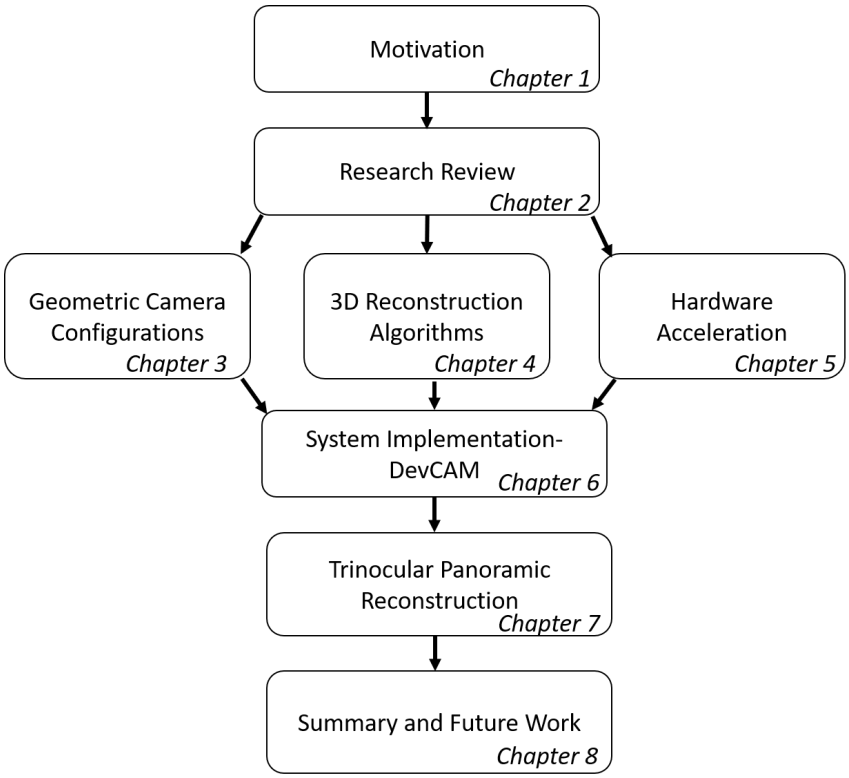

Figure 1.7: A summary of material organization within this thesis

Chapter 2 reviews the state of research in the field. It highlights the outstanding challenges and discusses the difficulties that will be addressed within this work.

Chapter 3 introduces the key epipoplar constraints that are discussed to be leveraged for the creation of efficient embedded multi-view systems. It discusses the effects of baseline and disparity on system reconstruction performance, as well as introducing the effect of disparity

orientation on feature orientation and geometric resolvability. This is discussed in the context of stereo, trinocular and partial ligh-field arrays. The multi-view configurations are then extended to cover spherical coverage, to meet the efficient data capture of full panoramic and spherical capture devices. The final part of this chapter discusses the importance of adequate camera array calibration and proposes a novel target design to address multi-camera calibration.

Chapter 5.1 introduces different types of image features which are triangulated into space to provide scene reconstructions. A novel semi-dense edge based feature set is introduced that is motivated by it's computational efficiency and the spatial accuracy in both image space and scene. The edge feature matching and triangulation is illustrated and the feature values are compared with results to sparse corner features and dense patch based reconstruction.

Chapter 5 introduces the considerations to hardware acceleration and how we can leverage the efficient algorithmic development to ensure efficient and fast reconstruction within a target architecture. Specifically, memory optimization is addressed in vision streaming settings, as well as the importance on control of image pre-processing. Within this Chapter, a novel pre-processing algorithm and implementation is also introduced to demonstrate the ability of improve reconstruction dataflow and data-dependency.

Chapter 6 describes the physical implementation considerations of a multi-view embedded system. It introduces an Open-Source infrastructure that facilitates the re-configurability of multiple small camera modules connected to processor boards capable of synchronizing all connected sensors, capturing at full resolution to onboard storage or which can be off-loaded over a high-speed networking interface. The implementation of a set of multi-camera arrays is demonstrated leveraging this novel development infrastructure.

The final Chapter 7 leverages the trinocular panoramic camera configuration built in Chapter 6 to capture and reconstruct real-world data on a moving vehicle. It extends the reconstruction work to include ego-motion estimation as well as full scene reconstruction. Evaluation of performance is completed in a number of comparisons. Firstly synthetic ground-truth data

is generated for an identical system, where pose and depth estimation is compared to ground-truth. The real-world system is co-located with a LiDAR unit and reconstruction performance is evaluated for a temporal sequence on an urban road setting.

This thesis is concluded in Chapter 8 by summarizing the contributions to the field, the limitations of the proposed methods and opens the discussion to future works.

1.7 Acknowledgements

This chapter is, in part, a reprint of the material as it appears, titled "A Survey of Dense 3D Reconstruction Methods for Multi-Sensor Embedded Vision" in UC San Diego Research Exam 2019. Meyer, Dominique. The dissertation author was the primary investigator and author of this material.

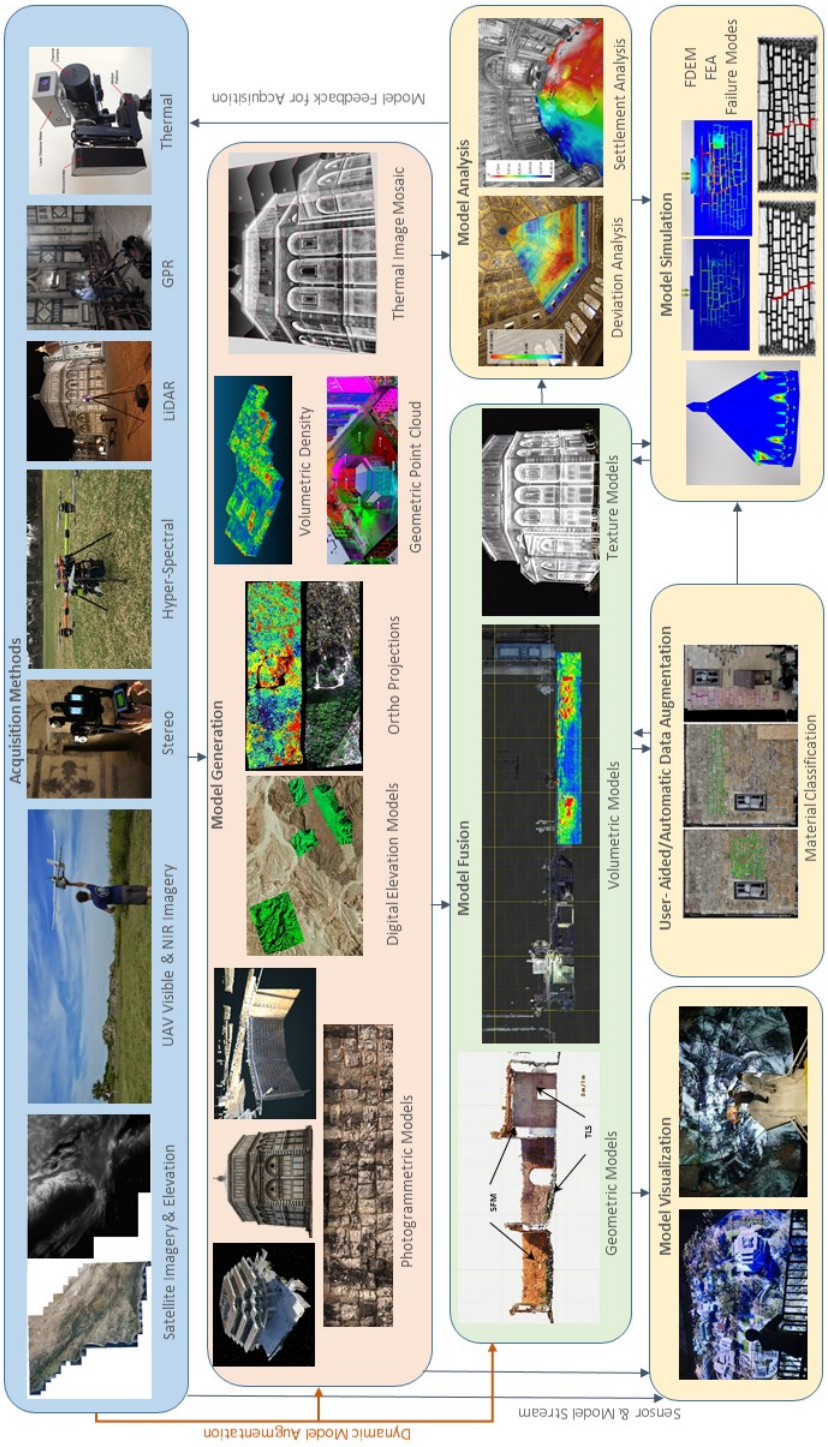

Figure 1.8: A workflow summary for the digital and non-destructive documentation of cultural heritage sites using a suite of sensory data acquisition platforms.

Chapter 2

State of Research of Multi-Sensor 3D Systems

This chapter reviews the state of research in embedded sensor arrays to reconstruct scene geometry. Sparse and dense multi-view reconstruction techniques have emphasized the computational complexity associated with scene reconstruction, which to date do not scale well to real-time processing on mobile processors. Hardware software co-design for embedded camera systems plays a critical role in dense scene reconstructions affecting accuracy, robustness and performance which will be reviewed in this work. The first part of the survey evaluates mechanical configurations of systems that consist of two or more camera pairs. Special attention is given to array layouts with respect to epipolar geometry and the resultant reconstruction difficulty and limitations. The second part of this chapter reviews the algorithmic methods to extract robust, dense depth from multiple sensors, followed by considerations on implementations and hardware acceleration techniques for different processor architectures. Existing solutions are evaluated based on these factors and a summary of state of the art system results are provided along with outstanding problems.

Sensor trade-offs for different environments and conditions are largely discussed in

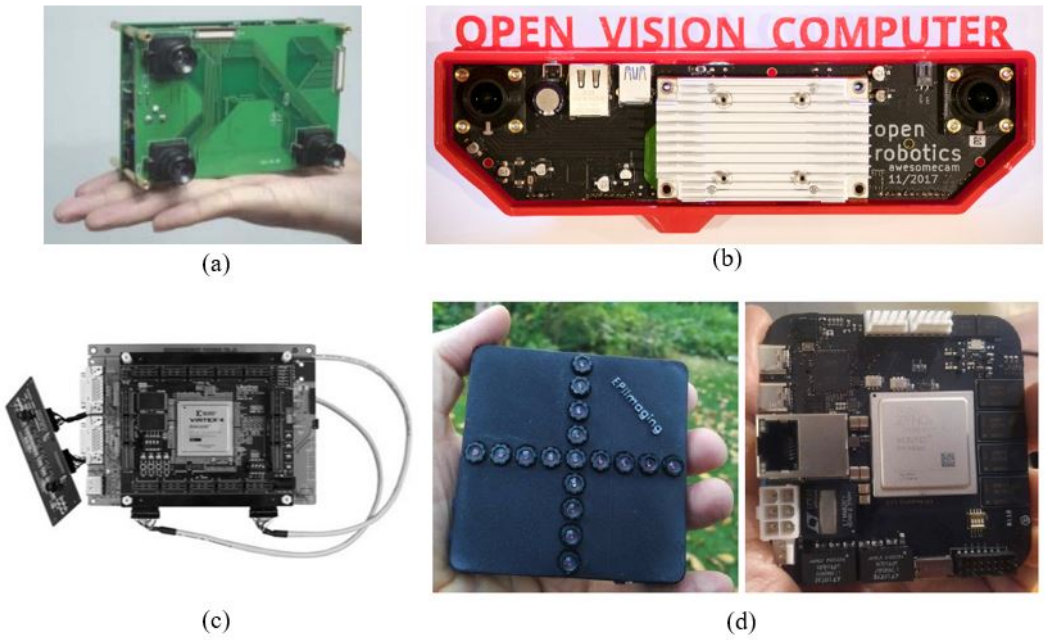

Figure 2.1: a) Trinocular FPGA camera introduced by [JZLA04].b) and c) traditional parallel configurations which leverage GPU and FPGA processing architectures respectively. d) lightfield array introduced by [BKM+19] with epipolar constraints as described in Figure 3.1.

[WSK+13, SLK15, NML+13, LD91] and will not be re-visited in this survey. Instead, only passive multi-camera vision systems will be evaluated with focuses on the following topics: 1) mechanical sensor layouts for algorithmic reconstruction ease, 2) multi-view algorithms and 3) underlying hardware architectures to accelerate the reconstruction algorithms.

2.1 Sensor Configurations

The most widely used multi-camera configuration is the stereo camera where two cameras are placed with an overlapping Field of View (FOV). Proposed systems by [WBJ+07b, Ano, HWLJ16, Yan06, GEM09, QMS+19, NJ04, How08] all leverage the traditional parallel stereo layout where the cameras are separated by a fix baseline and each have the same FOV. This dominant configuration is to uphold the epipolar constraint to reduce the search space in the stereo matching techniques. No self-contained real-time embedded system was found to have

been designed to leverage non-parallel configuration as the basis of the design, but many surround vision systems with large, overlapping FOVs leveraged stereo principles for Visual Intertial Odometry (VIO) [JO04, KKN+12].

Trinocular camera systems have the added benefit to allow for increased configuration options. They can be arranged in a co-planar L-configuration, as verteces of an equilateral triangle, and any configuration which is not co-planar. The L-configuration provides the benefit that both the camera to the side and the vertical camera with respect to the center camera are constrained by separate epipolar geometries [ZZW13, JZLA04]. In the case of cameras that are not co-planar, a common configuration is an object of interest centric set of camera poses (all cameras are configured inwards). This is shown in [WT99, WW06, MCMOM09, MID02, MK00] and provides increased viewpoint for object reconstruction when the object has an overall convex geometry facing the camera system. Finally, the research highlighted in [CCHL95] provides an approach to solve for an optimal trinocular configuration that minimizes the parallelogram defined search area of a central camera given the projections of the two epipolar lines from the separate cameras.

When considering more than 3 passive cameras that constitute a vision system, the degrees of freedom of the configuration grows with each added camera. The general set of configurations spans three trends: firstly bullet arrays have all cameras facing inwards creating rings [Xu18], secondly arrays that form a planar grid as seen in [WJV+05, VWJL04, VJM+15], and finally arrays with a subset of cameras from the grid arrays, where the cameras form co-linear series like in [BKM+19]. The benefit of inward facing arrays lies in the the ability to view all perspective of an object for N-view reconstruction whereas co-planar arrays provide the ability leverage epipolar geometry to robustly estimate geometry.

2.2 Pre-processing

Pre-processing steps in embedded systems serve to enhance raw image data in advance of view-point correlation, and play an important role in image point determination and hardware acceleration performance. The below tasks are a representative subset of demonstrated methods that are designed improve overall system performance.

- Debayering- For color sensors with a Color Filter Array (CFA), each wavelength color band is separated for the respective color channels. To reconstruct a color value for every pixel, a weighted interpolation is used which also acts as a low-pass filter [Reu17b]. Traditionally debayering is accomplished with a 3x3 or 5x5 weighted bi-linear interpolation with color weights alternating depending on the pixel location. Each color value is calculated accordingly, and the interpolation for the green channel is shown in Equation 2.1. More advanced filtering and enhancements methods are proposed by [MHC04] and [MAC06] which are widely used in open-source implementations and commercial software packages such as [Mat17] and [Bra00].

$$\begin{aligned} \widehat{g}(i,j)) &= \frac{1}{4}\sum_{(m,n)} g(i+m, j+n) \\ (m,n) &= \{(0,-1),(0,1),(-1,0),(1,0)\} \end{aligned} \tag{2.1}$$

  The proposed methods by [FZY09, RH13, ZWW13b] summarize FPGA acceleration methods for such debayering/demosaicing tasks through the use of highly parallelized bilinear interpolation implementations on FPGAs.

- Image enhancements- In many cases images captured in the world are subject to noise, areas of under or over exposure and motion blur. A set of filters can be used to improve image quality and achieve a variety of image enhancement effects such as in works [Lee80], [OR90]. This is commonly to reduce the noise characteristics and to increase robustness

in the point matching. Another critical step to finding good features and achieve good matching is adequate contrast and intensity values distributions. Image stretching techniques such as [Yan06] and [AF81] are used to improve the image histograms and maximize the range of values for a certain pixel bit-depth. [SP11b] reviews common image enhancement algorithms and how they can be accelerated using custom hardware configurations on FPGAs.

- Rectification- Images are rectified to accommodate for intrinsic parameters such as lens distortions. Additionally, in the case of multi-camera systems, the images are also rectified to parallel configurations to satisfy the epipolar constraint. Mechanical manufacturing limitations prevent systems to be perfectly aligned, so there always needs to be some amount of rectification to achieve a row-aligned search space. The rectification block in an embedded vision reconstruction pipeline frequently becomes the throughput bottleneck due to the memory requirement [HN15a]. Upon pixel stream receipt by the processor in a FIFO scheme, the processor needs buffer the number of rows of rows the define the maximum vertical remapping of the rectification. This in the case of sensors with an order of magnitude of Megapixels requires 10s of Megabytes of on-chip memory footprints per frame [HN15a]. The method provided by [OK08b] proposes the use of compressed 1D Look up Tables (LUTs) for the rectification, and a block-buffered implementation to achieve minimal memory utilization in the forward-backward rectification.

Table 2.1: This table summarizes the systems evaluated in this survey. It illustrates the disparity estimation rate achieved by the respective systems, the sensor layout and the hardware acceleration pairing for the respective parts of the algorithms

System	Year	Horizontal Resolution (px)	Vertical Resolution (px)	Frame Rate (fps)	Megapixel disparities per second (MPd/s)	Power (W)	Hardware Architecture				Sensor Layout
							Rectification	Matching	3D Projection	Interfacing	
[WBJ+07b]	2007	512	480	200	49.15	1	FPGA	ASIC	FPGA	CPU	parallel stereo
[WVH97]	1997	320	240	42	3.23	22.5	FPGA	FPGA	FPGA	CPU	parallel stereo
[BHF+10b]	2010	640	480	103	31.64	N/A	FPGA	FPGA	N/A	CPU	parallel stereo
[Ano] D400	2018	1280	720	90	82.94	1.44	ASIC	ASIC	N/A	ASIC	parallel stereo
[Ano] D420	2018	1280	720	90	82.94	1.12	ASIC	ASIC	N/A	ASIC	parallel stereo
[GEM09]	2009	750	480	25	9.00	3	FPGA	FPGA	FPGA	CPU	parallel stereo
[LS11]	2011	640	480	60	18.43	N/A	GPU	GPU	GPU	CPU	parallel stereo
[BKM+19]	2019	1280	960	60	73.73	5	FPGA	CPU	CPU	CPU	epipolar arrary
[MID02]	2002	659	494	2.5	0.81	N/A	CPU	CPU	CPU	CPU	converging trinocular
[JZLA04]	2004	640	480	30	9.22	N/A	FPGA	FPGA	N/A	CPU	L-shape trinocular
[MLR+07]	2007	320	240	150	11.52	N/A	FPGA	FPGA	FPGA	CPU	parallel stereo
[HTGT10]	2010	100	100	75	0.75	N/A	FPGA	FPGA	FPGA	CPU	parallel stereo
[HZW+10]	2010	450	375	7.74	1.31	5	N/A	DSP	DSP	CPU	parallel stereo
[JCDP+09]	2009	640	480	64	19.66	N/A	FPGA	FPGA	FPGA	CPU	parallel stereo
[KSY+99]	1999	320	240	20	1.54	N/A	FPGA	FPGA	FPGA	CPU	parallel stereo
[KMI+03]	2003	256	192	50	2.46	N/A	ASIC	ASIC	ASIC	CPU	parallel stereo
[DRM03]	2003	256	360	30	2.76	N/A	FPGA	FPGA	FPGA	CPU	parallel stereo

2.3 View-point Correlation

View-point correlation evaluates which points in one image match to image points seen by all other cameras. Sparse methods use a feature detector such as the Harris corner detector [Der04] or thresholded gradient eigenvalues [Shi94] along with a descriptor which describes the area in the image around that corner. By comparing descriptor vectors, it is possible to identify the set of most likely matches [Bau00]. Dense methods are tasked with finding a matching pixel with the use of a matching cost. The most basic matching can be done using simple pixel intensities, and more complex and robust methods evaluate textures and radiometric variation. These are commonly absolute differences, squared differences, sampling-insensitive absolute differences or truncated versions [BT98]. Window based approaches frequently use the sum of absolute or squared differences, normalized cross-correlation and rank and census transforms [ZW94]. Finally more complicated similarity measures leverage mutual information and approximative segment-wise mutual information [Egn00, Hir05, KKZ03, ZKU+04]. There is a large body of literature which reviews the trade-offs of different stereo correlation methods, such as the surveys completed by [SS02, SVdMG04]. This section however will briefly describe the dense stereo correlation methods that have been implemented on real-time embedded systems.

[HWLJ16] Proposes the use of a Sobel Operator to derive image gradient information prior to applying a Census operator with adaptive matching window sizes. The authors propose a slight alteration to the census matching by replacing the center pixel value by the window pixel average to reduce the center-weighing nature of the matching. The work by Gehrig et al. [GEM09] leverages the state-of the art semi-global matching algorithm introduced by [Hir05]. This method uses Mutual Information (MI) as a cost function to match un-occluded pixels with maximal MI while also minimizing the energy function over a global window size which provides a smoothness constraint. An approximation to minimizing the energy function and as such finding a match consistent with the global energy minimum is proposed by approaching the image point

by eight or sixteen symmetrical directions. [LS11]

The works by [BBM87, BB89] have introduce native correspondence between viewpoints by leveraging the Epipolar constraints. They construct a two-dimensional manifold defined at the zeros crossings of a 3-dimensional spatiotemporal Laplacian providing geometric surface continuity over time. While preliminary works demonstrated reconstructions from a single moving camera, this was later expanded to sensor arrays to achieve full three-dimensional scene reconstruction for real-time computation [BKM+19]. The benefit of this method is that the matching across sensors is inherently self-occurring due to the spatio-temporal continuity in the epipolar planes generated from all the sensors simultaneously.

2.4 Hardware Acceleration Design Optimization

This section will provide hardware acceleration methods which works have provided to generate real-time dense reconstruction. The work by [HWLJ16] leverages a FPGA architecture to accelerate the Sobel gradient operator, adaptive thresholding module and stereo matching module. The implementation uses an off-chip cache window to accelerate the computation and buffer larger data amount that cannot be contained within on-chip BRAM. [GEM09] Uses a Xilinx Virtex 4 FPGA linked to a PC CPU to accelerate the image data. The data is pushed using Direct Memory Access (DMA) to the fabric of the FPGA where one or two SGM blocks complete the left-right consistent matching. The implementation is pipelined and parallelized to achieve improved performance at minimal power and clock speed. The DeepSea G2 stereo system proposed by [WBJ+07b] uses a multi-architecture approach to optimize performance and leverage the benefits of each respective architecture for different tasks. The stereo rectification, background model generation (scene segmentation) and projection space (projection of disparity maps into point clouds) are all completed on an FPGA, while the stereo correlation block is accelerated in a custom ASIC, the pre-processing is done on a custom DSP and finally all system interfacing

and user application are running on a PowerPC CPU. The proposed system in [BHF+10b] pairs a Xilinx Virtex 5 FPGA with an Intel IXP 460 CPU to accomplish co-processing of the image data stream. The D400 and D420 Realsense systems [Ano] uses a dedicated ASIC to complete the full image processing and depth reconstruction. Finally Baker et al. [BKM+19] leverage an MPSoC to compute the depth images in real-time. Part of the preliminary processing such as the rectification, debayering and filtering happens on the FPGA while the epi-image creation and surface reconstruction happens in the ARM cores of the CPU. The DRAM where images are buffered, is shared between both architectures to improve data throughput.

2.5 Performance Evaluation and Discussion

A number of system goals have this far been introduced. The dominant objective is arguably the reconstruction accuracy and throughput. Due to the difficulty in knowing the correct world points X_i and hence derive the system *SSD*, the evaluation in geometric accuracy remains a largely unaddressed problem. Most algorithmic evaluation are done using benchmark data with ground-truth depth values/ scene geometries. The works reviewed within this survey are summarized in Table I. This provides a high-level overview of the different implementation approaches for system designs. It is seen that a majority of systems leverage a parallel stereo camera layout and many use hybrid architectures to achieve the desired frame-rates. The leading set of systems are by [Ano] and [BKM+19] which are both commercial endeavors leveraging ASICs and FPGAs respectively to accelerate the processing. Many publications did not state power consumption for their system making it difficult to consider that as well as cost in the

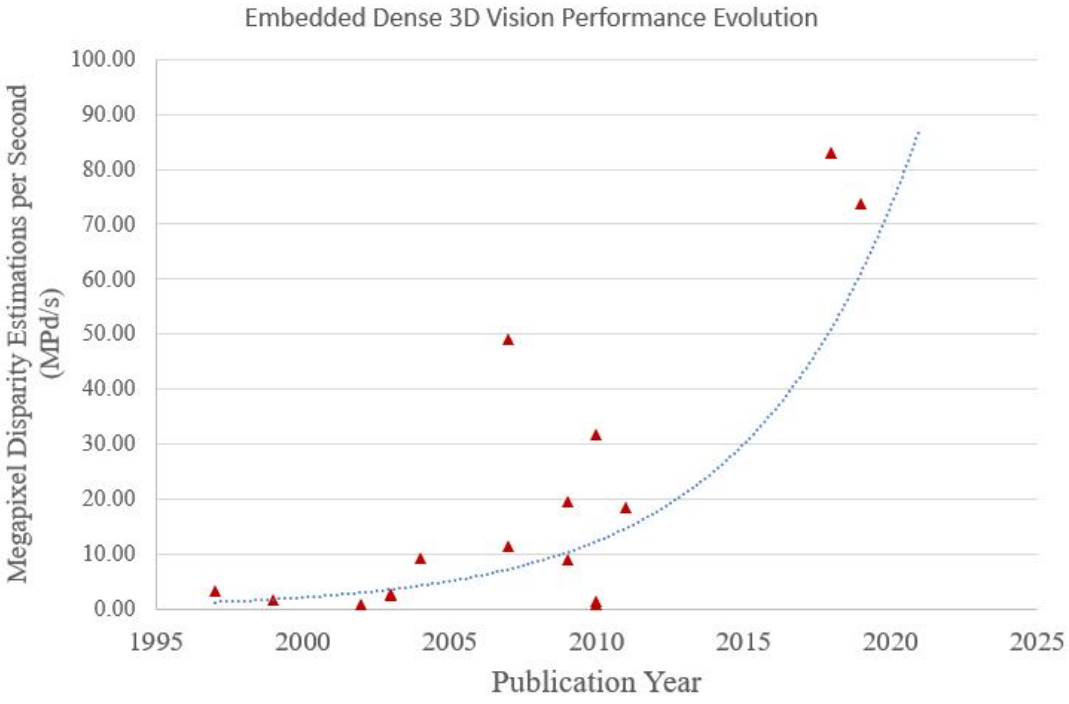

Figure 2.2: A plot of dense disparity rates found in cited systems over time, taken from Table I and complemented with an exponential trendline.

evluation of the systems. The system performance in Table I is plotted over time in Figure 2.2 to observe the improvement of performance over the last few decades. It is clear that the systems have gradually improved, but there appears to be no clear trend in standardization of processor architectures for such systems.

2.6 Discussion: Open Problems

The development of a fully integrated multi-camera system capable of processing depth/scene information on-board in real-time is proven to be a difficult task due to the multitude of problems being addressed. There needs to be a logical hardware-software co-design that maximizes system performance to achieve the specified requirements. Due to the disjunct efforts in system development, largely driven by varying applications, it is difficult to directly compare systems and methods to improve overall performance. This section will discuss the set of problems that remain unsolved throughout this research area, and highlight potential avenues for future research.

Stereo matching algorithms have been evaluated over standardized benchmarks [SSG+17] on both synthetic and real stereo data to provide a fair and platform agnostic ground truth comparison for performance. As every system is inherently different in the kind of data that it collects, it is difficult to standardize on a way of validating the geometric estimation performance. The world points X_i are not easily known and therefore techniques need to be developed to estimate these with some confiden through non-vision based methods to evaluate new systems. LiDAR has become a popular tool for creating ground truth depthmaps and scenes representations, but these have shown many limitations. It is expected that improved and standardized methods will be developed to easily and critically evaluate embedded multi-view systems. Additionally, an important part in depth estimation from closely co-located sensors is to understand edges, in what direction they go, and which of them contribute to occlusions. From a stereo system, an occlusion manifests itself in a non-match which is normally quantified as a geometric error. It would be important to introduce an evaluation metric that considers reconstruction performance at edges and their ability to understand the nature of edges and occlusions. Finally, it is difficult to evaluate photometric correctness of any camera system. While geometry is more easily approached with different methods, photometric evaluation requires a full light-field representation. This is most easily accomplished in simulation but would be important to evaluate in real-world systems.

A large problem in embedded research is the time commitment to building and validating systems. While theoretical research lends itself to a scalable approach of researching problems, involving hardware in the loop frequently limits the extent of research given the same resources. Many commercial endeavors also assume a faster development time requirement, pushing integration time as a factor into design optimization. It is important to evaluate whether a design for a custom ASIC should take place, which increases the development time by at least an order of magnitude when compared to FPGAs and two orders of magnitude compared to a traditional CPU implementations. It is therefore invaluable to factor in difficulty, feasibility and timing of system integration when developing systems.

The evolution of hardware architectures has played a important role in the development of embedded systems at large. Industry is fostering the continuous target of bringing increased computational power to smaller and more power efficient processors, largely driven by the mobile tele-communication industry. The future of embedded camera systems is largely dependent on the future of computation. Systems are still only operating at very small resolutions ($<$ 2MP) to achieve real-time performance when compared to sensor resolutions (8+MP) common in systems found at the time of this survey.

[GEM09] Introduced temporal smoothing in depth estimation to improve depth estimation over time, however there largely lacks research in multi-view dense optimizations over time. When only consider a single time step of points, these are subject to larger error than if they were observed from multiple cameras over time as the cameras move. SLAM and photogrammetry methods use dense optimization techniques to optimize reconstructions over time but due to the computational complexity is not even feasible in real-time on large computational systems. This will become of major interest for embedded research as processor performance improves over time.

The overall performance in Figure 2.2 show that the field of dense embedded vision is still largely not keeping up with regular video frame rate RGB captures. The problem for efficiently

solving depth and scenes in real-time given limited hardware resources will continue to rely on hardware-software co-design methods to maximize performance.

2.7 Acknowledgements

Chapter 2, in part, is a reprint of the material as it appears, titled "A Survey of Dense 3D Reconstruction Methods for Multi-Sensor Embedded Vision" in UC San Diego Research Exam 2019. Meyer, Dominique. The dissertation author was the primary investigator and author of this material.

Chapter 3

Camera Geometry

3.1 Epipolar Camera Geometry

The common camera converges a set of light rays in the world onto a planar sensor, where pixels discreetly sample the flux of the light field. Knowing exactly where the camera is, how the rays pass through the optical assembly, and how these get accumulated in electrical signals, allows us identify up to scale, where these rays originated. With two or more cameras, we are then able to find the point in 3D space which fits the intersection of two or more rays [HZ03]. The camera projection matrix P describes the mapping of 3D world points X_i to image points x_i in the 2D image plane and the Fundamental matrix F maps a point x to an epipolar line in the other image of a stereo pair, on which the corresponding point x' lies according to epipolar geometry. Figure 3.1 shows the point projections of a single camera, stereo cameras and multiple parallel cameras. Assuming we have a calibrated multi-camera system (we know the fundamental matrices for every respective camera pairs), to get a dense match across images, we need to find the matching points x_i' for every x_i. Inherently this poses a 1D search problem for every point that needs to be matched. The importance in geometric sensor layout affects the matrix F, and as such, the equation of the epipolar line in the second image [HZ03]. Searching across a single

row of pixels greatly improves performance when compared to searching for a match with a line that spans a diagonal in the image where any sub-pixel point may be the matching point. The computational implications will be further discussed in Section 1.2.1. As such, we aim to choose a sensor configuration that minimizes the computational cost to solve for multi-view matching. This almost always implies the requirement for sensors to be parallel with each other, with image centers co-located on a single axis or plane. Figure 3.1 shows the the effect of multiple sensors which are parallel and in-line imager centers. We can observe that all epipolar lines are parallel as well as covering the same row of pixels for sensor respectively. A point in space is therefore projected in the same pixel row of all sensors, given that it is visible by all cameras.

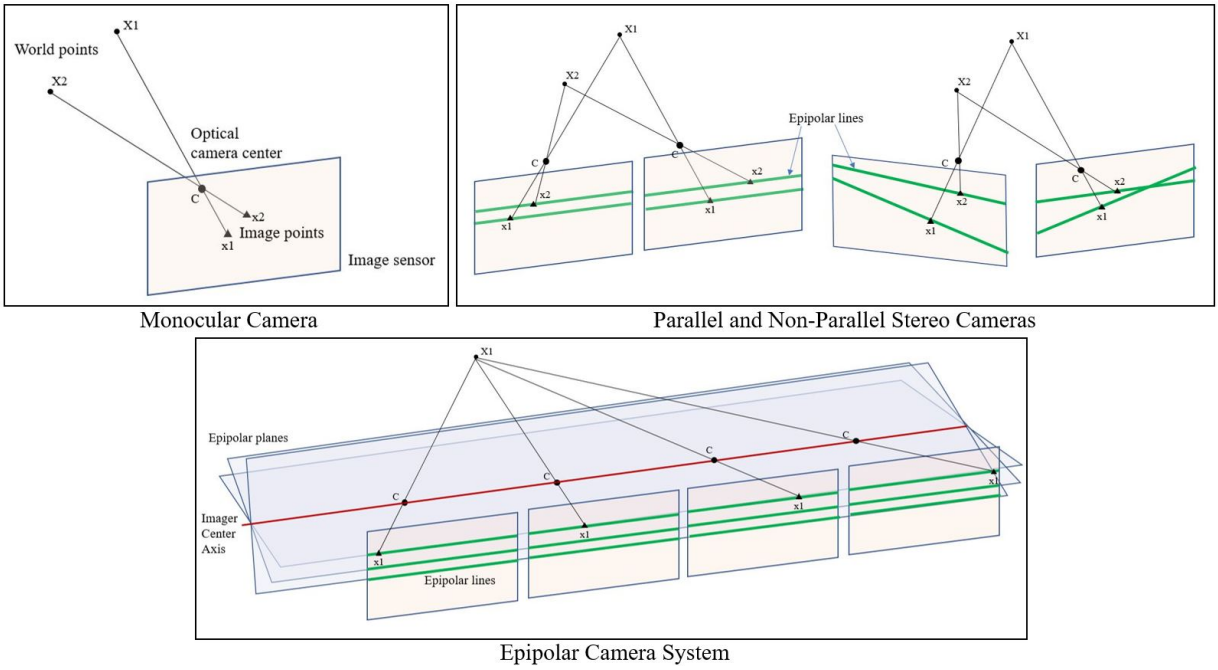

Figure 3.1: Epipolar principles in single, stereo and multiple camera systems. a) monocular camera observing two world points. b) stereo cameras with parallel and non-parallel configurations and their epipolar lines. c) co-linear camera array with epipolar planes constraining image points to the same pixel row.

The second effect of relative sensor placement has to do with a less commonly addressed issue, but an important note introduced by [MWS+19]. Every physical system assumes mechanical and electrical mis-alignments, noise and non-perfections and as such, even state-of-art

stereo camera integrators such as [Ano], cannot build an ideal stereo camera. Image rectification serves as a way to remap pixel values according to some transformation which in many cases is a combination of a perspective transformation and lens un-distortion. This pixel re-mapping involves either forward-backward mapping with interpolation on the source pixel values [LZ99], or a forward only mapping with splatting and interpolation on the output [CDK99a]. With increased pixel mapping distances, the image resolution is affected and therefore it is desired to minimize perspective transformations and strong distortions to maintain maximum ability to match the correct sub-pixel points between images. Figure 3.2 demonstrates the change in re-mapping distances when comparing parallel and outward facing stereo cameras. It is noticeable that during the rectification process, the resolution is reduced for the outward facing system.

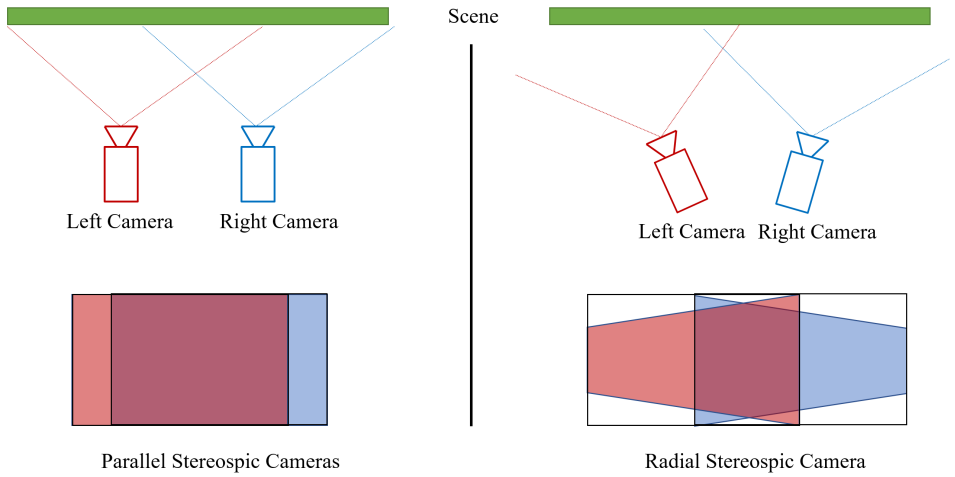

Figure 3.2: Comparison of the perspective transformation associated with non-parallel stereo cameras compared to parallel cameras from [MWS+19]

When deciding on a sensor configuration we try to achieve two optimizations: firstly we align the sensors so that the reconstruction accuracy is improved and as such the scene world points SSD is minimized; and secondly that the algorithmic matching is facilitated to reduce computational complexity, increasing system throughput and decreasing system latency, hardware requirements, power and size.

3.2 Disparity

3.2.1 Baseline, Disparity and Depth

This section will discuss the relationship of stereo camera baseline and it's affect on disparity and consequently depth estimation. Human inter-ocular distance is approximately 75mm, allowing us to derive a notion of depth from what we see. While many visual cues are derived from perceived flow of head movements, the view disparity is arguably one of the primary notions that allows us to comprehend the geometry of what we are looking at, and the position of our viewpoint. In machine vision, a similar principle applies, we use the disparity between observed features to triangulate it's location in 3D space. This relationship essentially comes down to the following key equation:

$$Z = f \cdot \frac{b}{d} \tag{3.1}$$

For an image disparity d that we observe between two or more observations, with reference camera of focal length- f, and stereo baseline b, we obtain a depth Z. This gives rise to a critical proportionality $Z \propto \frac{1}{d}$ assuming a constant scale factor $f \cdot b$. Given a human inter-ocular distance of 75mm for a pair of cameras, Figure 3.3 shows how the inverse proportional relationship of disparity affects the depth estimate. Consequently, near objects, demonstrate a disparity that can be bounded by a maximum disparity value while objects that appear further approach zero-disparity.

When deciding on a multi-camera baseline, the distance will affect a number of characteristics that we must consider. Firstly, it changes the disparity range in the search space is completed. With larger baselines, the maximum disparity for closer objects will be larger, increasing the 1D search space for a rectified stereo pair, and consequently costing more computational and memory resources. The other factor is the effect of depth precision. Assuming that it is possible to localize

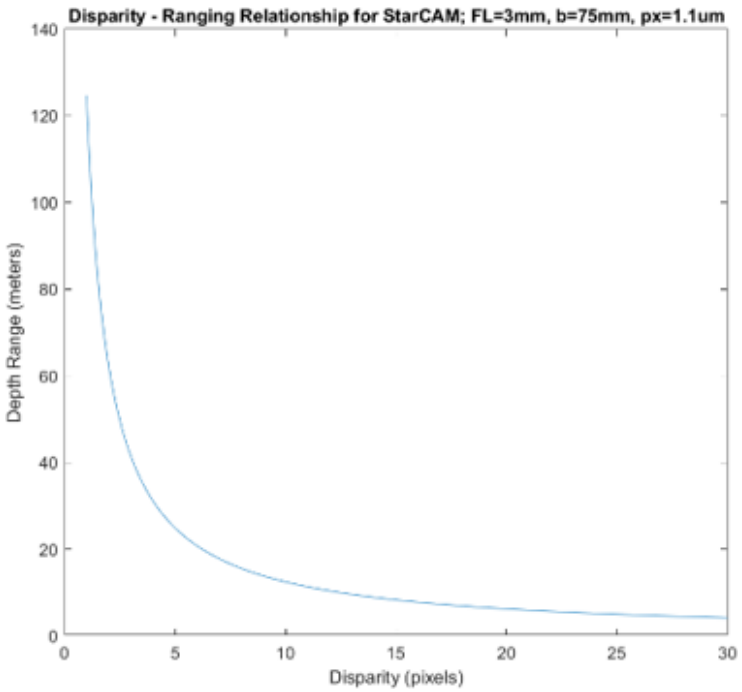

Figure 3.3: Disaprity- depth relationship of an example stereo camera system (dual See3CAM130) with a baseline of 75mm

image feature to some sub-pixel accuracy, depending on the baseline, that will impact the depth accuracy. As, such it is important to consider all these factors in the configuration of multi-view systems.

3.2.2 Directional Disparity

Feature types include corners, edges and patches. While these provide respective advantages in different image representations and will be discussed further in Chapter 5.1, one key aspect needs to be considered when deciding on camera geometry- disparity and feature orientation. In the case of corners, they are well defined in both horizontal and vertical directions since they lie on areas of both large x and y gradients. In the case of edges or patches however, it is possible that they are localizable only in one direction, which in the case of multi-view reconstruction, plays an important role in triangulation. Consider two road markings observed from both a horizontal and vertical pair of cameras located on a vehicle (presented in figure 3.4).

The *Ref. Camera* is shared between the horizontal and vertical stereo pairs, and both respective pairs are theoretically perfectly aligned with unit intrinsics (no lens distortions). The first lane marking, 1, is a horizontal line. In the case of of the *Ref. Camera* and the *Right Camera*, this line is coincident with the epipolar line between both cameras. A disparity estimate can therefore not be achieved, and consequently the depth estimate of that horizontal feature is not possible, except by interpolating from neighboring areas, where we can estimate the depth from unique feature matches. Consider now the stereo pair formed between the *Ref. Camera* and the *Top Camera*: the lane marking 1 is perpendicular to the epipolar lines, and can easily be uniquely defined in that axis, allowing for a robust disparity and depth estimate. While edge features are naturally ambiguous in one direction, it is also the case in patches that demonstrate weak textures. In the case of a Census transform that overlaps with a perfectly horizontal edge, unless there is a unique texture, it is difficult to estimate a disparity from a horizontal stereo pair. The second feature example in Figure 3.4) is a vertical line as observed from the three camera perspectives.

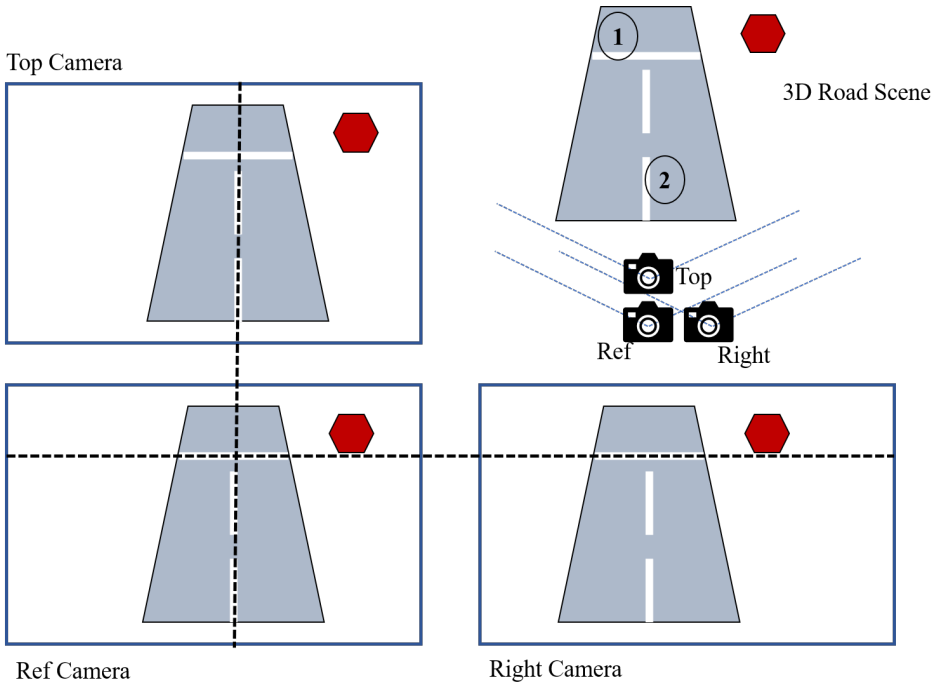

Figure 3.4: Demonstrating the effect of horizontal and vertical disparity

In that case, the *Ref. Camera* and the *Top Camera* have vertical epipolar lines which align with the vertical lane marking feature, unable to define a disparity and depth. The horizontal camera pair is however able to localize the feature perpendicularly to its disparity.

The road marking example has shown, that in cases where perfectly horizontal or vertical features are observed by horizontal or vertical stereo cameras respectively, it is difficult to estimate the depth directly. In the event that a diagonal world feature is observed, it should therefore be possible to constrain it using both a horizontal and vertical stereo pair. This is where we introduce trinocular camera configurations, leveraging multi-directional disparities to improve our depth estimation. This thesis introduces a novel constraint which enforces this multi-view prior:

For two ideal stereo pairs orientated perpendicularly, co-planar, and with equal baseline, directional image features observed from both must have equal disparity ($Disp_{hor} == Disp_{ver}$), equating to the same depth.($Z_{hor} == Z_{ver}$)

The introduction of this constraint will be further discussed in Chapter 7 where we can leverage it for outlier detection and improved disparity estimation. With two perpendicular observation pairs, it should be possible to resolve all directional features in a scene, but what happens when we over-observe a scene with redundant directional disparities? Figure 3.5 shows a grid array of 9 cameras (3x3) that observes a sphere in a scene. This configuration approaches the commonly used light-field arrays where there are upward of 9x9- individual observations. In this case, we see that the epipolar lines converge around the principal point of the reference camera, and that the sphere can be triangulated from the circle observations from all cameras. The problem however with this configuration is that the grid array implies that the baseline (distance between focal points) does not remain constant for the horizontal, vertical and diagonal observations, which consequently changes the disparity offset of the observed circles.

Let us now re-arrange these cameras in a circular pattern with equal baseline, Figure 3.6. What we suddenly see is that the circle observations are all offset by the same disparity, forming a disparity circle which through triangulation of the array baseline, allows us to estimate the depth.

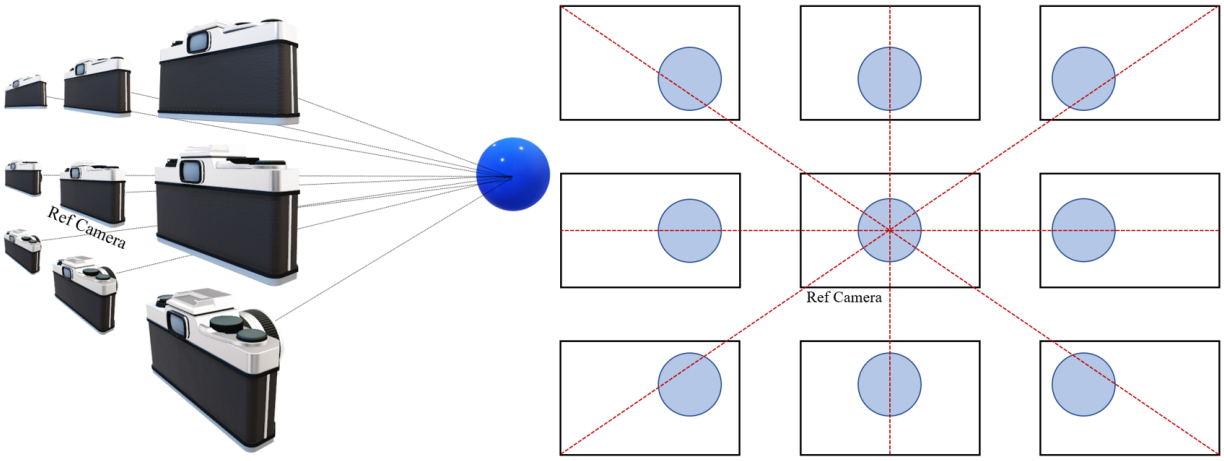

Figure 3.5: A grid array of 9 cameras that share a central reference camera. The surrounding cameras have epipolar lines that converge at the reference camera.

This observation formulation would enable any feature to be robustly triangulated, but with the disadvantage than diagonal and vertical epipolar lines will be more computationally demanding to search than horizontal only, since the image sensor dataflow is on a row basis, increasing the memory buffering requirements to meet both the rectification demands of the system as well as the rows to fill the maximum disparity search window about the search point in the reference image. Additional discussion will be provided in Chapter 5.

3.3 Spherical Configurations

Embedded systems often require fields of view (FoV) that extend beyond that of a single camera, or an array of cameras facing the same direction. While increasing the FoV through changing the optical lens element, is an option up to approximately 180deg for fisheye lenses, it comes at the cost of reduced spatial resolution. It is therefore favorable in many cases to increase the number of cameras and simultaneously capture different FoVs. In the case of a monoscopic panoramic configuration, it has often been desired to co-locate the focal points of all cameras as shown in SubFigure 3.7 a), which yields a system that is mechanically unfeasible to assemble, but that provides a strong constraint for the algorithmic stitching of the images. This constraint

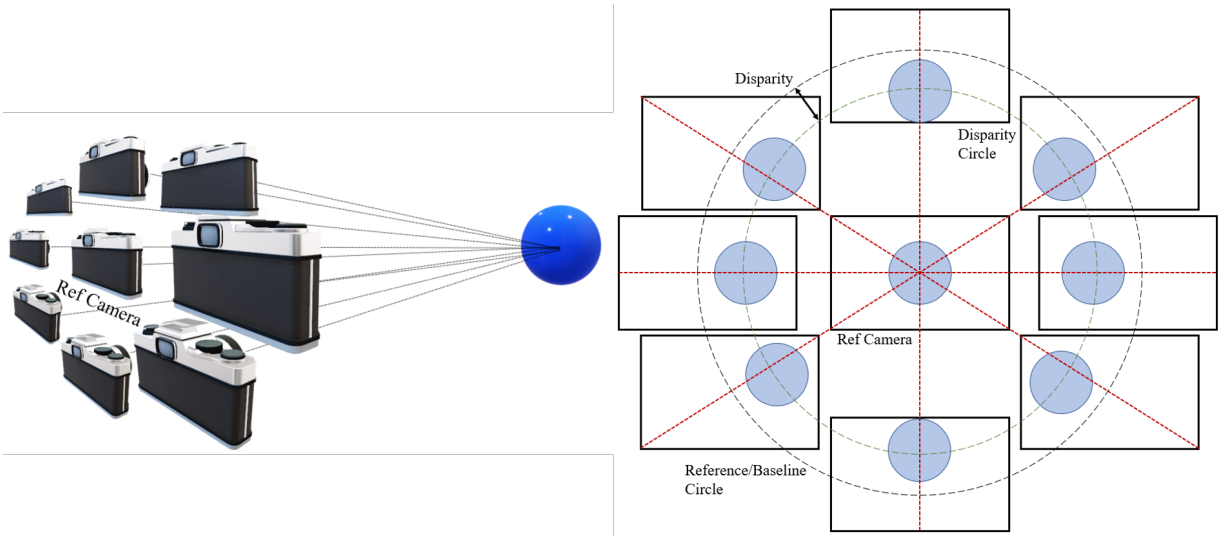

Figure 3.6: A 9 camera array in circular configuration with equal baseline. The implication is that the disparities must all be equal, and lie on a disparity circle offset from the central reference camera principle point projection into all other cameras.

is that the extrinsic parameters are solely defined by a rotation and no translation. Rotating the cameras with an offset that is as small as possible while allowing it to be physically feasible yields a configuration as shown in 3.7 b). The implications are however that the set of perspectives do not share a coincident focal point, and therefore require any algorithmic projection and stitching methods to consider both a rotation and translation between the respective cameras.

In the case of stereo pairs, the same principles apply. The main choice of stereo-panoramic configurations comes down to whether the stereo pairs are kept parallel or follow a spoke (radially outward facing) design. As discussed, parallel stereo pairs provide unique advantages for the search reduction to image rows (epipolar lines of parallel pairs), and reduce the loss of spatial resolution. When patterned spherically, these stereo configurations assume a position as shown in Figure 3.8. Parallel stereo pairs mechanically interfere when spherically patterned unless they are interleaved with an appropriate baseline.

Overlapping stereo interleaved pairs allows for a fully panoramic view field to be captured, with depth estimation across it. Figure 3.9 shows how two world points X_1, X_2 are observed from one of the 8 stereo pairs in the array x_1, x_2. When considering the projection into spherical

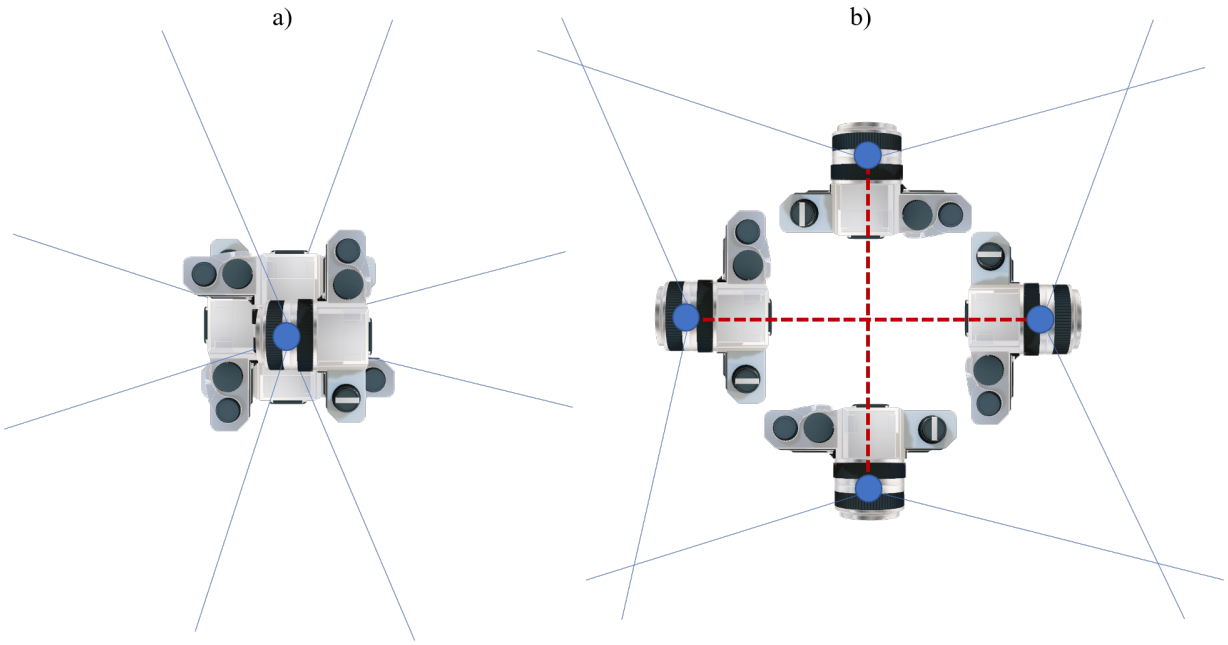

Figure 3.7: Monoscopic camera configuration of 4 cameras: a) Co-located camera focal-points (mechanically unfeasible), b) Offset rotation (mechanically feasible)

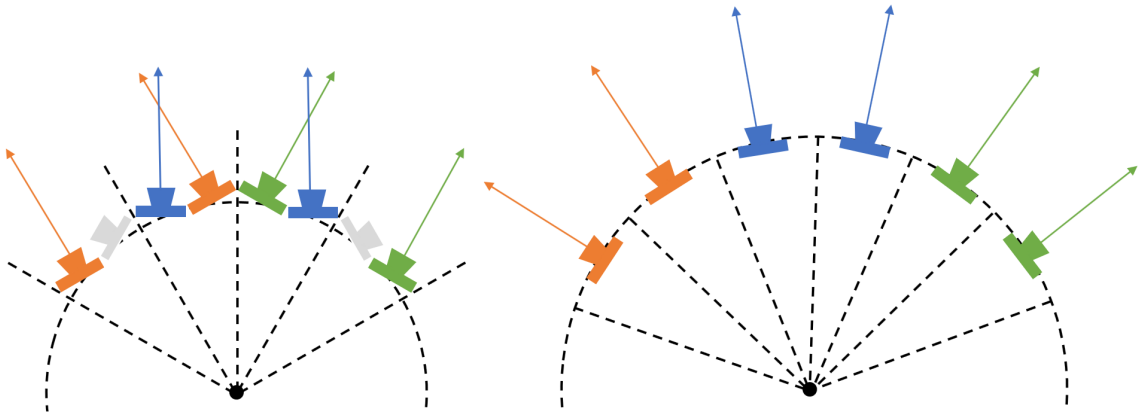

Figure 3.8: A comparison of parallel stereo and radial stereo panoramic camera configurations

coordinates, and unwarped to cartesian coordinates, one can see that the image observations form a continuous panoramic band, with all observed points, being epipolar constrained within each respective stereo pair.

The number of stereo pairs required to fully observe a panoramic view circle an be

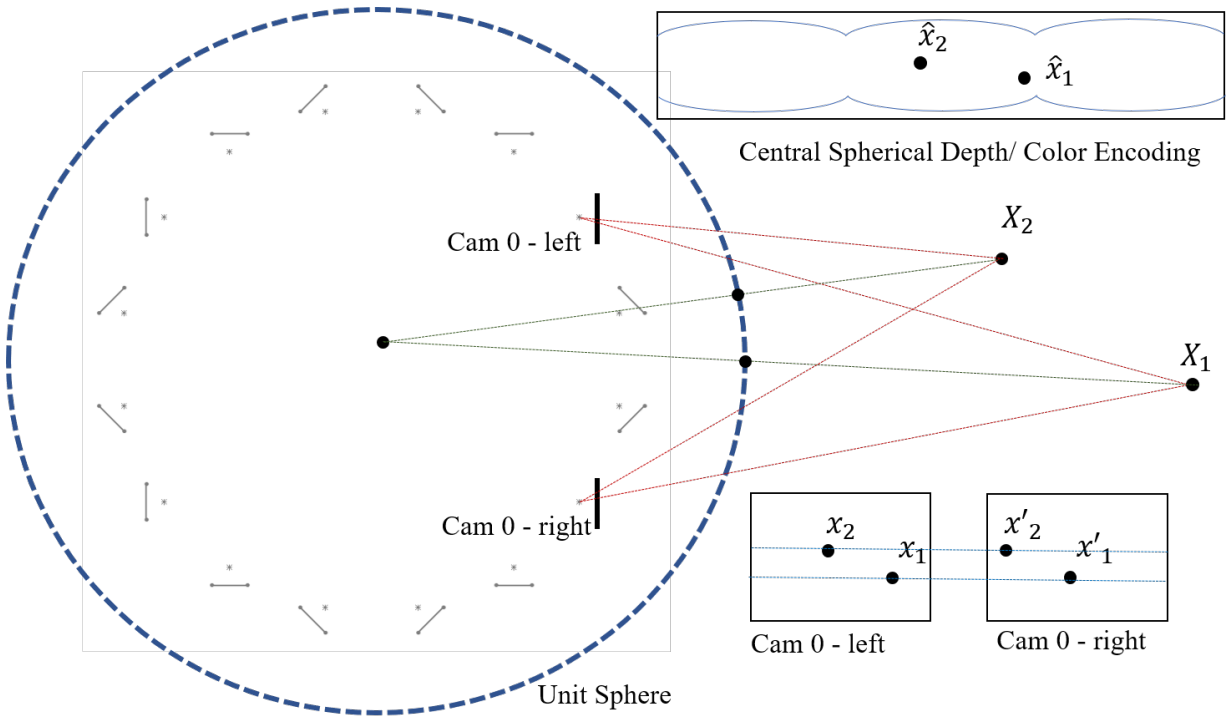

Figure 3.9: Stereo-Panoramic configuration in the context of a unit sphere projection model

calculated from the horizontal FoV of the sensor, lens pairs. If applications require larger vertical coverage, up to the a full Spherical FoV, additional camera rings can be tilted upwards and downwards respectively to increase that vertical coverage. A certain overlap is normally preferred to account for the lens vignetting that may slightly reduce FoV on corners. Examples of monocular and stereo, multi-ring panoramic configurations for a embedded cellphone camera modules are shown in Figures 3.10 and 3.11 respectively.

$$CamNo._{hor} = ceiling(\frac{360}{FOV_{hor} * Overlap\%})$$
$$RingNo._{ver} = ceiling(\frac{180}{FOV_{ver} * Overlap\%})$$

Figure 3.10: Panoramic configuration with 2 bands of See3CAM130 cameras for a larger vertical FoV

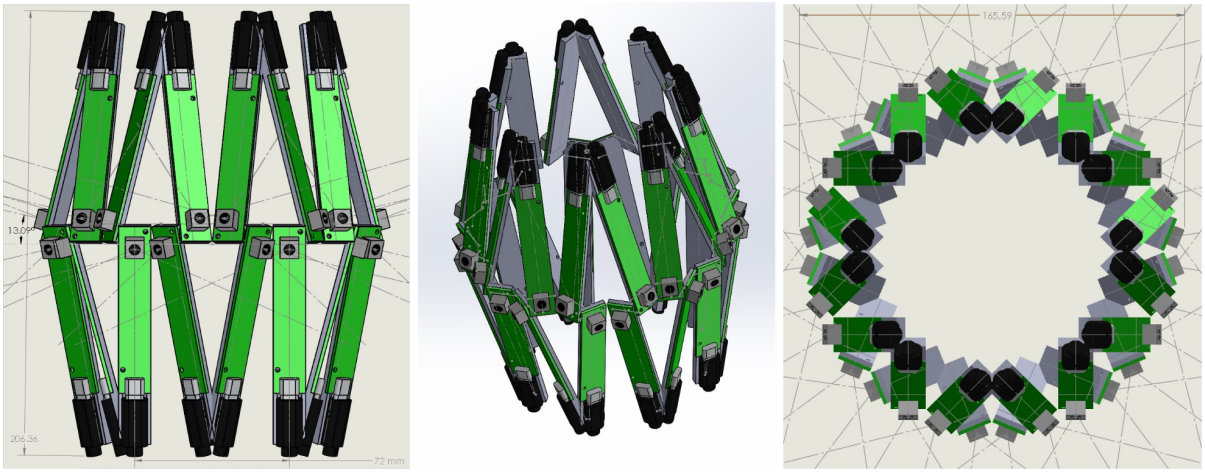

Figure 3.11: Stereo-panoramic configuration with 2 bands of See3CAM130 cameras for a larger vertical FoV

3.4 Calibration

Robust multi-camera calibration is a fundamental task for all multi-view camera systems, leveraging discreet camera model fitting from sparse target observations. Stereo systems, photogrammetry and light-field arrays have all demonstrated the need for geometrically consistent calibrations to achieve higher-levels of sub-pixel localization accuracy for improved depth estimation. This work presents a calibration target that leverages multi-directional features to achieve improved dense calibrations of camera systems. We begin by presenting a 2D target

that uses an encoded feature set, each with 12 bits of uniqueness for flexible patterning and easy identification. These features combine orthogonal sets of straight and circular binary edges, along with Gaussian peaks. Our proposed feature extraction algorithm uses steerable filters for edge localization, and an ellipsoidal peak fitting for the circle center estimation. Feature uniqueness is used for associativity across views, which is combined into a 3D pose graph for nonlinear optimization. Existing camera models are leveraged for intrinsic and extrinsic estimates, demonstrating a reduction in mean re-projection error of for stereo calibration from 0.2 pixels to 0.01 pixels when using a traditional checkerboard and the proposed target respectively.

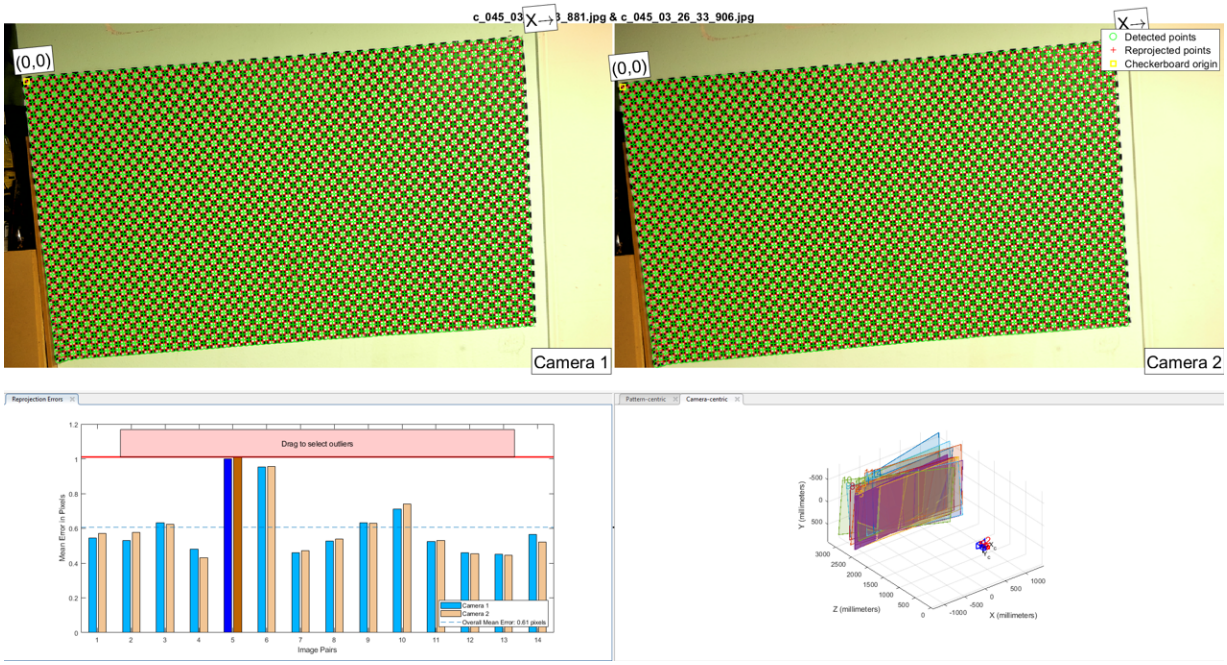

Figure 3.12: Matlab stereo calibration toolbox to calibrate a stereo-pair of a stereo-panoramic array

Geometric camera calibration allows a camera model to be fit to map the geometric transformation of light through the optical elements of a camera system in addition to relative extrinsics in the case of multi-camera systems. Such calibrations enable the mapping of observed world features to image space, often needed in the case of multi-view and geometric computer vision applications. Despite the emergence of auto-calibration methods which allow systems to

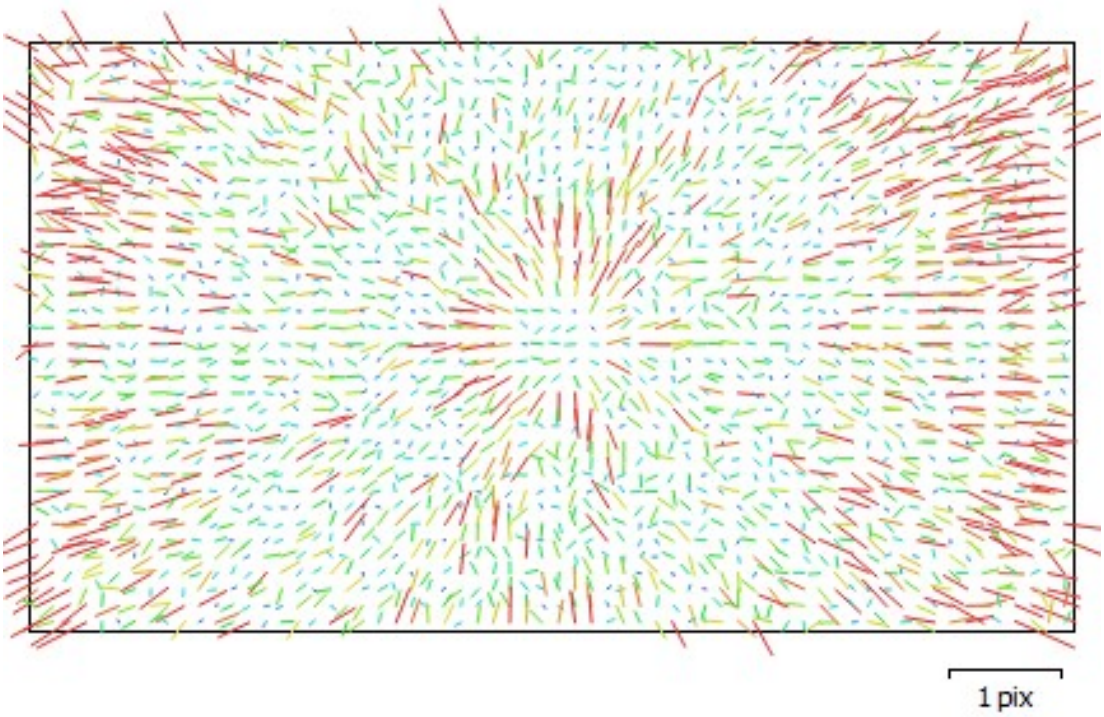

Figure 3.13: Camera re-projection residual averages from an auto-calibrated photogrammetry capture

solve for calibration parameters at the time of deployment, explicit calibration is required for many industrial camera systems with limited computational power, yet real-time estimations needs.

Camera calibration is divided into two major steps, including intrinsic and extrinsic camera parameter estimation. Intrinsic calibrations are widely used across applications requiring physical lens and non-perfect sensor-lens placements to be corrected. Lens distortions, de-centering and focal lengths are the primary variables of such intrinsic calibrations. For extrinsic parameters of a system, it is merely a relative pose $R|t$. Any vision problem involving the mapping or inverse mapping of 3D world features into the image plane is subject to these physical effects that need to be accounted for.

The proposed calibration target addresses a set of limitations associated with current

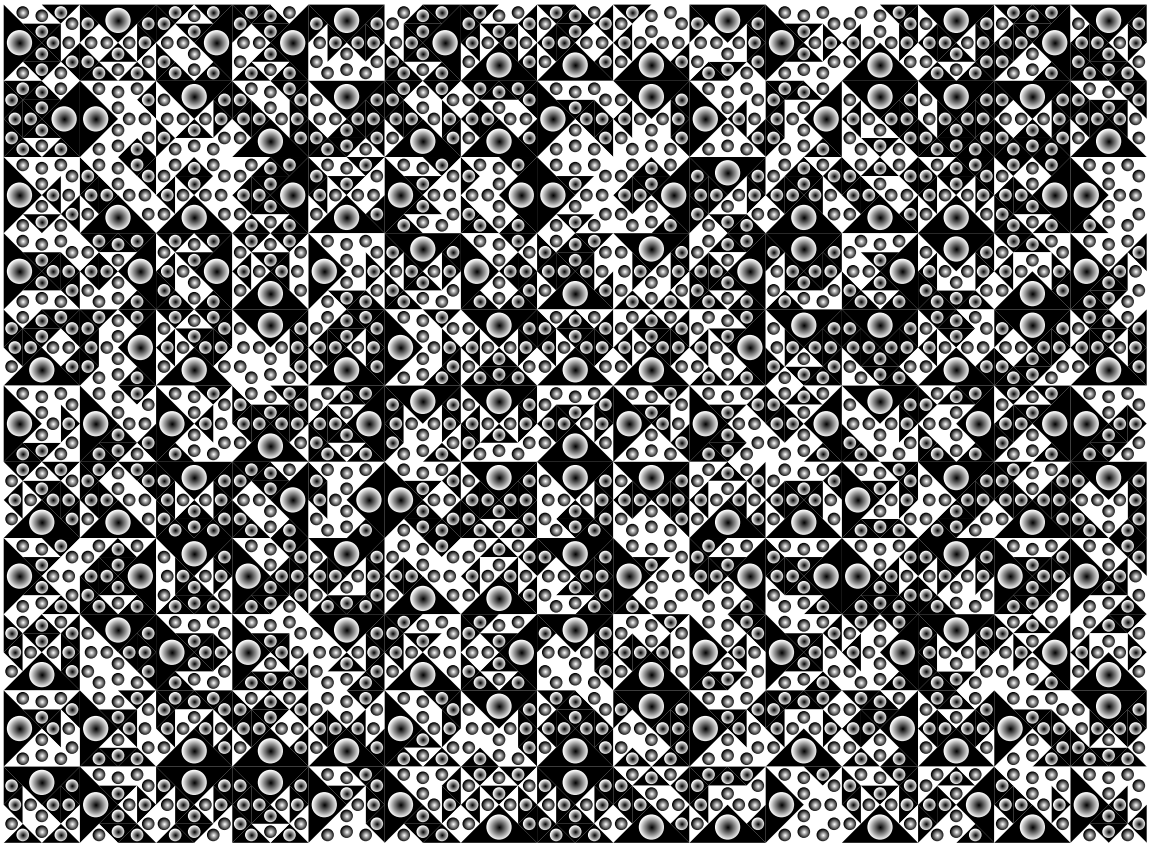

Figure 3.14: Proposed encoded calibration target.

calibration methods. Firstly, current targets lack dense features that can be uniquely identified across multiple-views, limiting the image-space feature distribution and hence suffering from larger calibration errors around the edges and corners of cameras. Furthermore, a predominance of strategies leverage local corner features that are subject to localization errors larger than if larger features are used. This work therefore proposes a dense and unique calibration target alongside a feature extraction algorithm, and formulates the optimization problem to obtain camera parameters for improved calibration.

3.4.1 Prior Work

Prior work covers a broad spectrum of approaches to fit a camera model from a calibration target, ranging from 2D planar targets to 3D rigs, and features of varying type. The most common approaches adopted across open-source vision libraries such as OpenCV [BK00] and Matlab use a calibration approach based on a singular planar checkerboard and a traditional corner feature detection algorithm [Zha00]. These detection methods achieve sub-pixel accuracy by solving for the local corner features within a window that best represents convergence of tangential gradient lines [FG87], or the minimum eigenvalue of the local gradient co-variance matrix [Shi94]. Such checkerboard patterns can however only be detected when all inner corners are visible, making acquisition of the pattern features difficult across the full Field of View (FoV) of a single camera, and as such even more challenging for multi-camera systems. Figure 3.15 demonstrates the coverage issues which arises from limited visibility. In the case of multi-view systems, it is often desirable to enforce calibration constraints dependent on the system baseline direction. Checkerboard edges only contain horizontal and vertical edges, which in the case of perpendicular arrays (trinocular systems in an L-configuration, or lightfield systems with both vertical and horizontal distribution of cameras) are insufficient to enforce geometric constraints. Augmenting calibration targets with diagonal edges and circular features, enables such constraints to be further increased.

Circular features have proven to improve localization accuracy due to their fitting from a larger neighborhood [Hei00]. While the improved feature localization can improve geometric calibrations, overall calibration quality is still subject to having full FoV coverage of features to ensure the calibration accounts for corner aberrations in a set of observations. In the case of circular patterns without unique encoding, these still require all features to be visible, again introducing the visibility limitation shown in Figure 3.15.

Alternatively, a fractal target encoding local square patches of decreasing size has been used [SDG+16], with the work emphasizing the importance of sampling density and uniqueness

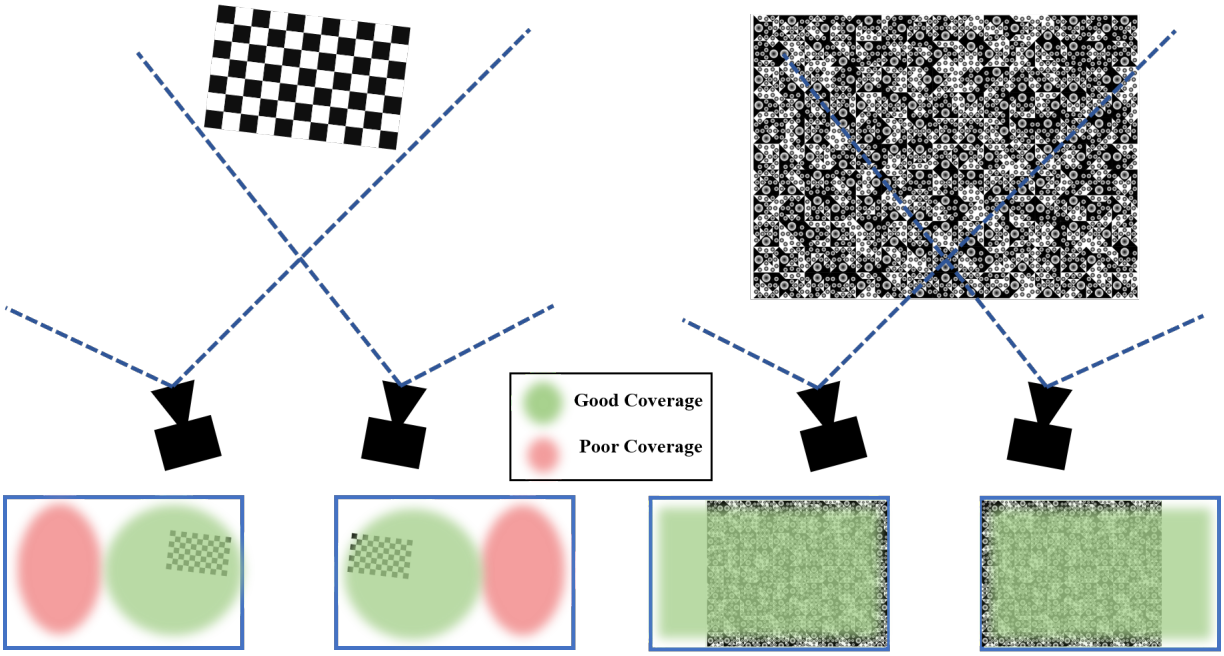

Figure 3.15: Calibration target visibility with partially overlapping stereo field of views. Left- traditional checkerboard target requiring full observations for detection. Right- proposed target with unique local feature patches allowing for partial visibility.

to enable feature samples across the full camera FoV to achieve improved calibration. Despite the coverage improvement from this target, the corner feature localization is still limited to the accuracy of corner detection methods.

3.5 Calibration Target

This paper proposes a calibration target (Figure 3.14) designed to address limitations of current calibration patterns and methods. Specifically it features uniquely encoded patches that allow partial visibility and enable complete sampling across camera FoVs. The target leverages diverse straight edges and circular features to improve the localization accuracy of feature fitting and enables multi-view constraint enforcement.

Each target is divided into a set of arbitrary number of user-defined patches with a spatial sizing chosen to match camera resolution and FoV for reliabale detection. Target dimensions

should be selected such that the size of the smallest circle features observed by a given camera system results in reliable feature detection. The square patches as shown in Figure 3.16 consist of large triangle-circle pairs that anchor the patch and enforce rotational in-variance. The rest of the patch is composited with smaller triangle sets with circles of opposite gradient. The encoding for each patch is achieved using this set of twelve smaller triangles and circles where the gradient direction defines the binary value. This approach supports up to $2^{12} = 4096$ unique patches, containing thirteen circular features each, resulting in up to 53,248 unique circle features.

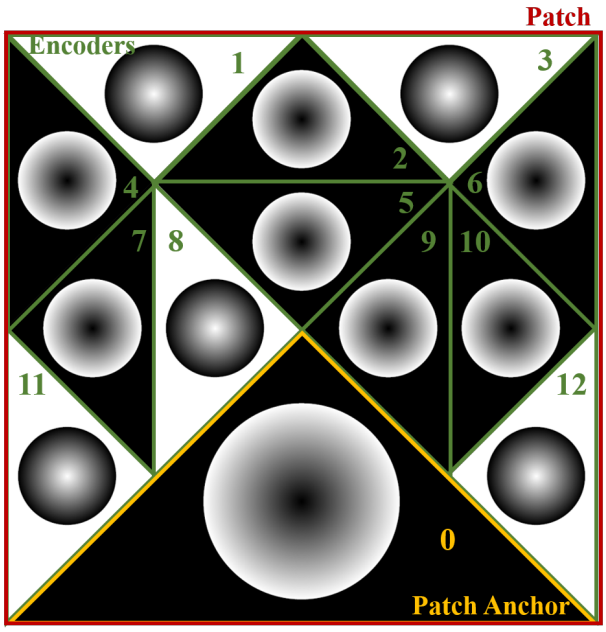

Figure 3.16: Annotated sections of the calibration target patches.

Additional features can be added in a fractal pattern by sub-dividing each small triangle by 4, enabling the spacing of 6 additional features per small triangle as can be seen in Figure 3.17. Since complete patches need to be detected to identify the patch identifier, and with the desired objective to have maximum image coordinate coverage of features, it is preferable to have more smaller patches over increased fractal levels of circular features.

The square patches contain diagonals introduced through the triangular features. These angled edges enable 8 different local gradient orientations (black to white transitions). Addition-

ally the circle edges have 2π angular range forming close contour ellipses when observed. The circular patches contain a continuous spherical gradients that assume a Gaussian distribution. This gradient defines the local orientation of each circle that is used to derive the directions of the peaks and may in the future serve as areas way for a secondary peak fitting method [WWW+18].

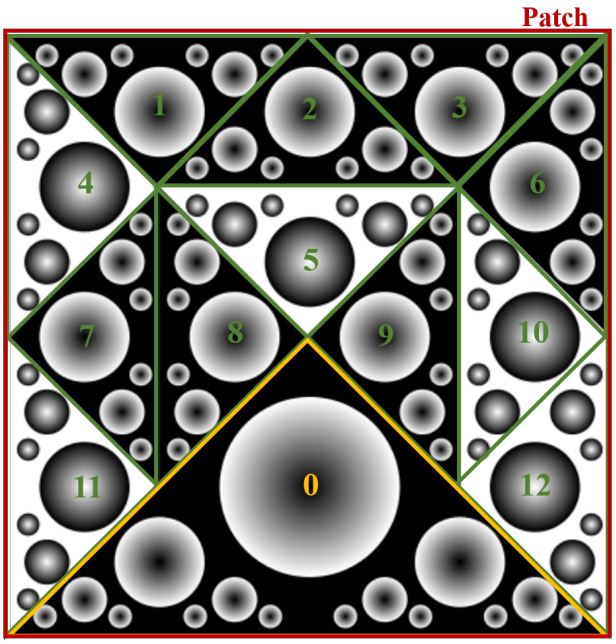

Figure 3.17: Example of a unique patch for a 3-level fractal target.

To ensure that the features are uniformly distributed with both white and black peaks and edges of all directions, the patch identifiers are randomly sampled without replacement. Furthermore each patch is randomly rotated *pi* radians to ensure a random distribution of large circles across the target. At the time of detection, this orientation is recovered from the large anchor circle and set of locations of smaller circles.

3.6 Detection and Calibration

The circular features in the proposed calibration target and patch encoding scheme require a detection pipeline which leverages high sub-pixel localization accuracy to achieve reliable

calibrations. We begin by an edge detection, followed by feature linkage to separate straight lines from ellipses. Once all features have been fit, the patches are recognized from the feature locations and identifiers to provide a dictionary of feature-coordinate pairs used by the model fitting within the calibration. The full pipeline is depicted in Figure 3.18 and will be discussed within this section.

3.6.1 Feature Detection

The importance of feature localization accuracy on calibration quality imposes the requirement for good edge fitting, in turn used for the ellipse fitting and centroid finding. Sub-pixel edge algorithms can use first and second order image gradients, with the first requiring additional optimization to find local ridges and the latter explicitly being defined as the roots of second order gradients. We use the steerable filtering approach introduced by [FA91] with user-tunable Gaussian kernel sizes to locally steer the second order derivatives to the locally dominant orientation. This provides an image gradient and an orientation image which is then used for solving for the real roots of the bi-linear surface yielding continuous edges. Edges are linked to provide edge point groups which belong to continuous feature contours as shown in Figure 3.19. With the help of the feature fitting described in the next step, these are then separated into ellipses and lines, rejecting any remaining outliers that do not meet fitting requirements and a minimum contour

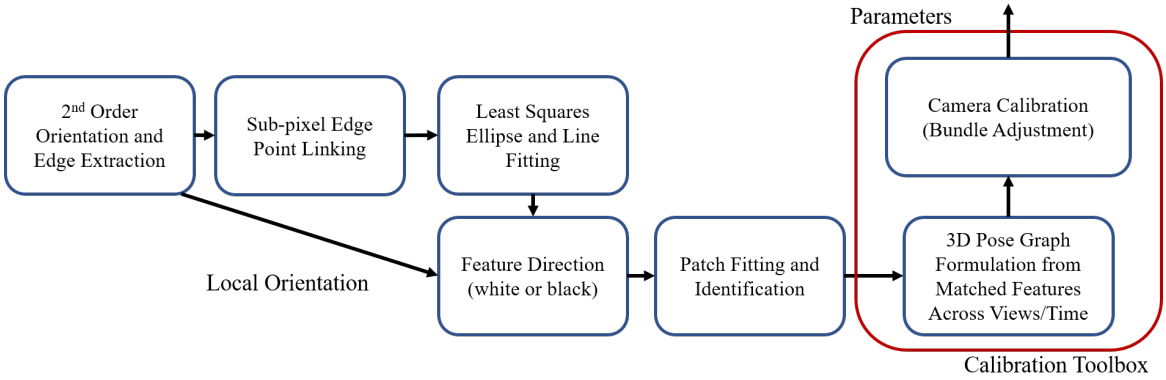

Figure 3.18: Target detection pipeline.

length.

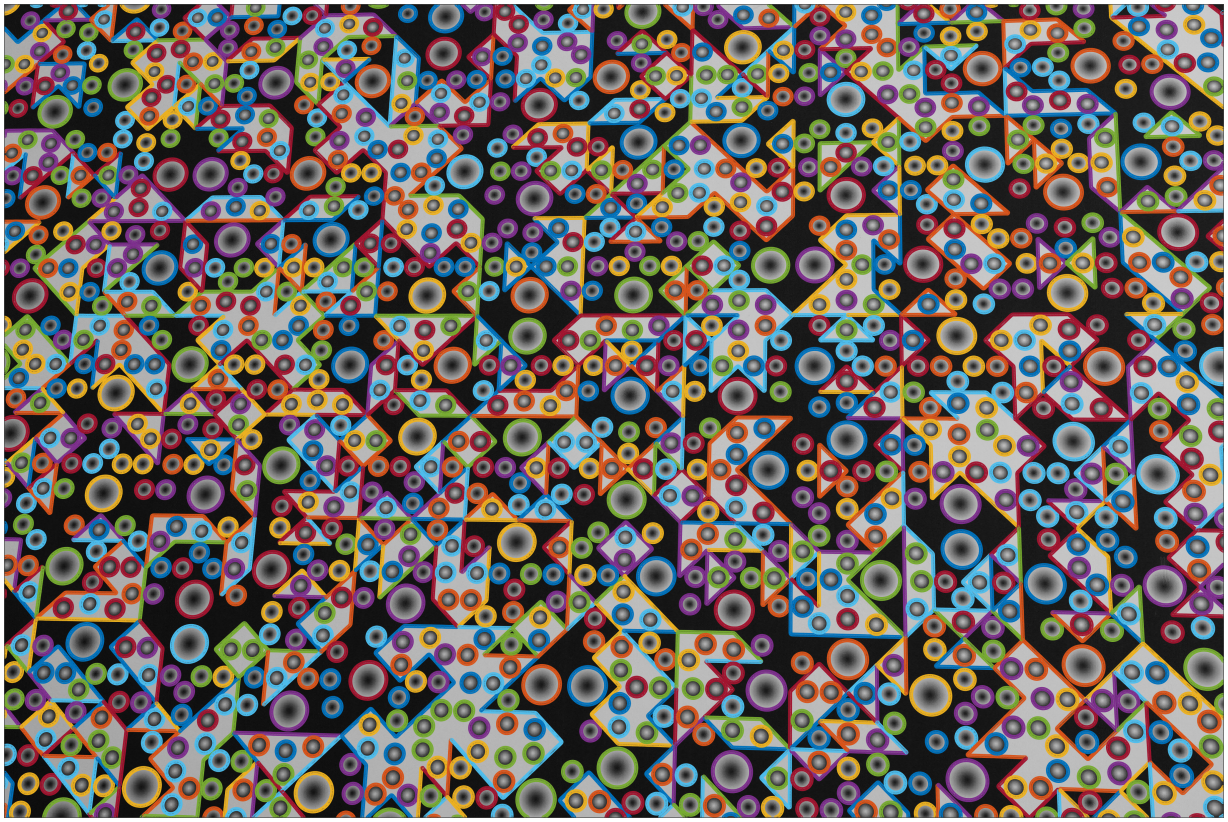

Figure 3.19: Proposed calibration target edge point clustering.

The target uses circles as its primary features, which when subject to a projective transformation assume elliptical geometries [Hei00]. While locally non-linear lens distortions may affect that assumption, it holds true as long as projected circles are small with respect to the overall lens distortions. The ellipse fitting is done using a direct least squares approach proposed in [FPF96]. All contributing points within a linked contour are used for the fitting, and a secondary pass is used to merge disjoint ellipse segments for an optimized fitting. All fit ellipses have a major and minor axis diameter, eccentricity, with a center point and a fitting quality metric (mean point to ellipse distance from edge points). These metrics are used to validate only reliable ellipse fits, and the diameter is used to separate large from small ellipses with an Otsu threshold biased by the ratio of large to small circles.

The edge gradient directions of the ellipses are used to extract whether the ellipse is white or black on an oppositely shaded triangle. This direction is identified from the ellipse curvature normal direction and the locally dominant orientation from the image filtering. If they are in agreement, both vectors point towards the ellipse center implying a white peak, and if they have opposite directions, they imply a black peak.

3.6.2 Target Extraction

To implicitly match features across image views for the calibration, a direct identifier is associated to each feature, stemming from the patch and identifier they lie in. These patches are extracted from the local distribution of ellipse centers, which are referenced from the large ellipses. Each patch is extracted only if all ellipses within the patch are confirmed valid. The features are

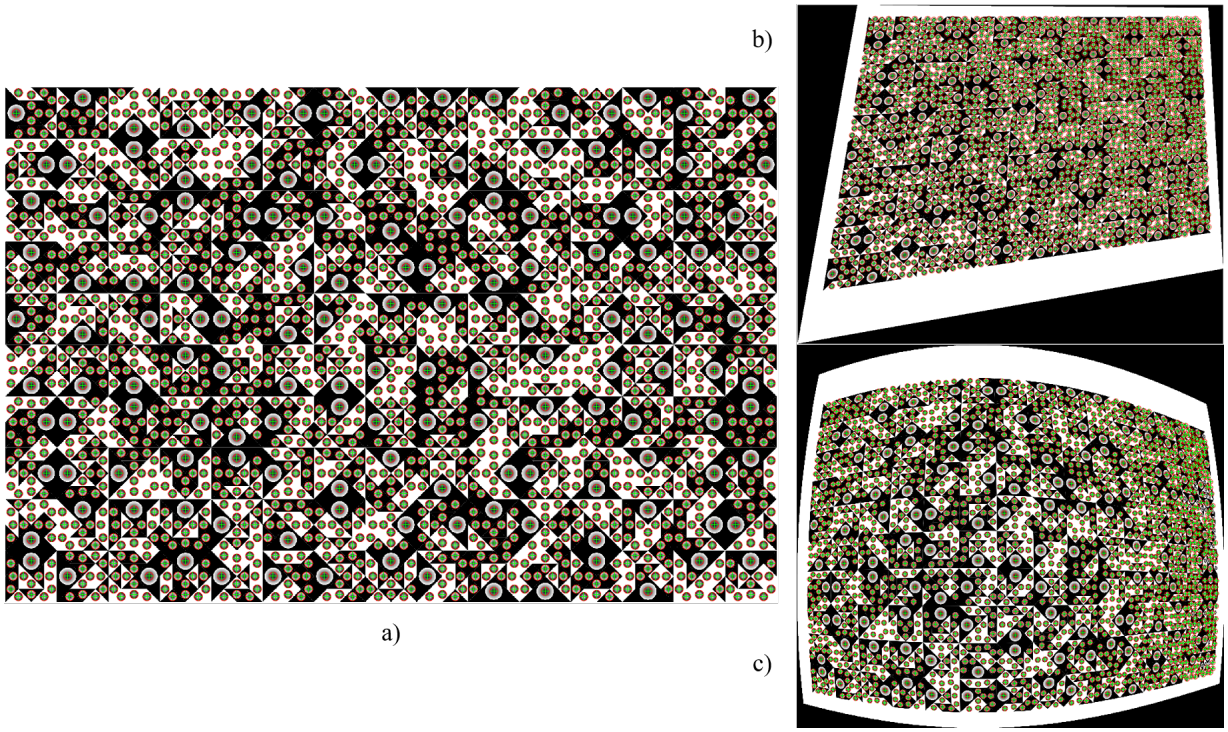

Figure 3.20: Evaluating localization accuracy of ground truth fractal target. The target circles are detected in the original (a), target subject to projective distortion (b) and target subject to radial distortion (c). Detected features are shown as green crosses in red circles, overlayed on the target.

linked to the 3D world points from the target plane by associating the feature image coordinates and world coordinates via their identifiers. A complete extraction of patches is demonstrated in Figure 3.20a.

3.6.3 Model Fitting & Calibration

We leverage the located and matched correspondences to formulate the optimization problem as a bundle adjustment problem: given the 3D points and the observed 2D locations in each view, simultaneously optimize for the camera parameters (intrinsics and distortion parameters) and the pose at each view such that the reprojection error is minimized. This bundle adjustment problem can be solved using a variety of non-linear optimization strategies, with the Levenberg-Marquardt optimization being commonly chosen to combine the convergence benefits of both Gauss-Newton and gradient descent [Zis04]. For this problem, we utilize OpenCV's implementation of the Levenberg-Marquardt based calibration to optimize over intrinsics and extrinsics.

Based on empirical evidence, it has been observed that it is beneficial to perform the calibration process in two stages: In the first stage, for each camera separately, we estimate and optimize the intrinsic parameters and pose at each view, minimizing the reprojection error computed by projecting the world points to image space using the intrinsics and pose estimates. Then, in the second stage, we use these intrinsics estimates to optimize for the relative extrinsics between the left and right camera, by estimating and optimizing the pose of each view in both cameras so that the relative pose between left and right camera remains constant, and the reprojection error using these pose estimates and the fixed intrinsics is minimized. It has been observed that separating the intrinsic and extrinsic calibration optimizations ensures convergence to a better optima.

3.7 Calibration Results

The results are divided into two sets of experiments with separate objectives: validating the localization accuracy of elliptical features and evaluating the camera calibration performance for monocular and stereo camera calibrations. We summarize these as:

- Sub-pixel localization accuracy of spatial images features given synthetically rendered images with varying degrees of geometric distortions, to demonstrate target and algorithmic reliability at extracting correct feature locations.
- Demonstrable and real-world results for the intrinsic and pose estimation of a stereo camera moved around a checkerboard and fractal target. Evaluating the reprojection error residuals across the images using a) only a monocular approach for intrinsic calibration and b) a stereo calibration to solve the relative pose using the monocular estimated intrinsics.

3.7.1 Localization of Features in Synthetic Targets

For the first experiment, we generated a synthetic fractal target and its ground truth sub-pixel level feature locations. To compare the feature localization accuracy, we subjected the target, and the ground truth feature locations, to different kinds of distortions: projective homographies (to yield targets similar to Figure 3.20b), radial and tangential distortions (to yield targets similar to Figure 3.20c), and then ran our feature extractor on the warped targets.

For projective distortions, we create a projective transformation matrix of the form:

$$T = \begin{bmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

and varied θ between -15 and 15 degrees.

For radial distortions, we considered the 2-variable model for radial distortion:

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{3.3}$$

$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{3.4}$$

$$r^2 = x^2 + y^2 \tag{3.5}$$

Where x, y are the undistorted normalized image coordinates. For our experiment, we set $k_1 = k, k_2 = k^2, k_3 = 0$, where k varied logarithmically between $1e-5$, and 0.33.

For each of the tests, we compute the ground truth feature locations in the warped image by applying the same transformations to the ground truth feature locations, and computing the mean L2 distance between the detected feature locations and the ground truth locations.

We present both qualitative and quantitative comparisons to demonstrate the performance of the feature extractor. Figures 3.20b and 3.20c show that our model can accurately detect most

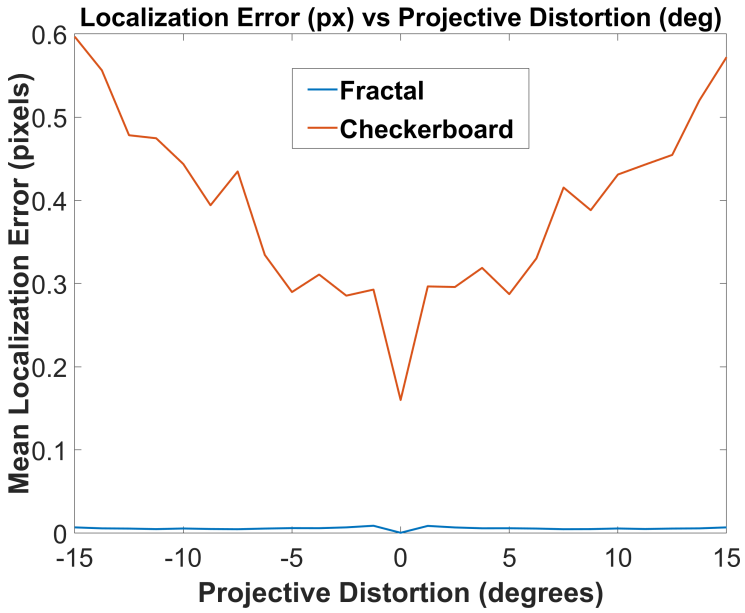

Figure 3.21: Feature localization error (in pixels) as a function of the image distortion, on our target vs min-eigen feature detector on checkerboard.

of the features seen in the ground truth target image (Figure 3.20a, even under large distortions. Quantitative results are shown in Figures 3.21 and 3.22, where we compute the mean feature localization error vs distortion magnitude.

We observe that our feature extractor can localize features to accuracies greatly smaller than 0.1 pixels even under extreme image distortions while the traditional corner extractor for the checkerboard only reached accuracies in minimal distortions conditions of 0.15 pixels with a rapidly degrading localization when either projective and radial distortions were increased.

3.7.2 Real-world Camera Calibration

The second set of experiments evaluated real-world performance of the proposed calibration routine. For data capture, we used dual TRI120S Lucid machine vision cameras (Sony IMX304 sensor) with fixed 12mm Computar lenses (V1226-MPZ), shot at an aperture of f/5.6. The acquisitions were separated into 3 datasets, each with 50 frames- one with the checkerboard (see Figure 3.23a), one with the fractal calibration target fully seen in all views (see Figure 3.23b),

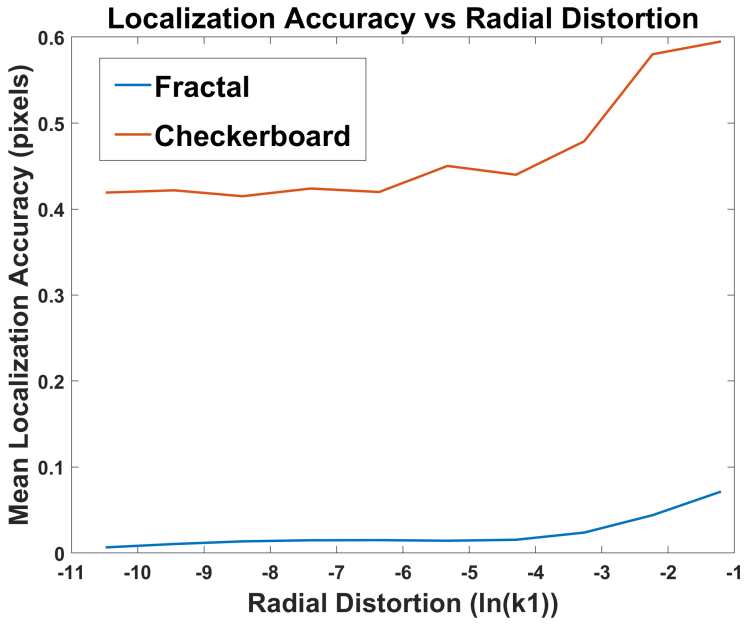

Figure 3.22: Feature Localization error (in pixels) as a function of the radial distortion, on our target vs min-eigen feature detector on checkerboard.

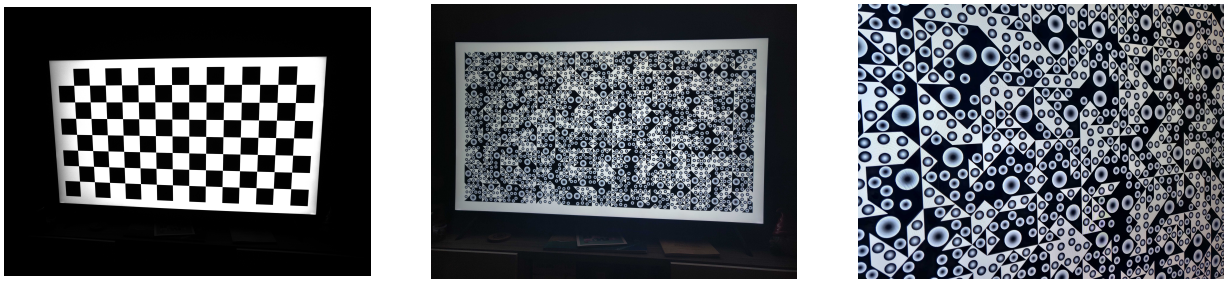

Figure 3.23: Sample calibration frames used to evaluate the checkerboard (left) and fractal (center- full, occluded- right) calibration targets.

and one with only a partial/occluded views of the fractal target (see Figure 3.23c). The targets were displayed using a 65 inch 4k TV with an aspect ratio of 16:9.

For each of the target frames, we extracted the features with their identifiers and matched them across different views to establish feature point correspondences. The checkerboard ones leveraged the OpenCV pipeline with the *findChessboardCorners* and *cornerSubPix* functions for feature extraction (these use the algorithm proposed by [Shi94]), while the fractal target ones used the proposed feature extractor from the previous section. Then, using OpenCV's *calibrateCamera* function, we estimated the intrinsic and extrinsic camera parameters for the cameras.

All evaluations were done with the True Pixel Error (TPE) metric, which compares the reprojected world points with the 2D image features given an estimated camera pose. This metric is used across multi-view bundle adjustment problems such as photogrammetry, and is considered a good measure of consistency for pose estimation and feature localization [FW16].

Monocular and Stereo Calibration

The monocular calibration considered the left and right cameras of the stereo captures independently, with the results presented in Table 3.24.

We pose the stereo camera calibration as a non-linear optimization of the camera intrinsics, extrinsics, and the relative poses between the left and right stereo cameras given the 3D world coordinates of the feature points and the 2D image coordinates of the features seen in each frame

Target		Avg points/frame	Monocular						Stereo		
			Left			Right					
			RMSE	μ	σ	RMSE	μ	σ	RMSE	μ	σ
Checkerboard		98	0.288	0.255	0.135	0.314	0.279	0.144	0.302	0.267	0.140
Fractal	All points	1017.4	0.242	0.191	0.149	0.255	0.200	0.159	0.249	0.196	0.154
	Subset	98	**0.059**	**0.049**	**0.032**	**0.059**	**0.049**	0.033	**0.059**	**0.049**	**0.033**
Fractal Occluded	All points	500.0	0.232	0.180	0.146	0.251	0.194	0.159	0.242	0.187	0.153
	Subset	98	0.065	0.053	0.036	0.060	0.050	**0.032**	0.062	0.051	0.034

Figure 3.24: Results table for the mean re-projection error (pixels) of checkerboard and fractal target calibrations in both monocular and stereo configurations.

by both cameras. For an efficient and accurate implementation, we first leverage the intrinsic parameters for both the left and right cameras using OpenCV's *calibrateCamera* previously calculated in a monocular seeting, and utilize these estimates to optimize for the extrinsics using a Levenberg-Marquardt based non-linear optimization, using OpenCV's *stereoCalibrate* function.

Both monocular and stereo calibrations are evaluated using the checkerboard and fractal targets. The mean reprojection error is calculated for each 3D world point back projected into image coordinates from observed views. The statistical distributions of the mean reprojection errors is summarized in Table 3.24 and the errors are plotted over the image coordinates in Figure 3.25.

We validate the coverage of the respective targets from the various calibration views in Figure 3.25. The plots include the image coverage of points across the full dataset of 50 frames for both the checkerboard and fractal data. The shear difference in number of points indicates that the fractal target better samples the space, providing improved data for the calibration. The color magnitude shows the reprojection error for the estimated points. Despite the significantly improved coverage with the proposed fractal target, there remain small areas in the corners that lack points. This is likely due to the patches in those areas not being fully detected, eliminating them from being used for the calibration.

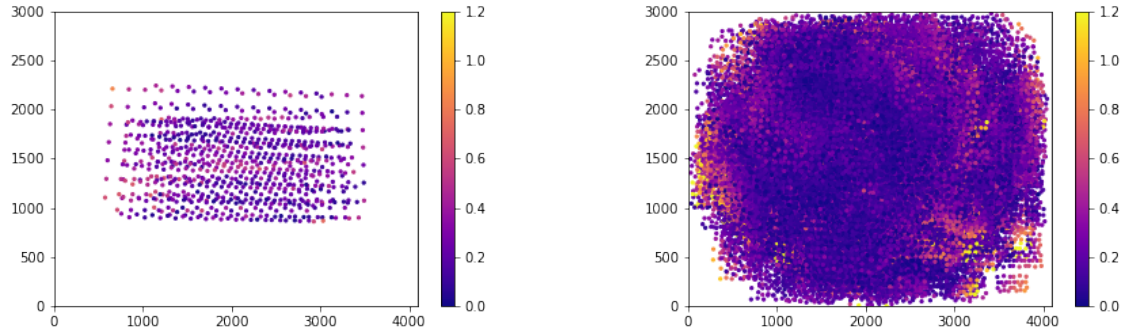

Figure 3.25: Image coverage of points detected from Checkerboard (left) and Fractal target (right) over accumulated 50 frames. Color magnitude represents the reprojection error (pixels).

3.8 Discussion

The previously introduced re-projection error is an important metric to validate geometric camera calibration. During the calibration optimization, pose and intrinsic parameters are adjusted to minimize this error resulting in the camera parameters and the respective error for them. Across the photogrammetry community, this error is widely used to evaluate to global consistency and quality of the reconstruction [FW16]- a lower error indicates improved reconstruction and camera localization. In the case of calibration, we can similarly use it to evaluate the calibration performance. Since this error is often on the order of sub-pixel values, it is necessary to verify both that the localized image features are accurately localized before evaluating the calibration. The first set of experiments that tested the effect of projective (Figure 3.20b) and radial distortions (Figure 3.20c) on the localization accuracy highlighted that traditional corner localization techniques only approach accuracies of around 0.2 pixels at best. The ellipse based detections from this work are able to reach localization accuracies nearly an order of magnitude better, localizing centers to around 0.01 pixels. Furthermore, the results suggest that increasing the magnitude of geometric distortions affects both localization performances of corner points and elliptical features, with the latter remaining more robust over greater distortions. It is likely that extreme distortions which would yield worse localizations are rejected through the feature quality

metric of elliptical fitting, a particular advantage over corner features that are more difficult to validate. These results conclude that the localization performance of elliptical features is greatly superior over corner features.

The second component of calibration evaluation focuses on the calibration quality which were evaluated separately on monocular and stereo calibrations in a real-world setting. From the results, we note that:

- Our feature extractor allows us to set up better feature point correspondences across different frames, which gives us a significantly lower mean reprojection error as compared to the standard checkerboard based calibrator (0.06 pixels vs 0.3 pixels).

- The uniquely identifiable patches allow the target to be partially visible, providing easier sampling of feature points across corners and edges of the FoV. Figure 3.25 shows the feature distribution for the checkerboard and fractal target respectively.

- Despite the improved and validated feature localization, there remains larger reprojection errors across corner and edge segments of the image coordinates, indicative of an outstanding systematic error in the camera model fitting. Additional radial parameters and local surface deformations may be a future improvement to address the calibration performance.

The overall reduction in this error is indicative of improved overall calibration since there is more local agreement to the solution. While the solution is limited to numerical optimization methods such as the Levenberg-Marquardt, there may be cases where the converged minimum is a solution that is not indicative to ideal geometry.

3.9 Conclusion

This paper introduced a fractal calibration target that uniquely encodes circular features for robust multi-view camera systems. Elliptical feature detection is used to improve the localization

accuracy over traditional corner features. Projective and radial distortion experiments confirmed the ability of the detector to localize to 0.01 pixels, improving from around 0.2 pixel localization accuracy in traditional corner detection methods. Monocular and stereo calibrations were tested using a proposed fractal target, improving the reprojection errors compared to a checkerboard calibration. Despite the improvements in calibration performance, systematic errors indicate the limitation of existing camera models for geometric calibration.

3.10 Acknowledgements

This chapter is, in part, a reprint of the material as it appears, titled "A Survey of Dense 3D Reconstruction Methods for Multi-Sensor Embedded Vision" in UC San Diego Research Exam 2019. Meyer, Dominique. The dissertation author was the primary investigator and author of this material.

This chapter is, in part, a reprint of the material as it appears, titled "StarCAM - A 16K stereo panoramic video camera with a novel parallel interleaved arrangement of sensors" in Electronic Imaging 2019. Meyer, Dominique Ernest; Lo, Eric; McFarland, Chris; Dawe, Gregory; Dai, Ji; Nguyen, Truong; DeFanti, Thomas; Wang, Haoyu; Sandin, Dan; Brown, Maxine; Kuester, Falko. The dissertation author was the primary investigator and author of this material.

This chapter is, in part, a reprint of the material as it appears, titled "Omniscopic Vision for Robotic Control" in IEEE Aerospace 2019. Meyer, Dominique Ernest; Lo, Eric; McFarland, Chris; Strawson, James; Drohobytsky, Danylo; Dawe, Gregory; Dai, Ji; Nguyen, Truong; DeFanti, Thomas; Wang, Haoyu; Sandin, Dan; Brown, Maxine; Kuester, Falko. The dissertation author was the primary researcher and author of this material. © 2019 IEEE. Reprinted, with permission, from [Meyer, Dominique Ernest; Lo, Eric; McFarland, Chris; Strawson, James; Drohobytsky, Danylo; Dawe, Gregory; Dai, Ji; Nguyen, Truong; DeFanti, Thomas; Wang, Haoyu; Sandin, Dan; Brown, Maxine; Kuester, Falko. "Omniscopic Vision for Robotic Control" in IEEE Aerospace 2019]

This chapter is, in part, currently being prepared for submission for publication of the material, titled "3D Multi-Camera Calibration with a Fractal Encoded Multi-Shape Target" in Electronic Imaging 2021. Meyer, Dominique Ernest; Verma, Pranav; Kuester, Falko. The dissertation author was the primary researcher and author of this material.

Chapter 4

3D Estimation and Localization

4.1 Feature Detection and Localization

At the core of all multi-view reconstruction algorithms lies the matching of image features across views or time. In this section, an introduction of a novel semi-dense feature space will be introduced, and compared to fully sparse and dense approaches. In the previous geometric section, we have introduced the relationship of disparity and depth for a given system. While the disparity is simply a distance between a feature of one image projected into another, and the matched feature of that other image, this disparity turns out to be highly influential on the derivative depth. Let us take a corner feature, that we localize to within 0.25px in both images, that gives us a disparity accuracy of $\pm 0.5 px$. This accuracy for an object at 10m distance from a stereo camera will yield a depth accuracy on the sub-meter order. Consider now a world point 50m in away, the same disparity accuracy will yield a depth accuracies of many meters. The accuracy relationship can be summarized in equation 4.2, and is plotted in figure 4.1. We can therefore conclude that for accurately triangulating points, we need to maximize our localization accuracy in image space, such as to minimize the disparity error, and therefore our depth error.

$$Z = f \cdot \frac{b}{d} \tag{4.1}$$

$$\delta Z = f \cdot (\frac{b}{d} + \frac{b}{d \pm \delta d}) \tag{4.2}$$

4.1.1 Feature Purpose

Features are chosen to be uniquely identifiable so that they can be matched across views and represent an estimation of the observed world. This opens the question of what features *best* represent the scene- points, lines, curves, surfaces or volumes? In the design of 3D multi-camera systems, we must choose our derivative representation to maximize our value in the data and with it, the quality of that data. In light of uniqueness and representation, semi-dense edge features provide a unique set of advantages over sparse corners and dense patch features:

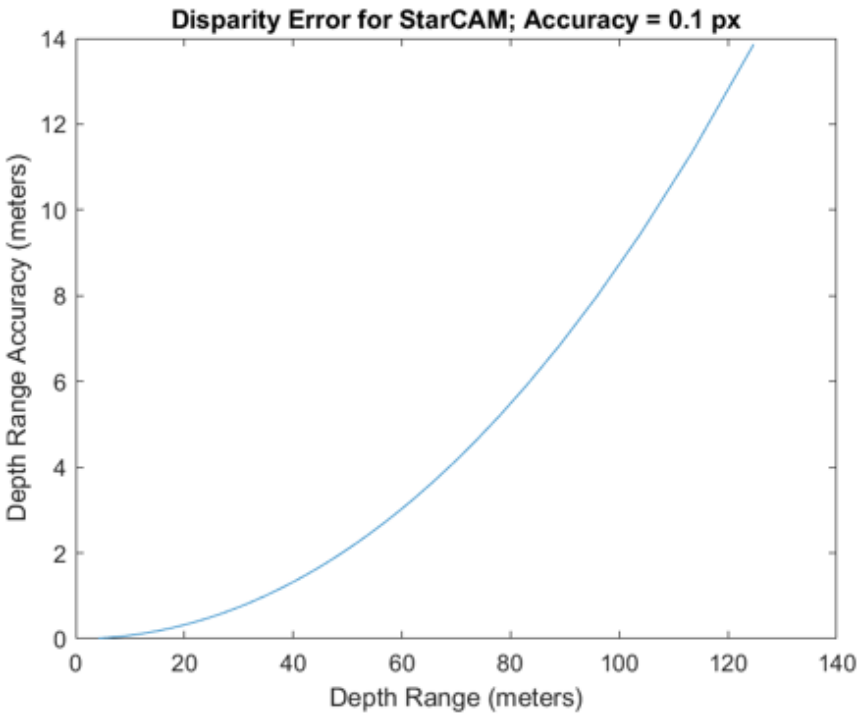

Figure 4.1: The effect of disparity accuracy on depth accuracy assuming a 75mm stereo pair with a 0.5 disparity accuracy

1. They are well localizable due to areas of high gradient.

2. They often form a perimeter of geometries in an observed scene, providing the strongest constraint on what that geometry is.

3. They are sparse enough to be handled scalably, while being dense enough to form a reasonable representation of the scene.

4. They form continuities by associativity in image space, as well as over-time.

4.1.2 Subpixel edge localization

To extract edges, we must find areas of high-gradient, and fit the exact 2d line/curve where this biggest change in the image happens. A number of first order derivative approaches have been proposed, including the popular Canny detector [Can86]. The challenge with first order methods, is that fitting splines to ridges is numerically more difficult than solving for the roots of second order derivatives. Figure 4.2 highlights the difference in edge behavior between original images, 1st order and 2nd order images. One can see that in the original image, it is hard to localize the exact location of the edge. In the 1st order derivative case, an optimization must be completed to fit a peak to the signal and consequentially, localize the maxima. Finally, the second order derivative clearly shows that the zero-crossing is at the point of maximum gradient, or the point in the image where the edge is located. As will be discussed, this root can be solved for analytically using a least squares solution rather than being optimized over, reducing computational complexity.

Image Derivatives

When taking image gradients, they have to be oriented to the local edge direction. The common approach to estimate image derivatives is through a convolution with a set of gradient kernels. Creating a kernel for each orientation is unrealistic, but Freeman [FA91] has shown

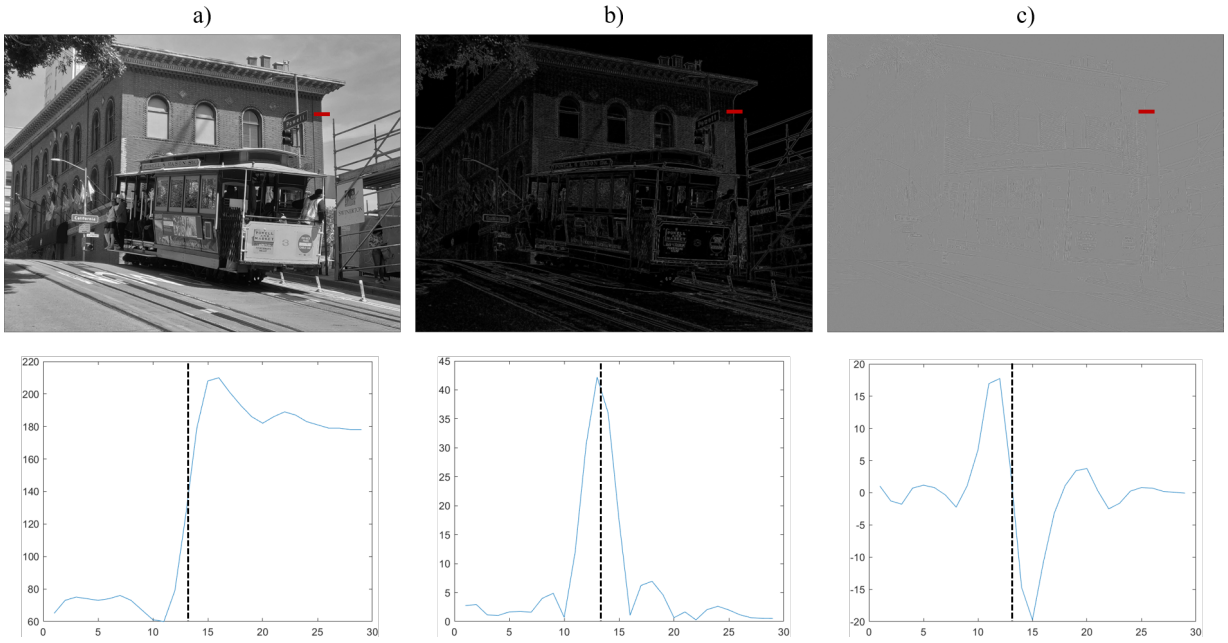

Figure 4.2: Comparing edges of a) original image, b) 1st order derivative and 2) 2nd order derivative. The second row plots the 1D samples of red lines depicted in the images.

that using of a set of basis filters to retrospectively steer the filtered response to any input kernel orientation, is feasible. This section leverages Freeman's steerable filter to derive both image gradients steered to the locally dominant orientation, and the image roots of these second order derivatives.

A set of 2D basis filters are designed using Gaussian derivatives with a standard deviation σ and are plotted in Figure 4.3. The larger σ, the more smoothing that takses place, reducing the filter sensitivity to noise. The input images are convolved with these basis filters and used to reconstruct the dominant orientation and filters

The locally dominant orientation is extracted from the second order basis functions and their respective Hilbert functions (used to offset DC component of the complex signals):

$$\theta_d = \frac{arg[C_2, C_3]}{2} \tag{4.3}$$

where $C_1, C_2, C_3 ...$ are the constants in the Fourier series of the oriented energy $E(\theta)$:

$$E_{G_2H_2}(\theta) = C_1 + C_2 cos(2\theta) + C_3 sin*(2\theta) + ... \tag{4.4}$$

In addition to directly obtaining the dominant orientation from the basis functions, we are also able to extract an orientation strength:

$$S = \sqrt{C_2^2 + C_3^2} \tag{4.5}$$

Finally, to obtain the 2nd order derivative image, we steer the 2nd order Gaussian to the locally dominant direction θ_d:

$$G_2^{\theta_d} = k_1(\theta_d)G_{2a} + k_2(\theta_d)G_{2b} + k_3(\theta_d)G_{2c} \tag{4.6}$$

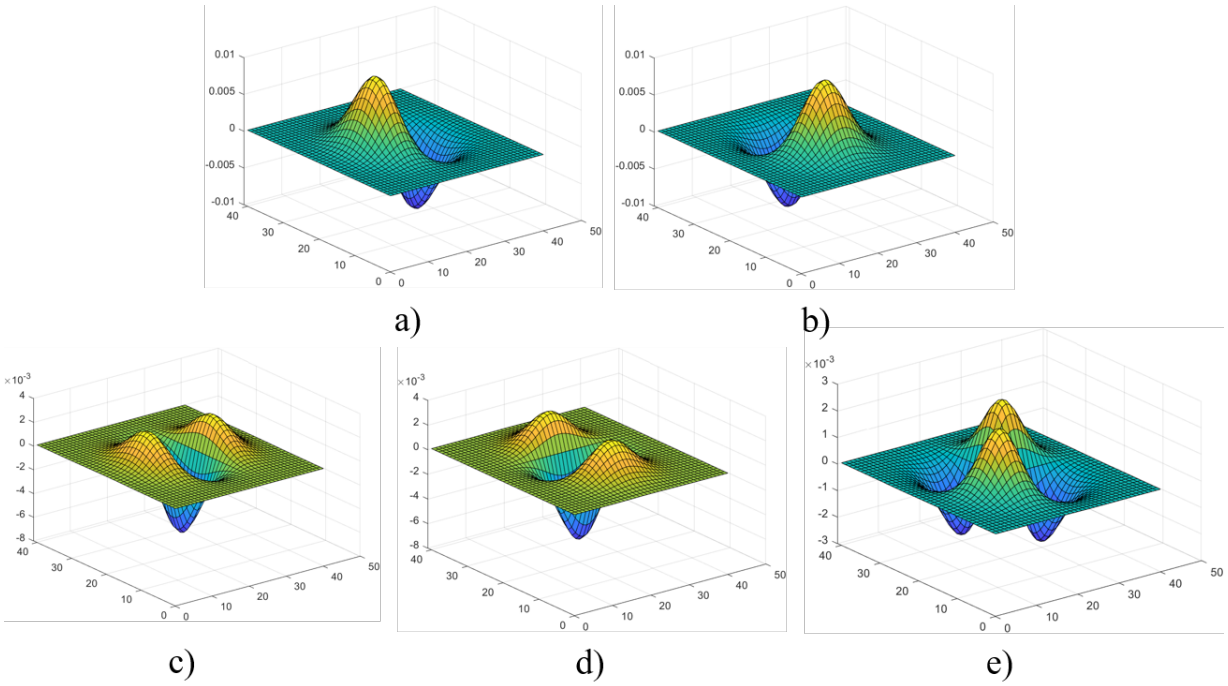

Figure 4.3: A set of basis filters used to convolve the input images to estimate the 1st and 2nd order gradients steered to locally dominant directions: a) G_{1a}, b) G_{1b}, c) G_{2a}, d) G_{2b}, e) G_{2c}

Root Finding

To find the roots of the image second derivatives, we consider a local interpolation which for simplicity is kept to bi-linear interpolation. While a cubic interpolation improves smoothing and reduces artifacting of the interpolation, it would require a 4x4 pixel window rather than a 2x2 for bi-linear interpolation. As such we leverage the linear intersection in both dimensions, in line of the pixel center values, which yields a non-linear surface within it. A linear interpolation with its root of the 2nd order of an image patch is shown in Figure 4.4.

For each unit square $w(i+0.5, j+0.5)$, cast by the four adjacent pixels, we solve the bi-linear system 4.7 of equations for the roots 4.9.

$$f(x,y) = a_0 + a_1 x + a_2 y + a_3 xy \tag{4.7}$$

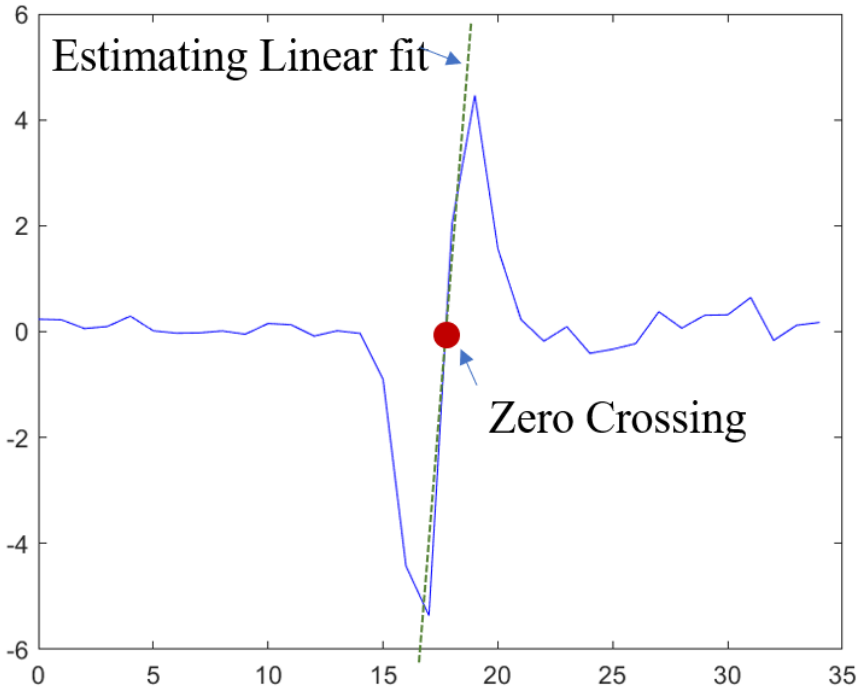

Figure 4.4: 1D Image root/ zero-crossing of second derivative signal

$$x = \frac{-a_0 - a_2 y}{a_1 + a_3 y} \tag{4.8}$$

The root of the image is guaranteed to be continuous to adjacent pixels if they share the same edge crossing of a set of 4 pixels, forming a set of roots that form a contour. The set of edges extracted from these image derivatives are shown in Figure 4.5 and 4.6.

Finally, this method also allows one to extract the curvature K and respective gradient K' of each root:

$$K = \frac{2a_3(a_1a_2 - a_3a_0)(a_2 + a_3x)^3}{((a_2 + a_3x)^4 + (a_3a_0 - a_1a_2)^2)^{3/2}} \tag{4.9}$$

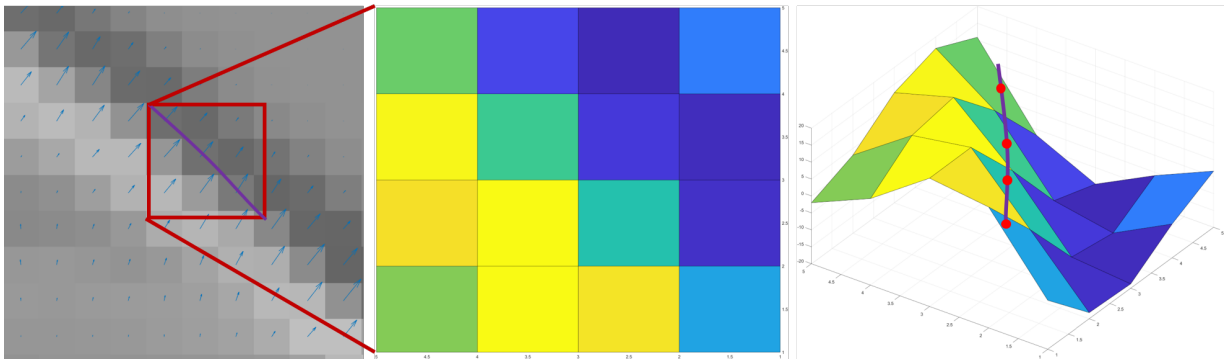

Figure 4.5: The location of an image root on 2nd order derivatives

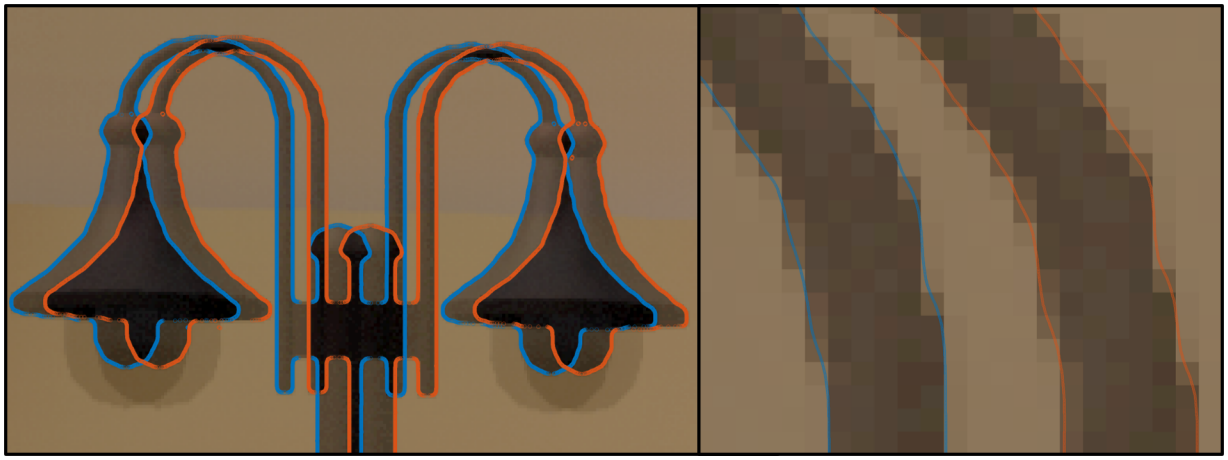

Figure 4.6: Image roots on a synthetic stereo light post image, localized to sub-pixel accuracy

The roots of the derivative of the curvature 4.10 are validated to be within the local window range $x \in [i : i+1], y \in [j : j+1]$ to guarantee function correctness. The window maximum curvature as the point along the contour with highest rate in change of local orientation, may in certain cases be identifiable as a unique corner feature.

$$x = -\frac{c}{d}, \frac{\pm\sqrt{bcd^2 - ad^3} - cd}{d^2}, \frac{\pm\sqrt{ad^3 - bcd^2} - cd}{d^2} \tag{4.10}$$

Most current methods leverage the Marching Cubes approach [LC87] which assumes straight lines within the intersecting areas. Given that this approach achieves to match contours to better sub-pixel accuracy in areas of high curvatures, it provides the benefit of reducing disparity when applied to multi-view methods.

4.2 Matching

Feature matching is used to estimate disparities across stereo viewpoints of the camera system, alongside temporal motion. We decompose the problem to a set of two independent matches- the stereo match of the contours, and the 2D track of the image points.

The computational nature of matching is expensive due to the search space and candidate matches. We therefore employ the epipolar constraint of the calibrated stereo pairs to constrain the 2D search to a 1D search for contour intersections. Since contours are continuous, we find the integer row intercept from the roots of the function that allow us to consider only a handful of match candidates within a maximum disparity range. A Census transform $\varepsilon(p, p') = \{0 \in p > p', 1 \in p \leq p'\}$ on a window centered about the root intersection candidates is obtained to compute the Hamming distance cost $C = \varepsilon_{right_n} - \varepsilon_{left}$. A winner takes it all strategy is used for independent curvelets, however, secondary matches are considered during a continuity enforcement. Since curves are likely to be continuous in disparity, any roots of neighboring curvelets, are re-evaluated to choose the consistent match candidate that minimizes cost and

disparity change within a local neighborhood.

4.3 Triangulation

We triangulate feature depths using the intrinsic focal length f, stereo baseline b and matched stereo disparity d using equation 4.11. Given that the contour disparities are continuous, we have the choice of discretely sampling the contours are vertical image values, or to analytically compute the 3D curvelets from the disparity.

$$Z = f \cdot \frac{b}{d} \tag{4.11}$$

The following equation is the analytical solution to the disparity $D(y)$ between two contour curvelet solutions ($a \rightarrow left, b \rightarrow right$). It is important to note as seen in Figure 4.7 that the valid range of the disparity is defined by the valid range of the curvelet. In the case where the left or

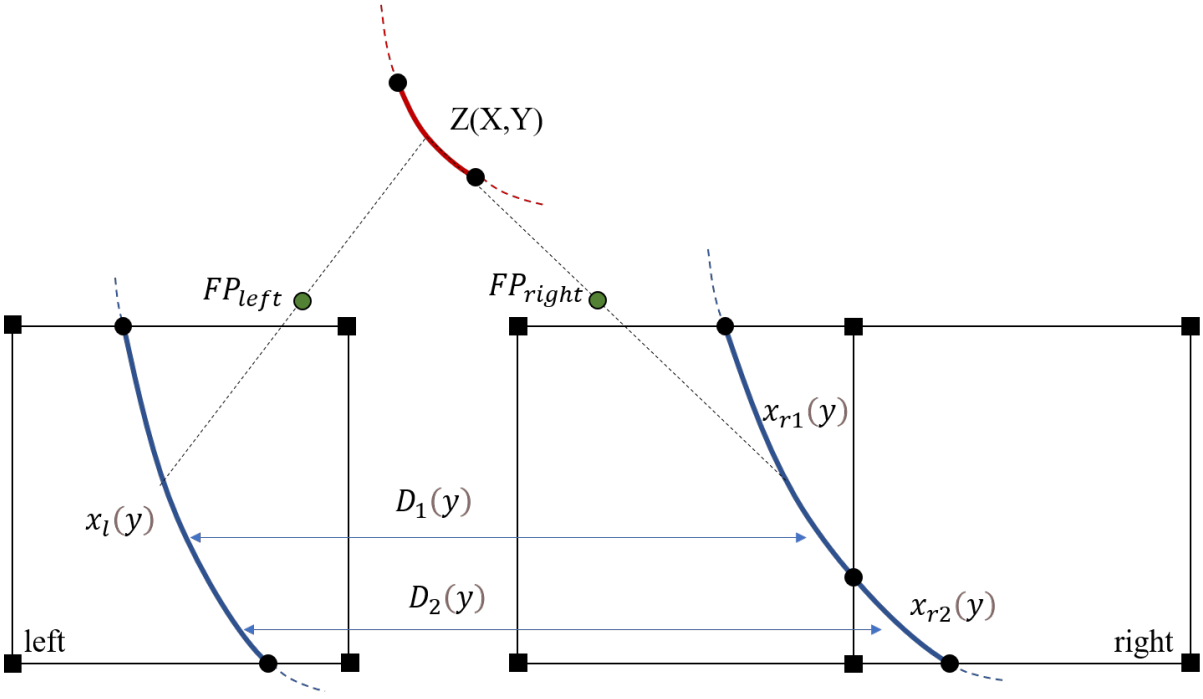

Figure 4.7: Curvelet reconstruction in 3D space from a set of matched image roots

right curvelet maps to a right or left curvelet respectively, that crosses multiple pixel boundaries, the disparities must be separated into two separate disparity/depth equations.

$$D(y) = \frac{(b_0a_1 + a_0b_1) + (b_1a_2 + b_3a_0 - b_2a_1 - b_0a_3)y + (b_3a_2 - b_2a_3)y^2}{b_1a_1 + (b_3a_1 + b_1a_3)y + b_3a_3y^2} \tag{4.12}$$

4.4 Ego-motion Estimation

We first use sparse feature matching between the cameras in the stereo pair, and using the computed disparity, compute the 3D locations of each 2D feature in the image plane. We then match the 2D features across different time steps and use the 2D-3D correspondences generated from stereo to get 3D point correspondences between two timesteps. We use Umeyama's algorithm ([Ume91]) to generate absolute pose estimates between the two timesteps, and use these as our initialization for the pose between each frame.

4.5 Acknowledgements

This chapter is currently being prepared for submission for publication of the material, titled "Stereo Panoramic Sceneflow: A Framework for Sensing and Perception of Dynamic Environments " as it may appear in IEEE Transactions on Pattern Analysis and Machine Intelligence 2021. Meyer, Dominique Ernest; Verma, Pranav; Niradi, Shreyas; Christensen, Henrik; Kuester, Falko. The dissertation author is the primary researcher and author of this material.

Chapter 5

Hardware Acceleration

5.1 Hardware Mapping of Vision Algorithms

Chapter introduced a set of algorithms that we would like to map to hardware architectures for efficient and real-time processing. While there are extensive arguments to go with one architecture over another, previously highlighted in Chapter 2, it is important to consider the data flow of a camera system. The two main types of sensors: rolling shutter and global shutter differ in the way they bin pixels. The first reads them one row at a time, while the latter bins all pixel charges simultaneously. When the pixel data is streamed out to a connected processor (often physically connected over high-speed data lanes on a PCB or serial cables), it is done sequentially. As such, we cannot receive the last pixel of an image (bottom-right most pixel), until we have received all other pixels in an image. When an algorithm operates on a set of pixels simultaneously, it needs to be able to access them all before it can complete the computational operation. This leads to an inherent data dependency issue, burdening any and every embedded vision system. Images are dense samplings of a light-field, with charges being synonymous to integration time of accumulated light flux. It is therefore necessary for us to keep intensity values at each pixel, and keep a certain number of pixel samples for our computation. These physically

translates to a per-pixel bit-depth (typically in the range of 8 to 12 bits) and an image resolution (hundreds to thousands in each dimension). When combined, this naturally gives rise a large data footprint $ImageSize(bits) = bitdepth(bits) \times resolution(pixels)$. These two fundamental principles of image processing is what we will uphold and worry about when designing and mapping our algorithms to hardware- 1) data is received sequentially from sensors, and 2) co-dependent data must be available at the same time to be operated on, often requiring large data buffers.

Figure 5.1 illustrates a general embedded vision system block diagram. We interface with a set of image sensors over a serialized data interface such as MIPI or SLVS, decode the data using de-serializers, and then have the choice of leveraging a set of operation blocks and memory to compute our image processing task, before outputting it to somewhere. The big question is- how do we design the algorithms, and map these to the resources (operators and storages) such as to maximize throughput and minimize the latency of the whole system? If we have to buffer whole frames, our on-chip storage is often too small, forcing us to cache to off-chip memory (DRAM), with increased latency, and limited memory bandwidth. If on the other hand we only cache to on-chip memory, we face the problem that we cannot store more than a few dozen image rows, limiting our algorithmic options.

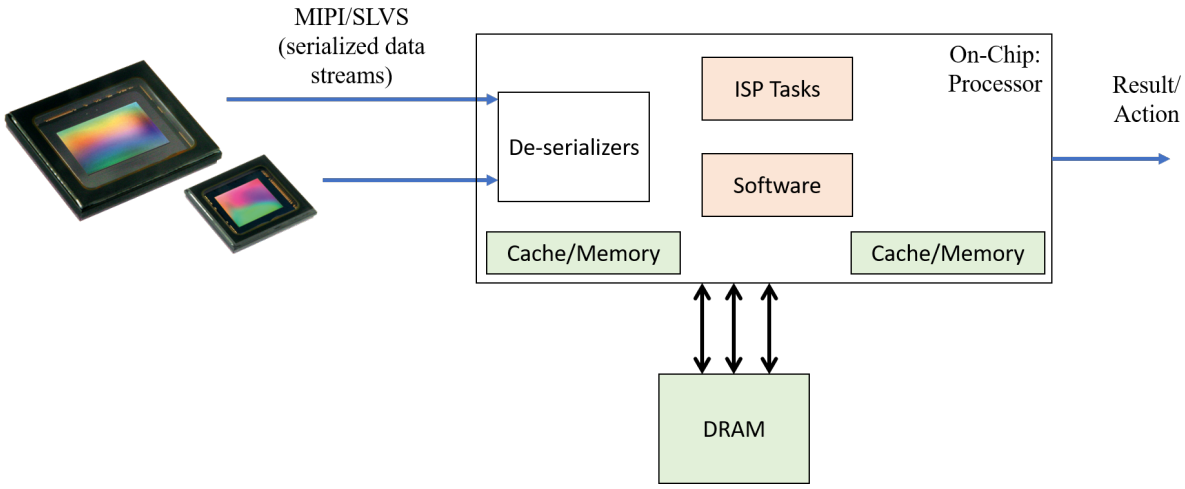

Figure 5.1: Dataflow of an embedded camera system

Recall our goal to reconstruct the 3D geometry from a set of images captured by multiple sensors simultaneously- we can now leverage the set of concepts introduced in the previous chapters to optimize specifically this system data-flow. To summarize, we have thus far concluded that: arranging our sensors in a parallel fashion reduces re-mapping requirements for image rectification to achieve horizontal epipolar lines in a stereo case; direct feature extraction and matching can be done on a sequential row basis, eliminating the need to buffer full frames for the processing; leaving us the final task of mapping this efficiently to our limited resources to minimize memory, latency and maximize the system throughput.

This chapter will begin by providing a high-level strategy to effective image processing handling for the purposes of system performance, followed by a detailed investigation and proposition of a joint pre-processing algorithm for multi-view systems, accelerated on the programmable logic of an FPGA. This pre-processing task is the largest data-dependency in the system since it aligns input images to meet the epipolar constraint, facilitating the remainder of the processing tasks.

5.2 Memory efficiency

Image streams arriving from the sensors are packaged in pixels chunks, or sets of pixel chunks. Since both Bayered and monochrome sensors output single channel data, these are represented with some bit-depth. In the case of 8-bits per pixel , this fits nicely in byte chunks, however for 12-bits per pixel, it requires 2 bytes per pixel, with the second byte only being half used, providing an overall packing efficiency of 75%. Alternatively, one can stack together 2 sequential 12 bit pixels, to use 3 bytes and have a packing efficiency of 100%. Once Debayered, multiple color channels are similarly packaged, with the same goal to improve our pixel packaging. In the case of embedded on-chip communication, it is common to use the Advanced eXtensible Interface (AXI) to transfer data efficiently, which supports direct data packet formatting to custom

or standard formats. For FPGAs specifically, we employ discrete or joint computation blocks linked by such communication interfaces, allowing for fully pipelined processing architectures with customizability on parallelism. Pixel values are cascaded down registers and operators, often without needing to cache them. In the case of general purpose processors (CPUs), operations are often handled asynchronously but sequentially with images stored in off-chip memory.

5.3 Splatty- unified demosaicing and rectification

5.3.1 Introduction

Image demosaicing and rectification are key tasks within multi-view computer vision systems. To date, however, their implementations have been plagued with large memory requirements and inconvenient dataflow, making it difficult to scale them to real-time, high resolution settings. This has motivated the development of joint demosaicing and rectification algorithms and implementations that resolve the backward mapping dataflow for improved hardware implementation. Towards this purpose, we propose Splatty: an algorithmic solution to pipelined image stream demosaicing and rectification for memory bound applications requiring computational efficiency.

We begin by introducing a polynomial Look-up-Table (LUT) compression scheme that can encode any arbitrarily complex lens model for rectification while keeping the remapping errors below 1E-10 pixels, and reducing the memory footprint to $O(min(m,n))$ from $O(mn)$ for an $m \times n$ sized image. The core contribution leverages this LUT for a unified, forward-only splatting algorithm for simultaneous demosaicing and rectification. We demonstrate that merging these two steps into a single, forward-only splatting pass with interpolation, provides distinctive dataflow and performance efficiency benefits while maintaining quality standards when compared to state-of-the-art demosaicing and rectification algorithms.

Vision algorithms inherently depend on input pixel values and correctness of model

assumptions that are used for higher-level cognitive and heuristic interpretations. A common use case of image preprocessing is stereo depth estimation where two sensor streams, as illustrated in Figure 5.2, are demosaiced and rectified before stereo matching can be used to extract pixel-wise depth estimates using the epipolar constraint [Hal10].

The data dependency should in theory allow for the data to be operated on sequentially, one pixel row at a time, without the need to temporarily store whole frames to calculate the output. In reality however, most of the preprocessing pipeline implementations separate the stages whereby full images are buffered between the steps. Additionally, both the demosaicing and rectification can each be interpreted as a lossy interpolation step, which when stacked together result in an unnecessary loss of data [HGG+11], which can be mitigated if we used one interpolation step which did both of these tasks simultaneously. This work is motivated with a two-fold set of objectives. Firstly, it is to create an algorithm that approaches ideal color and re-mapping results, and secondly, it is to have it be optimized for a streaming, forward only mapping and First-in, First-out (FIFO) data flow. By achieving this proposed flow, the hardware mapping becomes inherently efficient. This results in a computational strategy that allows the data to be processed using minimal buffer memory, thereby reducing the processing and memory requirements which are often limited in embedded systems.

This section proposes a novel preprocessing algorithm and architecture, Splatty, that simplifies rectification and demosaicing into a single forward mapping pass. At the time of this publication, this unification has not been explored and it is the goal that the advancement of video stream processing will enable accelerated image stream processing across systems with ever growing sensor resolutions and frame rates.

5.3.2 Rectification

Rectification is a general image re-mapping process, used for correcting a set of distortions that are introduced in the image capturing process: lens distortion, camera tilts, offset from focal

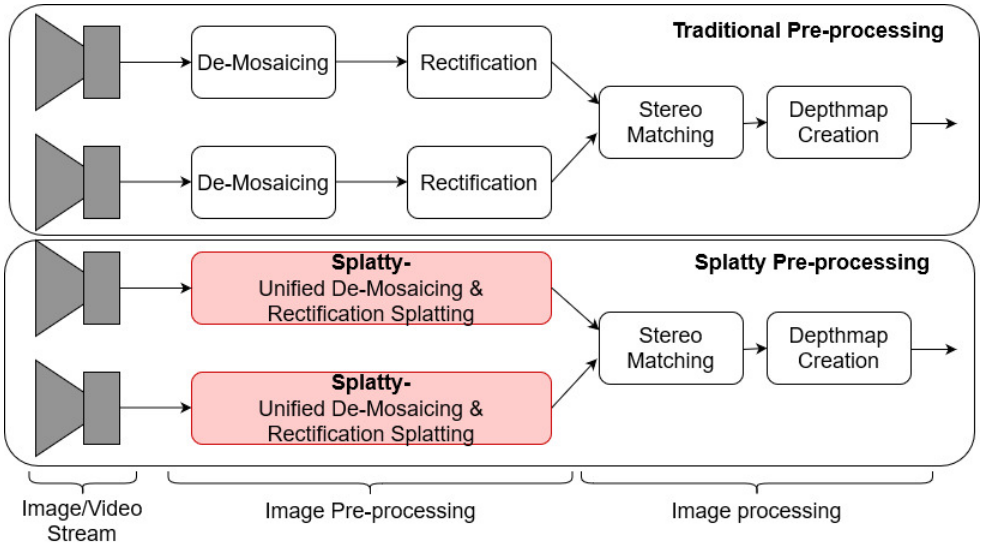

Figure 5.2: Stereo Pipeline- top: traditional preprocessing, bottom: Splatty preprocessing

axes; as well as compensating for non-perfect placement of stereo camera pairs. This is common in stereo depth systems [HJCE+16], where two cameras in a bifocal stereo apparatus are not coplanar or row-aligned, and require re-mapping so that the epipolar lines in both the images are pixel row aligned. It is usually the most computationally demanding step in stereo processing pipelines (for eg., see Table 1 in [GHGB11]), and hence the focus of optimization, to improve resource utilization.

There are two methods to perform rectification: forward mapping, and backward mapping, with the latter being the dominant method. In forward mapping, each pixel in the input image is mapped to a location in the output image, and the input pixel's intensity value is used to calculate the intensities at the image pixels in the neighbourhood of the output pixel, as shown in top part of Figure 5.3. There are various methods to perform this extrapolation: the simplest method is to map the output pixel to the nearest pixel location in the output image. However, this approach leads to so called "holes" in the output image [CW93]. An improvement over this approach is to use a bleeding or splatting technique, which extrapolates the pixel values from the forward-mapped pixel to all its neighbourhood pixels, and depending on the distance it bleeds over, it's value is more or less dampened using a decay function. For example, in the method

described in [She68], the intensity value falls off inversely with the square of the distance from the mapped location.

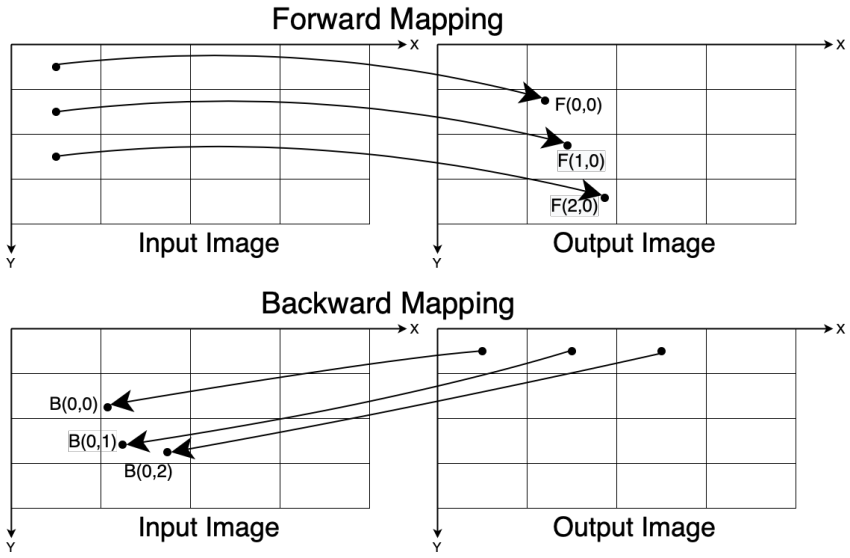

Figure 5.3: Forward and Backward Mapping

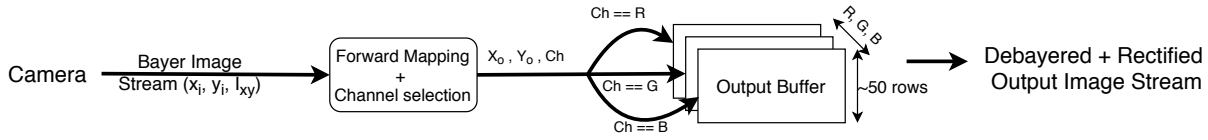

Figure 5.4: Proposed algorithm architecture

Backward mapping aims to overcome the problems of holes in the output image, by performing a reverse mapping: for each pixel in the output image, it finds the sub-pixel level accurate location of a point in the input image which maps to it. Then, it fills up the output image pixel by interpolating the intensity at the sub-pixel level accurate location from the intensities of the pixels in its neighbourhood [MSH04]. This is shown in Figure 5.3. This approach yields good results, but is difficult to implement in a streaming setting, where the input image is coming in one row at a time, and buffering large number of rows is not feasible. The locations of backward mapped pixels may not vary uniformly, and this necessitates buffering multiple rows for both input and output images. [FMR+08].

Adapting rectification to a streaming setting to operate in real-time is shown to be a difficult endeavor by many academic and industry efforts. [BHF+10a] attempts to create a real-time 3D vision system, where it uses rectification as a first step in their 3D vision pipeline. However, their results are limited to small image sizes (640x480 px). Larger image sizes at 30fps frame rates are not feasible on their systems, due to memory constraints. The work in [HN15a] demonstrates that on state-of-the-art FPGA systems, it is still necessary to buffer image data to off-chip DDR memory instead of having it all on the on-chip BRAM memory, which means there's signification I/O overheads involved. Furthermore, the remapping step of the full stereo pipeline, is by a factor of 5, the slowest stage in the system throughput. One implementation by [OK08a] has highlighted the feasibility to perform a radial only rectification with a backward mapping, but this implementation and architecture does not allow for more general rectifications to take place, and the output does not respect a FIFO order. [JHRN19] describe a lossy compression and subsampling based rectification method, which can handle more general cases of rectification, but it trades off reconstruction accuracy for reduced memory usage. We show in our results section that we utilise even less memory than [JHRN19] (we use $O(min(M,N))$ space for LUT storage vs $O(M \times N)$ for theirs, for an $M \times N$ pixels image), while keeping the reconstruction errors low.

5.3.3 Unifying rectification- Splatty

We propose that combining rectification and debayering into one will provide improved dataflow as it reduces the in between buffering requirements, and may lead to improved accuracy. This is because both debayering and rectification can be interpreted as interpolation steps, and both these steps aren't perfectly lossless. In that scenario, performing one lossy, joint debayering+rectification step can be better than performing those two lossy steps, one after another, especially if our performance on both these tasks is as good as the existing methods for each task.

We start off by decoupling the three channels and performing rectification on each channel separately through one forward splatting pass. A Lookup Table informs the destination location

of the splatting and a decay functions weighs the splatting to neighboring pixels in a set of output buffer rows. These get divided out depending on a count that is accumulated by the splatting contributions. Then, to improve the rectification result, we use the information of the green channel to improve the results of the blue and red channels, taking inspiration from the method proposed by [Cok]. This allows us to improve the reconstruction in areas with high frequency, such as sharp edges.

5.3.4 Implementation Overview

We implement rectification using a forward mapping based approach. Not only does this allows us to simplify the ordering of incoming stream of input pixels, it makes it easier for us to perform debayering along with rectification in one go on the stream of pixels, since they are ordered properly. It also allows us to map the input pixels as they come, and then discard them, allowing minimal buffering at the input side. For the output image, we need to buffer a few rows; the number depends on the extent of image distortion introduced by the rectification mapping. For most of the cases of stereo rectification that we've encountered, the maximum width of the output buffer does not exceed 50 rows. We present the algorithm flow in Figure 5.4, and lay out the complete algorithm in Algorithm 1

5.3.5 LUT Compression

We note that the LUT compression is important because the full look up table size grows with the size of the image. We compress LUT using a polynomial curve fitting scheme. In the most general setting, a LUT is the value of the function $f(x_{in}, y_{in})$ at discrete points x_{in}, y_{in}, which are the coordinates of the pixels in the input image. To perform LUT compression, we assume that the output y coordinate, (y_{out}) depends only on the input y coordinate (y_{in}) as an n^{th} order polynomial function of y_{in}: $y_{out} = a_0 + a_1 \cdot y_{in} + a_2 \cdot y_{in}^2 ... a_n \cdot y_{in}^n$. These coefficients will then be a

Algorithm 1 Forward mapping based debayering and rectification

```
for each row in input image do
  for each pixel, (x_i, y_i), in input row buffer do
    The color channel at (x_i, y_i) in the bayer pattern: c
    Find polynomial coefficients P_x = LUT_x[x_i], P_y = LUT_y[x_i]
    Compute output coordinates: x_o = P_x(y_i), y_o = P_y(y_i)
    Splat to the neighbourhood of (x_o, y_o), but only in the channel c
  end for
  if Top row in output buffer was not splatted to then
    Normalise row, adjust the values in 3 channels, and pop it out
  else
    Do nothing
  end if
end for
```

function of X_{in}, the input x coordinate. We store the coefficients in a table whose size is $max(X_{in})$ For compression of the x coordinate LUT, we assume that even the output x coordinate x_{out} depends on the input y coordinate, y_{in}, as a n^{th} order polynomial, whose coefficients depend on X_{in}, and can be found as above. We have assumed here that $max(X_{in}) < max(Y_{in})$. If that's not the case, then we can flip the values and coefficient dependencies to be on x_{in} and y_{in}, instead of on y_{in} and x_{in}, respectively. This way, we will be able to compress the LUT from size $(x_{max} * y_{max})$ to $(2 * min(x_{max}, y_{max}) * (n+1))$, where n is the order of the polynomial we use for curve fitting. For a full HD image (1080x1920), with even a 6^{th} order polynomial, we see the memory footprint decrease by $>$ 100x, with minimal increase in the computational expense for calculating the locations in the output image.

5.3.6 Splatting

We use a simplified version of the approach taken by [ZPVBG01], and [CDK99b] to define an isotropic splatting function, since computing the parameters for an anisotropic gaussian distribution at each pixel will be too computationally intensive for the demands of real-time processing on FPGAs. The splatting function, which describes how the a pixel's intensity

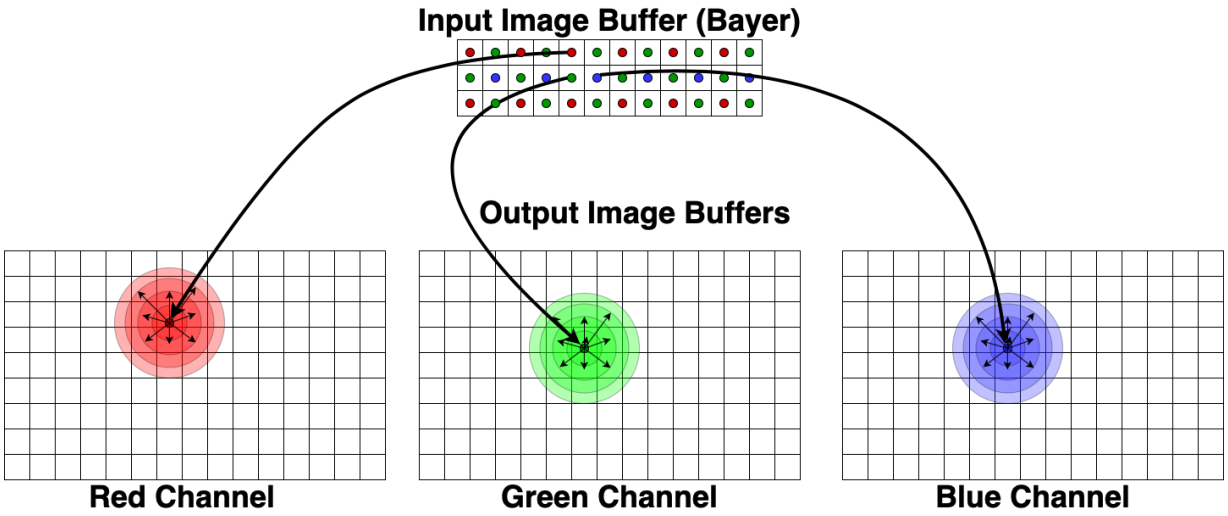

Figure 5.5: Illustration of splatting mechanism. Note that we splat each pixel only to it corresponding channel in the output image

contributes to the intensities of pixels in its neighbourhood, is defined as follows: For all pixels, P_o in the neighbourhood $N(P_o^{'})$ of pixel $P_o^{'}$, we can describe the contribution of P_o to the intensity of $P_o^{'}$ as

$$S(P_o^{'},P_o) = I_{P_o} * f(d(P_o,P_o^{'})) \tag{5.1}$$

where I_{P_o} is the intensity of the pixel at location P_o, $f(.)$ is the fall-off function, which describes the variation in the contribution as a function of distance, $d(.)$.

Once we have the contribution of all neighbouring pixels, to the intensity at pixel $P_o^{'}$, we normalize by dividing with the weights assigned to each of the neighbour's contribution. This can be expressed as follows:

$$I_{P_o^{'}} = \frac{\sum_{P_o \in N} I_{P_o} * f(d(P_o,P_o^{'}))}{\sum_{P_o \in N} f(d(P_o,P_o^{'}))} \tag{5.2}$$

We tested functions of the form $f(d) = e^{-(d^p)}$ and $f(d) = \frac{1}{1+d^p}$, for varying exponents p, and in section 5.3.10, we show how varying the functional forms affected the reprojection performance.

5.3.7 Rectification

The rectification step is automatically accounted for in the splatting destination. The decoded LUT table provides the destination pixel address to where the weighted values need to to be splatted to, and as such, there is no need for additional interpolation.

5.3.8 Debayering

To perform debayering, we start with the assumption that the channels can be decoupled, and each channel is debayered and rectified together independently. After getting an initial estimate of the channels, we correct our estimates for each channel by using the intensities of the other channels, in a fashion similar to the method proposed by [Cok] Since each channel is being filled out by the same input pixel, we can perform this correction as we are pushing the rows out of the buffer. This ensures that each pixel of the output image stream is coming out debayered and rectified.

5.3.9 Dataflow

The major novelty of Splatty manifests itself in the dataflow optimization of the algorithm that lends itself to streaming processing. Image streams are generally transmitted as pixel rows, which are buffered into row buffers. Our method operates sequentially on pixels as they are incoming, directly deriving forward mapping coordinates through the polynomial LUT decoding. The LUT memory footprint is of order $N \times O \times B \times 2$, where N is the image height, O is the polynomial order and B is the coefficient bit-depth. The decoding of the LUT is accomplished through a decoding block, which evaluates the polynomial at the pixel location. Using the destination pixel coordinates, a distance calculation block is used to calculate either the L1 or L2 distance norm. This gets passed to a decay function that calculates the splatting coefficient for the neighboring pixels, which in turn multiply accumulate the results in the output buffer. Once the

output row has been completely contributed to, a row state releases the row. This implies that the number of output buffer rows needed is equal to the largest vertical remapping difference per row defined by the LUT. As this can be precomputed after a camera calibration, logic synthesis and architecture can be optimally adapted. This architecture lends itself to pipelining and block parallelism that scales due to the eliminated data dependency issue that is present with a backward mapping approaches.

5.3.10 Results

Look Up table compression

In order to find the polynomial order which best fits the rectification maps, we find the mean squared error distance between the values predicted by the polynomial-compressed rectification maps and the uncompressed maps for all pixels in the input image for each polynomial order. This way we can find the smallest order polynomial which fits the mapping well (i.e. which has MSE below a threshold, which we set to 1e-10).

Once we find the polynomial coefficients which best fit the mapping, we don't have to compute them again till we change the mapping itself. Therefore, the polynomial compression of the Look Up Table needs to happen only when we're calibrating the system.

In Figure 5.6, we present the mean squared reconstruction error as a function of the order of polynomial used for polynomial curve fitting. We observe that a 6^{th} order polynomial is sufficient for compression of LUT for this type of rectification. For orders greater than 6, we see that the errors do not decrease, and hence, in order to be as memory efficient as we can, we use order = 6.

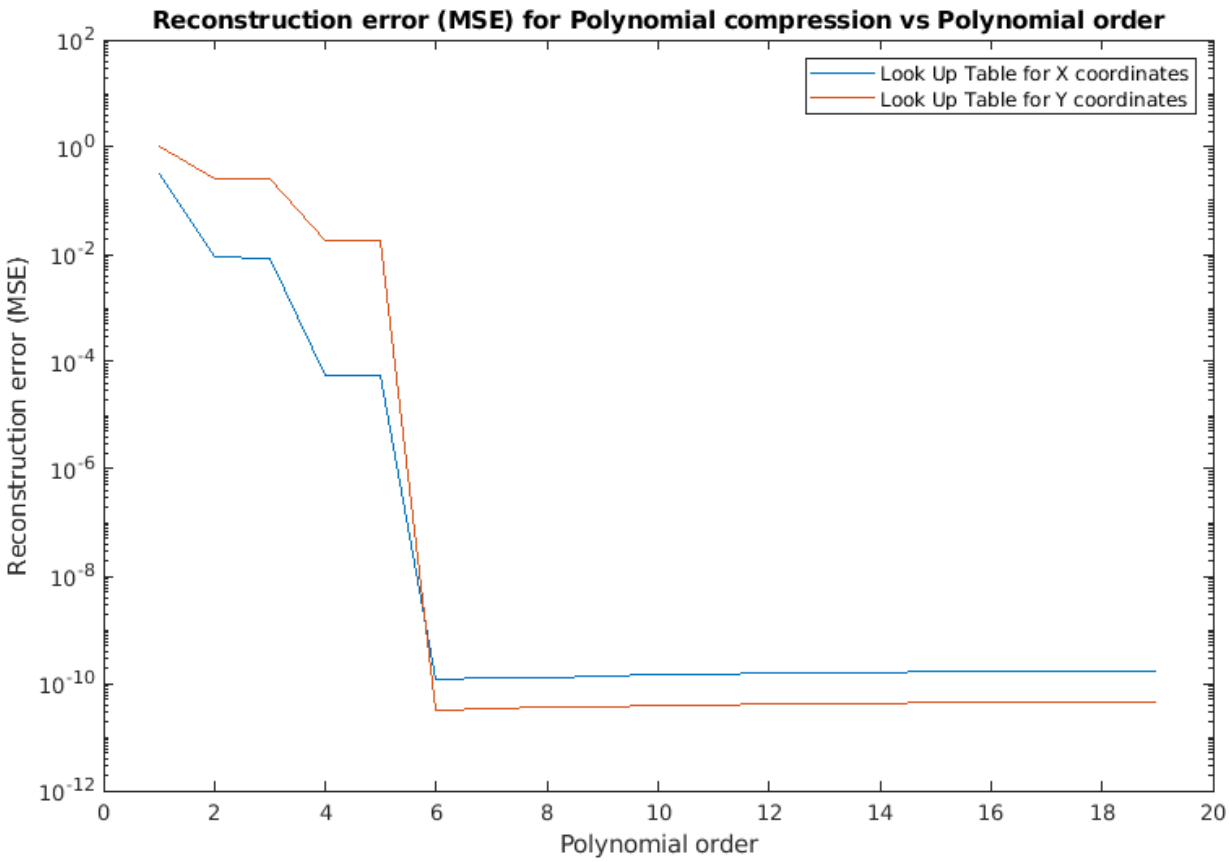

Figure 5.6: Mean squared Reconstruction Error vs polynomial order

Splatting Function

For our splatting function defined in Equation 5.1, we vary the falloff function f and the distance metric between two pixels, and choose the parameters which yield the highest average PSNR scores on the Kodak dataset [Com20].

For each value of any parameter, we find the average PSNR over all the possible values of the other parameters, over all images in the dataset. From these, the value of the parameter with the highest PSNR value is chosen. We show the results in Figure 5.7. From the figure, we find that the optimal splatting function would look like the following:

$$S(P_o^{'}, P_o) = I_{P_o} * exp(-\|P_o - P_o^{'}\|_1^4) \quad \forall P_o \in N_8(P_o^{'}) \tag{5.3}$$

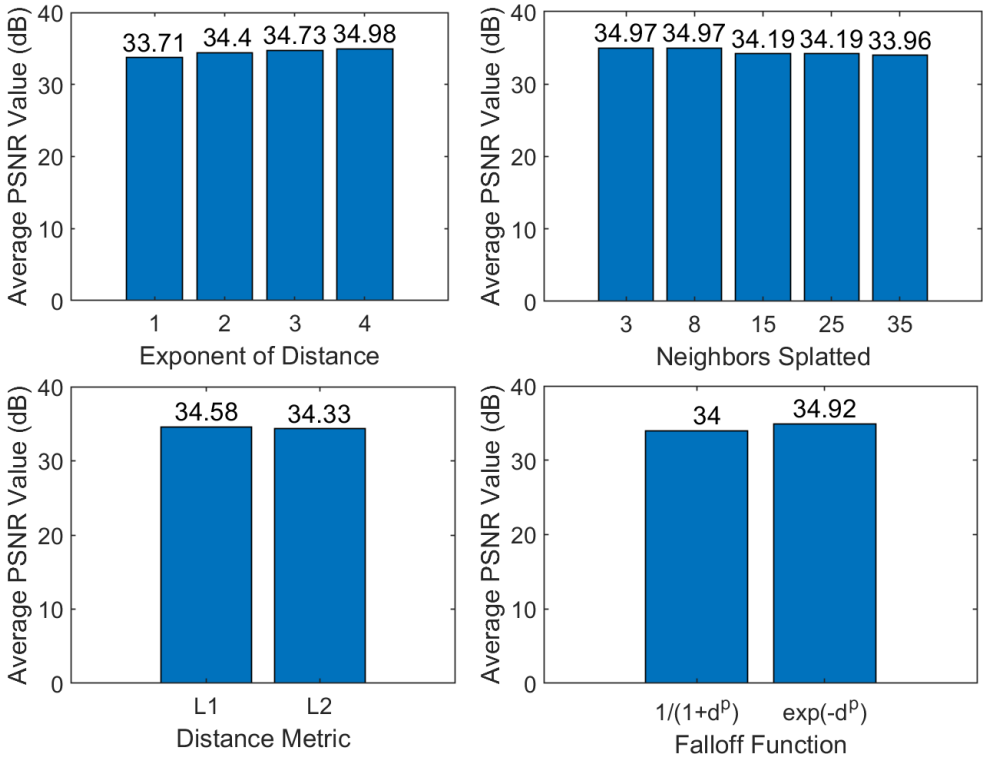

Figure 5.7: Variation of Average PSNR scores as we modify different parameters in the splatting function

where $\|.\|_1$ is the $l1$ norm, and N_k is the set of k nearest neighbours.

5.3.11 Debayering

Since most of the debayering algorithms in use have been proposed for the general CPU+GPU architecture, we compare them with our C++ implementation in Table 5.1, comparing their reconstruction accuracy (via Peak Signal to Noise Ratio values) on the Kodak Images Dataset [Com20], along with the memory usage as a function of the image size, and the effective number of passes that each algorithm takes to produce the final output. The last two metrics are important, since multi-pass algorithms which need the full intermediate images in memory to produce outputs cannot be used effectively in a pipelined setting, where we would want to buffer the data, and use as few input and intermediate rows to produce an output row as we can. Because of this reason we only present the PSNR vs passes and memory usage order-of-magnitude for spatial domain based demosaicing algorithms, and skip the frequency domain and neural network based algorithms, since passes and memory usage order-of-magnitude make little sense in those scenarios. We also provide a more thorough comparison between all the different algorithms, by looking at average PSNR score on Kodak dataset vs the memory usage for processing 1920x1080 images, in Figure 5.9. The results show that our algorithm is better than the existing single pass algorithms, and some of the multi-pass algorithms ([XO01]), while being orders of magnitude more memory efficient than every other algorithm.

In the interest of space, we show the qualitative results from our algorithm and [MLC04] on the complete Kodak dataset, and several close ups, in the supplementary material. The qualitative results show that our algorithm produces fewer, or about the same visual artifacts in high frequency regions (sharp edges) as the other approach ([MLC04]).

Table 5.1: Debayering performance of various algorithms on Kodak Dataset, measured by PSNR in decibels, the effective number of passes to produce the output, and the order of memory consumed.

Algorithm	Avg. PSNR (dB)	Num Passes	Memory
Bilinear	30.889	**1**	$O(mn)$
Cnst. Hue	33.259	2	$O(mn)$
Malvar	34.337	**1**	$O(mn)$
Li	34.388	2	$O(mn)$
Kimmel	35.61	2	$O(mn)$
Hamilton	**36.9**	2	$O(mn)$
Gunturk	35.8	2	$O(mn)$
Proposed	34.937	**1**	**O(min(m,n))**

5.3.12 Rectification

To test our algorithm's rectification performance in a real world setting, we use a series of images from a stereo camera pair introduced in [MWS+19], and test our algorithm's performance on performing stereo rectification. As a implementation reference, we created a pipeline that uses the works by ([MLC04], [Har99], and [Zha00]), implemented in the OpenCV library debayering and rectification functions. We also use [Har99] to create our uncompressed LUT. The rectified outputs from the reference pipeline and from our Splatty algorithm are showing in Figure 5.8. From the qualitative results from Figures 5.8 along with the reconstruction errors in Figure 5.6 we can see that our rectification is on par with the standard rectification algorithm in use today ([Har99]), having a variation of less than 1e-10 in RMSE between pixel locations calculated by [Har99] and ours, even though we use a fraction of the memory for rectification (as seen from Table 5.4).

We also present a quantitative comparison between [Har99] and our implementation, by comparing their performance on rectifying radial, tangential and projective distortions, and their combinations. We use OpenCV's initUndistortRectifyMap function to create uncompressed LUTs from lens distortion parameters, which we then invert for use in forward mapping setting, and [Har99] to create uncompressed LUTs from projective transformation matrices. We take a high density checkerboard pattern, apply said distortions, and use both rectification pipelines to

Table 5.2: Mean Absolute Error in corner detection after rectifying distortions using backward mapping based approach ([Har99]) and our forward mapping based approach

Distortions\Algorithm	Bwd Mapping ([Har99])	Proposed
Radial	0.8473	**0.7177**
Radial + Tangential	0.831	**0.6998**
Tangential	0.4831	**0.2021**
Projective	**0.7678**	0.9013
Average	0.7323	**0.630225**

undistort the images, and then compute the mean absolute distance between corner points in the ground truth and the undistorted outputs from both pipelines (Table 5.2). Images generated in this experiment are present in the supplementary material.

(a) Bayer Input (b) [MLC04] + [Zha00] (c) Our Algorithm

(d) RAW Bayer input (e) [MLC04] + [Zha00] (f) Our algorithm

Figure 5.8: (a),(d) Raw, unrectified image from right camera of a stereo pair, (b),(e) Debayered, then Stereo Rectified Image using [MLC04], and [Zha00] (c)Debayered + Stereo Rectified Output from our algorithm, (d)-(f) Close ups from respective images

Table 5.3: Splatty Implementation Resources on a Xilinx ZU15EG FPGA

Name	**BRAM**	**DSPs**	**FF**	**LUT**
Splatty	266	63	136954	113436
ZU15EG Total	1488	3528	682560	341280
Utilization %	17	1	20	33

5.3.13 Memory footprint

To the best of our knowledge, no other algorithm performs joint debayering and rectification in a streaming based approach. Therefore, we consider multiple rectification and debayering algorithms, and create a pipeline by stacking the most memory efficient rectification and debayering algorithms together. Since many of the debayering algorithms are multi-pass algorithms which cannot work efficiently in a buffering setting, it is infeasible to implement them on FPGAs and compare their memory utilisation directly. We present a theoretical estimate on the memory utilisation of each algorithm, taking into account only the memory needed to store the input, output (as unsigned 8 bit integer values) and intermediate stages or LUTs (if any) (as float32 values), and buffering when possible, to highlight the feasibility of each algorithm for use on an edge compute device. We then show the approximate usage for processing a 1920x1080 image for non-deep-learning based methods in Table 5.4. In order to also compare the recent deep learning based demosaicing methods, we consider the memory usage of storing the parameters, and any intermediate feature maps that need to be stored to produce output at inference time (for eg., feature maps used in skip connections). We compute the memory usage for these methods and present a plot of demosaicing + rectification performance (in terms of PSNR scores) vs memory utilization for processing 1920x1080 image, in Figure 5.9. We also show the total on-chip memory available on a commonly used reference FPGA, the Xilinx Ultrascale+ ZU15EG, and the commercially available FPGA with the largest amount of on-chip memory, the Xilinx Virtex Ultrascale+ VU19P. The algorithms on the right of these lines will need to store their parameters/intermediate outputs onto off-chip memory, which is a hinderance in real-time systems, since I/O from off-chip memory is slow. We validate the algorithm using High-Level-Synthesis (HLS) tools for synthesized and routed RTL on a Xilinx ZU15EG FPGA. The resource utilization is shown in Table 5.3, which delivers an overall throughput of 16FPS and can accommodate up to 3 full 1080p pipelines on a single device. No off-chip memory or UltraRAM is used for this implementation and the video streams are implemented using an AXi4-Stream. Our algorithm

takes 4x less memory than the next most efficient combined, debayering and rectification pipeline, and performs better than the most memory-efficient debayering algorithms, evident from the PSNR scores in Table 5.1. This proves that for edge scenarios, where efficiency and accuracy need to be balanced, our algorithm is the best choice for debayering and rectification preprocessing steps.

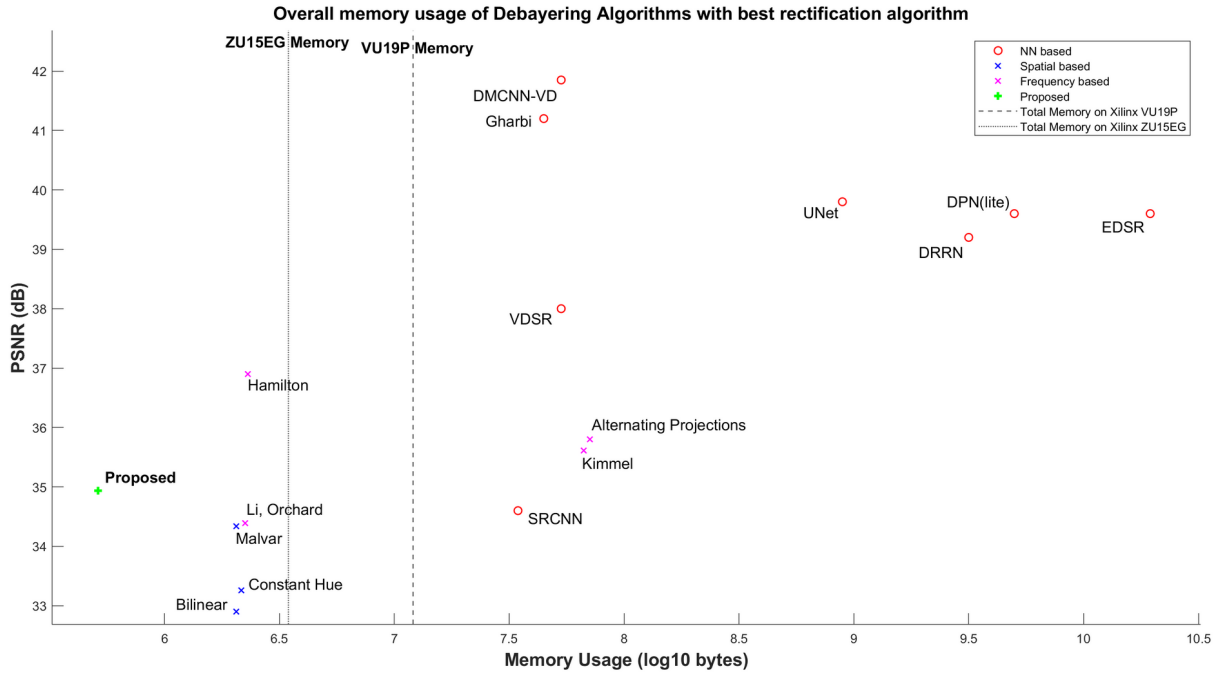

Figure 5.9: Memory Utilization by a theoretical debayering + rectification pipeline, constructed from various debayering algorithms and the most efficient rectification algorithm considered [OK08a]

5.4 Discussion

The results provide a valuable justification that rectification and demosaicing performance can be maintained despite the drastic gains in memory and dataflow efficiency. Firstly, we highlight that the splatting process is largely dependent on the splatting radius and decay function. Sharper fall-offs yield crisper images, but are slightly more expensive to compute in the decay function block, and larger splatting radii eliminate holes in more extreme rectification cases, but

Table 5.4: Theoretical estimates of memory consumption of different Debayering and Rectification algorithms, while processing a 1920x1080 pixels image, in kilobytes

Rectification		Debayering	
Bwd Mapping	8100	Bilinear	**15.12**
		Malvar [MLC04]	**15.12**
		Smooth Hue [Cok]	**15.12**
Oh, Kim	**2044.72**	Hamilton [JFHJ]	75
		Gunturk [GAM02]	15693
Junger	2120.63	Kimmel [Kim99]	14175
		Li, Orchard [XO01]	35.63
Best	2044.72	Best	15.12
Rectification + Debayering Best			2059.84
Ours			**513.00**

also comes at the cost of needing to splat to more pixels, that is hardware expensive. The LUT results are promising, and it can be concluded that we can achieve minimal reconstruction errors using even a low order polynomial approximation. Also, since our rectification method relies only on a general LUT to model distortions, we can incorporate more complex lens models, or refine a LUT created from simpler models to improve rectification accuracy. Future research may include color channel coupling to be adopted as a terminal stage of the pipeline and additional filtering operations to be incorporated within the decay function. While the works that outperform Splatty in terms of PSNR, all use more complex and multi-pass demosaicing, these concepts can in future also be applied to the post-splatting operations of Splatty, to similarly, improve output qualities.

We present a novel forward mapping algorithm that merges two common image preprocessing steps - demosaicing and rectification, for improved streaming feasibility. Embedded camera systems often are limited by their throughput capabilities due to hardware resource limitations, namely memory and computational blocks. We have shown that it is possible to maintain state of the art- demosaicing and rectification results by reducing the memory footprint to $O(min(m,n))$ from $O(mn)$ by merging these steps. We have also validated a polynomial rectification LUT that maintains remapping accuracies to 1E-10 RMSE. We hope that enabling efficient preprocessing through unified splatting will allow future systems to increase resolution

and frame-rate operations while reducing hardware resources.

5.5 Acknowledgments

This chapter is, in part, a reprint of the material as it appears, titled "Splatty- A Unified Image De-Mosaicing and Rectification Method" as it may appear in Winter Applications of Computer Vision (WACV) 2021. Verma, Pranav; Meyer, Dominique Ernest; Xu, Hanyan; Kuester, Falko. The dissertation author proposed the algorithmic approach to the work and oversaw the implementation in software and hardware.

This chapter is, in part, currently being prepared for submission for publication of the material, titled "DevCAM: An Open-Source Multi-Camera Development System for Embedded Vision" in Electronic Imaging 2021. Meyer, Dominique Ernest; Birlangi, Meher Akhil; Kuester, Falko. The dissertation author was the primary researcher and author of this material.

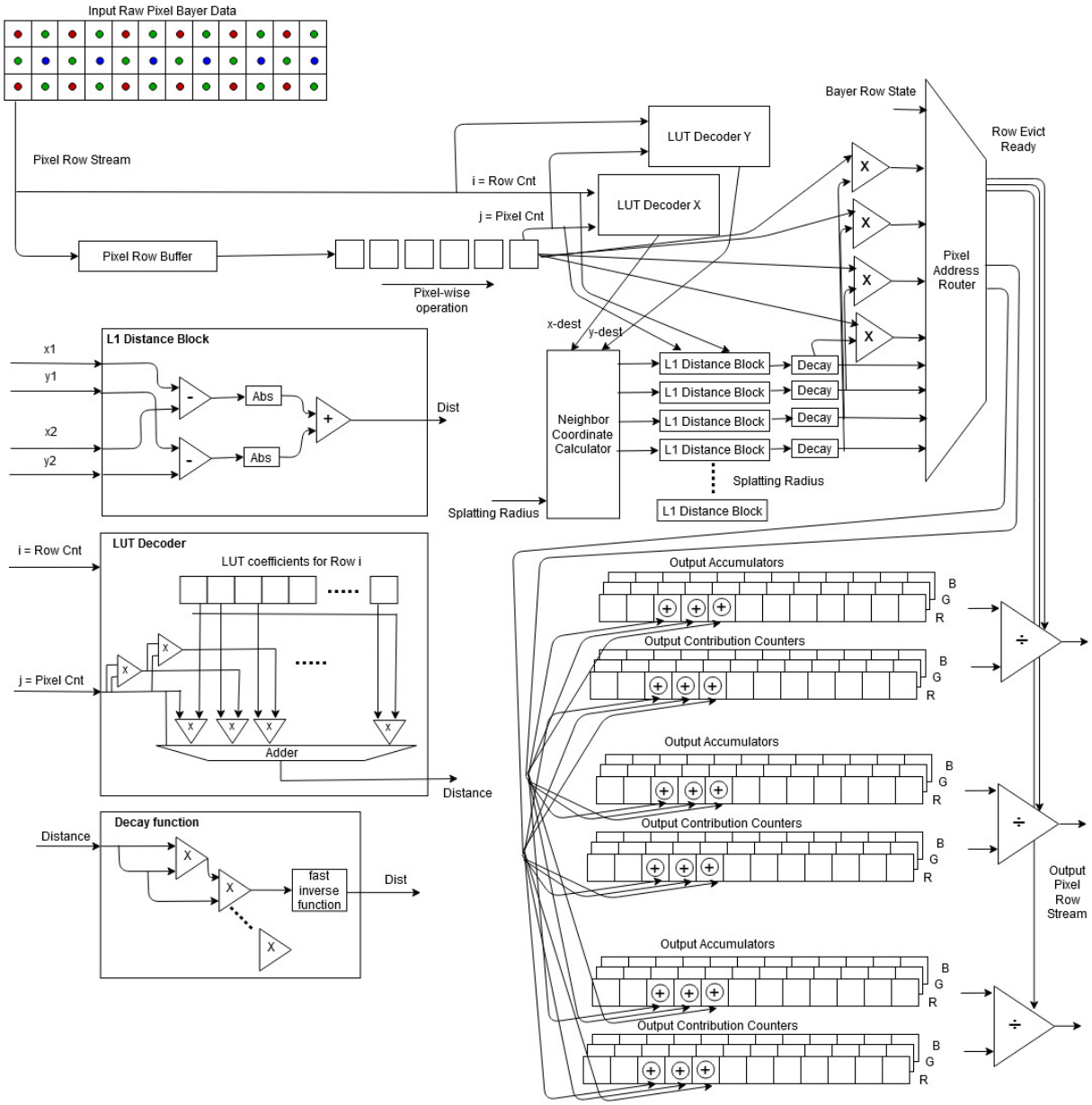

Figure 5.10: A dataflow architecture of Splatty implemented in an FPGA context

Chapter 6

Embedded Design of Multi-Camera Systems

6.1 System Development Objective

Computer vision algorithms are often burdened for embedded implementation due to the system development and complexity. Many commercial systems prevent low-level image processing customization and hardware optimization due to the largely proprietary nature of the algorithms and architectures, hindering research development by the larger community. This chapter highlights a hardware implementation of a research development system, designed to facilitate the research, development and deployment of such kind of systems: DevCAM- an open-source Multi-Processor System on Chip (MPSoC) based system, targeted at hardware-software research for vision systems, specifically for co-located sensor processor systems. The objective being to facilitate the integration of multiple latest generation sensors, abstracting interfacing difficulties to high-bandwidth sensors, enable user defined hybrid processing architectures on FPGA, CPU and GPU, and to unite multi-module systems with 40Gb/s networking and 12Gb/s off-chip NVMe storage. The system can accommodate up to six 4-lane MIPI sensor modules which

are electronically synchronized to an embedded RTK-GPS receiver and 9-axis IMU. Furthermore, the system is compatible with existing GPU/ARM CPU platforms such as the Xavier systems to allow for easy computational comparison and flexibility. We demonstrate a number of available configurations that can be achieved for stereo, 360, and partial light-field image acquisition tasks. The development framework includes mechanical, PCB, FPGA and software components for the rapid integration into any system. System capabilities are demonstrated with the focus on opening new research frontiers such as distributed edge processing, inter system synchronization sensor synchronization using GPSDO, and hybrid hardware acceleration of image processing tasks.

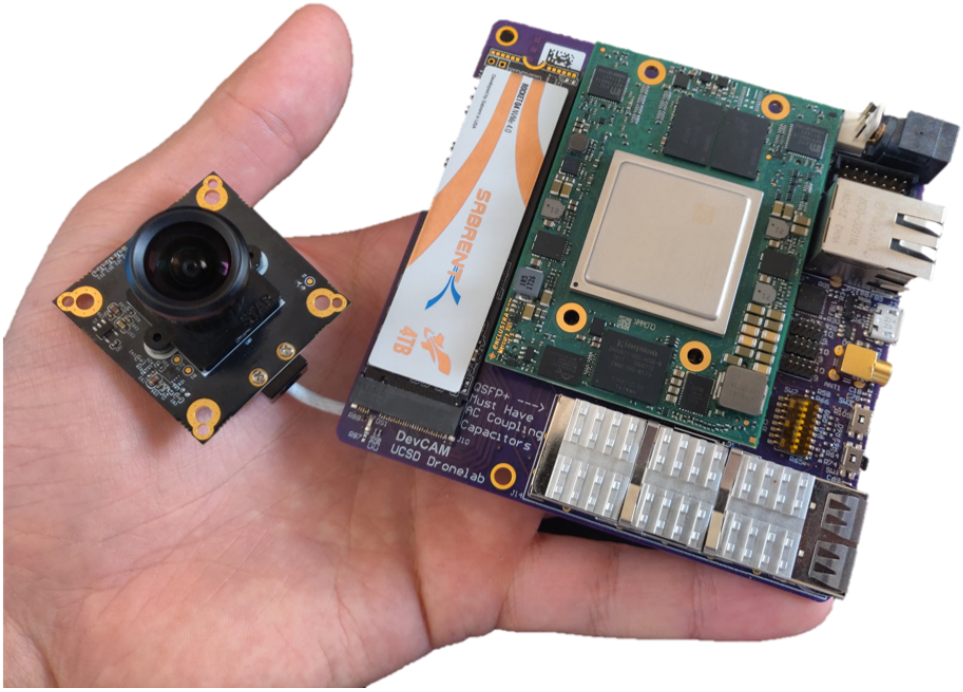

Figure 6.1: DevCAM V1.0 system with a single IMX577

This work spans a set of evolving fields within the electronic imaging community: Image Signal Processors (ISP), multi-view reconstruction, hardware acceleration, image/video compression, visual-inertial Simultaneous Localization and Mapping (SLAM), robotics and light-field arrays. Across these fields, there has been an ever evolving need to improve the hardware implementation of the latest algorithms, which are often demonstrated on desktop-class processors, but

unable to be run in real-time, and on mobile platforms. This limitation is two-fold: algorithms are commonly implemented with the goal to achieve superior qualitative results compared to state of the art rather than to perform within a deployed system, and underlying hardware architectures prevent cross-architecture acceleration. We are therefore after creating a development ecosystem that facilitates the porting of pre-developed algorithms to hardware platforms with minimal effort, and to leverage pre-established camera interfaces for systems to be deployed.

The DevCAM development platform has been designed as an application agnostic system, opening research possibilities beyond industry furnished development kits. The system leverages existing open-source tools for high-level synthesis of C/C++ code into RTL deployable in the FPGA fabric and easily configurable mechanical components to enable research at any level- array configuration optimization, hardware acceleration and algorithmic optimization for systems.

Commonly, embedded vision systems are designed for dedicated applications, by a team of engineers that each focus on their area of expertise- optics, mechanical design, electrical hardware, FPGA logic, and fimware/software. The problem being that the development stack implies the requirement of such teams, with information gaps across members, and global design goals that need to be set. The DevCAM system's approach is to unite the stack into a more approachable system design (Figure 6.2, where all components are pre-assembled, but remain configurable for any one person to tackle the full system. This is critical in environments such as academia and research, since limited resources make it difficult to tackle problems such as multi-view camera design.

The DevCAM Open-source project unites a set of full-stack features to enable advanced vision research within the electronic imaging community- specifically embedded vision. The major technical challenges that are addressed are: 1) Onboard synchronization between image sensors, IMU, GPS, and synchronization between non-connected systems leveraging GPSDO for simultaneous large baseline, multi-view capture. 2) Distributed image processing across multiple hybrid hardware architectures. 3) ISP algorithm development on multi-camera systems. 4) High

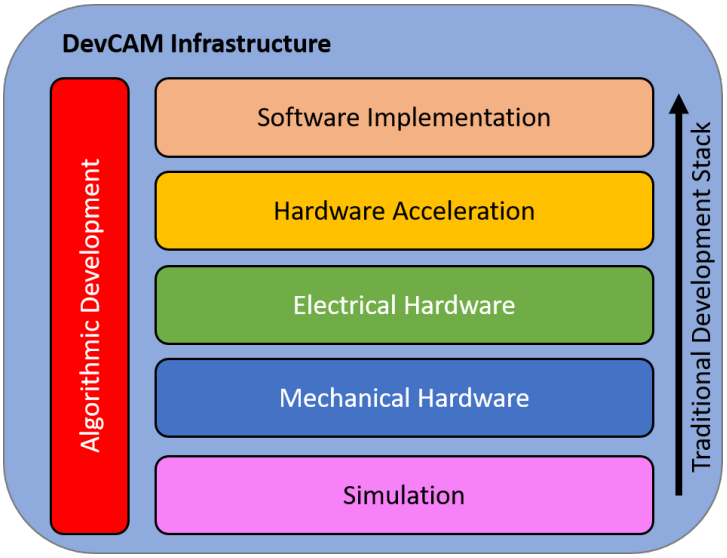

Figure 6.2: DevCAM Development Stack

bandwidth image acquisition and processing 5) Compact camera array configuration for epipolar constrained and spherical imaging.

6.2 Physical Integration

Many practical factors play a role in the effectiveness of an end system and it is with that in mind that this system was designed. Firstly, the mechanical compactness, weight and power efficiency all affect where the system can be used. Whether being deployed on an autonomous vehicle in an urban setting, or on a pole in Maya temples, the deployment envelope is defined by the system capabilities. We wanted the system to be as flexible as possible, enabling both data collection for post-processing and on-board compute opportunity to transition algorithms to run in real-time on the system. Finally, what is arguably the most important feature we targeted, is the co-location of the sensors to do our data acquisition. From the image sensors, IMUs, and GPS modules, these are normally found on separate modules and have to be connected, to form a bundle of sensors- cumbersome to design, operate and suffer from physical separation. The packaging that this system is able to achieve is an order of magnitude more compact that some

deployed research system, while resulting in similar resolutions, sensory acquisition packages and onboard storages/processing. An example is shown in Figure 6.3, comparing the Google street view spherical camera array to the DevCAM monocular panoramic system.

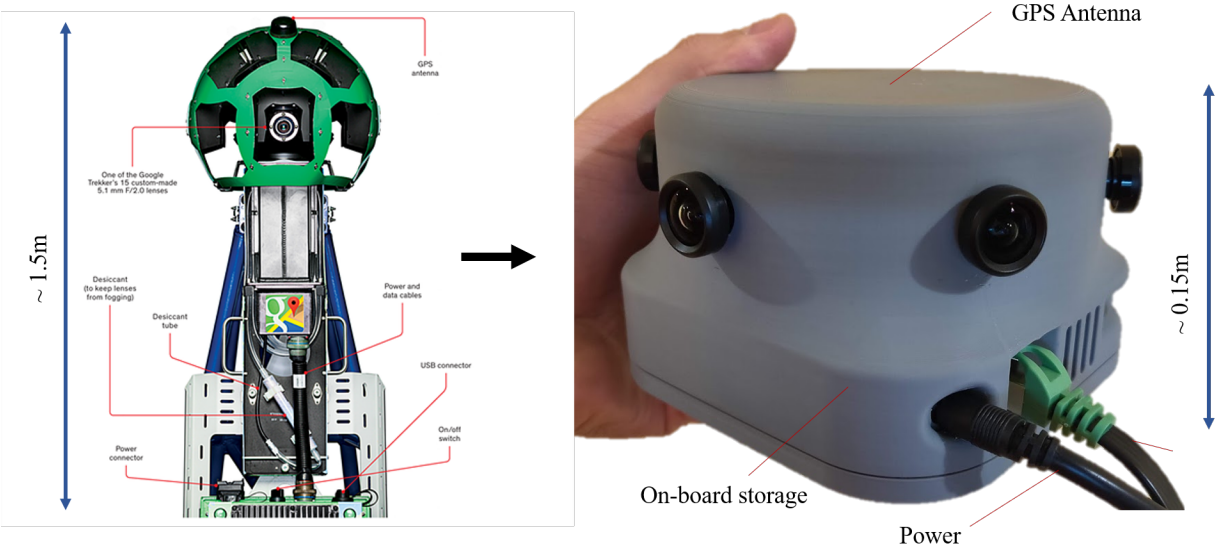

Figure 6.3: Comparison of the Google StreetView platform and the DevCAM panoramic implementation, with similar resolution, and sensor features

In camera systems, the most important components are the sensors and lenses, since they dictate resolution, FoV, sensitivity and dynamic range. The two main types of sensors include rolling shutter and global shutter sensors, that differ in the binning- the first binning pixel rows sequentially, while the latter bins whole frames synchronously. Up until recently, global shutter sensors were sparse, expensive and did not reach higher image resolutions compared to rolling shutter sensors. These global shutter sensors are greatly preferred in the computer vision community since they allow the assumption of a fix focal point over the exposure, while rolling shutters do not. The development of back-illuminated CMOS sensors has enabled the improvement of these pain points in global shutter sensors, bringing to the markets sensors with both favorable resolution and light sensitivity, at a reasonable price point.

The other choice in sensors is the sizing, which dictates the surface area available to integrate the light-flux. Generally, larger sensors with large pixel pitches, are able to acquire

better low-light imagery. The problem associated with that is that they consequently also require large optics, which are cumbersome and expensive to integrate, especially when you need many of them. As such, we focus on sensors that are able to fit M12 lens optics, small enough that they can be easily assembled into arrays, yet still capable enough that they can reliably resolve scenes in diverse conditions. These fit in the size range from 1/2.3" to 2/3" sensors.

Within the proposed systems, we highlight 2 sensors that have demonstrated excellent performance and that are easily integratable within multi-camera systems. Firstly the IMX264 is a global shutter sensor with 5 megapixels of resolution in a 2/3" form factor, leveraging 3.3 μm pixel pitch and a bit-depth of 12 bits per pixel. The second being the IMX577 rolling shutter sensor in a 1/2.3" form factor, with 1.55 μm pixerl pitch and a 12 bits per pixel bit-depth. The important note about why this rolling shutter sensor was deemed reasonable is due to the ability for it to be synchronized across multiple sensors to do synchronized rolling shutter captures. What this means is that the pixel rows are binned simultaneously across the sensors. This is important in cases where the camera system is in motion during acquisition, so that the relative extrinsic parameters are correctly assumed to be constant. The same applies to global shutter sensors, where the exposure and binning is synchronized.

To match these sensors, M12 lenses were selected to best meet the integration of multiple cameras into a single system. More sensor-lens modules allow for narrower FoVs, and higher spatial resolutions, but with increased processor count, cost and integration difficult to cover full panoramic and spherical FoVs. Wider lenses in fewer modules on the other hand suffer from larger lens distortions, increasing memory requirements in the rectification stage of multi-view systems and consequently enabling larger throughputs.

The final integration consideration is the use of multiple processing modules, in a synchronized fashion. The physical integration of more than 6 MIPI sensors per DevCAM processor board is unfeasible due to the limited high-speed I/O pins available on the package. Since we want to leverage the full capability of 4 MIPI lanes per sensor, that makes a total of 24 differential

pairs, it is not easily possible to add more sensor modules to the FPGA package. Furthermore, the computation resource limitations are quickly limiting the addition of more sensor processing pipelines, encouraging the distribution across multiple processor packages. This integration flexibility is enabled through timing synchronization interfaces between multiple processing boards, allowing unlimited boards to be daisy-chained for simultaneous data capture and processing.

6.3 Open-Source Environment

An important aspect of the design of this system, is that it applies to both research efforts and deployed systems across different applications. The DevCAM environment is fully open-sourced, to enable larger community adoption and facilitate the experiments in this field.

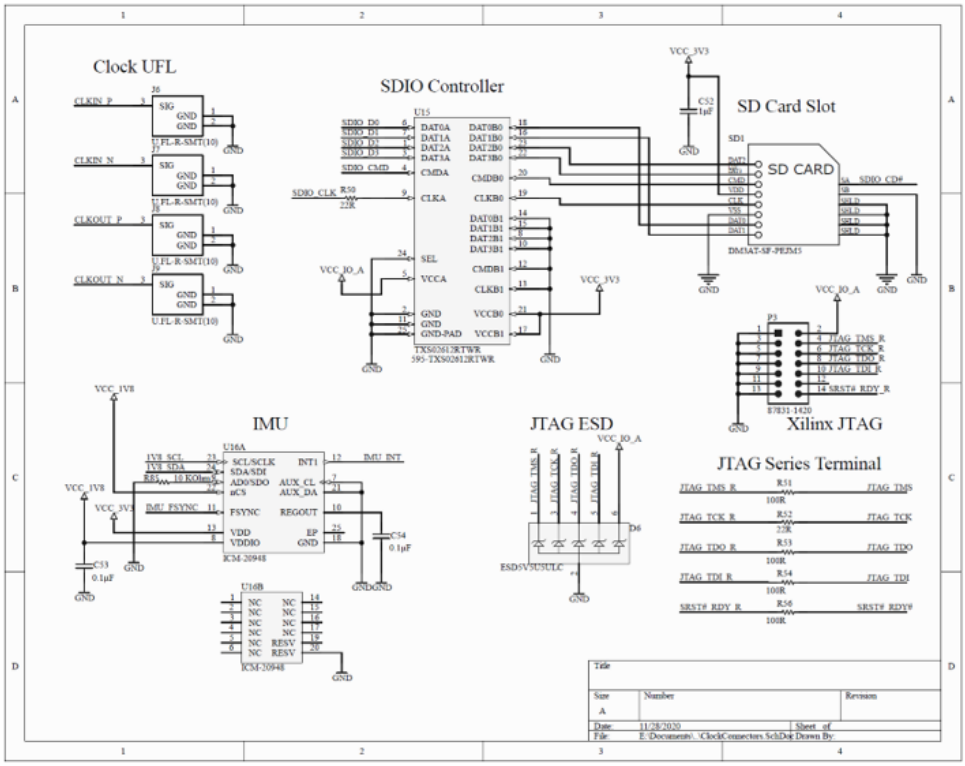

Figure 6.4: DevCAM V1.1 Schematics Example

The development infrastructure is broken down into physical, and software components,

which are tightly coupled. On the physical side, the design features all system schematics, linking the PCB components and connectors together- Figure 6.4. The design is implemented on a small form factor PCB board, with double sided population. The board has dimensions of 100 x 100mm, making it small enough to integrate into a compact package. All trace routes, vias, and PCB specifications have been carefully designed in a flexible, specification adherent manner to allow for easy re-configuration. The Altium design files are shared part of the open-source project, allowing researchers to leverage it for re-customization, or for direct re-production. Figure 6.5 highlights the high-density in the PCB design, and the trace design for high-speed buses such as the MIPI lanes or the PCIe interfaces. While the initial design complexity was on the order of designing a computer motherboard, future iterations and alterations can easily be done, without having to re-design the full system. Flexible power distribution enables additional sensors and components to be add on 0.8V, 1.2V, 3.3V, 5V and 12V power rails, with interfaces ranging from ethernet, I2C, CAN-BUS, PCIe and many more. The mechanical variants and integration guidelines are described later in this chapter and also allow for easy integration and re-configuration.

The software component of the environment leverages design templates in the Xilinx Vivado design suite as well as the open-source High Level Synthesis (HLS) tools. This allows users to use existing designs such as that shown in Figure 6.6 to create customized image processing pipelines. A set of IP blocks derived from hardware optimized C++ implementations with HLS are also contained within the environment, and can serve as backbone to developed systems.

The environment also features compatibility with the Nvidia Xavier systems, leveraging a carrier adapter board offered by Leopard Imaging. While these Xavier systems are mostly closed source, they offer high performance to power ratios, and dedicated ISP components to achieve high- performance video capture, with great ease of rapid integration. The Xavier AGX does support sensor synchronization on exposure but handles all ISP processing and compression

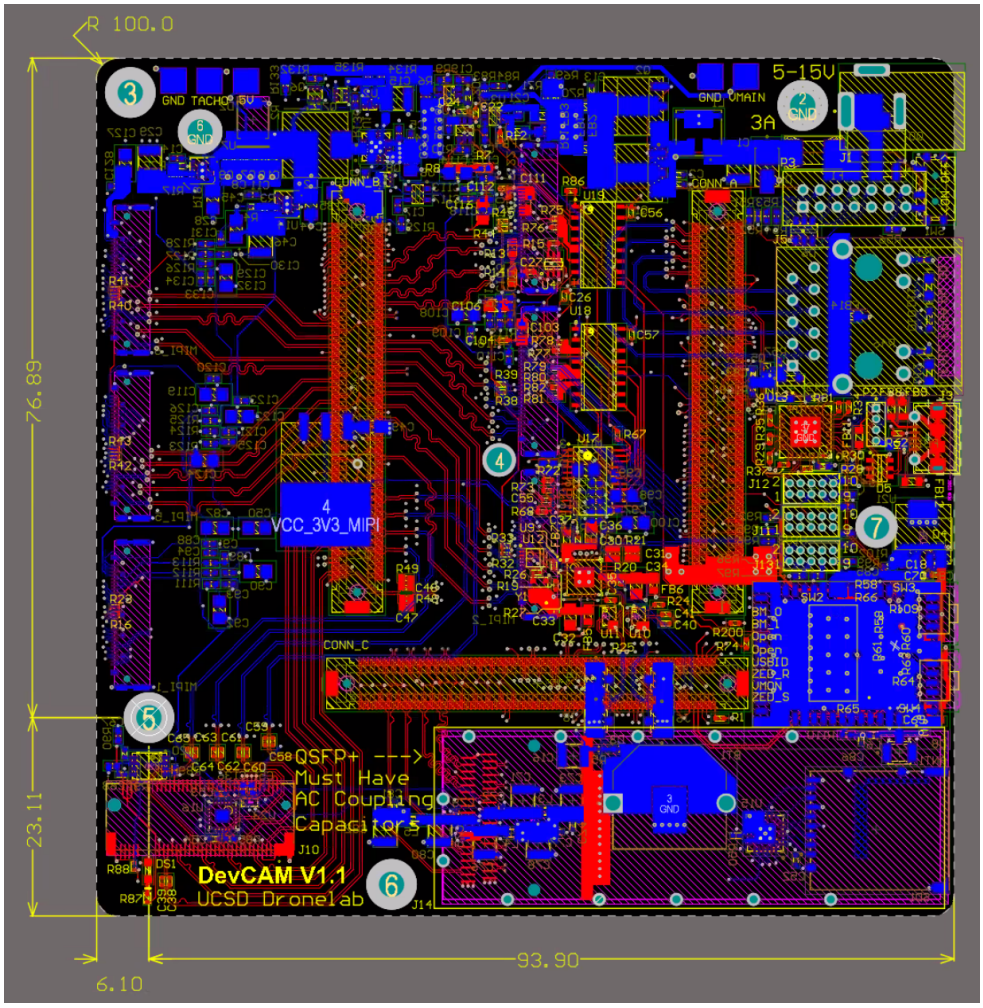

Figure 6.5: DevCAM V1.1 Electrical layout

asynchronously, implying a consistent pain-point to re-aggregate frames in a synchronized manner. Due to the closed source scheduling details of the ISP handler, it was deemed unfeasible to re-aggregate these frames at capture, leaving the requirement to do so after storage into compressed video frames. Figure 6.7 demonstrates the shutter synchronization of the IMX577 sensors to monitor timing precision, while Figure 6.8 emphasizes that as soon as you are dealing with >10E5 frames, captured by multiple cameras on multiple systems with dynamic video encoding, the re-aggregation of frames often requires an external strobe. This issue is directly addressed on the DevCAM FPGA system since capture is synchronized at both the exposure level and on the processing.

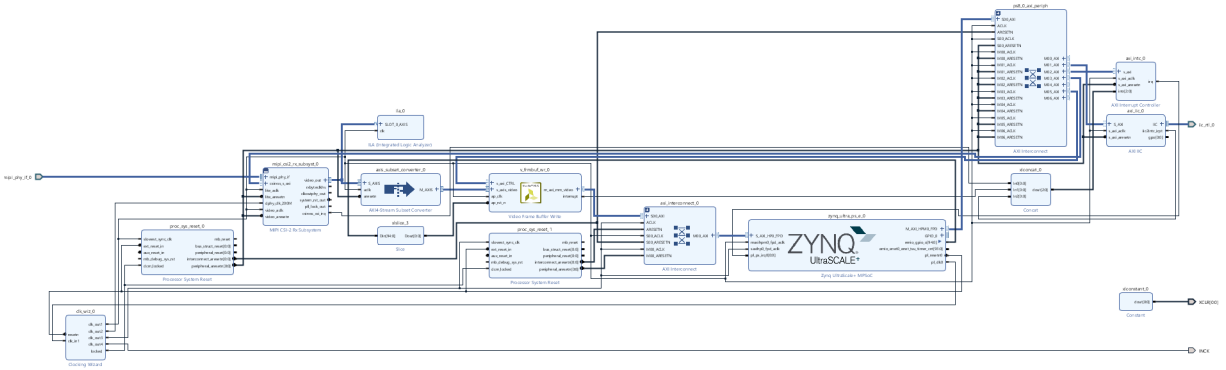

Figure 6.6: DevCAM V1.1 Vivado FPGA Pipeline Template for a single MIPI camera

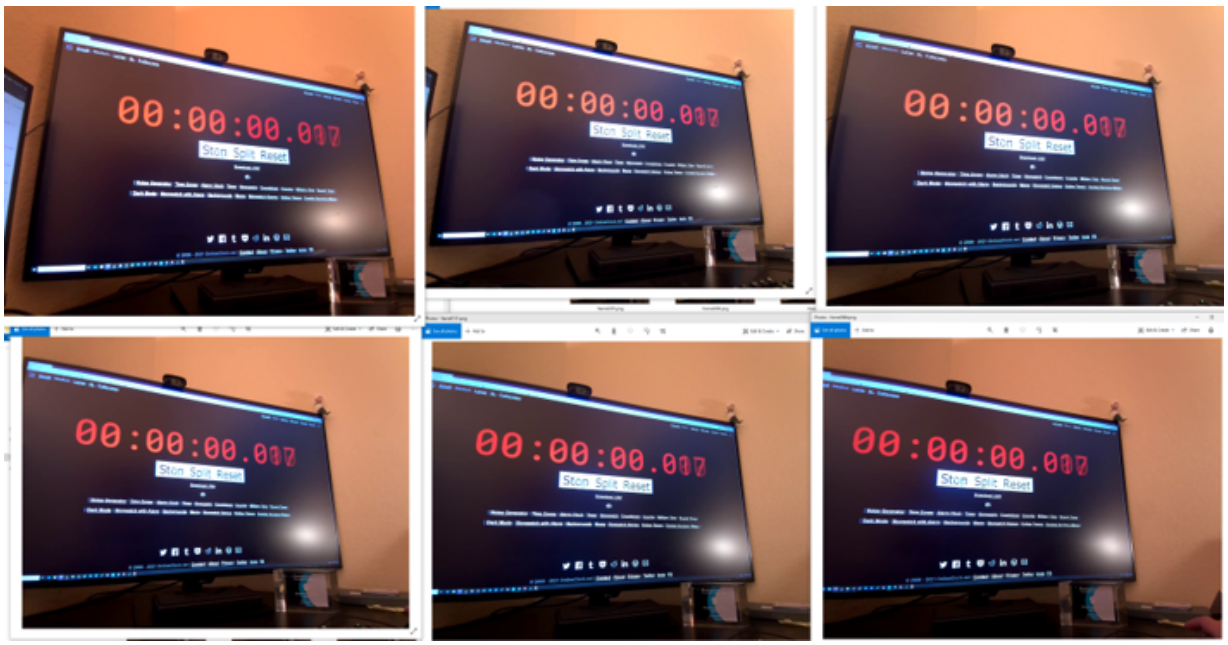

Figure 6.7: 6 camera synchronization Test using the Xavier Capture infrastructure. Due to a monitor frame-rate of 60 HZ, synchronization could only be physically validated to within +- 17ms

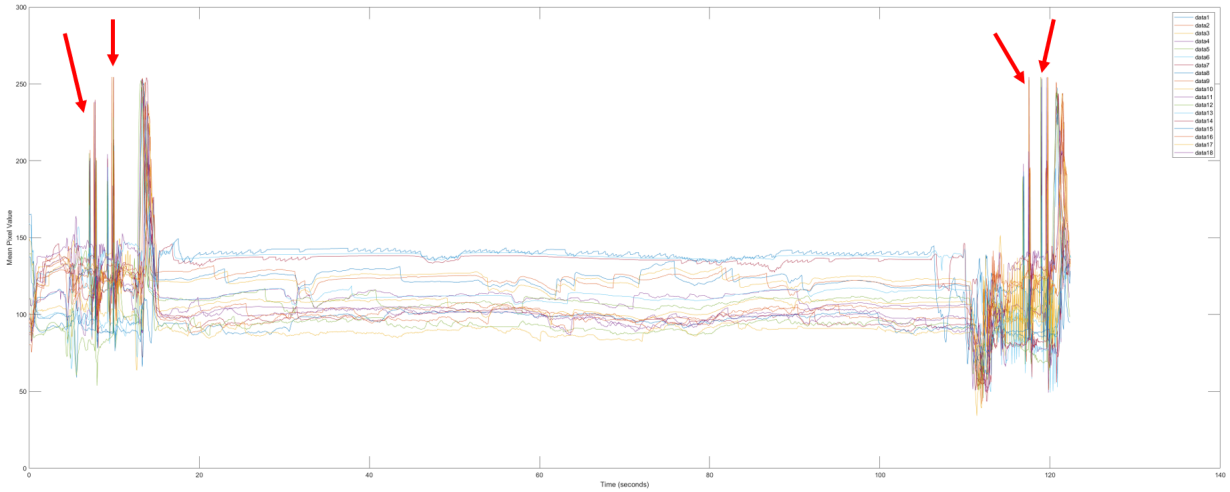

Figure 6.8: Time re-aggregation example from Xavier compatible system. Red arrows indicate the light intensity spikes associated with strobe light to align the timing of the individual captures. Frame timings are extracted from H.265 video encoded frames with variable encoding rate.

6.4 Processing Infrastructure

The DevCAM system uses Enclustra System on Modules (SoM) that are small-form factor PCBs with the Xilinx processor chip, high-quality power filtering, and off-chip memory. On those, are found the core chip architecture, based on an Xilinx Ultrascale+ System on Chip (SoC) that offers a offers a versatile reconfigurable hardware aspect for easy hardware-software co-development. A variety of SoMs are compatible with this development environment, enabling the easy alteration of FPGA/processor size, depending on the desires pipeline and required memory. All image sensors are connectable directly to the fabric where a MIPI interface converts the image streams to a customizable and standardized video streaming format. Furthermore, all these image sensors, IMU and RTK-GPS are also hardware synchronizable to achieve nanosecond timing synchronization. This can be expanded to achieve synchronization between multiple DevCAM systems, over direct hardware connections, or by the use of an inbuilt GPS disciplined oscillator (GPSDO) framework.

The heterogeneous Xilinx architecture has facilitated the interfacing of many different sensors to different parts of the chip. This is largely driven by the interface bandwidth: the

Programmable Logic (PL) is used where high parallelism and high-speed capture is required, and direct connection to the ARM cores is done for sensors with low data-rates, but that benefit from easier driver and software support. A high-level connection is outlined Figure 6.9.

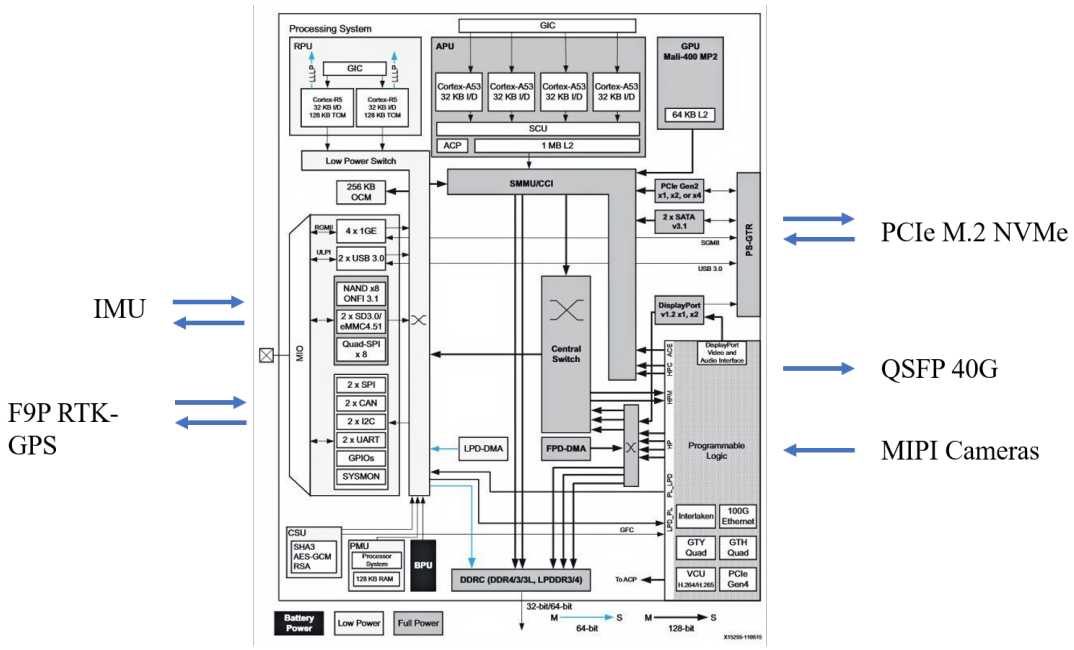

Figure 6.9: DevCAM V1.1 System interfaces and features

A full set of physical interfaces is specified in Figure 6.10. These include a M.2 connector with PCIe 2.0, 4 lanes, routed to the ARM cores (PS) PCIe host controller, and a QSFP module supporting 40G Ethernet networking via the PL. This networking interface requires MAC and PHY modules to be instantiated in the PL of the system, allowing for networking development. A lower-speed 1G ethernet interface is connected to the PS ethernet controller, allowing for easy monitoring and operation of the device. All MIPI camera connectors are directly connected to the FPGA so that the data can be de-serialized and synchronized. All system timing is done using clock domains derived from the GPS PPS signal, allowing for multi-module synchronization down to the microsecond. This time synchronization also allows the IMU, GPS and cameras to be perfectly synchronized.

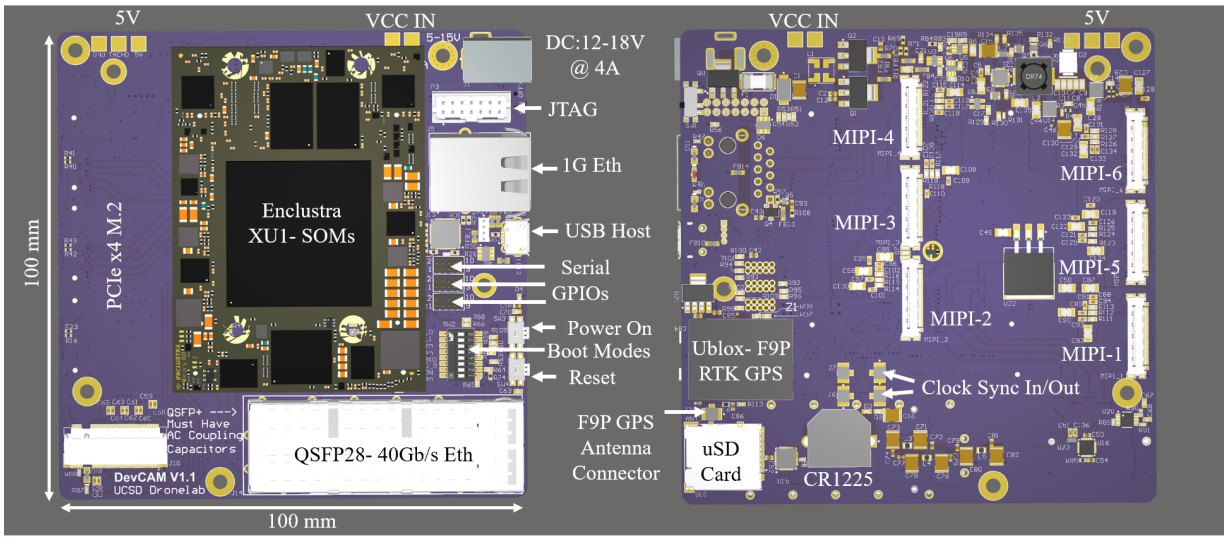

Figure 6.10: DevCAM V1.1 System interfaces and features

6.5 Variants

Figure 6.11: DevCAM Mechanical Configuration Variants- from left to right: Quadnocular, Co-linear Lightfield, Trinocular Panoramic, Panoramic

In this section, a set of mechanical system variants are introduced that are based on the DevCAM development environment. This enables research to be completed using these different

camera layouts, based on the geometries highlighted in Chapter 3. Figure 6.11 summarizes the variants, showing that using the same processor board and camera sensor-lens modules, a large set of experimental platforms can be developed. This section will highlight some configurations that have been built and demonstrated, with emphasis on research opportunities these configurations have to offer.

6.5.1 Quadnocular

The first configuration is a quadnocular array as depicted in Figure 6.12. It leverages 4 of the IMX264 or the IMX577 sensors arranged in a square pattern with a baseline of 70mm between adjacent modules. The system can act purely as a visual-inertial capture device, without an integrated GPS antenna. This configuration provides redundant vertical and horizontal disparity, allowing for the merging of directional disparities and redundant agreement of the observations. Works like [KEFW20] have emphasized the value of trinocular (L-shaped) arrays, which this is the extension of. The configuration allows for the creation of 6 individual stereo pairs (horizontal, vertical and diagonal) for drastic view-point constraining. The system depicted in Figure 6.12 features 4 IMX577s sensors with a square baseline of 8cm.

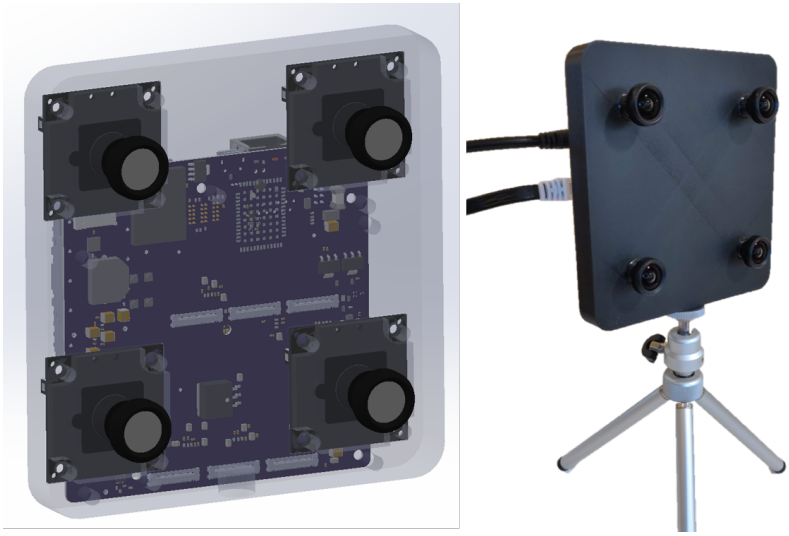

Figure 6.12: DevCAM system in a quadnocular configuration with a four IMX577 sensors

A sample capture of calibration data and real-world scene allowed for validation of the quadnocular system, shown in Figure 6.13. The cameras were calibrated and rectified as separate stereo pairs- horizontally and vertically, forming 4 pairs. The disparity maps were computed with Semig Global Matching (SGM), showing robust matching most areas of the image. These disparity maps were then aggregated based on their mean value across view, that were validly estimated. A 3D reconstruction of the point cloud is shown in the center of the figure.

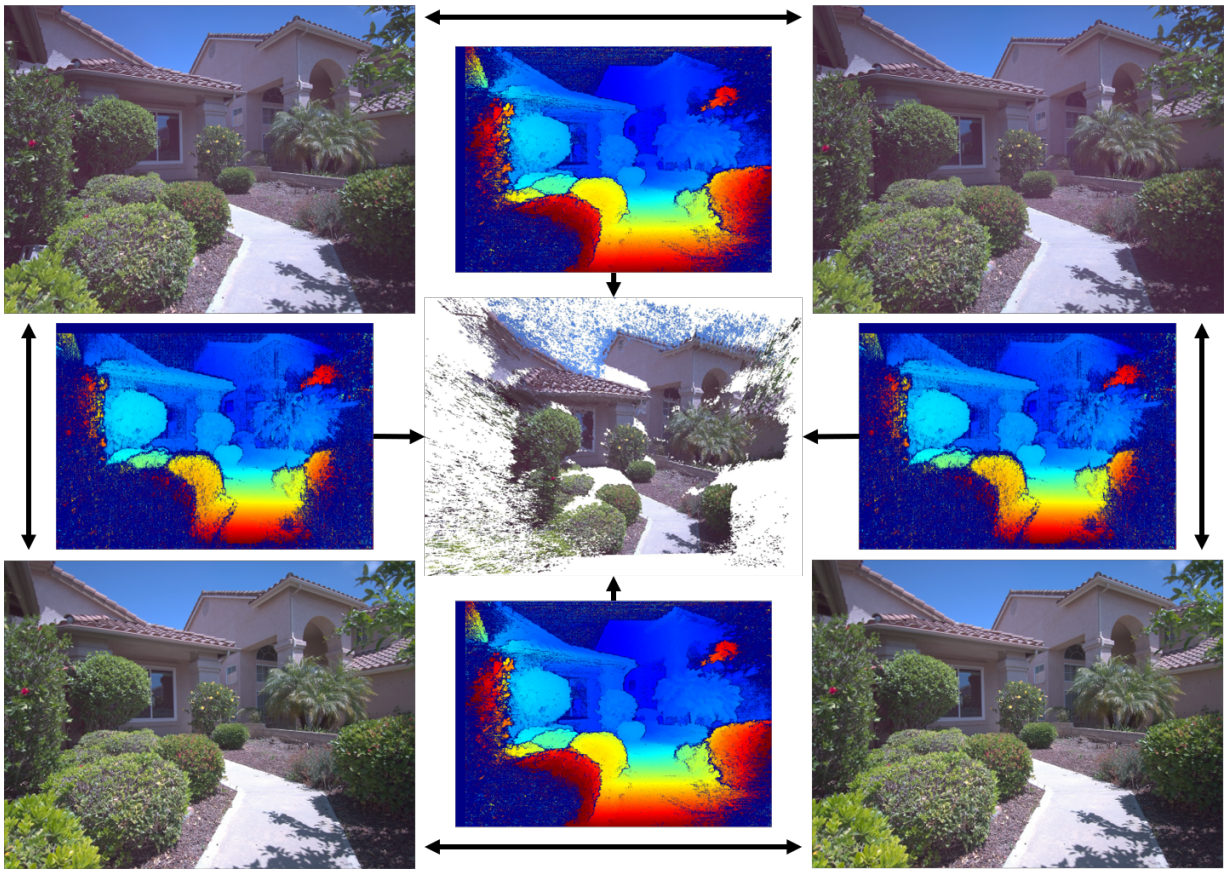

Figure 6.13: Quadnocular Camera Matching and Reconstruction Example Leveraging Semi-Global Matching.

6.5.2 Co-linear Array

Co-linear arrays are a natural extension to the stereo case, due to the constraint of epipolar planes forming epipolar lines across pixel rows of the sensors. This constrains the matching with

the benefit of having view redundancy. Diminishing returns have been noted in 4-camera multi-baseline cases [AN15] and [HSP17]. The proposed 6-camera array features a a 4cm inter-sensor spacing, forming a 20cm overall baseline. Due to the flexibility of connecting/disconnecting sensors, it can serve as an excellent research platform for continuing to evaluate co-linear camera arrays with varying baselines and number of cameras.

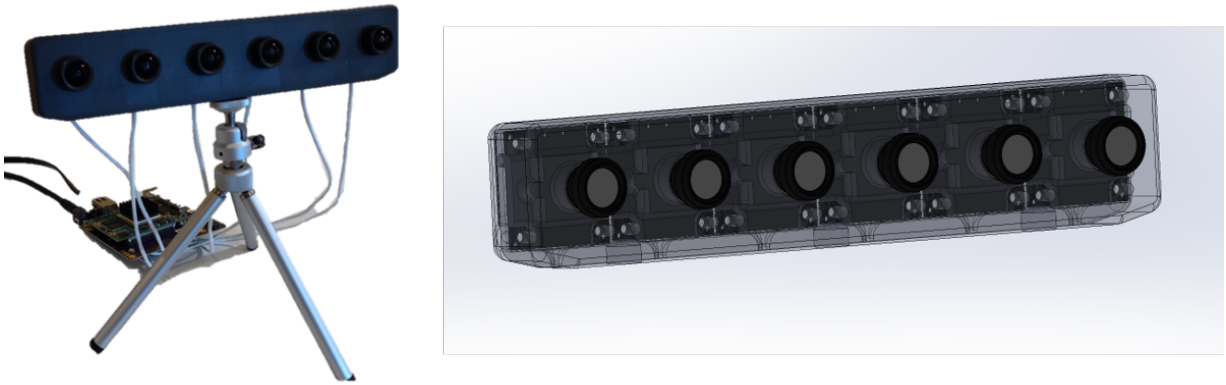

Figure 6.14: 6 Camera Linear Array to Capture single direction light-field scenes.

The linear distribution of sensors has been a favorable constraint in light-field research proposed by [BB89]. Forming Epi-polar Plane Images (EPIs) as shown in Figure 6.16 can be done by taking rectified image data (Figure 6.15) and re-shaping it into stacks of pixel rows from all sensors. The slope $\frac{\delta s}{\delta x}$ of any observed features manifests itself as epipolar lines with where the depth can be directly derived from the focal-length and the slope value (Equation 6.1). This often facilitates the matching, improving the depth estimation reliability over stereo.

$$Z = -f\frac{\delta s}{\delta x} \tag{6.1}$$

6.5.3 Panoramic

A monocular panoramic configuration shown in Figure 6.17. This simple, yet powerful configuration provides a simple array to capture a full surround scene with IMU and GPS data

Figure 6.15: Sample data capture of linear 6 camera array with the DevCAM system.

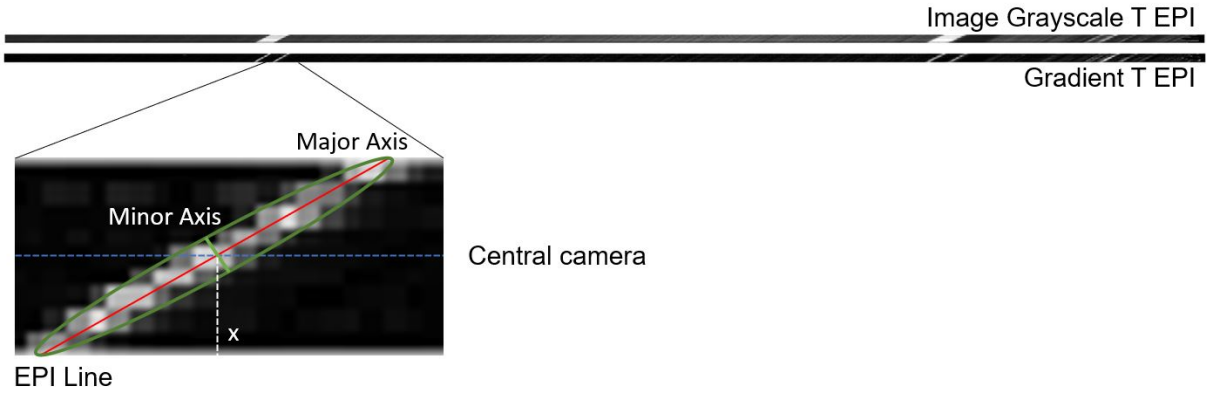

Figure 6.16: Epipolar line example demonstrating the disparity relationship across views. The slope of the features is directly proportional to the depth of the feature.

in a compact form-factor. Applications include street-view style captures for panoramic video creation, or indoor/outdoor scene capture. In this configuration, the sensors are rotated into portrait mode to allow for a wider vertical FoV to be captured in additional to the fully panoramic horizontal FoV. Every camera module's FOV overlaps a total of 24 degrees to ensure full coverage. Due to the high per-sensor resolution, an array of this sort is able to achieve signiicantly higher spatial resolution over conventional Pan-Tilt Zoom cameras, without any mechanically moving components, and at higher-frame rate. A prior intrinsic calibration can allow us to rectify frames with a naive projection to an infinite sphere using the CAD extrinsics.

Figure 6.18 shows an example of a single frame capture and stitch using the panoramic DevCAM system.

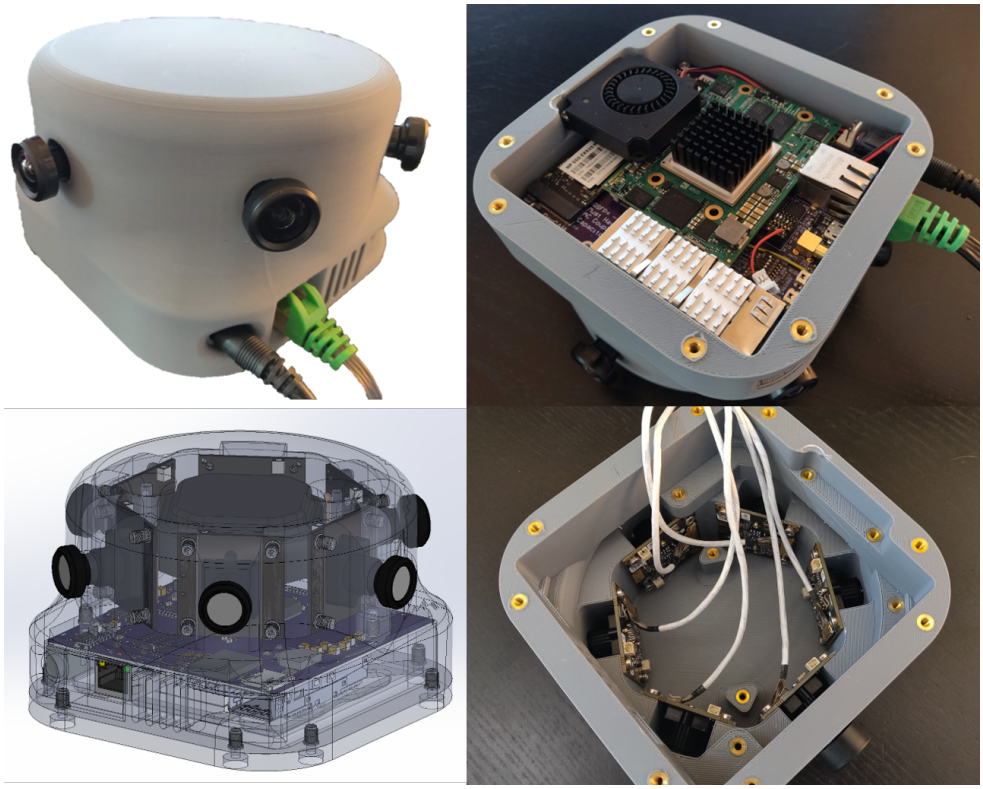

Figure 6.17: DevCAM system in a monocular panoramic configuration with a six IMX577 sensors

Figure 6.18: Monocular panoramic stitch result from the 6 camera, DevCAM system

6.5.4 Trinocular Panoramic

The final configuration introduced in this package is a trinocular panoramic system. This array leverages a total of 18 sensors packaged into 6 sets of trinocular views. This novel layout features the benefit that each viewpoint is observed from at least 3 cameras, that use a dual disparity constraint- horizontal and vertical. Similar to the panoramic configuration, this array has portrait oriented sensors to increase vertical FoV while keeping a full panoramic horizontal FOV.

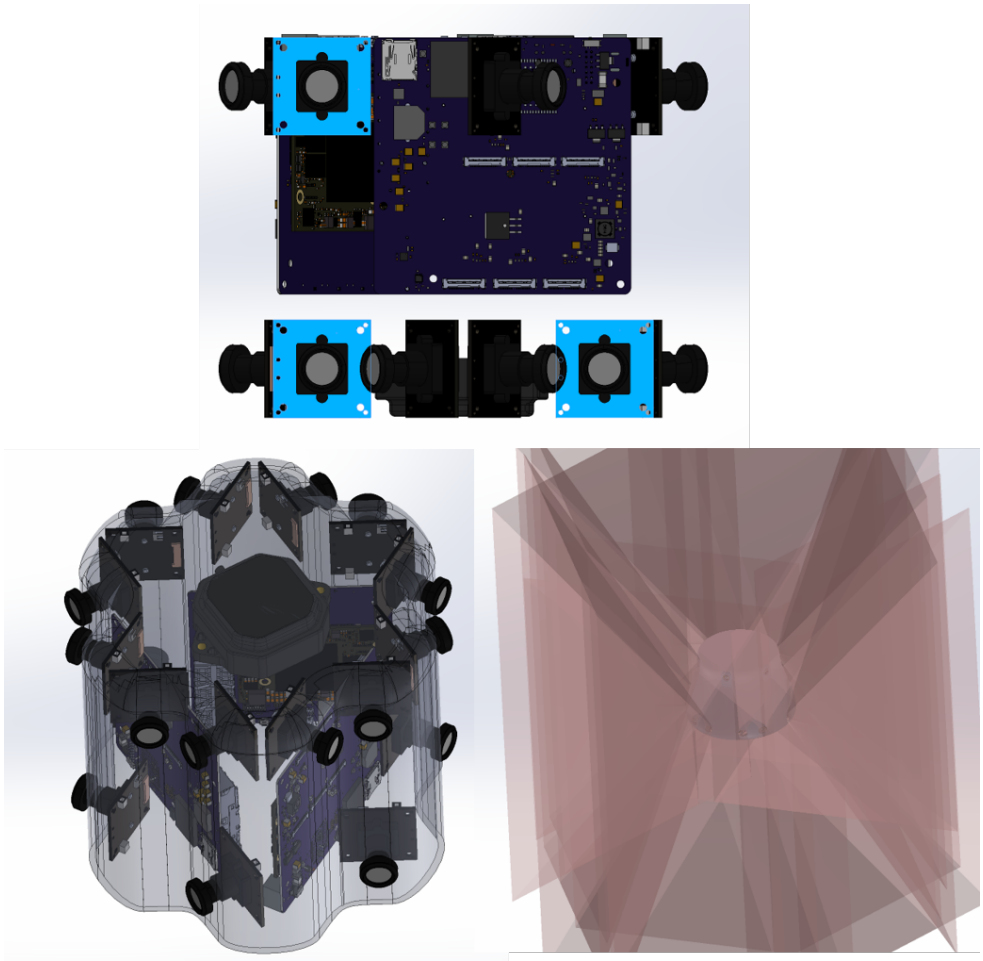

Figure 6.19: Trinocular Panoramic configuration with 18 IMX577 sensors, connected to 3 separate DevCAM processor boards which are can be electronically or network synchronized.

The array and it's respective FoV cones are shown in Figure 6.19. The sensor module disparity in this implementation is 110mm, limited by the camera arrangement to allow interleaving, as well as case manufacturing feasibility on an additive manufacturing printer. This system has to use a total of 3 DevCAM boards to interface with the 18 sensors, leveraging inter-board synchronization. Each board drives 6 cameras, which is 2 trinocular sets. The system can also feature a roof-mounted RTK-GPS antenna with a 100mm ground-plane and respective IMUs on the PCBs, resulting in a total of 6 IMUs in the system, mounted at 45 degrees to each other. A central air-flow column guarantees the evacuation of heat from the system. A mechanically rotating LiDAR can also be mounted in lieu of the GPS antenna, allowing for geometric comparison of

vision reconstructed data to active sensing. Figure 6.20 shows an Ouster OS0-128 co-located on the system- all timing synchronization is done over PTP network time.

The extrinsics of the system are known from the mechanical CAD design. This allows us to directly know the transformations between respective trinocular sets, as well as that of the LiDAR and any IMUs.

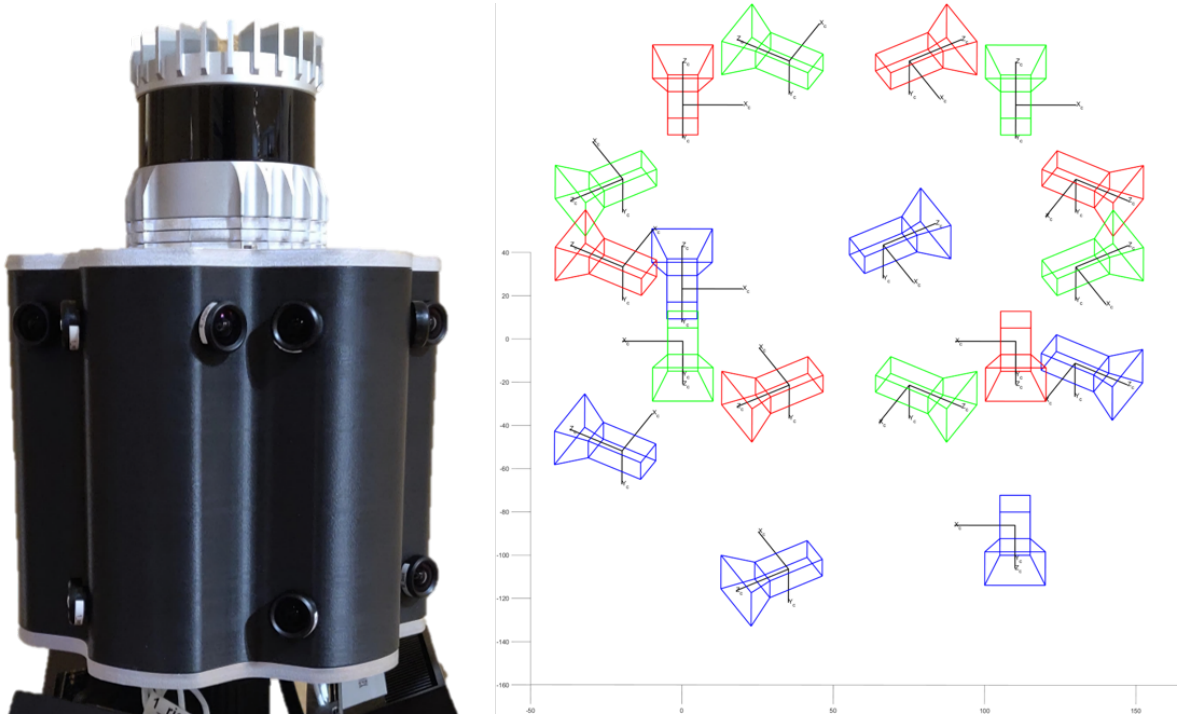

Figure 6.20: Trinocular Panoramic configuration as built. Left- system with an Ouster OS0-128 LiDAR. Right- CAD based extrinsic camera plotting.

This trinocular panoramic array serves as the foundation for all data capture and view reconstruction described in the final Chapter 7. Sample data and reconstructions will be shown there.

6.6 Acknowledgments

This chapter is, in part, currently being prepared for submission for publication of the material, titled "DevCAM: An Open-Source Multi-Camera Development System for Embedded

Vision" in Electronic Imaging 2021. Meyer, Dominique Ernest; Birlangi, Meher Akhil; Kuester, Falko. The dissertation author was the primary researcher and author of this material.

Chapter 7

Trinocular-Panoramic Reconstruction

7.1 Introduction

This final chapter unifies the various components formulated within this dissertation to validate the hypothesis and demonstrate real-world results. It leverages the trinocular epipolar geometry introduced in Chapter 3, the feature matching described in Chapter 5.1, on data captured using a custom multi-camera system proposed in Chapters 5 and 6.

Throughout the autonomous driving community, there has been an ever standing debate on the provided value of LiDAR and cameras for scene perception. A plethora of research areas have emerged from sensor development, including LiDAR, stereo cameras, and radar, through reconstruction algorithms such as Simultaneous Localization and Mapping (SLAM), Sceneflow and detection and tracking algorithms. Notable datasets [GLU12], [GLSU13], [CBL$^+$20], [COR$^+$16] have evolved to include an ever growing number of different sensors, all to encourage the exploration through an exhaustive search of methods to answer a set of simple questions: how is the system moving in the world, and what is surrounding it? Cameras have the disadvantage of providing only 3D world projections in the form of images over an optically constrained FoV, loosing the necessary 3D surround geometry to plan navigation. LiDARs on the other hand

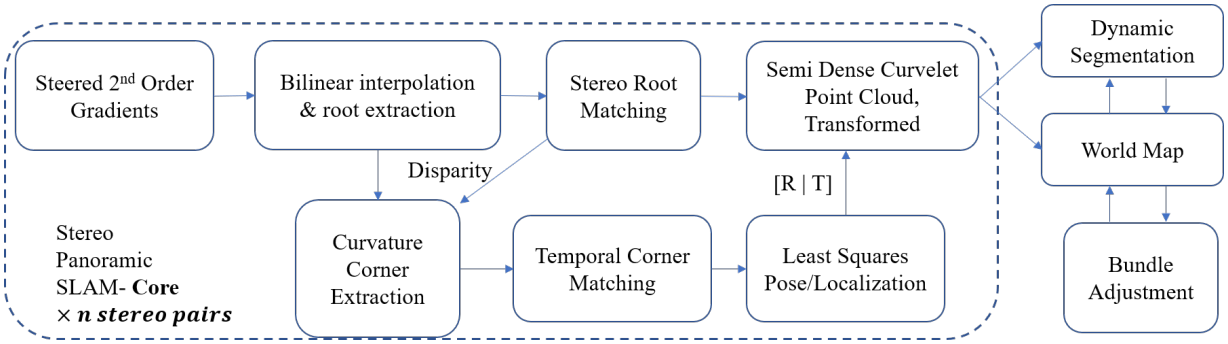

Figure 7.1: Semi-dense Variant of the Stereo Panoramic Mapping Workflow

suffer from poor spatial resolution, reducing the visual cues to semantically understand scenes. This work leverages a small-baseline, trinocular panoramic camera array to unite the existing pain-points and investigate the potential of vision-only sensing and perception.

Systems have up to now been dominated by LIDAR sensing, in order to meet the depth accuracy required for safe navigation. In multi-view systems, the reconstruction accuracy is directly dependent on the estimate of camera locations, and the matched feature localization in image space. Depth ranging is highly sensitive to disparity accuracy for stereo pairs, and as such, a key aspect to the proposed work focus is on achieving the highest feature localization accuracy to estimate right disparities, and consequently directly reconstruct geometric features in 3D space. While the degree of sub-pixel spatial accuracy in image processing varies between camera calibrations, image rectification and matching, we argue and demonstrate that it is possible to achieve spatially correct reconstructions up to more than 20 meters when leveraging small inter-ocular baseline cameras, all while reconstructing a full-surround scene, without sacrificing spatial resolutions and maintaining computational efficiency.

To begin, we must revisit the overarching objectives that have been covered thus far. We desire a geometry within our rig which facilitates computation, allowing us to strongly constrain the system such as to directly estimated geometry from a view-matching approach. To reduce mis-matching errors, we leverage view redundancy in the form of trinocular sets, that have a pre-defined baseline for mechanical integration. Since the array is contained within a single system,

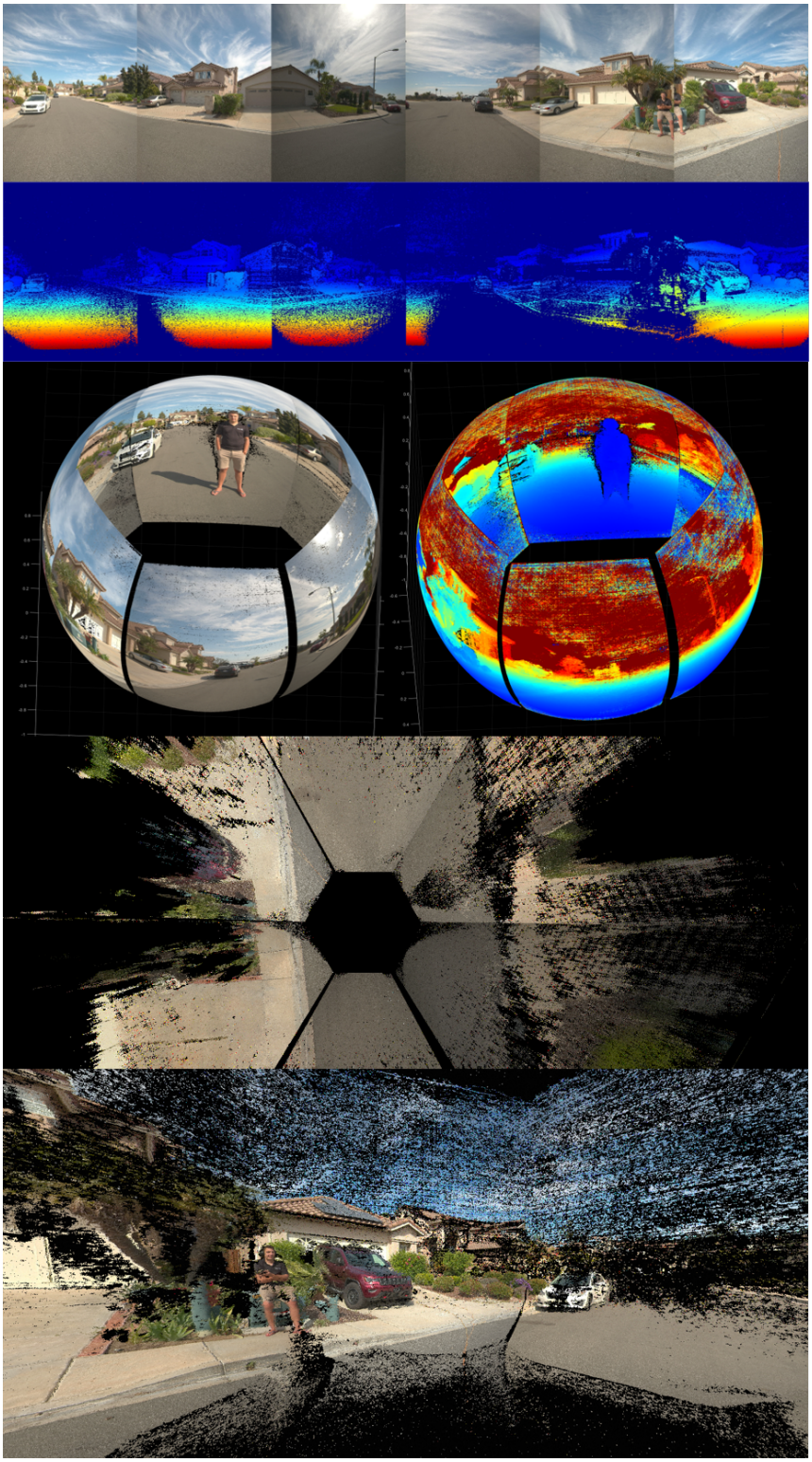

Figure 7.2: Data inputs and derivatives from the trinocular panoramic system. Top to bottom: Reference sensor views; Disparity Maps; Surround spherical projections of reconstructed scenes as colormaps and depthamps; Bird's Eye View of 3D point cloud; 3D point cloud rendering.

we are able to robustly calibrate it prior to deployment and predict the ranging performance as a cost function from the disparity accuracy, optics and baseline. While the baseline for this system was at 11cm- a comparatively narrow baseline to the 50+cm normally used on vehicles, this allowed us to package the system into that single unit. Despite this physically small baseline constraint, we are able to demonstrate that it proves sufficient for ranging up to approximately 20m due to the high sensor resolution of 12MP per. Finally, our desire for omniscopic vision prescribes our design to pattern the trinocular sets in an interleaved fashion to cover a surround view circle. In this work we explore the assembly of the system components, spanning hardware and software to evaluate the value of constrained multi-view arrays for efficient scene reconstruction in an outdoor navigation setting.

A unified vision framework for fast 3D localization and mapping of stereo panoramic camera systems using semi-dense and dense features is presented. Parallel interleaved trinocular arrays are used to geometrically constrain the stereo matching of semi-dense and dense features for ego-motion and scene geometry estimation. The semi-dense contours are formed from a novel set of spatial curvelets which were introduced in Chapter 5.1, derived from matched local roots of steered Gaussian second order derivatives. The reconstructions achieve favorable depth accuracy due to the improved spatial localization compared to state-of-the-art dense energy minimization approaches at reduced computational complexity. Dense evaluation is performed with the well accredited Semi-Global Matching (SGM) algorithm proposed by [Hir05]. Temporal associativity and full-surround localization is done using traditional sparse corner features, matched over frames, with a global bundle adjustment formulation for optimization. This final bundle adjustment is formulated to leverage the array constraint to optimize both pose and world reconstruction. Qualitative results are shown on both synthetic data rendered using CARLA as well as data captured from the trinocular panoramic camera sets to demonstrate reconstruction within a real-world setting.

Figure 7.3: Interior, high-angle calibration procedure for the trinocular panoramic camera array

7.2 System Calibration and Data recification

The trinocular panoramic system leveraged cameras with sensor-lens combinations that casted a wide, 86 degree horizontal FoV, with a minimum, fixed focus distance of 3m. As such the system was calibrated using a calibration target, with motion to maximize the FoV sampling coverage, with a minimum imaging distance of 3m. These practical requirements made it such that the Fractal target proposed in Chapter 3 was not feasible due to the resultant small size of the observed features which were unreliably detected in tests. A checkerboard calibration was therefore chosen, with the trinocular sets being calibrated independently. The inter-set extrinsics were assumed ideal from CAD design and were not calibrated.

The calibration resulted in real-world scaled intrinsic and extrinsic parameters which were sequentially used to rectify internal distortions κ, and secondly to align sensors due to mechanical manufacturing tolerances. This was done for every trinocular set sequentially resulting in image frames that have equal number of rows and columns of pixels for all images. All relative poses between trinocular sets remained un-calibrated prior to processing, and the CAD designs were used as as-built values. Calibration within each trinocular set defines the disparity accuracy

which directly affects the depth estimation. A small deviation from design would lead to a large reconstruction error, defining the need for correct multi-view calibration. The relative poses of the sets are however less stringent on this requirement as they are solely used for re-aligning the respective point cloud together.

7.3 Data acquisition

The datasets were split into synthetic and real-world datam, with the goal to quantify and validate the algorithms with ground-truth depths and motion of the synthetic, and to validate the system capabilities in a real-world setting. This section describes the approaches taken to capture/generate these datasets.

7.3.1 Synthetic Data

For synthetic data generation, we used CARLA ([DRC+17]) version 0.9.10, an open-source simulator, to render photo realistic scenes. We use the City 10 HD simulation model, and generate data from a vehicle mounted camera rig that is driven around the simulated city containing pedestrian, motorcycle and vehicle actors. For our panoramic camera module, we leverage the CAD designed camera parameters and data-sheet provided focal-length for FoV estimates. We collected 1,000 images at 30 frames per second. For each trinocular set and at every time-step, we also generate ground truth depth maps and semantic segmentation maps for the reference sensor (top-left), collect the absolute pose/dynamics of each camera and the vehicle in world coordinate frame, serving as our ground truth data.

7.3.2 Real-world Data

Real-world data was captured using the trinocular panoramic array introduced in Chapter 6. The system consists of 18 cameras, broken down into 6 separate trinocular sets. All images

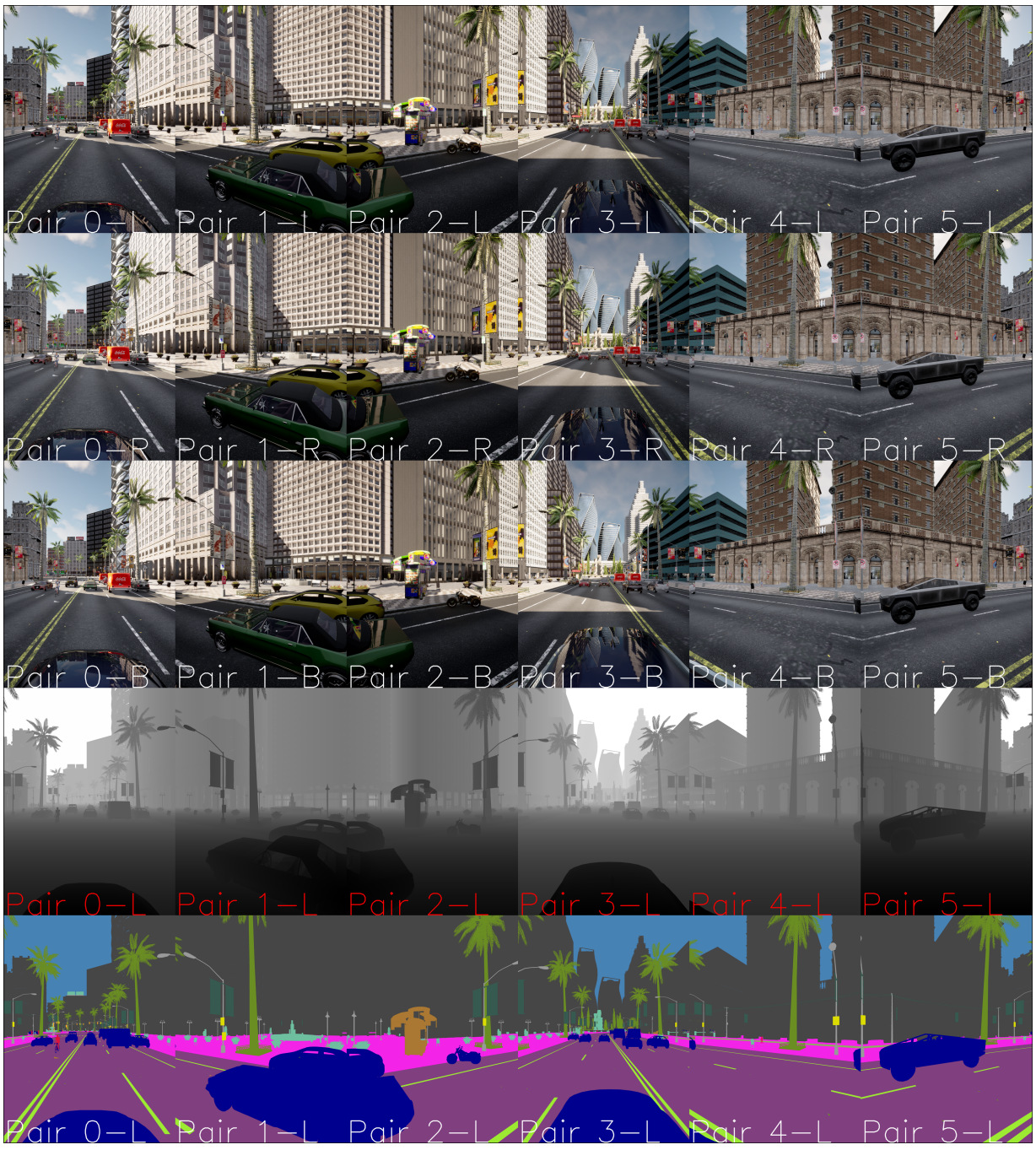

Figure 7.4: Synthetic Panoramic Data from CARLA simulator. From top to bottom: top-left camera views; top-right camera views;bottom-left camera views; ground-truth depthmaps; semantic segmentation maps

Figure 7.5: Trinocular Panoramic System Data Acquisition on an Urban Road

were captured simultaneously while the system was statically mounted on a rigid tripod (Figure 7.5. Further tests with high-frame rates captures were completed, but synchronization tests highlighted bugs which did not guarantee perfect synchronization. Future work will include processing data at the full capture capabilities of 30fps for the whole system, once the DevCAM hardware outlined in Chapter 6 has proven stable. At each capture timestep, frames are aggregated into a folder on a central compute host, alongside timestamps and a LiDAR scan. A sample dataset from a single time-step is shown in Figure 7.6. For this dataset, 2 sequences were captured: the first keeping the camera stationary with an actor moving away from the system, and with the second, the system moving in a linear fashion down a road, with no scene motion. All image data is rectified using the previously estimated parameters.

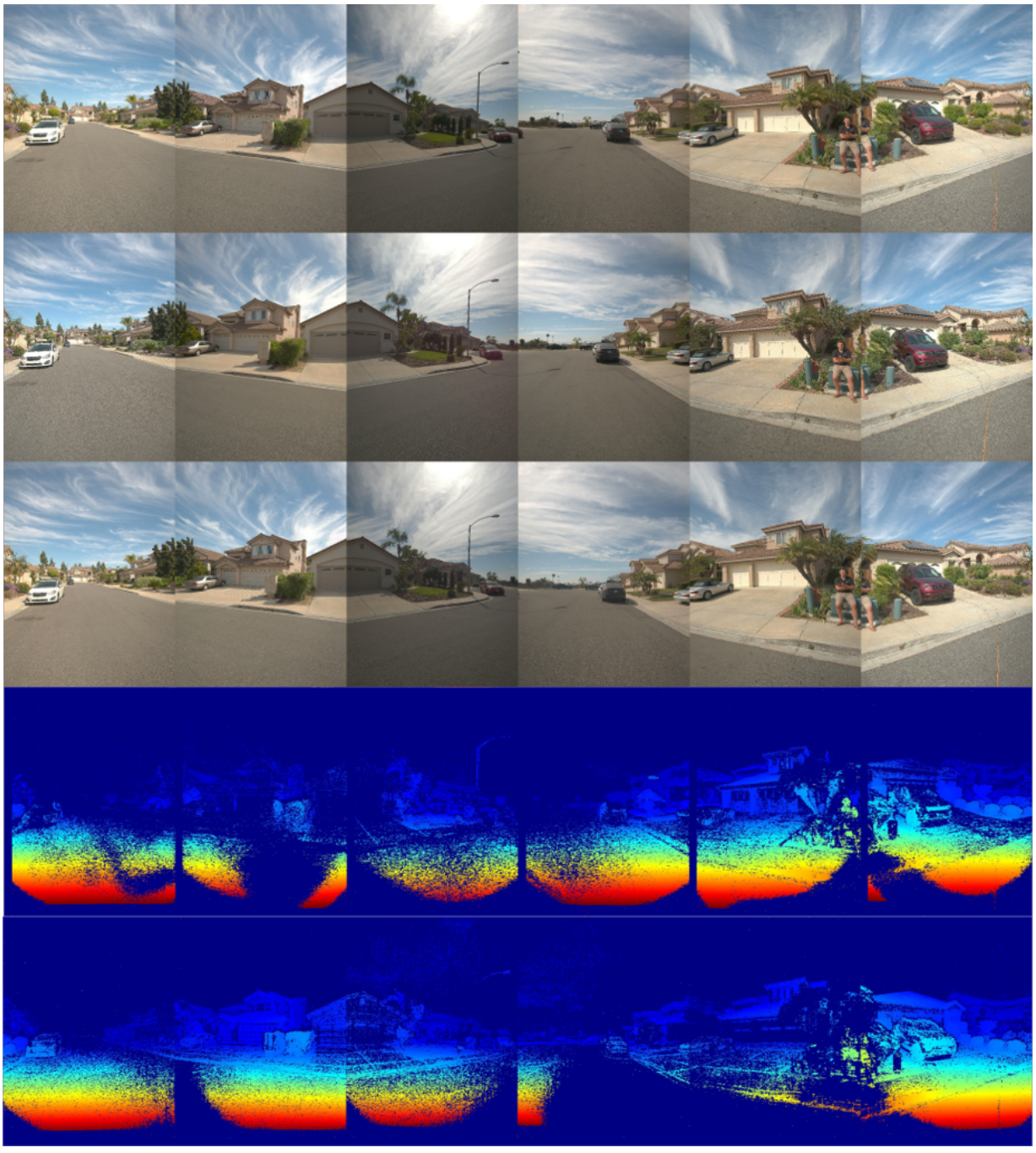

Figure 7.6: Real-world data at timestep 0. From top to bottom: top-left camera views; top-right camera views;bottom-left camera views; horizontal disparity maps; vertical disparity maps

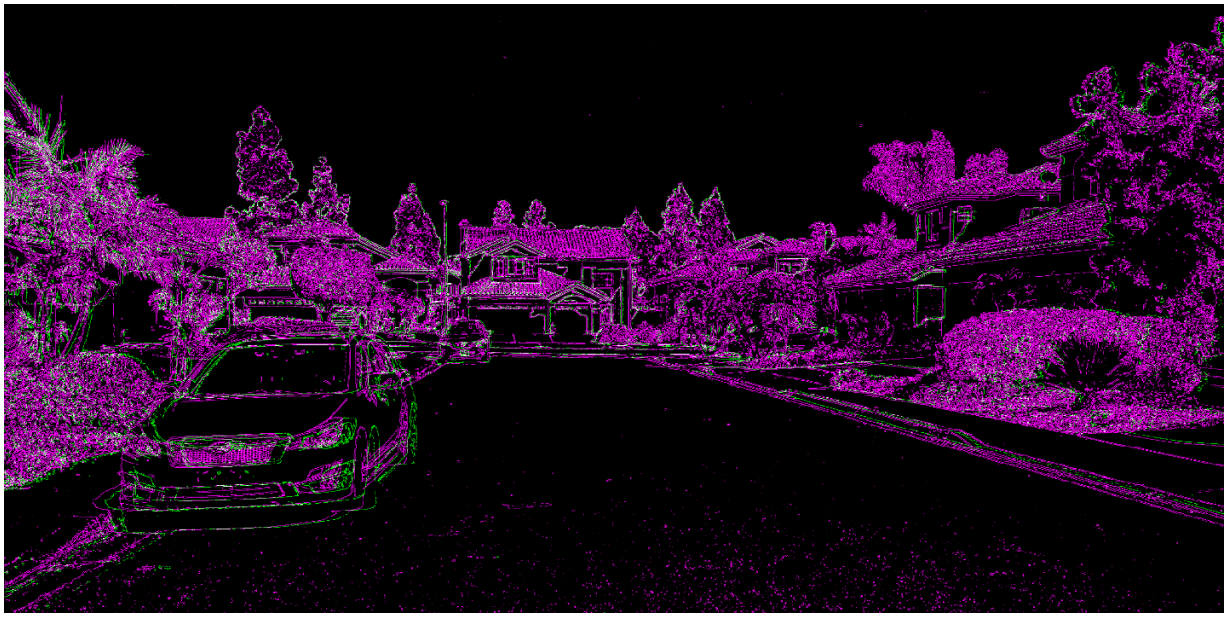

Figure 7.7: Semi-dense edge extraction using steerable filtering on top stereo set of trinocular panoramic camera

7.4 Reconstruction

This section is divided into a summary of multi-view matching approaches used for the surround reconstruction of the scenes

7.4.1 Semi-dense

The first matching approach used is that introduced in Chapter 5.1. Second order steerable filters are used with a sigma of 1.5 to find the roots of the derivative image. These roots are then matched in 1D, across epipolar lines within the disparity range, such as to provide a continuous disparities with sub-pixel accuracy. Figure 7.7 shows a sample crop of the root extraction for the top stereo pair of set 0 in the trinocular panoramic array. Due to ISP inconsistencies on the capture system, certain frames suffer from reduced sharpness, leading non-similar edges across views.

Figure 7.8 shows an example of this semi-dense reconstruction using a single stereo pair

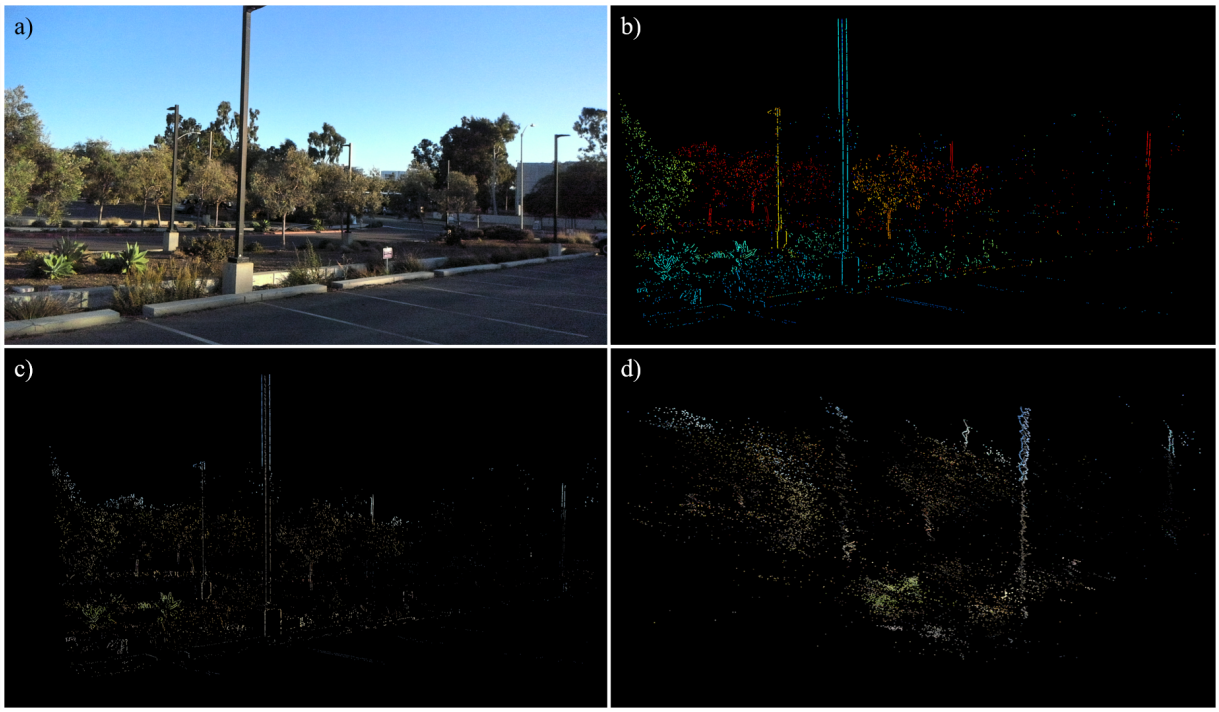

Figure 7.8: Sample Semi-Dense Stereo reconstruction: a) Original Left image from stereo pair, b) Semi-dense Depthmap of Contours, c) Original contours, d) Semi-dense Point cloud reconstruction.

from the Starcam system, outlined in Chapter 5.1. While the output is largely sparse, the main scene components such as the outlines of visible objects are clearly resolved.

7.4.2 Dense

The second core component of the reconstruction involves the commonly used Semi Global Matching algorithm (SGM) that matches patches to maximize the Mutual Information (MI). The global optimization stage of the algorithm imposes smoothness constraints such as to minimize the allowed change in disparities across local neighborhoods. While this smoothness constraint handles continuous surfaces decently, it often fails at discontinuities such as edges. Furthermore, this matching approach relies on texture to ensure the macthes are correct. If there is no texture information to work off, there is no reliable way of matching an area. In the case of the trinocular panoramic system, capturing a road environment, nearby surfaces such as

the tarmac and cracks reliably reconstruct, while distant, smoothed surface such as sky, house walls and reflective car surfaces often fail. This algorithm is readily implemented with hardware acceleration on GPUs, becoming a decent choice for intergation for a surround reconstrution mechanism.

7.4.3 Disparity Aggregation

The benefit of view redundancy is up to this point unused. We want to enforce the known constraint that the disparity for equal-baseline system, with horizontal and vertical views, must be equal. In reality, the slight difference in calibrated baselines makes us constrain the system as follows:

$$Z_{hor} = Z_{ver} = FL_{hor} * \frac{baseline_{hor}}{disparity_{hor}} = FL_{ver} * \frac{baseline_{ver}}{disparity_{ver}} \tag{7.1}$$

From this constraint, we can choose our matches in both semi-dense and dense matching cases, such as to minimize the combined cost of both horizontal and vertical matching. The work by [KEFW20] has demonstrated the added benefit of doing so, with the limitation that they do not consider a confidence and consequential outlier rejection. There are also cases where averaging the disparity cost would incorrectly weigh towards the less reliable direction. The most robust matches are generally those where the dominant gradient direction is perpendicular to the matching direction.

7.5 Results

Both the synthetic and real-world results were processed to evaluate the reconstruction performance. Ground-truth data only exists for the synthetic captures, so quantitative evaluation could only be performed on the synthetic set. Due to the difficulty in measuring the

Figure 7.9: Synthetic Reconstruction Evaluation in Bird's Eye View and Side-view: (top) Semi-dense contour reconstruction, (middle) Dense Semi-Global Matching, (bottom) Ground Truth.

correct reconstruction, we focus on the disparity correctness- the most common metric in stereo reconstruction.

7.5.1 Synthetic Validation

The first evaluation is done using the synthetic CARLA dataset, by looking at the depth estimation accuracy. The same timestep's data was reprojected using both the semi-dense and dense methods, and compared to the ground truth depths. Figure 7.9 shows projections in both Bird's Eye View (BEV) and side point cloud renderings. The semi-dense reconstruction is noticeably sparser than the dense reconstruction, While all sets of projections, including the ground-truth, suffer from the spatial resolution fall-off associated with distance, it is more drastically accentuated with the semi-dense due to the lesser features to begin with.

The evaluation of correctness of disparity is done on a per-pixel metric for both the dense and semi-dense. Since only certain pixels are valid for the dense reconstruction, and the same

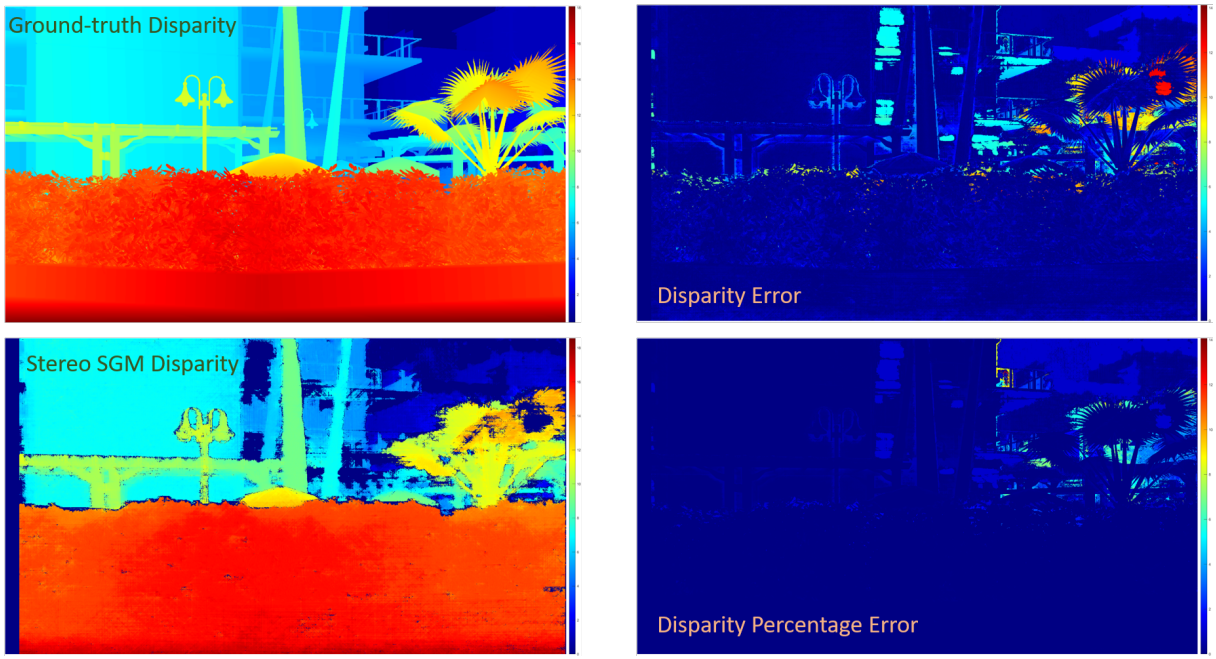

Figure 7.10: Evaluating SGM Dense Disparity by Comparison to Ground-truth Disparity

with the semi-dense, only those valid pixels are compared to the ground truth. Figures 7.10 and 7.11 summarize the metrics on a single frame. The 2 major considerations are as follows: firstly it is important to evaluate the number of pixels with a large error, which are cases of mis-matches. The second being the magnitude of the noise for small- error disparity values, these assume a correct match, but that are subject to the systematic noise, linked to the sub-pixel estimation accuracy of the disparity.

The histograms of both semi-dense and dense pixel disparity errors summarize statistics as follows: The SGM-dense maps has a mean disparity error of 0.67px, with a standard deviation of 1.49px, while the semi-dense have a mean error of 1.21px and a standard deviation of 2.57px. This shows that the average disparity error is more favorable with the SGM matching approach. Observing the histograms also emphasize that the semi-dense shows an increased outlier count with errors greater than 2px, compared to that of the SGM dense matching approach.

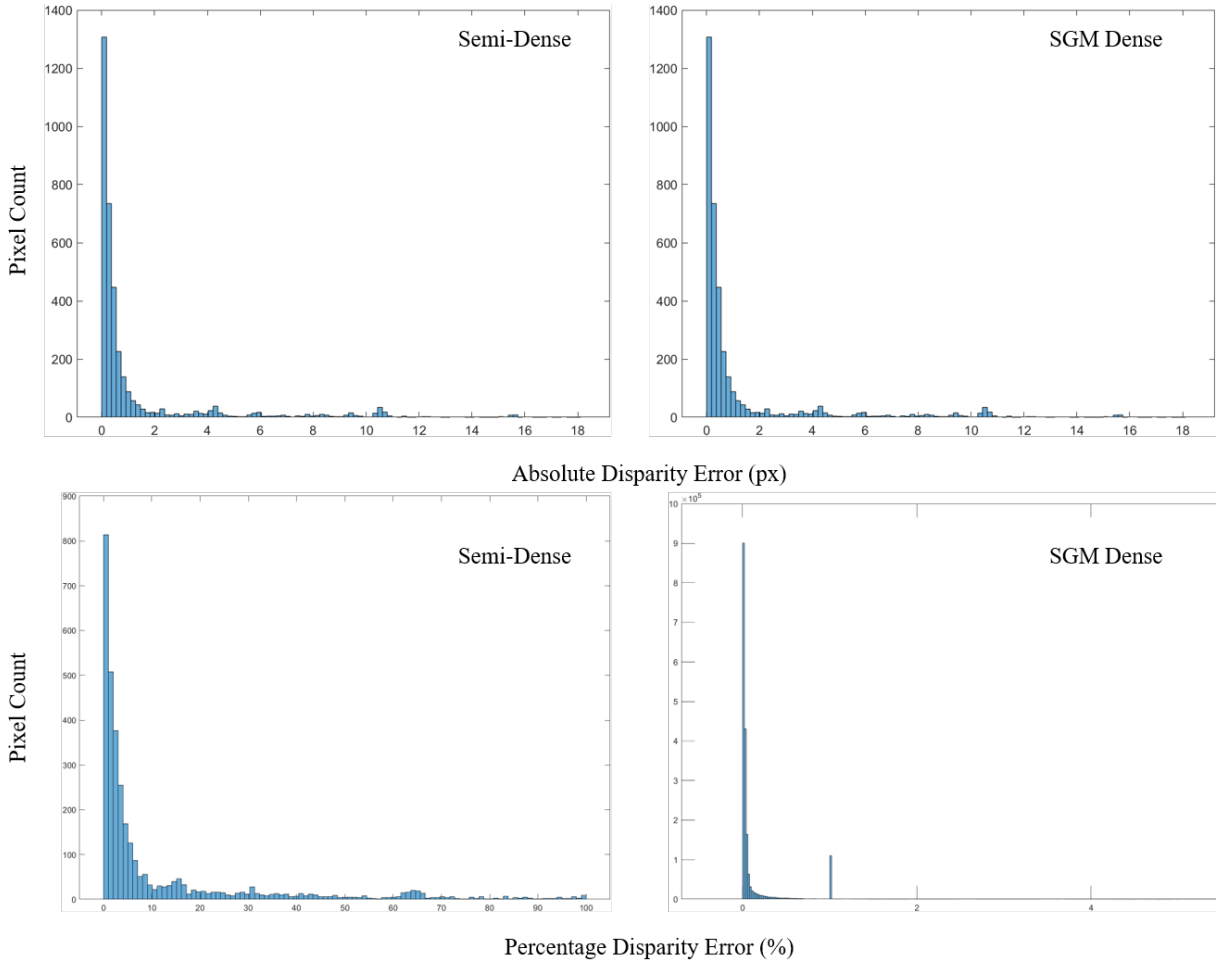

Figure 7.11: Histogram of disparity errors of valid semi-dense adn SGM dense disparities compared to ground-truth. Evaluation on a single frame.

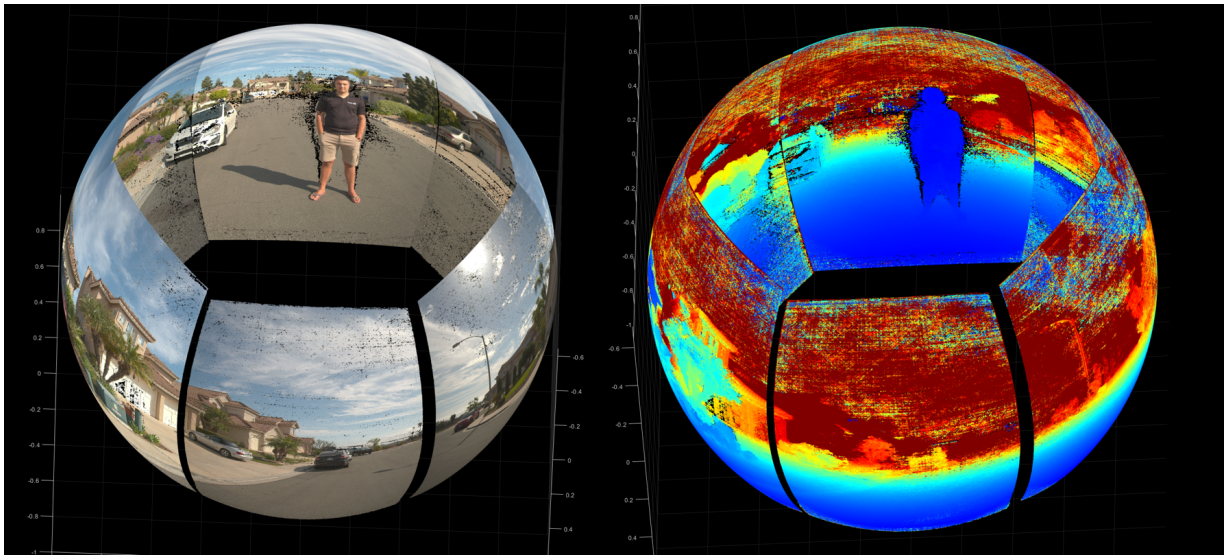

Figure 7.12: Trinocular Panoramic Image Data with Respective Depth Maps

7.5.2 Real-world Results

We now illustrate the real-world results from the trinocular panoramic system. In these, there are no ground-truth depths/disparities, so the evaluation will be constrained to a qualitative review. All reconstructed scenes are projected to 3D point cloud, which are then either back-projected to a unit sphere, centered about the system origin, or into a 3D rendering of the reconstructions. Figure 7.12 shows re-mapped RGB and Depthmaps in spherical projections. These are visual representations that easily allow for visual evaluation of estimation validity. These spherical projections can be plotted over time to show motion from the system's surround perspective.

The next valuable representation consists of a Bird's Eye View (BEV) map, thresholded to within a height range above the dominant plane in the scene. This representation is often used on 2D lidar clustering of objects. The geometric estimation and top-down rendering explicitly implies the homography to go from image projections into the ground plane. Figure 7.13 shows the bird's eyeview for 3 timesteps of the dynamic motion sequence- where the camera rig is moved between frames. The white car to the top of the system center is a useful representation

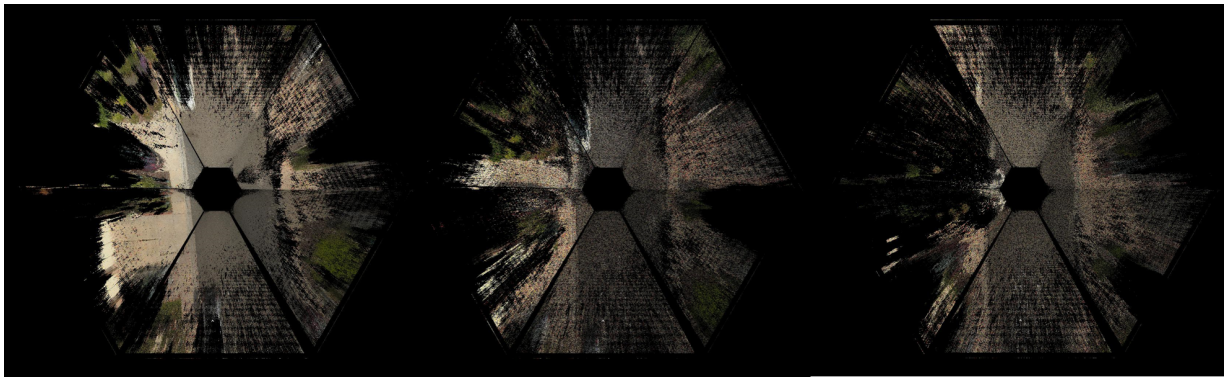

Figure 7.13: Bird's Eye View of Respective Timesteps- t= 1, 10, 20

point, which the system moves towards and next to. This data representation is synonymous to that of LiDARs and therefore may serve as direct drop-in replacements in future works.

The final real-world reconstruction results are renderings of the derivative point clouds. Figure 7.14 shows 2 different perspective renderings of the same point cloud reconstruction. All mis-matches are visible, resulting in cloudy patches. Despite the noise, the reconstruction is visibly representative of the scene, including the nearby vehicles, actor, road surface and structures.

7.5.3 Processing Performance

To evaluate the feasibility of embedded real-time deployment, a simple benchmark of the reoncstruction pipeline was performed at a timestep. The times stated in Table 7.1 were broken down into the following sub-tasks: Image rectification, Disparity Estimation, Point cloud projection and transformation to global coordinate system. The benchmark was completed on a i9-9900k CPU, with single threaded, non accelerated implementation within matlab. The timing results totalled 30.3 sec for a full surround reconstruction, with the by far largest time commitment being associated with the matching for the disparity estimation. Despite this timing, it remains feasible to consider this running in real-time on an FPGA, as was highlighted in Chapter 5.

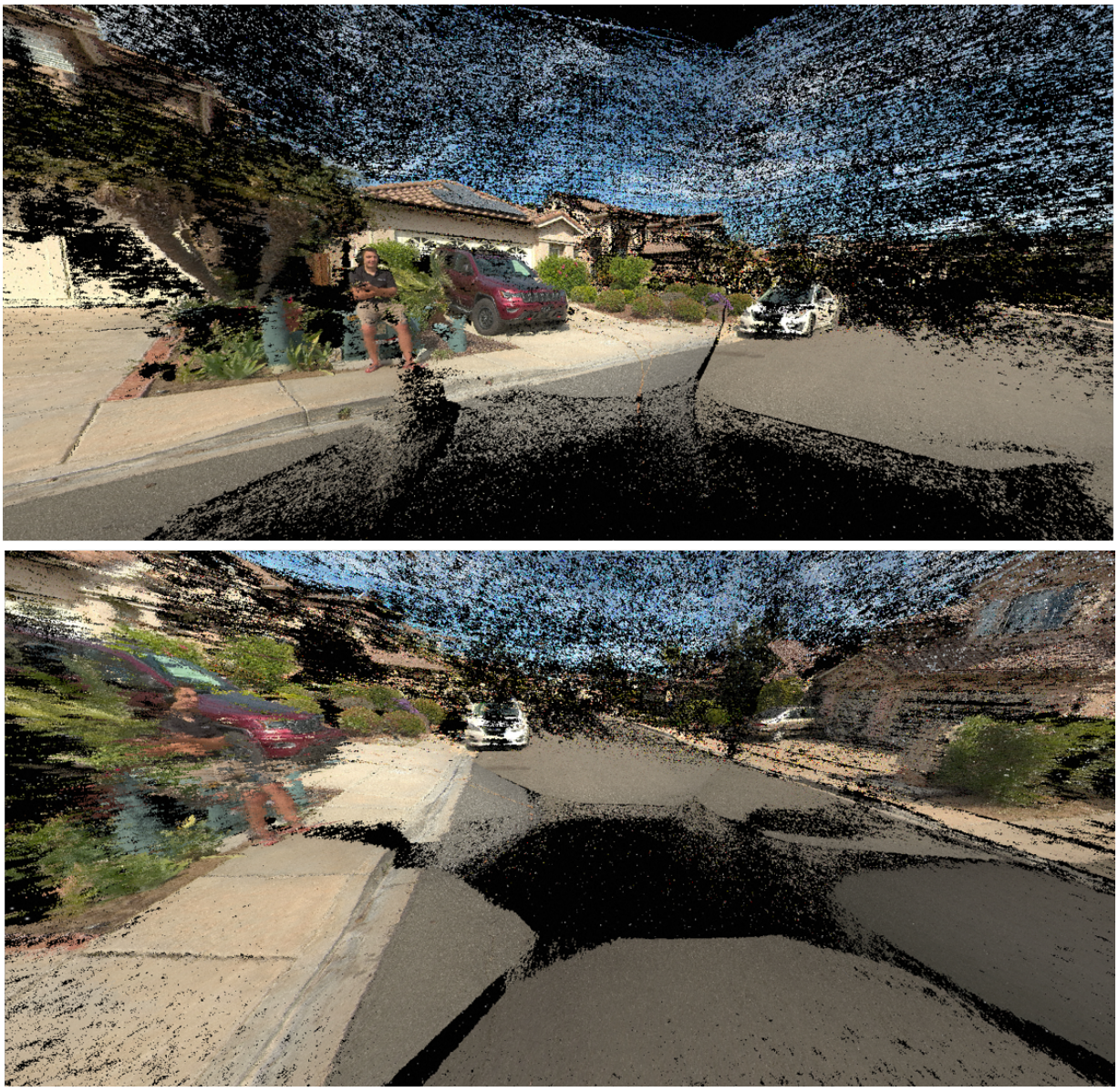

Figure 7.14: 3D Point Cloud Renderings

Table 7.1: Processing Time Breakdown for the Dense Recontruction of a single timestep of the trinocular panoramic system (all 18 images simultaneously).

Task	**Processing Time (sec)**
Rectification	3.57
Disparity Estimation	26.40
Projection to Point Cloud	0.615

7.6 Discussion

This work on trinocular panoramic vision for sensing and perception has opened discussion into a field largely uncharted research topics. Compared to leveraging a mechanically rotating LiDAR that provides reliable ranging measurements, matching images to estimate depth, definitely carries it's challenges. All the way from optics, mechanical configurations, sensor performance, through image ISP algorithms, and matching algorithms, the places where things may deviate from ideal are large. The final reconstruction renderings in Figure 7.14 are comparably noisy to LiDAR point clouds, but they do originate from consumer grade image sensors, at price points significantly more attainable than active sensors will likely ever be. The ability to reconstru a full surround view at every time-step highlights that there is a foreseeable path to vision only navigation on road environments. We can derive a holistic spatial view of objects, and there distances, up to some range defined by our sensor configuration and capabilities. In this case, we have been able to reliably range an actor up to a distance of 15m, simply using a system with a 11cm baseline.

Comparing the semi-dense and dense reconstructions, it is notable that the dense approach outperforms the direct contour matching of the semi-dense. While theoretically the sub-pixel localization of the contours to be superior in areas of high gradient, the texture matching approach performs all around better by reducing the number of outliers. It will be interesting to find future ways to improve dense performance on the edges, where gradient matching will play a bigger role. Furthermore, the importance of matching confidence becomes an important one when we want to solely keep the reliable point estimates in derivative models. Despite the superior performance of the SGM Dense reconstruction, considering the computational efficiency and inherent downsampling associated with dealing with edges, this remains a viable route for real-time implementaion.

7.7 Acknowledgments

This chapter is currently being prepared for submission for publication of the material, titled "Stereo Panoramic Sceneflow: A Framework for Sensing and Perception of Dynamic Environments " as it may appear in IEEE Transactions on Pattern Analysis and Machine Intelligence 2021. Meyer, Dominique Ernest; Verma, Pranav; Niradi, Shreyas; Christensen, Henrik; Kuester, Falko. The dissertation author is the primary researcher and author of this material.

Chapter 8

Concluding Remarks

8.1 Concluding remarks

This dissertation has explored the full design space of an embedded multi-camera system with the goal of efficiently and accurately estimating a 3D scene. The work has introduced key relationships between system disparities, directionalities and depth estimation accuracy. Building on the geometric priors, it has formulated a novel semi-dense feature triangulation approach which allows for the robust geometric estimation from a set of cameras. A physical FPGA processor system was designed to hardware accelerate tasks within the reconstruction pipeline, advancing the system capabilities closer to real-time operation. The final results of a fully integrated trinocular panoramic system are demonstrated within a reconstruction framework, which proves the benefits of leveraging multi-directional disparities to reconstruct scene geometry, and doing so efficiently.

8.2 Principle Contributions

The key research contributions presented within this dissertation are summarized as following:

- A novel concept of circular disparity is proposed in Chapter 3, providing distinct benefits when estimating world geometry of features with varying directionalities.
- A novel interleaved trinocular panoramic camera array is proposed, designed and built across Chapters 3, 6 and 7. This array is demonstrated to capture real-world data and outperform a stereo counterpart in terms of reconstruction accuracy and system localization.
- A dataflow architecture for efficient image demosaicing and rectification is presented and justified that enables the efficient multi-view processing on embedded systems.
- An open source multi-camera development infrastructure has been developed to enable future students, researchers and enthusiasts to test and improve on integrated visual-inertial multi-camera arrays.

8.3 Future Work

This work has opened many questions in the space of camera arrays, and it will undoubtedly take many papers and years of continued research work to expand on the discussed topics. A large portion of this work will surely become resolved with computational advances, improving the amount of processing that is achievable on an embedded system. What will however remain a key challenge, is the formulation of the model representations of the world in a way that they are most useful. Considering autonomous vehicles for examples, simply creating denser point clouds that span larger ranges is definitely not sustainable due to the increased data footprint and the requirement for further perception. Moving towards higher levels of abstraction, highly influenced by contextual cues will be critical.

The final results presented within this dissertation focused on semi-dense reconstructions. These 3D edge contours are excellent wire-frame representation of the world, but lack the visual density associated with fully dense reconstructions. It would be valuable to, in the future, explore filling uniform areas bounded by these contours with contextual information.

One of the major outstanding difficulties is auto-calibration on embedded devices. The calibration proposed in Chapter 3 requires a dedicated calibration pattern, and is computationally too expensive to run continuously in real-time while a system is in normal operation mode. It is with certainty, that solving this problem to be done on the fly, using real-world features, and achieving reconstruction performance equal or better than what are shown in this work, will be invaluable.

Another interesting research avenue which has up to now not been explored is the radial disparity circles that can be created by positioning in a circular pattern about a reference camera as seen in Chapter 3. This arrangement is a valuable alternative to linear arrays from light-field systems, and is guaranteed to augmented confidence on directional features.

8.4 Final Thoughts

This work is the culmination of many years of passionate systems engineering research. It has spanned about as full-stack as is possible- from mechanical design, electrical integration, hardware mapping, and finally software algorithmic innovation. It has been a fascinating exploration to evaluate every part of a camera system, which has resulted in building what turns out to be versatile and capable multi-camera systems with many diverse applications. I would like to thank all co-authors, advisors, colleagues and friends for your endless support to have made this happen, and I look forward to the continued worked with everyone to continue and evolve the presented concepts.

Bibliography

[ACM04] Adnan Ansar, Andres Castano, and Larry Matthies. Enhanced real-time stereo using bilateral filtering. In *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*, pages 455–462. IEEE, 2004.

[ACN+19] Min Sung Ahn, Hosik Chae, Donghun Noh, Hyunwoo Nam, and Dennis Hong. Analysis and noise modeling of the intel realsense d435 for mobile robots. In *2019 16th International Conference on Ubiquitous Robots (UR)*, pages 707–711. IEEE, 2019.

[AF81] Erhan Alparslan and Mr Fuatince. Image enhancement by local histogram stretching. *IEEE Transactions on Systems Man and Cybernetics*, 11:376–385, 1981.

[AM12] Sameer Agarwal and Keir Mierle. Ceres solver. 2012.

[AN15] Robert Andersson and Oskar Noresson. Utilization of quadnocular stereo vision for simultaneous localization and mapping in autonomous vehicles. Master's thesis, 2015.

[Ano] Anon. Tracking camera t265 – intel realsense depth and tracking cameras.

[APSB18] Kamel Abdelouahab, Maxime Pelcat, Jocelyn Serot, and François Berry. Accelerating cnn inference on fpgas: A survey. *arXiv preprint arXiv:1806.01683*, 2018.

[Aya91] Nicholas Ayache. *Artificial vision for mobile robots: stereo vision and multisensory perception*. Mit Press, 1991.

[Bau00] Adam Baumberg. Reliable feature matching across widely separated views. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 774–781. IEEE, 2000.

[BB89] H Harlyn Baker and Robert C Bolles. Generalizing epipolar-plane image analysis on the spatiotemporal surface. *International Journal of Computer Vision*, 3(1):33–49, 1989.

[BBM87] Robert C Bolles, H Harlyn Baker, and David H Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International journal of computer vision*, 1(1):7–55, 1987.

[BFO+20] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86–1, 2020.

[BFP18] Rhys Buggy, Marco Forte, and François Pitié. Neural net architectures for image demosaicing. In *Applications of Digital Image Processing XLI*, volume 10752, page 1075218. International Society for Optics and Photonics, 2018.

[BHF+10a] C. Banz, S. Hesselbarth, H. Flatt, H. Blume, and P. Pirsch. Real-time stereo vision system using semi-global matching disparity estimation: Architecture and fpga-implementation. In *2010 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation*, pages 93–101, July 2010.

[BHF+10b] Christian Banz, Sebastian Hesselbarth, Holger Flatt, Holger Blume, and Peter Pirsch. Real-time stereo vision system using semi-global matching disparity estimation: Architecture and fpga-implementation. In *2010 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation*, pages 93–101. IEEE, 2010.

[BK00] Gary Bradski and Adrian Kaehler. Opencv. *Dr. Dobb's journal of software tools*, 3, 2000.

[BKM+19] H Harlyn Baker, Gregorij Kurillo, Allan Miller, Alessandro Temil, Tom Defanti, and Dan Sandin. Epimodules on a geodesic: Toward 360° light-field imaging. *Electronic Imaging*, 2019(3):636–1, 2019.

[BLZ+18] Chaim Baskin, Natan Liss, Evgenii Zheltonozhskii, Alex M Bronstein, and Avi Mendelson. Streaming architecture for large-scale quantized neural networks on an fpga-based dataflow platform. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 162–169. IEEE, 2018.

[Bra00] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[BRL15] Donald Bailey, Sharmil Randhawa, and Jim Li. Advanced bayer demosaicing on fpgas. pages 216–220, 12 2015.

[BT98] Stan Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, 1998.

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[Can86] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[CBL+20] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020.

[CCHL95] Rong-Nan Chiou, Chin-Hsing Chen, King-Chu Hung, and Jau-Yien Lee. The optimal camera geometry and performance analysis of a trinocular vision system. *IEEE Transactions on Systems, Man, and cybernetics*, 25(8):1207–1220, 1995.

[CDK99a] Baoquan Chen, Frank Dachille, and Arie Kaufman. Forward image mapping. In *Proceedings Visualization'99 (Cat. No. 99CB37067)*, pages 89–514. IEEE, 1999.

[CDK99b] Baoquan Chen, Frank Dachille, and Arie Kaufman. Forward image mapping. In *Proceedings Visualization'99 (Cat. No. 99CB37067)*, pages 89–514. IEEE, 1999.

[Cok] David R. Cok. Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal.

[Com20] Eastman Kodak Company. *Kodak LossLess True Color Image Suite*, 1999(accessed 23 June 2020).

[COR+16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[CS87] James L Crowley and Arthur C Sanderson. Multiple resolution representation and probabilistic matching of 2-d gray-scale shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):113–121, 1987.

[CW93] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 279–288, 1993.

[DA90] Umesh R Dhond and Jake K Aggarwal. Binocular versus trinocular stereo. In *Proceedings., IEEE International Conference on Robotics and Automation*, pages 2045–2050. IEEE, 1990.

[Dav03] Andrew J Davison. Real-time simultaneous localisation and mapping with a single camera. In *null*, page 1403. IEEE, 2003.

[DD19] Bruno César Douady and Sandeep Doshi. Image signal processor for local motion estimation and video codec, September 26 2019. US Patent App. 16/303,428.

[Der04] Konstantinos G Derpanis. The harris corner detector. *York University*, pages 2–3, 2004.

[DLHT14] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[DRC+17] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[DRM03] Ahmad Darabiha, Jonathan Rose, and JW Maclean. Video-rate stereo depth measurement on programmable hardware. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–I. IEEE, 2003.

[Dub05] Eric Dubois. Frequency-domain methods for demosaicking of bayer-sampled color images. *IEEE Signal Processing Letters*, 12(12):847–850, 2005.

[Egn00] Geoffrey Egnal. Mutual information as a stereo correspondence measure. 2000.

[EKC17] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017.

[ESC14] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014.

[ESC15] Jakob Engel, Jörg Stückler, and Daniel Cremers. Large-scale direct slam with stereo cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942. IEEE, 2015.

[FA91] William T Freeman and Edward H Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern analysis and machine intelligence*, 13(9):891–906, 1991.

[Fel19] Dieter W Fellner. Virtual reality in media and technology. In *Digital Transformation*, pages 19–41. Springer, 2019.

[FG87] Wolfgang Förstner and Eberhard Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proceedings of ISPRS intercommission conference on fast processing of photogrammetric data*, pages 281–305. Interlaken, 1987.

[FMR+08] Johannes Fuertler, Konrad J Mayer, Michael Rubik, Harald Penz, Joerg Brodersen, Gemeiner Christian, Christian Eckel, and Herbert Nachtnebel. Streaming warper with cubic spline interpolation for rectification of distorted images on fpgas. In *Real-Time Image Processing 2008*, volume 6811, page 68110H. International Society for Optics and Photonics, 2008.

[FPF96] Andrew W Fitzgibbon, Maurizio Pilu, and Robert B Fisher. Direct least squares fitting of ellipses. In *Proceedings of 13th International Conference on Pattern Recognition*, volume 1, pages 253–257. IEEE, 1996.

[FPS14] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE, 2014.

[FW16] Wolfgang Förstner and Bernhard P Wrobel. *Photogrammetric computer vision*. Springer, 2016.

[FZG+16] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. Svo: Semidirect visual odometry for monocular and multi-camera systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2016.

[FZY09] Ivan Olaf Hernandez Fuentes, Miguel Enrique Bravo Zanoguera, and Guillermo Galaviz Yanez. Fpga implementation of the bilinear interpolation algorithm for image demosaicking. In *2009 International Conference on Electrical, Communications, and Computers*, pages 25–28. IEEE, 2009.

[FZYL17] Weikang Fang, Yanjun Zhang, Bo Yu, and Shaoshan Liu. Fpga-based orb feature extraction for real-time visual slam. In *2017 International Conference on Field Programmable Technology (ICFPT)*, pages 275–278. IEEE, 2017.

[GAM02] Bahadir K Gunturk, Yucel Altunbasak, and Russell M Mersereau. Color plane interpolation using alternating projections. *IEEE transactions on image processing*, 11(9):997–1013, 2002.

[GCPD16] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Transactions on Graphics (TOG)*, 35(6):1–12, 2016.

[GEM09] Stefan K Gehrig, Felix Eberli, and Thomas Meyer. A real-time low-power stereo vision engine using semi-global matching. In *International Conference on Computer Vision Systems*, pages 134–143. Springer, 2009.

[GHGB11] Pierre Greisen, Simon Heinzle, Markus Gross, and Andreas P Burg. An fpga-based processing pipeline for high-definition stereo video. *EURASIP Journal on Image and Video Processing*, 2011(1):18, 2011.

[GLSU13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[GLU12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[GW02] Rafael C Gonzalez and Richard E Woods. Digital image processing, 2002.

[Hal10] Hachem Halawana. *Partial demosaicing of CFA images for stereo matching*. PhD thesis, Ph. D. dissertation, Université de Lille 1, 2010.

[Har99] Richard I Hartley. Theory and practice of projective rectification. *International Journal of Computer Vision*, 35(2):115–127, 1999.

[Hei00] Janne Heikkila. Geometric camera calibration using circular control points. *IEEE Transactions on pattern analysis and machine intelligence*, 22(10):1066–1077, 2000.

[HGG+11] Simon Heinzle, Pierre Greisen, David Gallup, Christine Chen, Daniel Saner, Aljoscha Smolic, Andreas Burg, Wojciech Matusik, and Markus Gross. Computational stereo camera system with programmable control loop. *ACM Transactions on Graphics (TOG)*, 30(4):1–10, 2011.

[HI16] Rostam Affendi Hamzah and Haidi Ibrahim. Literature survey on stereo vision disparity map algorithms. *Journal of Sensors*, 2016, 2016.

[Hir05] Heiko Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814. IEEE, 2005.

[HJ11] Adam P Harrison and Dileepan Joseph. Maximum likelihood estimation of depth maps using photometric stereo. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1368–1380, 2011.

[HJCE+16] Daniel Hernandez-Juarez, Alejandro Chacón, Antonio Espinosa, David Vázquez, Juan Carlos Moure, and Antonio M López. Embedded real-time stereo estimation via semi-global matching on the gpu. *Procedia Computer Science*, 80:143–153, 2016.

[HN15a] Ástvaldur Hjartarson and Klas Nordmark. Implementing stereoscopic video processing on fpga. 2015.

[HN15b] Ástvaldur Hjartarson and Klas Nordmark. Implementing stereoscopic video processing on fpga. 2015.

[How08] Andrew Howard. Real-time stereo visual odometry for autonomous ground vehicles. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3946–3952. IEEE, 2008.

[HSP17] Dominik Honegger, Torsten Sattler, and Marc Pollefeys. Embedded real-time multi-baseline stereo. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5245–5250. IEEE, 2017.

[HSR15] Eduardo E Hitomi, Jorge VL Silva, and Guilherme CS Ruppert. 3d scanning using rgbd imaging devices: A survey. In *Developments in Medical Image Processing and Computational Vision*, pages 379–395. Springer, 2015.

[HST+14] Felix Heide, Markus Steinberger, Yun-Ta Tsai, Mushfiqur Rouf, Dawid PajÄ…k, Dikpal Reddy, Orazio Gallo, Jing Liu abd Wolfgang Heidrich, Karen Egiazarian, Jan Kautz, and Kari Pulli. Flexisp: A flexible camera image processing framework. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2014)*, 33(6), December 2014.

[HTGT10] Stavros Hadjitheophanous, Christos Ttofis, Athinodoros S Georghiades, and Theocharis Theocharides. Towards hardware stereoscopic 3d reconstruction: a real-time fpga computation of the disparity map. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 1743–1748. European Design and Automation Association, 2010.

[HWLJ16] Jianhui Han, Zhen Wu, Lanying Li, and Ying Ji. Fpga implementation for binocular stereo matching algorithm based on sobel operator. *International Journal of Database Theory and Application*, 9(4):221–230, 2016.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[HZW+10] Martin Humenberger, Christian Zinner, Michael Weber, Wilfried Kubinger, and Markus Vincze. A fast stereo matching algorithm suitable for embedded real-time systems. *Computer Vision and Image Understanding*, 114(11):1180–1202, 2010.

[IHR+16] Sunghoon Im, Hyowon Ha, François Rameau, Hae-Gon Jeon, Gyeongmin Choe, and In So Kweon. All-around depth from small motion with a spherical panoramic camera. In *European Conference on Computer Vision*, pages 156–172. Springer, 2016.

[IKH+11] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, and Andrew Davison. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568, 2011.

[IVGT20] Andrey Ignatov, Luc Van Gool, and Radu Timofte. Replacing mobile camera isp with a single deep learning model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 536–537, 2020.

[JCDP+09] Seunghun Jin, Junguk Cho, Xuan Dai Pham, Kyoung Mu Lee, Sung-Kee Park, Munsang Kim, and Jae Wook Jeon. Fpga design and implementation of a real-time stereo vision system. *IEEE transactions on circuits and systems for video technology*, 20(1):15–26, 2009.

[Jes09] Jeppe Barsøe Jessen. *Real time sparse and dense stereo in an early cognitive vision system using cuda*. PhD thesis, Citeseer, 2009.

[JFHJ] James E. Adams Jr. John F. Hamilton Jr. Adaptive color plane interpolation in single sensor color electronic camera.

[JHRN19] Christina Junger, Albrecht HeA, Maik Rosenberger, and Gunther Notni. FPGA-based lens undistortion and image rectification for stereo vision applications. In Maik Rosenberger, Paul-Gerald Dittrich, and Bernhard Zagar, editors, *Photonics and Education in Measurement Science 2019*, volume 11144, pages 284 – 291. International Society for Optics and Photonics, SPIE, 2019.

[JO04] Hong Jeong and Youns Oh. Real-time three-dimensional image processing system for non-parallel optical axis and method thereof, November 18 2004. US Patent App. 10/795,777.

[JU04] Mathews Jacob and Michael Unser. Design of steerable filters for feature detection using canny-like criteria. *IEEE transactions on pattern analysis and machine intelligence*, 26(8):1007–1019, 2004.

[JZLA04] Yunde Jia, Xiaoxun Zhang, Mingxiang Li, and Luping An. A miniature stereo vision machine (msvm-iii) for dense disparity mapping. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 1, pages 728–731. IEEE, 2004.

[KEFW20] Jan Kallwies, Torsten Engler, Bianca Forkel, and Hans-Joachim Wuensche. Triple-sgm: Stereo processing using semi-global matching with cost fusion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 192–200, 2020.

[Kim99] Ron Kimmel. Demosaicing: image reconstruction from color ccd samples. *IEEE Transactions on image processing*, 8(9):1221–1228, 1999.

[KKLML16] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[KKN+12] Tim Kazik, Laurent Kneip, Janosch Nikolic, Marc Pollefeys, and Roland Siegwart. Real-time 6d stereo visual odometry with non-overlapping fields of view. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1529–1536. IEEE, 2012.

[KKZ03] Junhwan Kim, Vladimir Kolmogorov, and Ramin Zabih. Visual correspondence using energy minimization and mutual information. In *null*, page 1033. IEEE, 2003.

[KM07] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007.

[KMI+03] Michael Kuhn, Stephan Moser, Oliver Isler, Frank K Gurkaynak, Andreas Burg, Norbert Felber, Hubert Kaeslin, and Wolfgang Fichtner. Efficient asic implementation of a real-time depth mapping stereo vision system. In *2003 46th Midwest Symposium on Circuits and Systems*, volume 3, pages 1478–1481. IEEE, 2003.

[KMR13] Bryan Klingner, David Martin, and James Roseborough. Street view motion-from-structure-from-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 953–960, 2013.

[Kon98] Kurt Konolige. Small vision systems: Hardware and implementation. In *Robotics research*, pages 203–212. Springer, 1998.

[KRD08] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. isam: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

[KS04] Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II. IEEE, 2004.

[KSC13] Christian Kerl, Jürgen Sturm, and Daniel Cremers. Dense visual slam for rgb-d cameras. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2100–2106. IEEE, 2013.

[KSC+19] Irina Kim, Seongwook Song, Soonkeun Chang, Sukhwan Lim, and Kai Guo. Deep image demosaicing for submicron image sensors. *Journal of Imaging Science and Technology*, 63(6):60410–1, 2019.

[KSY+99] Shigeru Kimura, Tetsuya Shinbo, Hiroyoshi Yamaguchi, Eiji Kawamura, and Katsuyuki Nakano. A convolver-based real-time stereo machine (sazan). In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 1, pages 457–463. IEEE, 1999.

[KW16] Philip Koopman and Michael Wagner. Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety*, 4(1):15–24, 2016.

[LC87] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.

[LCTZ07] Nai-Xiang Lian, Lanlan Chang, Yap-Peng Tan, and Vitali Zagorodnov. Adaptive filtering for color filter array demosaicking. *IEEE Transactions on Image Processing*, 16(10):2515–2525, 2007.

[LD91] Martin Lange and Jurgen Detlefsen. 94 ghz three-dimensional imaging radar sensor for autonomous vehicles. *IEEE Transactions on Microwave Theory and Techniques*, 39(5):819–827, 1991.

[Lee80] Jong-Sen Lee. Digital image enhancement and noise filtering by use of local statistics. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (2):165–168, 1980.

[LGS+14] Ragnar Langseth, Vamsidhar Reddy Gaddam, Håkon Kvale Stensland, Carsten Griwodz, and Pål Halvorsen. An evaluation of debayering algorithms on gpu for real-time panoramic video recording. In *2014 IEEE International Symposium on Multimedia*, pages 110–115. IEEE, 2014.

[LGS+15] Ragnar Langseth, Vamsidhar Gaddam, Håkon Stensland, Carsten Griwodz, Pål Halvorsen, and Dag Johansen. An experimental evaluation of debayering algorithms on gpus for recording panoramic video in real-time. *International Journal of Multimedia Data Engineering and Management (IJMDEM)*, 6:1–16, 07 2015.

[LK81] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. 1981.

[LMY10a] Olivier Losson, Ludovic Macaire, and Yanqin Yang. Comparison of color demosaicing methods. In *Advances in Imaging and Electron Physics*, volume 162, pages 173–265. Elsevier, 2010.

[LMY10b] Olivier Losson, Ludovic Macaire, and Yanqin Yang. Comparison of color demosaicing methods. *Advances in Imaging and Electron Physics - ADV IMAG ELECTRON PHYS*, 162:173–265, 07 2010.

[Low04] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[LS11] Sang Hwa Lee and Siddharth Sharma. Real-time disparity estimation algorithm for stereo camera systems. *IEEE transactions on Consumer electronics*, 57(3):1018–1026, 2011.

[LSK+17] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[LTQB18] Chang Liang, Yun Tie, Lin Qi, and Cheng Bi. Deep vidar: Cnn based 360° panoramic video system for outdoor robot visual navigation and slam. In *Tenth International Conference on Digital Image Processing (ICDIP 2018)*, volume 10806, page 1080663. International Society for Optics and Photonics, 2018.

[LZ99] Charles Loop and Zhengyou Zhang. Computing rectifying homographies for stereo vision. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 1, pages 125–131. IEEE, 1999.

[MAC06] Daniele Menon, Stefano Andriani, and Giancarlo Calvagno. Demosaicing with directional filtering and a posteriori decision. *IEEE Transactions on Image Processing*, 16(1):132–141, 2006.

[MAMT15] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[MAT15] Raúl Mur-Artal and Juan D Tardós. Probabilistic semi-dense mapping from highly accurate feature-based monocular slam. In *Robotics: Science and Systems*, volume 2015. Rome, 2015.

[Mat17] The Mathworks, Inc., Natick, Massachusetts. *MATLAB version 9.3.0.713579 (R2017b)*, 2017.

[MCMOM09] Carol Martínez, Pascual Campoy, Iván Mondragón, and Miguel A Olivares-Méndez. Trinocular ground system to control uavs. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3361–3367. Ieee, 2009.

[MHC04] Henrique S Malvar, Li-wei He, and Ross Cutler. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages iii–485. IEEE, 2004.

[MID02] Jane Mulligan, Volkan Isler, and Kostas Daniilidis. Trinocular stereo: A real-time algorithm and its evaluation. *International Journal of Computer Vision*, 47(1-3):51–61, 2002.

[MK93] Sanjit K Mitra and James F Kaiser. *Handbook for digital signal processing*. John Wiley & Sons, Inc., 1993.

[MK00] Jane Mulligan and K Kaniilidis. Trinocular stereo for non-parallel configurations. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 1, pages 567–570. IEEE, 2000.

[ML00] Don Murray and James J Little. Using real-time stereo vision for mobile robot navigation. *autonomous robots*, 8(2):161–171, 2000.

[MLC04] H. S. Malvar, Li-wei He, and R. Cutler. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages iii–485, May 2004.

[MLM+19] Dominique Meyer, Eric Lo, Chris McFarland, James Strawson, Danylo Drohobytsky, Ji Dai, Gregory Dawe, Truong Nguyen, Tom Defanti, Falko Kuester, Haoyu Wang, Dan Sandin, and Maxine Brown. Omniscopic vision for robotic control. In *2019 IEEE Aerospace Conference*, pages 1–13. IEEE, 2019.

[MLR+07] Chris Murphy, Daniel Lindquist, Ann Marie Rynning, Thomas Cecil, Sarah Leavitt, and Mark L Chang. Low-cost stereo vision on an fpga. In *15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM 2007)*, pages 333–334. IEEE, 2007.

[Mor78] Jorge J Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer, 1978.

[MSH04] Martin Matousek, Radim Sara, and Václav Hlavác. Data-optimal rectification for fast and accurate stereovision. In *Third International Conference on Image and Graphics (ICIG'04)*, pages 212–215. IEEE, 2004.

[MWS+19] DE Meyer, H Wang, D Sandin, C McFarland, E Lo, G Dawe, J Dai, T Nguyen, H Baker, MD Brown, T DeFanti, and F Kuester. Starcam-a 16k stereo panoramic video camera with a novel parallel interleaved arrangement of sensors. *Electronic Imaging*, 2019(3):646–1, 2019.

[NJ04] Gerhard Neukum and Ralf Jaumann. Hrsc: The high resolution stereo camera of mars express. In *Mars Express: The Scientific Payload*, volume 1240, pages 17–35, 2004.

[NJ19] Maikon Nascimento and Dileepan Joseph. System-on-chip design flow for the image signal processor of a nonlinear cmos imaging system. *Electronic Imaging*, 2019(9):363–1, 2019.

[NLD11] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE, 2011.

[NML+13] Rahul Nair, Stephan Meister, Martin Lambers, Michael Balda, Hannes Hofmann, Andreas Kolb, Daniel Kondermann, and Bernd Jähne. Ground truth for evaluating time of flight imaging. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, pages 52–74. Springer, 2013.

[OK08a] Sungchan Oh and Gyeonghwan Kim. An architecture for on-the-fly correction of radial distortion using fpga. *Proceedings of SPIE - The International Society for Optical Engineering*, 03 2008.

[OK08b] Sungchan Oh and Gyeonghwan Kim. An architecture for on-the-fly correction of radial distortion using fpga. In *Real-Time Image Processing 2008*, volume 6811, page 68110X. International Society for Optics and Photonics, 2008.

[OR90] Stanley Osher and Leonid I Rudin. Feature-oriented image enhancement using shock filters. *SIAM Journal on numerical analysis*, 27(4):919–940, 1990.

[Pop01] Voicu Sebastian Popescu. *Forward rasterization: A reconstruction algorithm for image-based rendering*. PhD thesis, Citeseer, 2001.

[QMS+19] Morgan Quigley, Kartik Mohta, Shreyas S Shivakumar, Michael Watterson, Yash Mulgaonkar, Mikael Arguedas, Ke Sun, Sikang Liu, Bernd Pfrommer, and Vijay Kumar. The open vision computer: An integrated sensing and compute system for mobile robots. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1834–1840. IEEE, 2019.

[Reu17a] E. Reuss. Beyond the limits of visual acuity: The real reason for 4k and 8k image resolution. *SMPTE Motion Imaging Journal*, 126(2):33–39, March 2017.

[Reu17b] Edward Reuss. Beyond the limits of visual acuity: The real reason for 4k and 8k image resolution. *SMPTE Motion Imaging Journal*, 126(2):33–39, 2017.

[RF10] Joao GP Rodrigues and Joao Canas Ferreira. Fpga-based rectification of stereo images. In *2010 Conference on Design and Architectures for Signal and Image Processing (DASIP)*, pages 199–206. IEEE, 2010.

[RH13] K Sudha Rani and W Jino Hans. Fpga implementation of bilinear interpolation algorithm for cfa demosaicing. In *2013 International Conference on Communication and Signal Processing*, pages 857–863. IEEE, 2013.

[RRKB11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[SCC18] Nai-Sheng Syu, Yu-Sheng Chen, and Yung-Yu Chuang. Learning deep convolutional networks for demosaicing. *arXiv preprint arXiv:1802.03769*, 2018.

[SDG+16] Hendrik Siedelmann, Maximilian Diebold, Marcel Gutsche, Hamza Aziz-Ahmad, and Bernd Jähne. A fractal calibration pattern for improved camera calibration. In *Forum Bildverarbeitung 2016*, volume 84, page 1. KIT Scientific Publishing, 2016.

[SGB18] Eli Schwartz, Raja Giryes, and Alex M Bronstein. Deepisp: Toward learning an end-to-end image processing pipeline. *IEEE Transactions on Image Processing*, 28(2):912–923, 2018.

[She68] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference*, ACM '68, pages 517–524, New York, NY, USA, 1968. Association for Computing Machinery.

[Shi94] Jianbo Shi. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

[SHW+14] Yi Shan, Yuchen Hao, Wenqiang Wang, Yu Wang, Xu Chen, Huazhong Yang, and Wayne Luk. Hardware acceleration for an accurate stereo vision system using mini-census adaptive support region. *ACM Transactions on Embedded Computing Systems (TECS)*, 13(4s):132, 2014.

[SKK+18] Roman A Solovyev, Alexandr A Kalinin, Alexander G Kustov, Dmitry V Telpukhov, and Vladimir S Ruhlov. Fpga implementation of convolutional neural networks with fixed-point calculations. *arXiv preprint arXiv:1808.09945*, 2018.

[SLK15] Hamed Sarbolandi, Damien Lefloch, and Andreas Kolb. Kinect range sensing: Structured-light versus time-of-flight kinect. *Computer vision and image understanding*, 139:1–20, 2015.

[SP11a] S. Sowmya and R. Paily. Fpga implementation of image enhancement algorithms. In *2011 International Conference on Communications and Signal Processing*, pages 584–588, Feb 2011.

[SP11b] S Sowmya and Roy Paily. Fpga implementation of image enhancement algorithms. In *2011 International Conference on Communications and Signal Processing*, pages 584–588. IEEE, 2011.

[SS02] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002.

[SSG+17] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3260–3269, 2017.

[SVdMG04] Harris Sunyoto, Wannes Van der Mark, and Dariu M Gavrila. A comparative study of fast dense stereo vision algorithms. In *IEEE Intelligent Vehicles Symposium, 2004*, pages 319–324. IEEE, 2004.

[SZC+18] Amr Suleiman, Zhengdong Zhang, Luca Carlone, Sertac Karaman, and Vivienne Sze. Navion: a fully integrated energy-efficient visual-inertial odometry accelerator for autonomous navigation of nano drones. In *2018 IEEE Symposium on VLSI Circuits*, pages 133–134. IEEE, 2018.

[SZC+19] Amr Suleiman, Zhengdong Zhang, Luca Carlone, Sertac Karaman, and Vivienne Sze. Navion: A 2-mw fully integrated real-time visual-inertial odometry accelerator for autonomous navigation of nano drones. *IEEE Journal of Solid-State Circuits*, 54(4):1106–1119, 2019.

[TYL17] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3147–3155, 2017.

[Ume91] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.

[VA13] Rohit Verma and Jahid Ali. A comparative study of various types of image noise and efficient noise removal techniques. *International Journal of advanced research in computer science and software engineering*, 3(10), 2013.

[VJM+15] Kartik Venkataraman, Amandeep S Jabbi, Robert H Mullis, Jacques Duparre, and Shane Ching-Feng Hu. Camera arrays incorporating $3\times$ 3 imager configurations, June 2 2015. US Patent 9,049,411.

[VMDS19] Nishchal K Verma, Aquib Mustafa, Narendra Kumar Dhar, and Vibhav Sarraf. Surf–mser based 3d mapping using rgb-d camera on automated vehicle. In *Computational Intelligence: Theories, Applications and Future Directions-Volume II*, pages 373–386. Springer, 2019.

[VWJL04] Vaibhav Vaish, Bennett Wilburn, Neel Joshi, and Marc Levoy. Using plane+ parallax for calibrating dense camera arrays. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I. IEEE, 2004.

[WBI+19] Frank Worcester, Philip Breedon, Kira Iaquinta, Mick Anderson, Steve Brooks, and James Sprinks. Low cost, user friendly embedded machine vision system implementation for high-speed industrial manufacture. 2019.

[WBJ+07a] J. I. Woodfill, R. Buck, D. Jurasek, G. Gordon, and T. Brown. 3d vision: Developing an embedded stereo-vision system. *Computer*, 40(5):106–108, May 2007.

[WBJ+07b] John Iselin Woodfill, Ron Buck, Dave Jurasek, Gaile Gordon, and Terrance Brown. 3d vision: Developing an embedded stereo-vision system. *Computer*, 40(5):106–108, 2007.

[Wes90] Lee Westover. Footprint evaluation for volume rendering. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 367–376, 1990.

[Wes91] Lee Alan Westover. *Splatting: a parallel, feed-forward volume rendering algorithm*. PhD thesis, University of North Carolina at Chapel Hill Chapel Hill, NC, 1991.

[WHC+18] Fu-En Wang, Hou-Ning Hu, Hsien-Tzu Cheng, Juan-Ting Lin, Shang-Ta Yang, Meng-Li Shih, Hung-Kuo Chu, and Min Sun. Self-supervised learning of depth and camera motion from 360 videos. In *Asian Conference on Computer Vision*, pages 53–68. Springer, 2018.

[WJV+05] Bennett Wilburn, Neel Joshi, Vaibhav Vaish, Eino-Ville Talvala, Emilio Antunez, Adam Barth, Andrew Adams, Mark Horowitz, and Marc Levoy. High performance imaging using large camera arrays. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 765–776. ACM, 2005.

[WSK+13] Junqing Wei, Jarrod M Snider, Junsung Kim, John M Dolan, Raj Rajkumar, and Bakhtiar Litkouhi. Towards a viable autonomous driving research platform. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 763–770. IEEE, 2013.

[WT99] Todd Williamson and Charles Thorpe. A trinocular stereo system for highway obstacle detection. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 3, pages 2267–2273. IEEE, 1999.

[WVH97] John Woodfill and Brian Von Herzen. Real-time stereo vision on the parts reconfigurable computer. In *Proceedings. The 5th Annual IEEE Symposium on Field-Programmable Custom Computing Machines Cat. No. 97TB100186)*, pages 201–210. IEEE, 1997.

[WW06] Justin M Wanek and Chin H Wu. Automated trinocular stereo imaging system for three-dimensional surface wave measurements. *Ocean engineering*, 33(5-6):723–747, 2006.

[WWW+18] Xiaowei Wan, Gangyi Wang, Xinguo Wei, Jian Li, and Guangjun Zhang. Star centroiding based on fast gaussian fitting for star sensors. *Sensors*, 18(9):2836, 2018.

[XO01] Xin Li and M. T. Orchard. New edge-directed interpolation. *IEEE Transactions on Image Processing*, 10(10):1521–1527, Oct 2001.

[Xu18] Jie Xu. System and method for bullet-time photography, August 14 2018. US Patent 10,051,170.

[Yan06] Ching-Chung Yang. Image enhancement by modified contrast-stretching manipulation. *Optics & Laser Technology*, 38(3):196–201, 2006.

[Zha00] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[Zis04] Richard Hartley Andrew Zisserman. Multiple view geometry in computer vision. 2004.

[ZKU+04] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. In *ACM transactions on graphics (TOG)*, volume 23, pages 600–608. ACM, 2004.

[ZPVBG01] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 371–378, 2001.

[ZSC+17] Zhengdong Zhang, Amr AbdulZahir Suleiman, Luca Carlone, Vivienne Sze, and Sertac Karaman. Visual-inertial odometry on chip: An algorithm-and-hardware co-design approach. 2017.

[ZW94] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. In *European conference on computer vision*, pages 151–158. Springer, 1994.

[ZWW13a] B. Zhu, D.-S Wen, and F. Wang. Improved bayer-pattern demosaicking and its hardware design. *Guangdianzi Jiguang/Journal of Optoelectronics Laser*, 24:1211–1218, 06 2013.

[ZWW13b] Bo ZHU, Desheng WEN, and Fei WANG. Improved bayer pattern demosaicking and its hardware design [j]. *Journal of Optoelectronics Laser*, 6(24):1211–1218, 2013.

[ZZW13] Dong-Qing Zhang, Jiefu Zhai, and Zhe Wang. System and method for trinocular depth acquisition with triangular sensor, October 3 2013. US Patent App. 13/991,636.