TECHNISCHE UNIVERSITÄT BERLIN

3D reconstruction with structured light using event-based vision and a video laser projector

vorgelegt von

Simon Baumann

von der Fakultät IV – Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades
MASTER OF SCIENCE (MSC)
genemigte Masterarbeit

Prüfungsausschuss:

Prüfer der Masterarbeit: Prof. Dr. Guillermo Gallego, Technische Universität Berlin
Prof. Dr.-Ing. Peter Eisert, Humbolt Universität zu Berlin

Berlin 2022

Abstract

This Masterthesis implements a structured light depth sensing system employing an event-based camera and a video laser projector which can simultaneously project arbitrary content for spatial augmented reality applications. The event-based camera enables the use of a laser scanner at higher speeds compared to conventional frame-based solutions while still benefiting from the advantages of the laser scanner like high accuracy and high robustness through high light source efficiency. In addition, the high dynamic range of the event camera improves usability in challenging ambient light situations. The working principle of the disparity search allows for the laser beam to have varied intensity and color as long as it is bright enough to trigger the event camera. This provides the ability to project content independent of the structured light pattern.

The structured light system is implemented as a camera-projector stereo pair. For the calibration of the camera and the projector, a method and application considering the dynamic nature of the camera is designed. To identify the time interval of a projected frame a trigger algorithm is implemented. Nonlinearity in the projector scan speed can be accounted for with a calibration for the temporal behavior of the projector. The disparity search takes place in the temporal domain as opposed to the intensity domain of frame-based systems. The disparity map is processed to a point cloud or depth map and used in two augmented reality demonstrations in real-time. The first is a live color map of the depth projected onto the scene and the second is an image that corrects its projection by warping to appear undistorted and the same size on angled screens.

Measurements taken with the system show that the main principle of projecting content while still registering the depth is possible. This ability depends on the brightness of the projected image and the ambient illumination, which limits the augmented reality applications a bit. Analysis of the depth map reveal notable noise and little systematic error which stem from jitter in the projector as well as camera timing and inaccuracies in the calibration, respectively. This noise puts limits on the detail of the recovered depth map, although still being clean enough for the example demonstrations. An integrated product based on this principle while addressing the limitations through purpose build hardware and software presents a lean yet versatile setup for interactive and real-time augmented reality installations.

Zusammenfassung

Diese Masterarbeit implementiert ein Structured Light System, das eine Event Kamera und einen Video Laser Projektor verwendet und ist gleichzeitig in der Lage beliebige Bilder zu projizieren. Die Event Kamera ermöglicht the Verwendung eines Laser Scanners bei höheren Geschwindigkeiten als konventionalle Frame basierte Systeme. Trotzdem bietet der Laser Projektor Vorteile wie hohe Genauigkeit und hohe Robustheit durch effiziente Nutzung der Lichquelle. Ein weitere Vorteil ist der hohe Dynamikumfang der Kamera. Das Arbeitsprinzip der Disparitätssuche erlaubt dem Laserstrahl verschiedene Helligkeiten und Farben, solange diese hell genug sind, um ein Event auszulösen. Dies bietet die Möglichkeit Bilder zu projizieren, die Unabhänging von dem Structured Light Muster sind.
Das Structured Light System ist als Kamera-Projektor Stereopaar implementiert. Für dessen Kalibrierung wurde eine Anwendung designt, die die dynamischen Eigenschaften der Event Kamera berücksichtigt. Um den Zeitintervall eines projizierten Bildes zu bestimmen, wurde ein Trigger Algorithmus implementiert. Eventuelle Nichtlinearität in der Scangeschwindigkeit des Projektors können ausglichen werden, indem auf das zeitliche Verhalten des Projektors kalibriert wird. Die anschließende Disparitätssuche verläuft in der temporalen Domäne, nicht in der Intensitäts Domäne, wie bei frame basierten Systemen. Das Disparitätsbild wird zu einer Tiefenkarte oder Punktwolke verarbeitet und dann in zwei Augmented Reality Beispielen verwendent. Zum Einen wird die Tiefenkarte farbkodiert live auf die Szene selber projiziert. Zum Anderen, ein Bild, dass sich durch Transfomration selbst korrigiert, um unverzerrt und in der gleichen Größe auf gedrehten Leinwänden zu erscheinen.
Messungen, die mit dem System aufgenommen wurden zeigen, dass es möglich ist bliebigen Bildinhalt zu projizieren und gleichzeitig die Tiefeninformation zu berechnen. Diese Möglichkeit hängt von der Helligkeit des projizierten Bildes und von der Umgebungsbeleuchtung ab, was die Einsatzgebiete leicht reduziert. Eine Analyse der Tiefenkarten zeigt, das diese von Rauschen und einem kleinen systematischem Fehler betroffen sind, welche auf Grund von Projektor und Kamera Timing Rauschen beziehungsweise Fehler in der Kalibrierung auftreten. Dieses Rauschen limitiert die Detailwiedergabe der gewonnenen Tiefenkarte, wenn gleich die Qualität für die implementierten Augmented Reality Beispiele ausreicht. Ein integriertes Produkt, das auf diesem Prinzip aufbaut und die genannten Limitationen mit anwendungsspezifischer Hardware und Software adressiert, bietet ein schlankes als auch flexibles Setup für interaktive und Echtzeit Augmented Reality Installationen.

Acknowledgments

At first, I would like to thank my partner Trang Tran and son Keno Bao Linh for the support, inspiration and happiness given to me every day.

I would like to thank Wieland Morgenstern for his very helpful and friendly supervision during research and writing.

Finally, I would like to thank the Heinrich Hertz Institute, Prof. Peter Eisert, and Prof. Gallego for giving me the opportunity to research and write this thesis as well as Prophesee for providing the event camera.

Ich versichere an Eides statt, dass ich diese Masterarbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe. Alle wörtlich oder inhaltlich übernommenen Stellen habe ich als solche gekennzeichnet. Die vorliegende Arbeit wurde weder in der vorliegenden noch einer modifizierten Fassung einer dritten, in- oder ausländischen Fakultät als Prüfungsleistung, oder zum Erlangen eines akademischen Grades vorgelegt und wurde nicht veröffentlicht.

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources. I have explicitly marked all material which has been quoted either literally or by content from the used sources. The submitted thesis was neither in this nor in modified form presented to another examination or to earn any academic degree and has not been published.

Berlin, 9.9.2022 Sim Bau

Datum Simon Baumann

Contents

Abstract **i**

Acknowledgments **iii**

1 Introduction **1**

1.1 Problem Statement 1

1.2 Contributions 3

1.3 Thesis Outline 5

2 Related Work **6**

2.1 Structured Light 6

2.2 Event Cameras 8

2.3 Event-Based Structured Light 12

2.4 Structured Light based Augmented Reality 13

3 Foundations **16**

3.1 Pinhole Model 16

3.2 Stereo Vision 18

3.2.1 Epipolar Geometry 18

3.2.2 Rectification 19

3.3 Current Implementations 20

3.3.1 MC3D 20

3.3.2 ESL 21

4 Setup **25**

4.1 Hardware 25

4.1.1 Camera 25

4.1.2 Projector 26

4.1.3 Physical Setup 27

4.2 Software . . . 30

5 Implementation **31**

5.1 Calibration . . . 31

5.1.1 Camera . . . 32

5.1.2 Projector . . . 32

5.1.3 Integration . . . 34

5.2 Structured Light System . . . 36

5.2.1 Trigger and Time Map . . . 36

5.2.2 Projector Time Map Calibration . . . 40

5.2.3 Disparity and Depth . . . 44

5.2.4 Projectors Point of View . . . 48

5.3 Demonstrations . . . 49

5.3.1 Depth-based Color Map . . . 50

5.3.2 Warping Correction . . . 51

6 Results **53**

6.1 Measurement Methodology . . . 53

6.2 Projector Time Map . . . 54

6.3 Ambient Illumination and Content Level . . . 55

6.4 Specular Material & Inter-Reflections . . . 59

6.5 Noise . . . 62

6.6 Point Cloud Comparison . . . 64

6.7 Demonstrations . . . 66

6.7.1 Depth-based Color Map . . . 66

6.7.2 Warping Correction . . . 68

6.8 Limitations . . . 69

7 Conclusion and Outlook **71**

7.1 Summary . . . 71

7.2 Future Directions . . . 72

Bibliography **73**

List of Figures **76**

List of Tables **78**

List of Listings **78**

Chapter 1

Introduction

1.1 Problem Statement

Spatial augmented reality applications (SAR) in which a projector project digital assets onto a real scene profit a lot from 3D knowledge about the scene. Especially when the SAR application can react to changes in the scene, like in object tracking and gesture detection, it is necessary for the 3D information to be accurate and captured in real-time. For detecting a depth map and therefore 3D information about a scene there are methods like time of flight cameras, stereo vision, and structured light. The advantages of structured light can be high accuracy, high speed or high robustness, but these come at a trade-off to each other [1]. Also, since a structured light system also employs a projector, the projected light disrupts the AR projection and vice versa. To address this, existing AR implementations which use structured light systems use light in the IR spectrum, hence invisible to the human eye. This creates complex hardware setups, especially if high performance is desired [2]. Integrating an event camera and a video laser projector as the structured light system allows for accurate depth maps at usable speeds and can use the same projector for both, the structured light pattern and augmented reality projection concurrently.

Recent publications on the topic of event-based structured light [1, 3] aim to improve three main problems with conventional structured light systems with the help of event-based cameras. These can be summarized in two trade-offs the systems have to make plus general limitations of the camera systems.

Generally, the best results in terms of accuracy and robustness in structured light methods are achieved with laser scanning devices, either point or line. These implementations outperform other methods. Namely, they outperform gray code, binary code, or phase shift in robustness and single shot methods in accuracy and resolution but at the cost of speed. These two trade-offs, speed-robustness, and speed-resolution, stem from the bandwidth necessary to take and process a complete camera frame for each

laser point or line position. Especially for the laser point, there is a lot of redundant information in the frame to capture and process.

Moreover, a conventional camera system has known limitations in dynamic range. The dynamic range does not only need to cover the dynamic range of the scene, but also the added brightness of the active projection. This is even worsened by a scene with high inter-reflectance.

Event-based cameras, also known as dynamic vision cameras, can alleviate these problems. These cameras register only a change in light intensity, i.e. photocurrent, per pixel and send an asynchronous stream of change detections or events with the information about pixel position, time, and polarity of the change. This reduces the necessary bandwidth drastically as only the parts of the image that changed, i.e. the laser point moved to a new position, is produced and needs to be processed.

These kinds of vision systems have several advantages, most notably the reduction in bandwidth necessary, the time resolution of the events and an increase in dynamic range [1, 3, 4].

To go further, structured light systems are basically a stereo setup in which the correspondence between the two images is not created by matching image features, but by projecting known features and detecting them to establish stereo correspondence. This implies that the projector needs to project a pattern, which can be a laser line sweeping the scene, a gray-code sequence, or a complex image with distinguishable features for every pixel like in single shot methods.

Implementations using an event camera rely on the timing of the scanning point of a video laser projector to find the correspondence. The event camera is able to register the position and timestamp of the moving laser point. It does not need to register its brightness nor its color, as long as the projected brightness at a given pixel is high enough to trigger a change detection. This separates the structured light pattern, in this case, the scanning laser point, from the image content projected.

Finally, being able to project onto a scene, while simultaneously being able to retrieve depth information, opens up SAR applications mentioned before. If the projector used is a video laser projector which is scanning the scene in a raster fashion, adding an event camera additionally creates a structured light system which can provide 3D information for the SAR application, while also benefiting from the advantages of event-based structured light.

1.2 Contributions

In preparation for any 3D reconstruction using a camera and in this case a projector these components need to be calibrated. The calibration allows to project a point in space onto the image plane and project a point on the image plane into a ray in space, respectively. Additionally, in a stereo setup, it allows changing from the camera's coordinate system to the other camera's coordinate system. In a structured light system, the second camera is replaced by a projector which is then modeled as an inverse camera. Although there are established tools for camera and camera-projector calibration, these methodologies work with frame-based cameras. The asynchronous event stream of an event camera needs to be processed first to be able to use conventional tools. For this thesis, a tool to calibrate intrinsic and extrinsic parameters of the camera, as well as the projector, was implemented to streamline the calibration process.

For the main motivation for implementing an event-based structured light system and projecting content independent of the structured light pattern, a tool using the event camera as in [1] was implemented, which operates in real-time with limited frame rates. Typically, in structured light implementations, this depth map is calculated from the camera's point of view. But to accurately project content using the depth information, this depth map needs to be from the projector's point of view. To account for this, the disparity search algorithm is modified. Furthermore, to compensate for inaccuracies in timing of the projector as well as the camera, an algorithm to calibrate for these errors is implemented.

Two demonstrations show possible applications of this principle. Firstly, projecting the live colormap of the depth onto the scene explains the concept directly. Secondly, the depth information can be used to create, modify or correct other content being projected. For example, perspectively correcting a projection mapped image, to appear without distortion, even if the scene is changing. A demonstration is implemented which projects a circle onto a plane, transforming the source circle to appear without distortion even when the plane is rotated.

Finally, measurements were taken to validate the working principles. The accuracy, robustness and influence of content brightness and ambient illumination are evaluated with the results.

To summarize:

- Implementation of a tool to calibrate projector and event camera
- Design and implementation of a structured light tool to measure the depth of a scene,
  - either from the camera's or projector's point of view,
  - including the calibration for the timing behavior of the projector,
  - while projecting mostly arbitrary content
- Implementation of two example applications, live colormap and live perspective correction

1.3 Thesis Outline

The rest of this thesis is organized as follows:

- In Chapter 2, the existing approaches and related works are discussed.
- In Chapter 3, the mathematic and geometric foundations and underlying principles are explained.
- In Chapter 4, the hardware setup is presented.
- In Chapter 5, the calibration, structured light and demonstration implementations are presented.
- In Chapter 6, measurements for evaluation are taken, analyzed and contextualized.
- In Chapter 7, the contributions and limitations of this implementation, and an outline of the directions for future work is summarized.

Chapter 2

Related Work

2.1 Structured Light

Modern camera systems are able to image a scene with steadily increasing performance, for example in terms of spatial resolution. But in the typical construction of a camera system every point of the scene which emits light into the camera system gets projected onto a 2D image plane, the image sensor. This results in the unavoidable loss of depth information. Without further processing or information a point on the image plane can only be reconstructed as the ray on which the original 3D point was located.

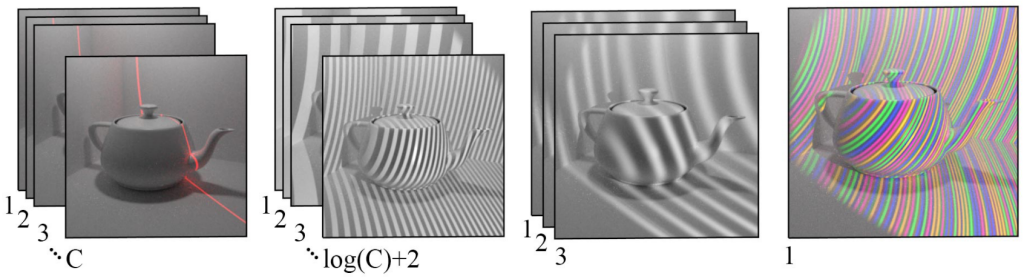
For that reason, depth estimation is a big part in computer vision with numerous applications in 3D reconstruction of a real scene, robotics, or augmented reality. There are multiple solutions for this problem like specialized imaging systems including time of flight sensors and lidar systems, which measure the time an emitted light signal needs to return to a receiver in the system.

Another principle is stereo vision. This principle is based on the way our eyes perceive depth, as our brain recognizes the differences in the images our left and right eyes see. An object close to the eyes or more general to the vision system, i.e. cameras, has a different relation to objects further away or to the background. In a simplified arrangement both cameras are placed next to each other in the x-axis and looking straight ahead. An world point projected on both image planes has a difference in the x-direction on both image sensors, also called disparity, which is inversely proportional to the distance to the camera or depth. [5]

Modeling a camera with a pinhole model, calculating the depth with epipolar geometry, and accounting for misalignment in the cameras with stereo rectification is well understood. The main problem with stereo vision working on two dense images like the ones from two frame-based cameras, is the corre-

spondence problem. Which point in camera two corresponds to a point in camera one? For a dense disparity image, ideally all corresponding points must be found, which is not a trivial task. Especially since world points which project into one camera can be occluded or projected outside the image frame of the second camera. This problem is called stereo matching and has been widely studied whilst still being in active research today. [5]

A popular method of solving this correspondence problem is structured light. Here, one camera is replaced by a projector, which projects a known pattern onto the scene, making this an active method. The projector can be modeled as an inverted camera using the same pinhole model as before. To find the correspondence a coded image or image sequence is projected onto the scene and viewed by the camera. Image processing can then find the properties of the projected images in the camera image and create reliable correspondences [6, 7]. The resulting problem is to find a way to code the image plane of the projector and to reliably detect and match the code in the camera image.



(a) Line Scan (b) Binary SL (c) Phase Shift (d) Single-Shot

Figure 2.1: Examples of structured light systems. Figure (a) shows a line scan system that needs take as many images as desired columns C in the resulting depth map. (b) is a binary structured light implementation which needs to take $\log_2(C)+2$ images. (c) A phase shifting system projects a sinusoidal pattern and needs at least three images. (d) Single shot methods take one image with a densely coded projection. [1]

There are different aspects to optimize for while choosing a coding pattern. Firstly, there is resolution, which implies finding the exact pixel correspondences. Resolution also directly helps with depth accuracy, since the disparity of two corresponding pixels is expressed as their spatial distance in pixel. Secondly, there is robustness to avoid errors from high ambient illumination or scene inter-reflections, where a high light-intensity and light-efficiency from the light source is advantageous. Thirdly, there is speed to get a depth estimation quickly or even being able to capture moving objects in the scene, which implies projecting information dense coding patterns. These three come with a trade-off to each other. [1]

Laser scanning optimizes for accuracy and robustness. Either a point scanner scans the scene in a raster

fashion or a line scanner sweeps the scene. As there is an exact laser position for each image captured, the accuracy of the correspondences is high. The laser concentrates its light output very strongly achieving a high light-efficiency, hence being able to be detected even with a high ambient illumination of the scene. This approach comes at the direct cost of speed, as a full image must be taken and processed for every laser position. This includes a lot of redundant information. Since the light source is concentrated on a point or a line, a lot of the image taken is left without information. The data transfer and processing bandwidth becomes the limiting factor [1]. Current implementations optimizing for these specifications include [8] improving robustness against ambient illumination as strong as sunlight by increasing light efficiency.

To increase speed at the cost of robustness, denser image patterns can be projected to increase the amount of correspondences, which can be detected with a frame. Important examples of this are the graycode pattern, binary pattern and phase shifting pattern. With these image sequences, each pixel in the projector image plane is identified by a code contained at the pixel's position over the sequence. For instance in the binary pattern, each column has an increasing binary number. In each frame of the sequence a column is illuminated or not corresponding to each digit in that number. To find the correspondence, the sequence of binary digits is read for every pixel in the camera image sequence [5]. By reducing the amount of images taken, the speed is increased in these methods. However, the light efficiency is reduced by spreading the projected light out to the whole frame. As the light intensity per pixel is decreased, the robustness against high ambient illumination is reduced. [1]

Going further, single shot implementations like the *Microsoft Kinect v1* use a very dense single image in which all necessary correspondences are coded. This allows for the acquisition of a depth map in every projected and recorded frame. Therefore, it allows for higher speeds than other methods. But similar to previously mentioned coding methods, this spreads the projection area and does not use the available light efficiently, resulting in lower robustness [1]. To add to that, to code the correspondence for every pixel in a single frame becomes increasingly difficult with higher resolutions. As a compromise single shot methods often use lower resolutions and/or periodic coding methods which limit the ability to correctly detect depth discontinuity, which can exceed the period length of the coding method. [6]

2.2 Event Cameras

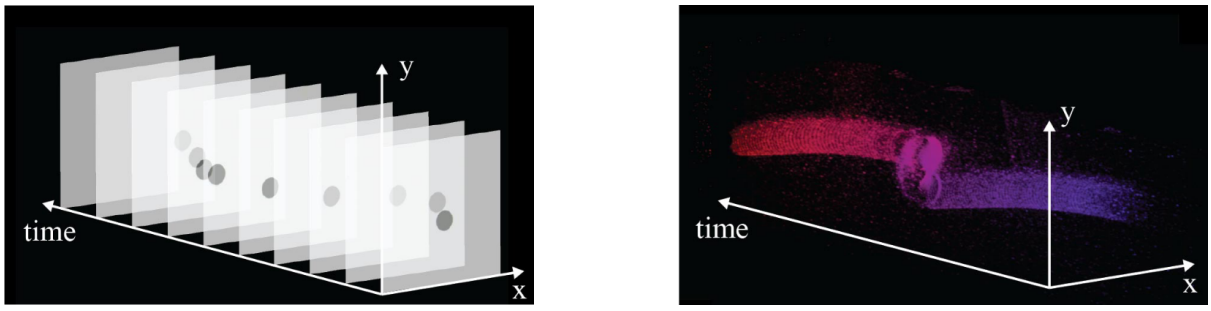
Image sensors and camera systems are well established in every aspect of the modern world. From commodity hardware in toys to highly specialized and performant scientific tools, camera systems fulfill a huge array of applications. Today the active pixel sensor or CMOS-sensor is by far the most commonly

used technology, with the CCD-sensor (charge coupled device) being used in some special cases.

These systems work on a frame-based principle. The light is projected through a lens onto a plane, i.e. the image sensor with a pixel array. In every pixel the incident light is converted to a photocurrent. The pixel integrates the photocurrent over a set period of time and the sensor outputs a frame which represents the absolute brightness at that point in time for every pixel, before starting the next interval of integration. [9]

Depending on the different applications, these sensors are often designed for specific characteristics. These can be a high frame rate, a high low-light sensitivity, a high dynamic range, and a smaller pixel pitch resulting in a high resolution and/or a small form factor. With some of these often playing out as a trade-off, like for example small pixels and high dynamic range.

Alternatively, there is a completely different principle for image sensors than a frame-based sensor. These are dynamic vision sensors (DVS) or often called event cameras. Inspired by and mimicking the working principle of the eye, the main two differentiators are that it works on the scene dynamics and every pixel works asynchronously and independently of each other. The survey paper [4] provides a comprehensive insight in this kind of vision system and is summarized in this section.



(a) frame-based camera

(b) event-based camera

Figure 2.2: Comparison of frame-based and event-based data output. (a) shows the data output of a frame-based camera on a XYT-coordinate system. At constant intervals the camera takes a dense image. (b) the data output of the event camera is sparse in the X- and Y- as well as the T-axis, with a higher temporal resolution. [1]

The basic physical structure of a dynamic vision sensor is similar to a conventional digital image sensor, as it also has a pixel grid on a silicon die, which is the image plane. But instead of integrating the brightness, i.e. induced photocurrent per pixel over time, the pixel monitors for a change in the log photocurrent per pixel. If the photocurrent rises above or falls below a set threshold, the pixel triggers an event, which contains the information about position, time and polarity. After that the pixel memorizes

the current log photocurrent and again monitors for change.

A frame-based camera starts and stops the integration at the same time for all pixels, or in the case of a rolling shutter with a very short delay line by line. Here every pixel outputs a data point at a frequency independent of the changes in the scene. Each pixel in a dynamic vision sensor works independently, so that a data point is only generated when a part of the scene has changed. This reduces the amount of redundant data and therefore the bandwidth and processing. Moreover, each pixel works asynchronously. In an analog circuitry the detection of a brightness change is fast. Since the output data is still digital, an internal clock of 1 MHz timestamps the event, which results in a microsecond resolution. Hence, the sensor can detect very fast changing scenes, like fast motion, vibration and flicker without artifacts like motion blur.

The reset of a pixel after an event takes more than a few microseconds so that the effective sampling rate is lower and in some cases adjustable. Additionally, although the analog circuitry is fast, it has a limited bandwidth. Illuminated with a sinusoidally-varying light source at low frequencies a pixel outputs a certain amount of events. As the frequency of the light source increases, the analog circuitry starts to filter out the changes and the number of events generated lowers. At some cutoff frequency the pixel is not able to keep up and no more events are generated. This cutoff frequency is a monotonically increasing function of the light intensity. With high intensity the pixel bandwidth is typically at 3 kHz (300 µs), while at low intensity this is 300 Hz (3 ms). The total bandwidth is as well limited by the digital read out speed of the sensor and the hardware interface with an event-rate ranging from 2 MHz [10] to 1200 MHz [11]

There are different implementations of dynamic vision sensors, namely the DVS, the ATIS and the DAVIS [12]. The DVS or dynamic vision sensor [10] works just as described above with a pixel that detects only light changes. In some applications though, it is advantageous to also have an absolute brightness measurement, like a conventional camera provides. An implementation that combines the two is the ATIS or Asynchronous Time Based Image Sensor [13]. This sensor integrates per pixel a DVS subpixel that triggers another subpixel upon an event, which creates an absolute light measurement by discharging a capacitor with a photodiode. The pixel sends out two more events which code the time at which the capacitor crosses two threshold voltages. A shorter time between these events means more leaking current and consequently, more light intensity at the photodiode. This way an event also contains the absolute brightness at that pixel. The ATIS achieves high dynamic range at the cost of resolution since each pixel needs more space. Also in dark scenes the long discharge can be interrupted by new events. Finally, the DAVIS or Dynamic and Active Pixel Vision Sensor [14, 15] combines a conventional pixel from an active pixel sensor with a DVS pixel. For an absolute brightness measurement, the whole

frame is read out, either at a constant rate or on demand. A smaller pixel is of advantage, since the photodiode is shared and the active pixel readout only adds little area. But as the active pixel sensor part is not as optimized, the dynamic range is limited.

The advantages event cameras bring are manifold. Firstly, event cameras have a high temporal resolution as a detection is fast and is timestamped with microsecond resolution. This allows for the detection of very fast motion. Secondly, low latency is achieved as each pixel works independently and a change detection is triggered as it occurs. The pixel does not have to wait for a global exposure time. Thirdly, only a brightness change is detected, which leads to low redundancy. Only the pixel at changing brightness need to reset and only their data is transmitted. Moreover, this causes lower power consumption and lower bandwidth. Fourthly, the pixels use a logarithmic scale and each one can independently adjust for its current absolute brightness. This results in a high dynamic range as high as 120dB, which is notably higher than the typical 60dB of frame-based sensors [4].

The complexity and size of the pixel results in a lower resolution compared to current frame-based cameras, which is the most obvious technical disadvantage. In addition, the new representation of visual data, which comes with event cameras, imposes the task of developing new algorithms to process such data. Since event camera data is inherently sparse and asynchronous, conventional computer vision methods, which rely on dense images and image sequences, are not usable with event data. Furthermore, the information event data contains (increase/decrease) needs to be processed differently than the grayscale or color information in a traditional camera. Absolute brightness, brightness change and motion are coded differently. Lastly, noise and dynamic effects behave differently and these effects in event cameras are not completely characterized. Consequently, algorithms need to be adjusted or redesigned to take the advantage of the unique properties of event data and event cameras.

In conclusion, the improvements event cameras and dynamic vision sensors bring can be applied in a wide field of computer vision tasks. The main advantages, high dynamic range, high temporal resolution, low latency and low power, bring real-time applications like robotics to mind. But in all fields of computer vision these advantages can be exploited, like object tracking, gesture recognition, optical flow estimation, high dynamic range image reconstruction, Simultaneous Localization and Mapping (SLAM), and depth estimation, in particular structured light 3D scanning which is the main topic of this thesis.

2.3 Event-Based Structured Light

As explained in 2.1, building a structured light system includes designing around a trade-off between resolution, robustness and speed. The limitations in speed mostly result from the necessary bandwidth of the image acquisition and processing. Two recent publications, MC3D (Motion Contrast 3D) [1] and ESL (Event-based Structured Light) [3], aim to alleviate these restrictions by combining the structured light technology with an event-based camera and are the basis for this thesis.

Pioneering the idea MC3D points out the mentioned trade-offs and shortcomings of frame-based structured lights systems and summarizes these with:

- A laser scanning device, specifically a point scanner using a set light budget most efficiently, maximizes SNR, and therefore enhancing robustness.
- Frame-based systems do not utilize most of the captured frame in point scanning systems.
- In scenes with low as well as highly specular BRDFs[1] the range of brightness values reaching the sensor may exceed its dynamic range.

For the first point a video laser projector that scans the image with a single laser beam in a raster fashion can be used. This combines the benefits of a laser scanner with the speed of video projector. To tackle the latter two points the advantages of an event camera can be used. The event camera only registers the changes in reflectances, i.e. motion, or the changes in brightness like a changing light source. As the laser point scanner scans a scene, the camera registers and transmits only the areas where the point scanner moved its laser point at a given time, assuming no other motion. This reduces the necessary bandwidth immensely. The camera is fast enough to follow the laser point scanning a scene at 60 Hz, without generating to much and redundant data to transmit and process. In addition to that, the high dynamic range of an event camera helps with otherwise challenging scenes with high global illumination and high scene inter-reflectance. Experimental result confirm the improvements proposed.

ESL iterates on this approach and introduces an additional processing step which considers the spatial-temporal neighborhood of width W and time T around already found correspondences to find a better matching one in that neighborhood. This approach leads to several advantages.

- Reduction of event noise and jitter by considering a spatial-temporal neighborhood.
- Less data by using the spatial neighborhood. There is no averaging over multiple scans like point wise methods use to improve accuracy.

[1] Bidirectional Reflectance Distribution Function: function that outputs the ratio of exiting radiance to the irradiance incident on a surface point depending on the incoming and outgoing light direction.

- Depth parametrization and stereo matching are combined in a single step. Classical approaches like Semi-Global Matching (SGM) [16] are using two-steps, establishing correspondences and then triangulating depth.
- The parameter W controls a trade-off. Lower values, i.e. searching in a smaller neighborhood, produce finer detail depth maps, while larger values suppress more noise effects.

For evaluation ESL implements an event-based structured light system. Results using this algorithm is compared to the MC3D and SGM algorithms. All algorithms are applied on a single time map pair, meaning a 16ms acquisition time. On static scenes a ground truth is defined as the pixel-wise average over 1s, i.e. 60 frames, using the MC3D algorithm. Measuring the RMSE of all algorithms against the ground truth, ESL outperforms both, sometimes significantly. It is also shown that, just like MC3D, ESL works reliably in different ambient light conditions, compared here to the conventional frame-based implementation *Intel RealSense D435*. Qualitative comparison in dynamic scenes also lead to a favorable result for ESL with better noise suppression and less artifacts on edges with depth discontinuity. The trade-off using different values for the neighborhood size W as anticipated by the paper is also confirmed experimentally.

This section has summarized the papers MC3D and ESL which implement a structured light system for depth map acquisition with a point scanner. Inherent shortcomings of conventional frame-based structured light systems and how an event camera provides solutions were discussed. The systems were implemented and their performance was confirmed experimentally. A more detailed look into their implementations is in section 3.3.

2.4 Structured Light based Augmented Reality

Augmented reality describes the practices of enhancing real world scenes with virtual assets and is a topic of ongoing research. Fields that may benefit from this are for example entertainment, education, or collaborative work [17, 18, 19, 20]. This can include all senses like for example hearing, though most applications of augmented reality are in the visual realm. Examples are placing virtual objects in 3D space through a heads-up display like the Google Glass in 2013 or inside of a (live) video of a real scene as easily accomplished in today's smartphones. These are called tracking based augmented reality, because they track the view point of the user or viewing device, e.g. smartphone to adjust the content displayed [21].

A distinction of augmented reality is called spatial augmented reality (SAR) in which the virtual assets are directly rendered in the real world without the need for the user to wear additional hardware [22]. A way to accomplish this among others is projecting textures, text or shapes onto surfaces in

the real scene, enhancing scenes with information and adding interactive abilities. This can be done in various scales from room size, to theater, or projecting onto buildings. If contents of the projection are mapped onto surfaces and objects in the real scene this method is often called projection mapping.

Example implementations on static scenes include among many others interactive projector-guided painting in [23] and augmenting the area around a television with computer graphics to enhance gaming experiences [24]. Projection mapping on moving objects is explored in for example [25], which projects textures and patterns on a moving actor on a musical stage, by real-time masking the actor against the background. Integrating a *Microsoft Kinect* [26] and [27] implement projection mapping systems on and around moving objects or a steerable projection system correcting for surface angles in real-time, respectively. These are just examples of a wide range of research in this field.

For a user to accept the virtual content as part of the real environment high accuracy of the placement and in the case of dynamic scenes stability and low latency are mandatory. Hence, depending on the application it is advantageous to necessary to know the 3D properties of the scene, the objects and shapes on which the video is projected. This can be done beforehand and implies that the scene is static as it is done large scale projection mapping onto buildings and infrastructure objects. Projecting on dynamic scenes with varying depth or in applications in which interaction with the user is intended 3D information about the scene greatly increases the system performance and needs to be acquired on demand or live in real-time.

As mentioned before some implementations include depth sensing devices like the Microsoft Kinect. Another good example for this is the SARndbox [28]. Here a video projector is mounted on top a Sandbox and is projecting vertically down. The Kinect depth camera is placed with a little offset and measures the 3D shape of the sand landscape. This dept map is turned into a topographic maps including contour lines. This map is projected onto the sand landscape and adapts in real-time to changes in the sandbox. Additionally, the topographic map is used for fluid simulations, which are then also projected onto the sand. The depth map also allows for gesture detection, with which a user is able to add the simulated water to the landscape. This is a good example for spatial augmented reality which is based on the depth map of the scene acquired live as it enables rich features and interactive content.

Considering this thesis, while looking at structured light systems as a solution for this problem it comes to mind that this system already employs a projector to project the coded images for depth reconstruction. This interferes with the projected content used for augmented reality. One possible solution could be injecting the coded images into the projection mapping, but even very little distractions destroy the acceptance of the augmented reality as real. To resolve this issue, the structured light images can be

projected in a non visible spectrum of light, i.e. infra-red (IR) light, as it is done with the Microsoft Kinect and in [2].

In [2] a system is set up with two high frame rate projectors, RGB and IR respectively, and a high frame rate IR camera as well as processing on a GPU. The IR projector projects a gray-code pattern onto a level plane to calculate the depth map with the help of the camera. The RGB projector is placed next to the structured light system, with all optical axes being approximately parallel. To achieve good results for the depth map even at fast moving dynamic scenes the IR projector and camera capture the scene synchronously at 1000fps, which leads to a 512 depth map at 500fps through highly parallel processing in the GPU. The RGB projector displays an 1024×768 image at 60Hz for various augmented reality applications. Examples of applications implemented in [2] are a depth-based color map, an AR spirit-level and an AR wrist-watch projected onto a live tracked wrist. Due to the structured light system this implementation provides good resolution with a very good speed, but comes at the cost of a complex physical setup.

In conclusion, spatial augmented reality employing projection mapping relies on various methods, most notably 3D knowledge about the scene, to achieve believable integration of virtual content. Structured light systems provide these depth maps at usable speeds with the caveat of also projecting onto the scene. To circumvent this, the structured light projection might be moved into non-visible infra-red light.

Chapter 3

Foundations

This chapter lays out the underlying mathematical and geometric principles, which are referred to later in the thesis. First, the model of the camera and projector is explained which allows to establish a geometric framework to project points from world coordinates in the image plane of the camera or projector and back. Secondly, this geometry is used to derive the depth triangulation using two cameras or a camera and projector. Lastly, the current implementations of event-based structured light on which this thesis is based are explained im more detail.

3.1 Pinhole Model

For a structured light system to work the camera and the projector need to be calibrated, i.e. the parameters to map a 3D point onto the image plane of the camera or the projector need to be calculated. For this the so called pinhole-model is used and the parameters are for one the intrinsic parameters that describe the focal length and the image sensor and the distortion coefficients which model the lens distortion non-linearly. Furthermore, the position and rotation of the optical systems relative to each other need to determined for the stereo relation to be used, these are called the extrinsic parameters.

The pinhole model of a camera is defined as

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \, [R|t] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.1}$$

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

where K are the intrinsic parameters and Rt are the extrinsic. The product $P = K\,[R|t]$ is called the projection matrix.

The extrinsic parameters $[R|t]$ map a 3D point $(\,X\;\;Y\;\;Z\;\;1\,)^T$ in normalized homogeneous coordinates to a 2D point $(\,x\;\;y\;\;1\,)^T$ as well in homogeneous coordinates. This achieves a change of basis to the camera coordinate system.

At this point it is possible to apply a non-linear model to correct for lens distortion and this thesis uses a standard model with five coefficients k_1, k_2, p_1, p_2 and k_3 which is

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 xy + p_2(r^2 + 2x^2) \\ y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y^2) + 2p_2 xy \end{pmatrix} \tag{3.2}$$

with

$$r^2 = x^2 + y^2$$

The intrinsic parameters then map the distorted 2D point in the camera coordinate system $(\,x'\;\;y'\;\;1\,)^T$ onto the image sensor which includes the focal length (f_x, f_y) and the principle point (x_0, y_0).

As a remark, this transformation reduces the dimensions by one, since $[R|t]$ is a 3×4 matrix. While projecting a 3D point to the image plane, the depth from the camera's point of view is now represented by the scaling factor λ. To get the actual position on the image sensor the result needs to be divided by this factor. Going the other way, from a pixel on the image sensor to the 3D world, naturally, this scaling factor is not known, so the position of the point in 3D space can only be described up to a ray coming from the camera's optical center going through the point $(\,x'\;\;y'\;\;1\,)^T$.

A projector can be modeled in just the same way as an inverse camera. With the parameters known a pixel in the image plane of the projector can be projected out onto a ray in 3D coordinates. The 3D point is where this ray hits an object.

The parameters of this model are found through calibration, which is done by capturing multiple images of a reference object with known or defined world coordinates and solving for the intrinsic parameters, distortion coefficients and extrinsic parameters by using an algorithm called Zhang's method [29].

3.2 Stereo Vision

As the depth of a point is lost in the projection onto the image sensor, this depth needs to be triangulated with the use of more than one camera, in the simplest case two cameras. The geometry which defines this is called epipolar geometry and relies on a corresponding point triple, point in 3D, pixel in camera one and pixel in camera two. To more easily find such correspondences a transformation called rectification is employed, which simplifies the search and depth relationship of correspondences.

3.2.1 Epipolar Geometry

The depth estimation relies on the basic principles of stereo vision. In the simplest configurations this consists of two cameras looking straight ahead mimicking the left and right eye. The principle behind the mathematics is called epipolar geometry and is well understood. Here the distance between the two pixels on the respective image planes, which correspond to the same 3D point in front of the cameras is proportional to the depth. Matching pixel from one camera to the other and finding the stereo correspondences is often called epipolar search. To create a dense map of pixel pairs and therefore a dense image with depth information is not a trivial task. [5]

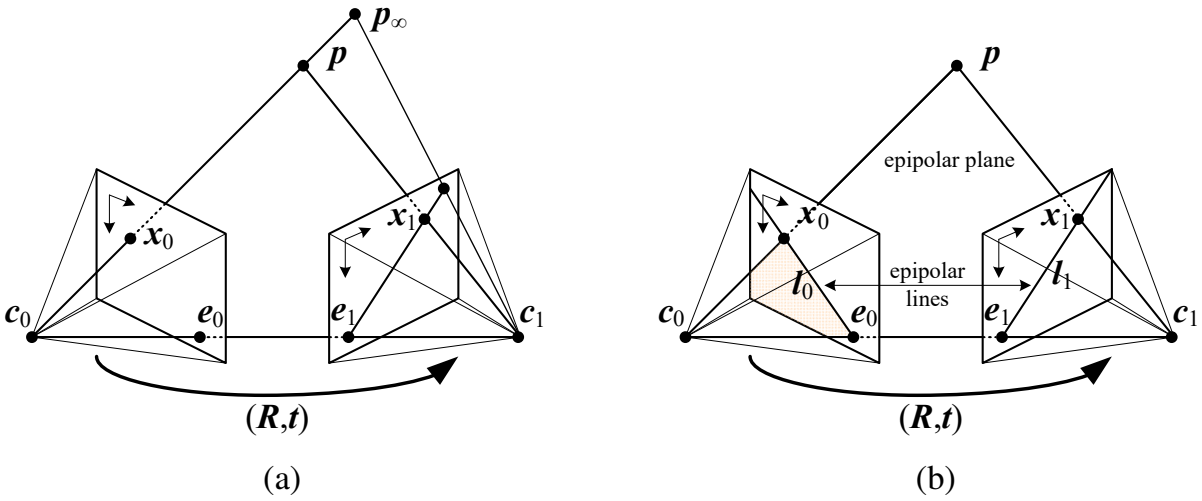


Figure 3.1: Principle of epipolar geometry. (a) Depending on the distance of point p to the left camera, its projection in the right camera lies on an epipolar line. (b) Doing the same for the left camera, creates another epipolar line. Both lines lie on the epipolar plane, intersecting the plane spanned by c_0, c_1, and p with the image planes. [5]

The principle of epipolar geometry as explained in [5] and [30] and as implemented in OpenCV is displayed in figure 3.1. Two cameras with optical centers c_0 and c_1 and a relative rotation and translation (R, t) view a point p in world space. If this point moves on a ray coming from c_0 through p, it is always projected on the point x_0 in the left image plane. In other words the distance to c_0 can not be determined

by its projection. But depending on the distance to c_0, the projection of p in the right image plane lies on a line, the epipolar line. The epipolar line is the projection of the ray from c_0 through p. On one side this line is bounded by the optical center c_0 which is projected onto e_1. This point is called an epipole. On the other side the epipolar line is bounded by an infinitely far away point p_∞. Conversely, moving the point p on a ray from the optical center c_1, its projection in the left image planes also lies on an epipolar line. One epipolar line is the projection of the other in its image plane and both are the intersection of the respective image planes and the plane spanned by c_0, c_1, and p. It is possible to see that the distance of p to the optical center of one camera is proportional to its position on the epipolar line of the other camera. An image on the image plane of one camera can be projected onto the image plane of the second camera through the intrinsic and extrinsic parameters. Therefore, the epipolar geometry of a stereo pair can be defined through calibration.

3.2.2 Rectification

To find the distance of a point, it is now necessary to find its correspondence on the epipolar line in the second camera. This search can be simplified with a process called rectification. In a horizontal stereo setup, as used in this thesis, the epipolar lines are mostly horizontal. Rectifying means to warp the images in a way that the epipolar lines are the horizontal scanlines, that corresponding epipolar lines have the same position on the y-axis and that a point infinitely far away, would project onto the same pixel, i.e. zero disparity. This is achieved by rotating both cameras around their optical center such that their image planes are parallel to each other and to the baseline vector. This achieves the horizontal epipolar lines. Further, the image planes are scaled such that they lie same plane, which results in equal y-coordinates for corresponding epipolar lines and a disparity of zero when the 3D point is infinitely far away.

For each camera a projection matrix ($P_{\text{rect},1}$, $P_{\text{rect},1}$) as in equation 3.1 and an additional rotation matrix ($R_{\text{rect},1}$, $R_{\text{rect},2}$) are defined. The rotation matrices are both considered to be the rotation from their respective camera coordinated system to their rectified camera coordinate system to transform captured images to their rectified equivalence and back. Since the optical center stay in place $t_{\text{rect},1}$ and $t_{\text{rect},2}$ are not necessary. The rotation matrices are found by defining the new x-axis parallel to the baseline vector b, the new y-axis orthogonal to the x-axis and the old z-axis of the left camera[1] and the new z-axis orthogonal to the previously chosen x- and y-axis. To keep the warping from this rotation to a minimum it is advantageous to configure the two cameras like a rectified pair as best as possible in the first place.

[1] This vector can be chosen arbitrarily and is chosen as the left cameras old z-axis to keep rotation to a minimum.

The projection matrices are considered projections from the first cameras rectified coordinated system into the respective image planes. Since the coordinate system is the rectified one and both image planes are parallel to each other, the rotation is the identity matrix I. The translation from camera one to two is the length of the baseline vector b along the x-axis $(|b|, 0, 0)^T$ and only applies to the second camera. The new camera matrices can be chosen arbitrarily as long as they are the same. Here it is advantageous to scale up a smaller image to a bigger one to avoid aliasing or loss of information. The resulting projection matrices have the form

$$P_{\text{rect},1} = K_{\text{rect}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad and \quad P_{\text{rect},2} = K_{\text{rect}} \begin{bmatrix} 1 & 0 & 0 & |b| \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

After rectification the camera systems follow simple geometry and a simple relationship from disparity to depth can be formulated as

$$d = f\frac{|b|}{Z} \tag{3.3}$$

where d is the disparity in pixel, f is the focal length, $|b|$ is the length of the baseline vector b in m, and Z depth of the 3D point in m in the rectified coordinate system. With the depht and $P_{\text{rect},1}$ the 3D point can be calculated. This 3D point differs from the same point in the unrectified camera coordinate system only up to a rotation since the optical center of the cameras stay in place.

3.3 Current Implementations

This thesis is based on two recent publications, MC3D [1] and ESL [3], which are explained in more detail in this section.

3.3.1 MC3D

The main principle of this implementation and of ESL is the conversion of spatial projector-camera disparity to temporal correspondences between the laser point and the point in time an event is generated. Assuming that each frame the laser point scans the image plane of the projector at a constant or known speed, each pixel within that frame can be assigned a timestamp. Conversely, if within the event stream, generated by the camera, the start and end of a scanned frame are detected or coded through trigger events, the timestamps of the containing events can be normalized to the time of one frame. Then the correspondence of projector and camera pixels can be found through the corresponding timestamps.

As mentioned in section 2.3 MC3D uses a laser scanning implementation for good results in resolution and robustness. As a laser scanner the *SHOWWX* laser video projector from microvision [31] with a resolution of 848x480 was used, which scans a red, green and blue laser diode in a raster pattern with help of a MEMS micro mirror at 60 Hz. This acts as a self contained 60 Hz fixed frequency point scanner. Additionally, a variable frequency line scanner was used for some experiments. As an event camera MC3D uses the iniLabs DVS128 [10] with a resolution of 128×128 pixel and 120dB dynamic range.

In MC3D the setup is explained in a simplified manner such that there is no distortion, blurring, or aberration and an already rectified projector camera pair with equal focal lengths f and are separated by a baseline b. Both projectors are assumed to be a line scanner, scanning a vertical line over the scene. Looking at one row figure 3.2 shows the model and how the correspondence is created. From the correspondence the disparity and the depth are calculated according to stereo vision and epipolar geometry.

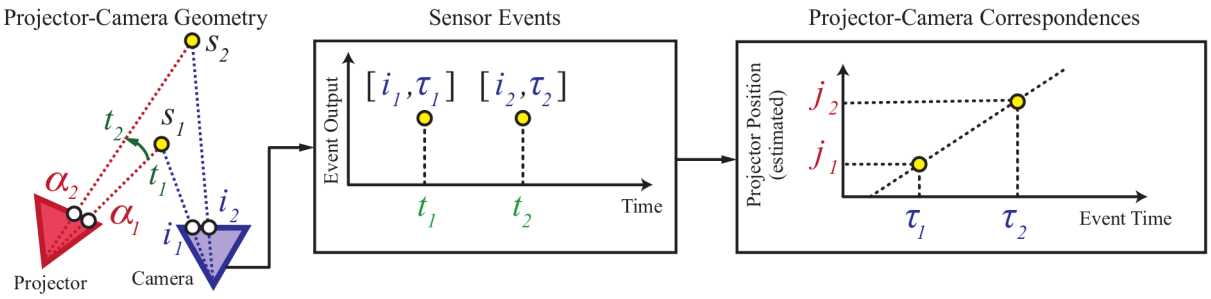


Figure 3.2: MC3D model of a row: Projector and camera are viewed from the top. At times t_1 and t_2 the projector laser is at positions α_1 and α_2 and is striking scene points s_1 and s_2, respectively. As the scanning point moves, the camera sees changes at column positions i_1 and i_2 at the respective times and generates the events $[i_1, \tau_1]$ and $[i_2, \tau_2]$. Since there is a proportional relationship of timestamp τ to scan angle α and therefore projector column j, the corresponding columns j_1 and j_2 can be determined. [1]

In experimental evaluations MC3D shows that the system creates a viable depth map of simple and complex shapes, while outperforming a *Microsoft Kinect v1* [32] in "fidelity" and a traditional laser scanning system in speed. The system also surpasses the *Kinect* in strong ambient light, despite having a much smaller light source energy (1mW vs. 60mW). In scenes with strong inter-reflections and with specular materials MC3D achieves better shape reconstruction than Gray code structured light systems.

3.3.2 ESL

The implementation in the paper ESL [3] iterates on these results as well as explaining more details of the implementation. This is described in the following paragraphs. For the light source ESL also uses a

video laser projector, specifically the *Sony Mobile projector MPCL1A* [33]. As a video laser projector it scans the whole frame in a raster fashion with 60Hz and has a resolution of 1920×1080 pixels. The event camera is a *Prophesee Gen3* DVS camera, with a resolution of 640×480 pixel [13, 34]. The projector and camera are connected via an external jack to synchronize the frames and the event timestamps.

For the depth estimation ESL considers an ideal stereo pair setup where each laser scan point illuminates one scene point which triggers one camera pixel to show limitations in this simplification. This is obviously not directly possible due to the mismatch in resolution and focal length of projector and camera. Additionally, ESL points out that the sweeping time of the projector per pixel is considerably faster than the temporal resolution of the camera. The projector is rotated $90°$ to counteract this limitation on epipolar lines. More on this in section 4.1.3. Lastly, the event noise sources and types are mentioned. Event latency describes the time from brightness change to the triggered event. Since this is mostly the same for all events, it can be considered a constant offset. Jitter on the other hand is random noise in the event timestamps and depends on scene and illumination conditions. BurstAER mode is a read-out mode in which whole rows or groups of rows of the sensor are read out at once and all containing events get assigned with the same timestamp. This can lead to banding effects in an event time map and is also detrimental to the depth estimation.

An ideal stereo pair for processing is created by calibrating and rectifying the projector camera pair. ESL achieves this by converting the event stream of the camera to a frame-based representation and using typical structured light calibration tools including [35] on the basis of [29].

The frames used for the stereo correspondence search are time maps generated by the projector and the event camera. A time map is a type of representation of an event stream in a given time interval. It is a sparse image with the camera's resolution in which each pixel is assigned the timestamp of the last event at that pixel in that interval, zero if no event occurred. Here, the time interval is naturally one scanning period of the projector. In the case of 60Hz the period is 16ms. Every pixel in the image plane $\mathbf{x}$ gets a timestamp assigned $\tau(\mathbf{x})$. The projector can be modeled as an inverse event camera. It creates an event for each pixel the scanning laser traverses in a raster manner, each event being about 8ns apart[2]. Theses events can also be combined to a time map for the projector which stays the same for every depth estimation. Figure 3.3 shows these time maps.

[2]The laser scans the frame at 60Hz with a resolution of 1920×1080. This results in a time per pixel of $\delta t = (f \cdot W \cdot H)^{-1} = (60Hz \cdot 1920 \cdot 1080)^{-1} \approx 8ns$

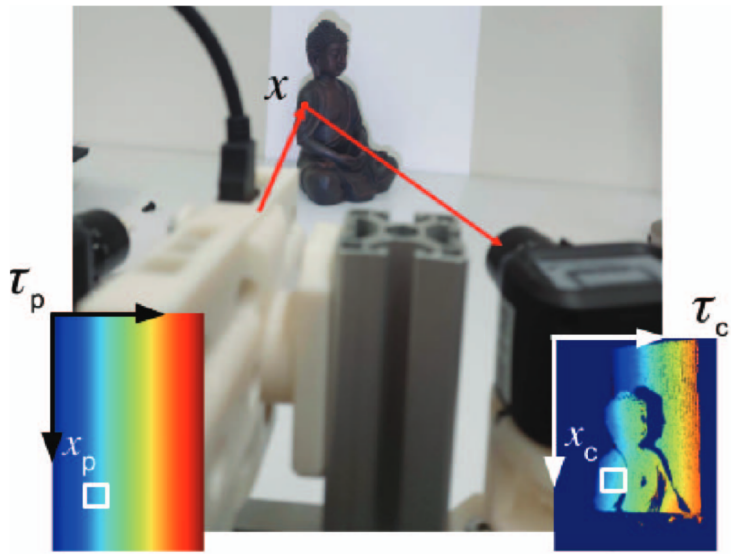


Figure 3.3: Representation of time maps used for stereo vision in ESL. The rotated projector projects a white frame onto the scene, which is captured by the event camera. On the lower left the time map of the projector τ_p is depicted. A colormap shows the timestamps from start to end of a frame increasing linearly in a raster fashion from bottom left to top right. On the lower left the time map of the camera τ_c uses the same colormap and shows the displacement of corresponding pixels $\mathbf{x_p}$ and $\mathbf{x_c}$ due to the depth of the scene. [3]

The stereo setup is modeled as the chained application of a reverse projective transformation from the camera into 3D coordinates π_c^{-1}, a change of basis to the projector's coordinates system T_{pc} and a projective transformation into the projector's image plane π_p. With this a point $\mathbf{x}_c$ on the event camera's image plane corresponds to a point $\mathbf{x}_p$ in the projector's image plane. As a projective transformation in the pinhole model loses one degree of freedom, i.e. the depth, this information must be added in the reverse transformation and it is expressed as $\pi_c^{-1}(\mathbf{x}_c, Z(\mathbf{x}_c))$. Z is the depth of pixel $\mathbf{x}_c$ from the camera's point of view. This results in

$$\mathbf{x}_p = \pi_p \left(T_{pc} \pi_c^{-1} \left(\mathbf{x}_c, Z \left(\mathbf{x}_c \right) \right) \right) \tag{3.4}$$

As the projector is modeled as an inverse event camera, an "outgoing event" from the projector creates an event in the camera. Considering the relationship from $\mathbf{x_p}$ to $\mathbf{x_c}$ in equation 3.4, it is possible to say that the time map of the projector induces a time map in the camera, expressed with

$$\tau_c(\mathbf{x}_c) = \tau_p(\mathbf{x}_p) \tag{3.5}$$

ESL calls this *time constancy principle* and it plays the same role as the brightness constancy assumption $I_1(\mathbf{x_1}) = I_2(\mathbf{x_2})$ in passive multi-view stereo.

To create the depth map, first the correspondence is searched along the epipolar lines, which finds the $\mathbf{x}_c$ and $\mathbf{x}_p$ along this line with the smallest difference in timestamp. This generates a complete disparity map, which can be further processed to depth using the cameras calibration.

Addtionally, to improve the time-constancy per camera pixel the best matching projector pixel is searched inside a $(W \times W \times T)$ neighborhood with $T = 1/f$. The expressions $\tau_c(\mathbf{x}_c, W)$ and $\tau_p(\mathbf{x}_p, W)$ are representing this neighborhood around $\mathbf{x}_c$ and $\mathbf{x}_p$, respectively. With the error function

$$C(\mathbf{x_c}, Z) \doteq ||\tau_c(\mathbf{x}_c, W) - \tau_p(\mathbf{x}_p, W)||^2_{L^2(WxW)} \tag{3.6}$$

the correspondence and therefore Z^* is searched which minimizes said error function:

$$Z^* \doteq \arg\min_Z C(\mathbf{x}_c, Z) \tag{3.7}$$

This processing step improves the quality of the depth map by reducing noise without averaging over multiple scans. This enables the high scan speed of the projector and the accurate depth map of moving objects. Unfortunately, this step is processing heavy and done offline, thus making it unpracticable in a real-time application and is not used in this thesis.

Chapter 4

Setup

This chapter includes an overview over the hardware, namely the camera and the projector, and the software used. Further, it describes properties of the camera and projector and how underlying settings for the camera are found and set.

4.1 Hardware

A structured light system consists in its simplest form of a directed light source, i.e. a laser scanner or a projector, and a camera. These devices are arranged in a fixed relative position to each other. This may range from a stereo camera rig in experimental setups to a sealed enclosure in a finished product. The optical center of projector and camera are necessarily off axis to each other for the stereo vision to work. This leads to the camera picking up shadows in the projected image where no depth reconstruction is possible. To mitigate this more elaborate systems use two cameras on each side of the projector and stitch the resulting images together.

In this thesis an experimental setup on a stereo rail with one projector and one camera like MC3D [1] and ESL [3] is used.

4.1.1 Camera

For the event camera a *Prophesee* Evaluation Kit 1 (EVK1) [34] with a Gen3.0 [36] camera is used. The kit provides an USB3.0 interface which combines data exchange and power delivery. The dynamic vision sensor has a resolution of 640×480 pixel with a 15 µm pixel pitch and provides contrast detections only. The sensor has a dynamic range of greater than 120 dB, a typical latency of 200 µs and timestamps the events with microsecond precision. The sensor is 9.6mm $\times$ 7.2mm and paired with a 8.5mm CS-mount lens it covers a $59°$ horizontal and $46°$ vertical angle of view. The aperture is fixed at

$f/8$ which provided a good compromise between exposure and depth of field.

The behavior of the pixel in the sensor can be adjusted with several bias voltages. In case of the Gen3.0 camera these bias voltages are represented in the actual mV values which can be sent to and read from the camera via the *Prophesee Metavision SDK* [37]. The bias voltages are:

bias_diff: This voltage sets the reference voltage from which the change is measured. It is not recommended to be changed and was left at its default value set by *Prophesee* in this implementation.

bias_diff_on: This voltage is greater than bias_diff and sets the upper threshold to detect a positive change. A value closer to bias_diff means a smaller difference in rising illumination to trigger an event with positive polarity, hence a higher sensitivity as well as higher noise and vice verse.

bias_diff_off: This voltage is lower than bias_diff and sets the lower threshold to detect a negative change. Similar to bias_diff_on a value closer to bias_diff means a smaller difference in falling illumination to trigger an event with negative polarity, hence higher sensitivity as well as higher noise.

bias_fo and **bias_hpf**: These voltages set the cut-off frequencies of a first order low pass filter and a first order high pass filter, respectively. The frequencies refer to the frequency of the change in illumination. This can filter out a light source flickering due to pulse-width-modulation, vibration or motion of objects with a certain speed and size.

bias_refr: This voltage sets the ”refactory period” in which a pixel is blind after it triggers an event or its deadtime. With this the maximum sampling rate of the sensor can be set.

bias_pr: This voltage sets the bias current of the pixel amplifier and with that the effective bandwidth. It is not recommended to be change this voltage and was left at its default value set by *Prophesee* in this implementation.

These biases strongly influence the behavior of the sensor and therefore the event stream. Combined with the projector a few considerations lead to the optimal bias values which are described in section 4.1.3.

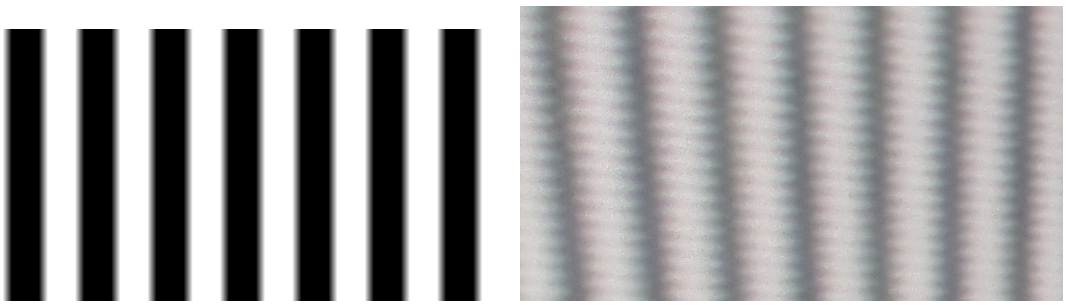
4.1.2 Projector

The main idea in this thesis is to use a laser video projector that produces the image by scanning a laser in a raster scan manner. This projector works by mixing three LED lasers for red, green, and blue and reflecting the resulting laser with a MEMS (micro-electro-mechanical systems) mirror. The mirror moves in such a way that the laser scans the screen in the desired raster fashion. For every laser direction the mirror is the source and can be viewed as the pinhole in a pinhole camera model, albeit a reverse

camera in this case. With this there is no need for a lens and no need to focus the image, which is very advantageous for the structured light system.

For this the *Nebra Anybeam MEMS Laser Projector* was used. It has a resolution of 1280×720 and a fixed refresh rate of 60 Hz. The laser consists of three laser diodes and the projector is a class 1 laser product, resulting in a brightness specification of 30 ANSI lumens. The angle of view is $39°$ horizontally and $22°$ vertically.

The projector had an unexplained jitter in the horizontal speed while scanning the lines. This leads to a periodically recurring zigzag pattern in vertical lines in most of the image as seen in figure 4.1b. No setting in the projectors menu had an effect on the jitter. Also different frames per second in the video feed to the projector ha no effect. It was not possible to improve this behavior and this artifact is present in all measurements using this implementation, which are evaluated in chapter 6.



(a) Part of original image with horizontal stripes (b) Photo of that part projected onto a screen

Figure 4.1: A test image (a) with vertical alternating black and white lines each four pixels wide is projected onto a flat screen. Figure (b) shows a photo of the projection of that image. Jitter in horizontal scan speed is visible. The lines are distorted in a zigzag pattern with an amplitude changing over time. The maximal amplitude was reached irregularly but at least once a second. With a width of four pixel per vertical line in mind the maximal amplitude of the zigzag pattern is about two to three pixel wide and the zigzag pattern repeats after two lines.

4.1.3 Physical Setup

The camera and projector are placed on a stereo rail with a variable width or baseline. To enable fine control of at least one component the camera was placed on two ball heads to be freely moved in position and orientation. Both camera and projector were roughly oriented to have a parallel optical axis and no offset of their optical centers in the y- and z-axis. The baseline b, i.e. the distance on the x-axis, is roughly 5.2 cm. The angle of view of the camera and of the projector allow for the whole projected frame to be captured by the camera at any practical distance.

To test this setup initially a full white frame was projected onto a projection screen approximately par-

allel to image planes. A few problems emerged with this setup. The camera was not able to capture the whole projected frame with any sensible bias setting, dropping events even through the theoretical maximal event rate was not reached. The dropped events were on the right side of the projected frame, i.e. dropping events on a line in the camera after a certain amount of events on that line were triggered. The projector scans a frame in about 13 ms with 3 ms of time to reset the laser for the next frame. With a vertical resolution of 720 lines one line is scanned in 18 µs. To many events triggered on one line is probably a limit of the sensor.

Furthermore, the correspondence of two pixels in the camera and projector is searched along the epipolar lines, which in this setup is approximately along the lines of the camera and the projector. A projector line is 1280 pixel wide, so one pixel is scanned in about 14ns, which is faster than the temporal resolution of the sensor. The same problem and solution is mentioned in ESL [3].

To solve these problems, the projector is rotated $90°$ counter-clockwise like mentioned in section 2.3 in the ESL paper and as seen in figure 3.3. The lines of the projector are now scanned starting from the lower left vertically up and left to right with increasing line numbers. Per scanned projector line the laser point now moves over multiple cameras lines. This increases the time difference of the laser point triggering two horizontal adjacent camera pixels, i.e. two pixel on a line, from horizontally adjacent pixels to vertically adjacent pixels in the projector. The increase of the time difference is to least 18 µs and is inside of the cameras temporal resolution, as well as it does not overload the camera line readout speed.

With the angles of view described in section 4.1.1 and 4.1.2 the projector fills an area of about 225×400 pixel in the camera image if projected onto a plane screen. As the rotated projector has a native resolution of 720×1280 one camera line covers 3.2 projector columns as well as one camera column covers 3.2 projector lines. This implies that while scanning three lines only one pixel per line of the camera sensor gets hits and triggers an event. The reset time for an pixel after triggering an event can be adjusted from roughly 30 µs to 50 ms with the `bias_refr` voltage. Very small values may increase noise since the events can trigger more often. Therefore the pixel triggers per projector frame only once and is blind to immediately subsequent triggers from the laser point as it scans the next lines.

The biases of the camera need to be tuned to match the application. Firstly, the voltages `bias_fo` (low pass filter) and `bias_hpf` (high pass filter) can be used to filter events from unwanted movement or vibration or light flicker. In this setup neither of these effects very present. Hence, both bias voltages were adjusted so that they do not filter out the periodic triggering of the events with the 60 Hz refresh rate, while still filtering events with a higher or lower frequency to suppress noise effectively. Secondly,

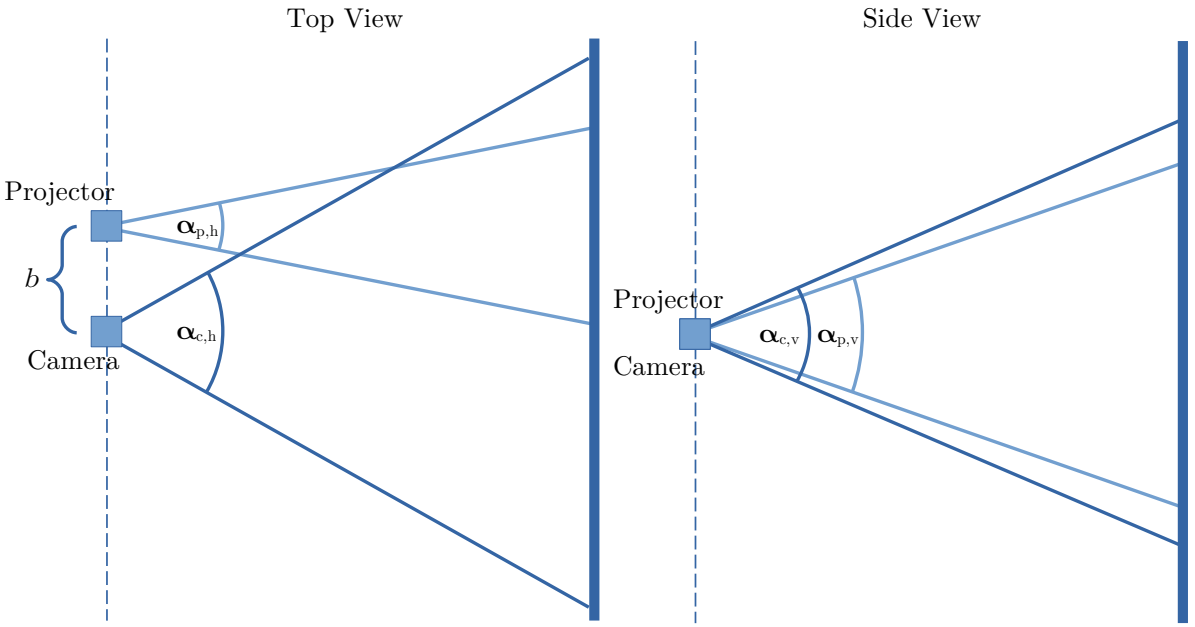


Figure 4.2: Representation of the physical setup and angles of view (AoV). Left: Top view with projector and camera separated by the baseline b on the x-axis. Right: Side view with projector and camera on the same point in the y-axis. The angles for camera and projector are $\alpha_{c,h}$ and $\alpha_{p,h}$ for the horizontal AoV and $\alpha_{c,h}$ and $\alpha_{p,h}$ for the vertical AoV, respectively.

for the correspondence search to be exact only one type of event is considered, in this case the positive polarity event, i.e. the moment the laser point illuminates a point in space. A negative event follows afterwards after the laser left that pixel. As the laser moves faster over the pixel than the reset time of a pixel, the negative event comes after that reset time. If the reset time is set longer, i.e. half of the frame time, the negative event may not be registered at all, since at activation of the pixel the laser moved along further and the brightness without laser point is the current reference for that pixel. Therefore, negative events offer no extra information in this case. To filter out negative events the threshold for negative contrast detection was increased to its maximal value. For this the voltage `bias_diff_off` was set to 0 mV, which means the maximal difference to the fixed 300 mV of `bias_diff` and therefore a very high threshold to trigger a negative event. At this point the event rate can be adjusted by the threshold for positive events, i.e. `bias_diff_on`. This value was chosen to generate the most events while still having low noise in areas where the projector is not projecting. Lastly, the voltage `bias_refr` was chosen so it does not create to much noise from a short reset time, but is fast enough for a pixel to be ready when the next frame is projected. With this the event rate was at about 5-6 million events per second. This matches the amount of pixel triggered in one second which is the width of the projection on the camera's image plane times its height times the frames per second:

$$255 \cdot 400 \cdot 60 \frac{1}{s} = 5400000 \frac{1}{s}$$

This confirms that each camera pixel which is covered by the projected frame is triggered once per frame on average.

4.2 Software

The choice for the software was mainly motivated by the hardware used. Specifically, the event camera prompted the choice for libraries and SDKs. The camera is the *Prophesee* evaluation kit 1 with a Gen3.0 sensor [34], which enabled the use of the *Prophesee Metavision SDK* [38]. This SDK is developed for the use with Prophesee cameras and the major version 2 of the SDK comes in an evaluation as wells as a professional license. The newly released version 3 of the SDK puts the full Python and C++ API under a free license. At the time of its release it was to late to incorporate version 3 into the implementation of this thesis so version 2 is used in the evaluation license.

The calibration application is build in `C++14` on an Ubuntu 18.04 LTS platform. The Metavision SDK is in version 2.1.0 because at the time of implementing the calibration application it was the only version which included the calibration API and open samples in the evaluation license. From this SDK the communication with and control of the camera, processing pipeline, specialized event filtering, and frame generation is used. To detect the chessboard patterns and the actual calculation of the intrinsic and extrinsic parameters as described in 5.1 the `OpenCV` library was used in version 4.5. Additionally, the libraries `Boost` (3.23.1) and `Eigen` (3.3.7) were used.

The structured light application is build in `Python 3.8` on an Ubuntu 20.04 LTS platform. The Metavsion SDK is version 2.3.2 and used for the communication with and control of the camera and a simple processing pipeline. The epipolar search as described in 5.2.3 is implemented with the help of `CUDA` in version 11.4 and a *NVidia* GTX Titan (700 Generation). For stereo vision, cuda programming and further signal processing the packages `OpenCV` (4.6.0), `NumPy` (1.19.5), `Numba` (0.55.2), `CuPy` (10.5), `SciPy` (1.8.1) and `Open3D` (0.15.2) were used.

Chapter 5

Implementation

The implementation of the structured light system requires the calibration of camera and projector and there geometric relationship to each other. A full calibration application is implemented which works with the event stream of the camera. The structured light system is implemented using the principles established in section 3.3. The components necessary for this are the trigger and time map generation, calibration of the projector time map, disparity search and depth calculation. In addition a simple method is explained to switch from the camera's point of view to the projector's. Finally, the implementation of the demonstrations are explained.

5.1 Calibration

In modern applications the intrinsic parameters of a conventional camera can be found with standard software tools like *Kalibr* [35] by viewing a checkerboard from different perspectives and using Zhangs method [29]. Calibrating an event camera like this can be done by converting the event stream of a moving camera filming the checkerboard to images or video with tools like *E2VID* [39] as done in [3].

Calibrating the projector introduces more complexity due to the projector not measuring its surrounding. The projector is modeled as an inverse camera. Here standard structured light tools like [40] are used in combination with a camera. The camera sees a checkerboard, to detect set points in 3D space. The projector projects a structured light pattern, e.g. gray code pattern, onto the checkerboard with which correspondences between the camera points and therefore the 3D points and the projector points can be created. Then known methods for the intrinsic and extrinsic parameters can be used.

In this thesis a more straightforward approach to calibration is used for ease of implementation, in which the camera is calibrated first. Then features projected by the projected are detected by the camera

and backprojected into world coordinates using the camera calibration and known 3D geometry. Lastly, the projector is calibrated using the defined features on the image plane and the projected 3D points.

5.1.1 Camera

Firstly, to find the intrinsic parameters of a camera an algorithm called Zhangs method [29] is used which is implemented in the *OpenCV* function `calibrateCamera` in the module *calib3d*. Two problems come up with this approach when an event-based camera is used. For the checkerboard to be detected by the event camera, the camera either needs to be moving, as it is done in [3], or the checkerboard needs to change in brightness, e.g. blink. Here the checkerboard is displayed on a 60 Hz computer screen and is alternating with an all white image at 30 Hz. The camera biases are chosen to filter higher frequency changes so no unwanted events are created during the screen refresh. The second problem is that the OpenCV algorithm was created for conventional cameras so the event stream needs to be converted to a frame-based representation. With the checkerboard blinking, this is simply done by aggregating all events in the time of one frame at a desired frame rate and creating a binary image with either no event present or at least one event present for each pixel. For this frame conversion an algorithm from the *Prophesee Metavision (2.1.0)* software is used and in this specific case further criteria like ratio of positive and negative events and event clustering are used to filter noise and unwanted events.

The plane of the monitor and therefore the checkerboard is assumed to be the XY-plane with $Z = 0$ and the side length of a square is known by multiplying the width of a square in pixel by the pixel pitch of the monitor. With the first corner point being for example $(\,0\;0\;0\,)^T$ every other corner point in 3D space is set. With the OpenCV function `findChessboardCorners` the correspondences of these points to the points on the image sensor per view are found. The function `calibrateCamera` calculates the intrinsic parameters and the distortion coefficients from these pairs.

It is possible to work out the extrinsic parameters from at least 4 3D-2D correspondences and the intrinsic parameters. Because the blinking checkerboard and corner detection are already set up, these functions provide the detection. The camera is fixed at a point where the full monitor is visible and filling the frame and its optical axis is approximately parallel to the Z-axis. Several sets of checkerboard corners are detected and a per point average over all sets is calculated to mitigate the low resolution of the event camera and the noise in the generated frame. The OpenCV function `solvePnP` is used with the `SOLVE_ITERATIVE` flag, which uses the Levenberg-Marquardt method [41] for optimization.

5.1.2 Projector

Calibrating the projector adds an additional level to the process, as it does not have an image sensor with which the reference object, in this case the checkerboard, can be detected. But the projector can

be modeled as an inverse camera for which the same pinhole model equations (3.1) hold. In contrast to the camera, the image at the image plane is given and a ray outwards from the optical center through a given pixel projects that pixel out. Where these projections are in world coordinates need to be detected to create the necessary correspondences for calibration. For this the previously calibrated event camera can be used.

The projector projects a checkerboard onto the monitor used to calibrate the camera and for which the extrinsic parameters already exist. The checkerboard corners are detected by the camera in much the same way as before. The only difference with this implementation is that the projected checkerboard is not blinking, due to the scanning laser already creating enough contrast change detections for the camera with biases adjusted for this scenario. The projected white squares are recognized by the camera, whereas the black squares output no or not enough light for the camera to register.

The plane of the monitor is set as $Z = 0$ in world coordinates. With this information, we can recalculate the 3D coordinates from the 2D camera detections by effectively applying the pinhole model in reverse including the distortion model. Firstly, the pixel on the camera sensor are normalized to the camera coordinate system by multiplying with K^{-1}.

$$K^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}$$

To undistort the points the OpenCV funtion `undistortPoints` is used which uses an iterative algorithm to approximate the original position before distortion.

To change the coordinate system back to world coordinates it is necessary to include the known position, $Z = 0$ in world coordinates, to account for the missing depth in a 2D camera image. This is done by reducing the 3×4 matrix $[R|t]$ with the known information about Z. With

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} r_{11}X + r_{12}Y + r_{13}Z + t_1 \\ r_{21}X + r_{22}Y + r_{23}Z + t_2 \\ r_{31}X + r_{32}Y + r_{33}Z + t_3 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13}Z + t_1 \\ r_{21} & r_{22} & r_{23}Z + t_2 \\ r_{31} & r_{32} & r_{33}Z + t_3 \end{bmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

the matrix $[R|t]$ can be reduced to 3×3. Using $Z = 0$, further simplifies to

$$\begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = [R|t]_{Z=0} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

The point in world coordinates is now in 3D homogeneous coordinates and still carries the scaling factor λ.

$$[R|t]_{Z=0}{}^{-1} \cdot \texttt{undistortPoints}\,(\, K^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \,) = \lambda^{-1} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{5.1}$$

Dividing by the third component of the result, i.e. λ^{-1}, will give the wanted X and Ycoordinates.

It is now possible to determine the intrinsic parameters, distortion coefficients and extrinsic parameters of the projector by matching each detected point in world coordinates with the corresponding known point on the projector image plane with help of the OpenCV functions `calibrateCamera` and `solvePnP`.

Lastly, while calculating the extrinsic parameters of the projector, the set of point triplets (Projector, World, Camera) in combination with the intrinsic parameters can be used to directly calculate the relative position and rotation from the camera to the projector using the OpenCV function `stereoCalibrate`. This relative position and rotation are used for all further stereo calculations ending the dependence on the reference object, i.e. the computer screen.

5.1.3 Integration

The physical setup for the implementation consists of the camera and the projector detachable on a stereo rig, pointed at a computer monitor. A sheet of white diffusion paper is applied directly onto the monitor screen. This lets the monitor reflect the projection more easily and acts as a canvas. Because the diffusion sits directly on the screen there is no notable effect on the sharpness of the displayed checkerboard. The XY-plane as detected by the camera is the display panel itself, but the XY-plane as projected on by the projector has additionally the glass pane and the diffusion paper. This introduces a small error since the plane as seen by the camera is assumed to be Z = 0. But this error is neglected as it is not directly measurable.

The calibration is implemented as a C++ application and lets the user calibrate all parameters of the camera and the projector. For a full calibration, meaning intrinsic and extrinsic parameters of the camera as well as projector, several steps are necessary.

Firstly, a video file of the blinking checkerboard is displayed on the monitor and the camera is detached from the stereo rig. As the size of the checkerboard is known and the plane is $Z = 0$, the 3D part of the correspondences is constant and known at all times. The camera is moved around the monitor and so it detects the checkerboard corners from varying perspectives and records all detections. Additionally, it is advantageous to cover the whole camera frame with detections. This provides more information for the function to determine the correct distortion coefficients for every part of the frame.

As a side note, the corners of the checkerboard are always recorded row-wise, left-to-right, top-to-bottom, starting with the top left corner. Due to the checkerboard being point symmetric, the detection algorithm in OpenCVs `findChessboardCorners` sometimes orders the corners starting from the bottom right, reversing the order exactly and effectively rotating the camera 180 degrees. As it is not expected to rotate the camera during calibration, a simple rule is implemented that reverses the order of the point set if the first point is lower and further right than the last.

Secondly, the extrinsic parameters of the camera need to be known to be able to reproject any detection of a projected checkerboard back to the world coordinate system. For this the camera is fixed to the rig and the monitor fills the whole camera frame. The camera detects several sets of displayed checkerboard corners from a fixed position. These points are averaged point-wise and the extrinsic parameters of the camera can be calculated.

For the intrinsic parameters of the projector, the monitor is now turned off or just displays a static solid color like black or white. The projector now projects an image of the checkerboard with known positions of the corners in pixel. In contrast to the displayed checkerboard, with the projected checkerboard the 2D part of the correspondences is constant and known at all times. The projector is already rotated 90 degrees counter clockwise as stated in 4.1.1. The checkerboard is projected onto the monitor now functioning as a screen. As before the projector is moved around the monitor to varying perspectives and the camera detects multiple sets of points. After reprojecting the detections to world coordinates with equation 5.1 the intrinsic parameters of the projector can be calculated.

It is noteworthy that the distortion coefficients of the projector may not be reliably calculated for two reasons. First of all, the corners in the image plane used for calibration stay the same on the image plane for every point set and therefore only provide limited information. Moreover, the laser system of the projector does not use a lens as described in 4.1.2 and possible distortions may not be adequately modeled with the equation 3.2 used by the OpenCV algorithm. In this thesis the distortion coefficients of the projector were only applicable for small r and resulted in very distorted results when used in stereo rectification. Thus, for any further use, the distortion coefficients of the projector were assumed to be

ideal, i.e. $k_1 = k_2 = p_1 = p_2 = k_3 = 0$.

Lastly, the projector is also fixed on the stereo rig pointing in the same direction as the camera. Most likely after this the extrinsic parameters of the camera need be calculated again, as it might have moved with respect to the monitor, i.e. the reference frame. Now the same process as described above to get the extrinsic parameters is repeated for the camera with a displayed checkerboard and for the projector with a projected checkerboard.

In the very last step a point set triple is detected: reference corner points on the image plane of the projector, detected corner points on the image plane of the camera and reprojected corner points in world coordinates. This triple in combination with the intrinsic parameters can be used in the OpenCV function `stereoCalibrate` to get the relative position and rotation from the camera to the projector, which are used for every subsequent stereo calculation. Hence, the monitor as a reference object is not needed any more. All resulting parameters are written to a configuration file in yaml-format.

For this setup it is advantageous to place the optical centers of the camera and the projector with as little a difference in the z- and y-direction as possible, i.e. keep the vector from one optical center to the other as parallel as possible to the image plane of both and to the X-axis of both. This minimizes the distortion from projecting and resultant error during the following stereo rectification as the plane in which the rectification projects is in part spanned by this vector.

The application can calculate every combination of intrinsic and extrinsic parameters of either camera and projector if the necessary information is provided from a previous calibration. The most useful example is to only calibrate the extrinsic parameters of both, camera and projector, if their relative position on the stereo rig has changed. Here the intrinsic parameters need to be read in from a configuration file.

5.2 Structured Light System

In this section the components making the structured light system as implemented in this these are presented.

5.2.1 Trigger and Time Map

There are many ways to leverage the unique properties of event cameras and most use various representations an event stream can take. Events can be processed individually as the are produced to get low

latency in applications like a spiking neural network. Going further the event stream can be processed in packets of a spatio-temporal neighborhood of various sizes and may be converted to other representations. An event frame or 2D histogram integrates the number of events or their respective brightness delta per pixel in a 2D grid. Moreover, this can be used to reconstruct classic intensity images from an event stream which retain the advantages of no motion-blur, high dynamic range and high temporal resolution. The events can be considered a 3D point set with $(x, y, t)^T$ or a voxel grid. This is a 3D histogram where each voxel integrates the events per pixel in a time span. A time map or time surfaces takes the last timestamp of all events per pixel in a time span and create a 2D map. Here a motion history is recorded with higher intensity representing more recent activity. [3]

All of the following underlying geometry is based on the theory of stereo vision. Classical stereo vision relies on two raster images taken by two calibrated cameras at the same time. The previously mentioned event representations may possibly be used in classical computer vision algorithms is they are grid based. As explained in 2.3 the correspondence between the projected image and the event stream is found trough their timestamp. The best representation to achieve this is the time surface or time map as it creates a motion history of the scanning laser point over the image plane. An ideal time map for the projector which is scaled to 1 can be created easily by assigning linearly increasing values from 0 to 1 in a raster manner to a frame. This time map can be seen in figure 3.3 with the values 0 to 1 mapped to the colors blue through red and is the same for all projector frames under ideal conditions. Reset time of the laser per line and timing jitter as described in 4.1.2 and figure 4.1b are not considered in this case.

The time maps for the event camera are created out of the event stream. To create a time map that corresponds to a projected frame, the start and end time inside of the event stream need to be found. Most event cameras provide external trigger pins with the ability to attach an electrical trigger signal, e.g. a pulse or square wave for periodic triggers. When triggered the camera creates an additional trigger event with the according internal timestamp which appears in the event stream. This is used in some ESL [3] datasets by outputting a 60Hz square wave audio signal from the projector. The square wave is adjusted with additional circuitry to fit the logic levels of the camera. A trigger event is now created on each start a projector frame. Unfortunately, this method is needs additional hardware and as described on the ESL github page [42] is subject to drift. In ESL another algorithm is implemented as well and which is also used in this thesis.

The laser point creates an event for every pixel it scans during the frame. As the projector takes approximately the same time per pixel, the timestamp of subsequent events increases linearly. However, the projector finished the frame faster than 16ms in about 13ms and uses the remaining 3ms to reset the laser position. In this time the laser beam is off and does not create new events. This gap is visible in the

event timestamps and can be used to detect the frame start and end. Figure 5.1 shows the timestamps of an event stream capturing the laser beam.

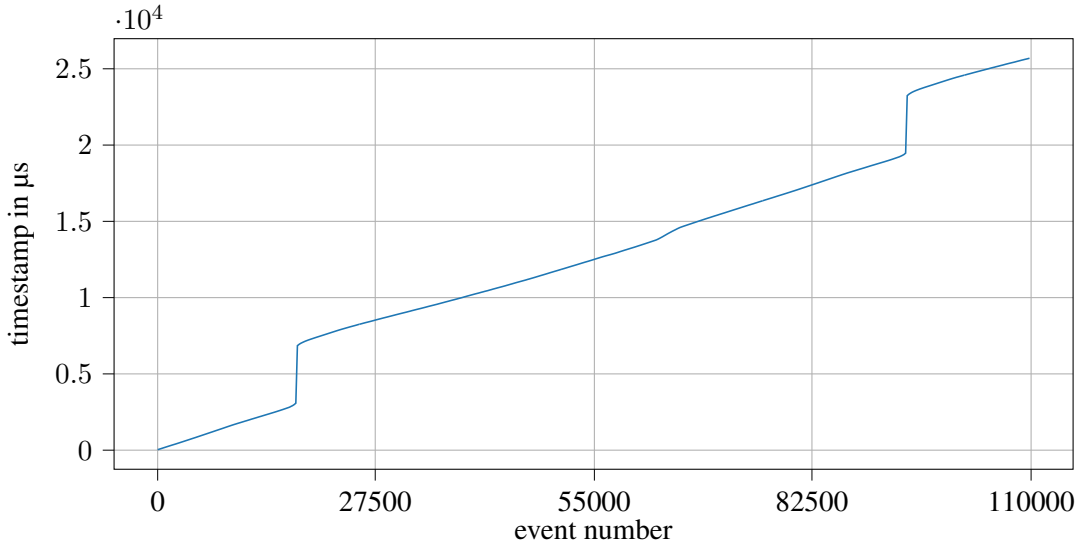


Figure 5.1: Timestamps of event stream. On the y-axis are the timestamps of every event in the event stream. As the events are ordered by timestamp and during the scanning of the frame the events come in at a constant rate, a mostly linear line emerges. While the laser is off during the reset time, very few events occur, which results in bigger time differences in subsequent events. Two of these jumps are visible in this figure.

The event stream comes of the camera and is processed in packets in the form of arrays. Inside these arrays the events are sorted by time, meaning subsequent events have a monotonically increasing timestamp. To detect the gap a packet of events in a time span greater than 16ms is processed to contain at least one complete frame. The time difference between each subsequent event timestamps in this time frame is calculated. The typical time difference between two events during the scan is 0µs or 1µs as the laser point moves over multiple pixel in 1µs. This time difference can be longer if parts of the projected image is blocked for the event camera by the shape of the scene but is generally small.

The indices of all events with a time difference to the previous event greater than a certain threshold are collected. This threshold is chosen so it filters out every event inside the scan as they have a small time difference. But it can not be chosen to high, i.e. at about the 3ms of the reset time, because there may be noise events or events induced by motion during this time. Hence, the smallest possible threshold value that still reliably filters out events created by the scanning laser should be chosen to be robust against noise and motion. Here the threshold was chosen empirically at 40µs or the time to scan two lines. On the one hand, in the test scenes recorded it is was unlikely that two complete vertical lines[1] are

[1]Lines are vertical, since the projector is rotated 90°

blocked to the camera by the shape of the scene. On the other hand, this value is low enough to captured all events with larger time differences in the 3ms reset time, even with moderate noise and motion.

According to this, all collected events are part of the 3ms reset time. Now the subsequent time difference between these events is calculated. If this difference is greater than half of the frame time, these two events are considered the last event in the preceding reset time e_0 and the first event in the succeeding reset time e_1. The exact position inside their respective reset times is not certain. Thus, one event after e_0 in the complete event stream is chosen to be the start timestamp, as this is the first event during the scan. Conversely, the timestamp of the event before e_1 is set to be the end timestamp, as this is the last event during the scan. In listing 5.1 an example implementation in python is given.

```python
import numpy as np

def find_trigger(event_buffer: np.ndarray, threshold_us: int, frame_time_us: int)
    -> list[int]:
  # calculate time difference for subsequent events
  diff_ts = np.diff(event_buffer['t'])
  # filter for time differences over threshold and get their indicies
  diff_idx = np.nonzero(diff_ts >= threshold)[0]

  # initialize empty list
  trigger = []
  # loop over collected indices, i.e. events within the reset time
  for idx, next_idx in zip(diff_idx[:-1], diff_idx[1:]):
    # check if the time difference between two events with these indices
    # is bigger than half the frame time
    diff_time = event_buffer['t'][next_idx] - event_buffer['t'][idx]
    if  diff_time > frame_time_us//2:
      # append the index of the event after idx
      trigger.append(idx+1)
      # append the index of the event before next_idx
      trigger.append(next_idx-1)

  return trigger
```

Listing 5.1: Example of trigger detection function

Next, with these boundaries the time map can be created. Firstly, a 2D array with the camera resolution is initialized with all zeros. All events within the defined time range are iterated upon and their timestamp value t is set as the pixel value at the x and y position of that event and the polarity is positive. Events with negative polarity are mostly filtered out be the bias voltages, but some noise events appear nonetheless. If an event has the same x and y position as a previous event, the pixel value is over-

written with the newer timestamp. Afterwards, the complete 2D map has a range of 13ms and all pixel with a non-zero value are scaled to the interval 0 to 1. How this is done in code is displayed in listing 5.2.

```python
import numpy as np

def gen_time_map(event_buffer: np.ndarray, camera_width: int, camera_height: int)
    -> np.ndarray:
  # initilize 2D map with zeros of camera shape
  time_map: np.ndarray = np.zeros((camera_height, camera_width))
  # loop of events in event buffer, i.e. all events from in one projector frame
  for event in event_buffer_frame:
    #if polarity is positive
    if event[2] == 1:
      # place timestamp at x,y coordinate of the pixel
      time_map[ev[1], event[0]] = event[3]
  # normalize: since event_buffer is in order the first and last element contain
    min and max time
  min_time = event_buffer[0][3]
  max_time = event_buffer[-1][3]
  # subtract min_value from time map
  normalized_time_map = time_map - min_value
  # scale interval [min_value, max_value] to [0,1]
  normalized_time_map /= (max_value - min_value)
  # reset all value lower than 0 back to 0
  normalized_time_map[ret_time_map < 0] = 0

  return normalized_time_map
```

Listing 5.2: Listing: Example of the time map generation function. The argument `event_buffer` is a 1-D array which contains all event from the start to the end of a frame projected by the projector. An element of that array is an `event` which contains in order (x, y, polarity, timestamp). The time map is afterwards scaled to 0 and 1.

All complete frames that are found in the current event array are created. Finally, all events from the start of the array to the end timestamp of the last frame are not relevant anymore and discarded. The remaining events are the start of the next frame but are not complete. They are prepended to the next incoming array of events when that array arrives and is processed.

5.2.2 Projector Time Map Calibration

The generation of the time map of the projector is described in section 5.2.1. However, this method creates a completely ideal time map for the projector, which assumes that the laser moves over every pixel in a constant speed and jumps without extra time from one the end of one line to the start of the

next. It does not consider non linear scan time for the projector as well as effects the readout behavior of the camera adds. As seen in 5.1 the timestamps of the events do not increase linearly as is assumed by the ideal projector time map. Some deviations from the ideal are due to noise and jitter as shown in figure 4.1b and therefore dynamic in nature. It is not possible to filter out these events with a more exact time map, since the time map used for the projector is static and does not change. Other deviations such as the non-linear behavior of the laser scan speed is mostly static over a certain amount of time and can be calibrated for. Moreover, it turns out that the laser beam as captured by the camera does not strictly follow the presumed raster pattern. This makes it necessary to generate a more accurate reference time map for the projector.

If the projector projects a white image onto a non moving plane which is mostly parallel to the image planes of projector and camera, the time maps created from the events show a rectangle as shown in figure 5.2. This corresponds directly to the actual time map of the projector as seen by the camera. This time map can be used as the reference time map of the projector in the disparity search. The exact physical setup of the plane on which is projected is not important as long as it is a plane and the projected image can be detected as a rectangle.

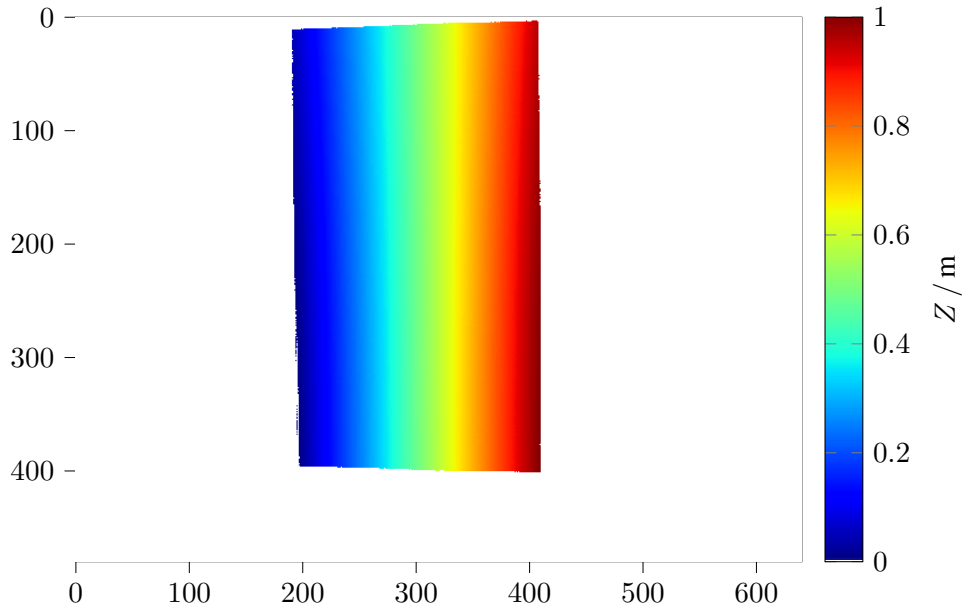


Figure 5.2: Time map of a full white frame projected onto a plane. x- and y-axes state the pixel number of the camera image. The timestamps are scaled to 1 and 0 means no timestamp, i.e. no event, which is colored white.

As the camera has a completely different resolution, additional processing is needed. Firstly, multiple time maps are generated, scaled to 1 and the pixel wise average is computed. This filters out dynamic

effects previously mentioned like noise and jitter. Next, a binary map is created from the average time map with a 1 where there is a non-zero timestamp and 0 otherwise as in figure 5.3. In this binary map the corners of the rectangle in the time map that represent the projected frame are detect. To avoid missing values at the edges of the projected frame, the corners are moved inwards by one pixel. Then the projective transformation from this irregular rectangle to a rectangular 2D map with the projectors width and height of 720×1280 is calculated and subsequently used to transform the time map.

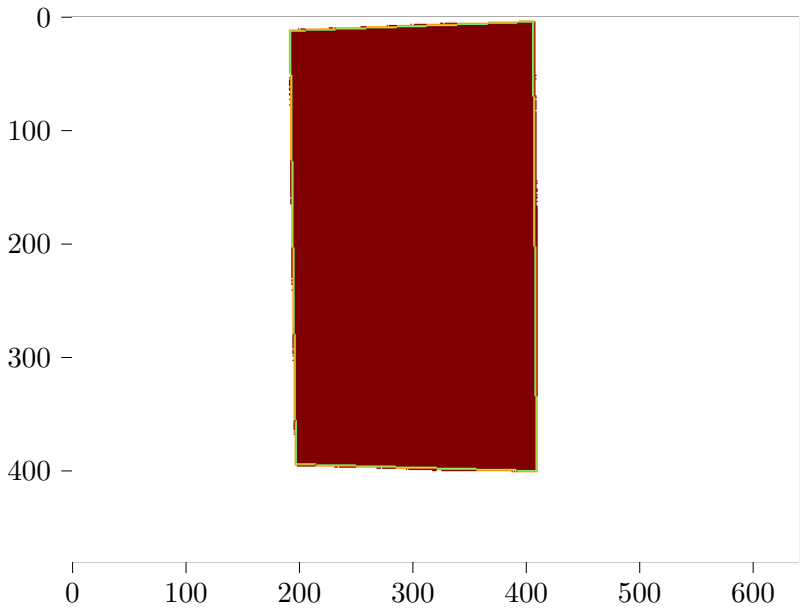


Figure 5.3: Binary map of figure 5.2 used to detect the area onto which the projector is projecting. The green border shows the detected rectangle and the are which is transformed to the projector time map.

In figure 5.4 the calibrated projector time map is flattened to one dimension to show the non linearity in time. This shows more detail than the colormap. It is visible that at the start of a frame, the projector is slower than the ideal curve in the beginning, while accelerating and finishing the frame faster than the ideal. Since the projected frame fills an area of only 225×400 pixel on the camera image and is not aligned with the camera lines and columns, most pixel in the resulting time map are interpolated and warped. To make it possible to differentiate between the projector lines, the missing values in between two actual lines are interpolated linearly as opposed to have the same values as their nearest neighbor. The warping uses the OpenCV function `warpPerspective` with the `INTER_LINEAR` flag to interpolate linearly. As a result of this, the time-stamps are not monotonically increasing pixel by pixel, line by line like in the ideal time map or as it would be in a perfectly accurate time map. This is displayed in figure 5.5 on the left. Additionally, it is visible that the time it takes to scan one projector line is decreasing up to the middle of the frame, where it starts to increase again. More interestingly, the timestamps

decrease during one line after the middle of the frame. This is only possible if the laser reverses the direction. The actual inner workings of the projector are not clear, but unexpected effects like this can be mitigated by the calibrated time map.

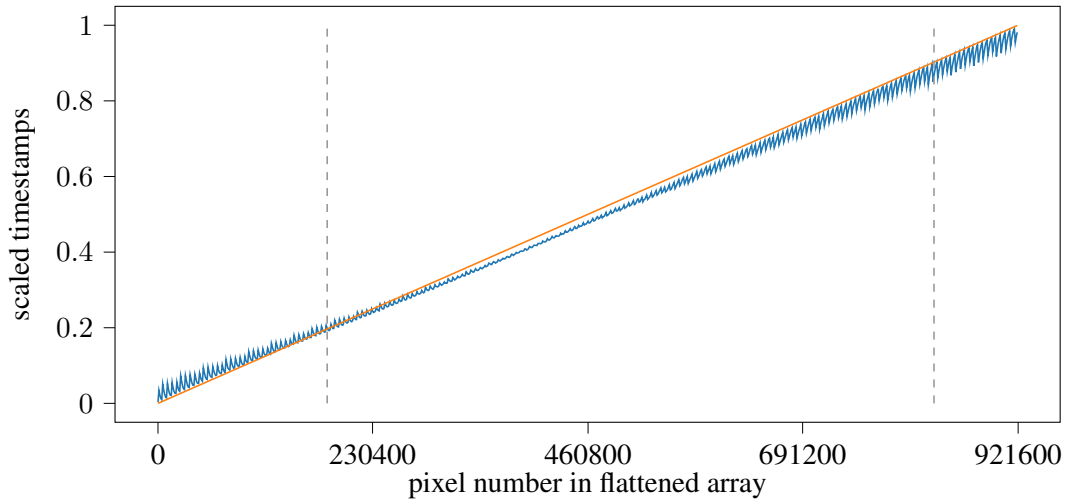


Figure 5.4: Timestamps of calibrated projector time map in blue. The 2D map is flattened to one dimension starting in the lower left going up, then left to right, since the projector is rotated 90° counterclockwise. The orange line marks the ideal linear timestamps. (Note: The spikes visible in the blue curve are not the actual individual lines, but the lines overlayed with aliasing due to the subsampling for plotting.) The gray marker denote where the detailed views in figure 5.5 are located.

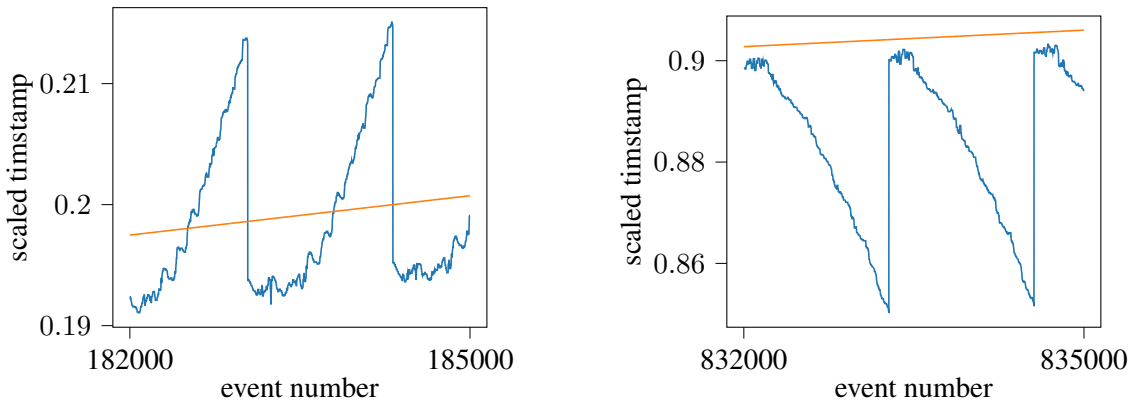


Figure 5.5: Detailed view of figure 5.4. The left plot shows lines at the beginning of the frame. The timestamps increase per line as is expected. The jump down after one line is the result of multiple camera lines being triggered at once as well as interpolating during the warping. The right plot shows lines in the latter half of the frame. The timestamps per line are decreasing indicating that the laser beam is going in the reverse direction.

5.2.3 Disparity and Depth

To apply this idea with existing images, these images need to be rectified or transformed according to the rotation and old and new projection matrices. The transformation to do so is defined by

$$T_i = K_{\text{rect}} \cdot R_{\text{rect}.i} \cdot {K_i}^{-1} \quad \text{with } i = 1, 2$$

The rectifying rotations and new projection matrices for both cameras are found with the OpenCV function `stereoRectify` in the calib3d module using the described principles. To easily apply the transformation to existing images the OpenCV function `remap` is used. This function utilizes two 2D maps in the shape of the new image, one for the x- and one for the y-coordinate. At a specific pixel the maps contain the position of that pixel in the original image to look up.

$$\texttt{dst}(x, y) = \texttt{src}(map_x(x, y), map_y(x, y))$$

The maps are calculated beforehand according to $P_{\texttt{rect},i}$ and $R_{\texttt{rect},i}$ and can take more complex and non-linear projections like the distortion coefficients into account. The necessary maps are found with the function `initUndistortRectifyMap`.

In this thesis the projector is modeled as an inverse camera. The complete theory of epipolar geometry applies just as well. Here the camera is considered the first camera and the projector is considered the second. The projector time map is rectified once, since it stays the same for all disparity searches. The camera time maps are each rectified after generation and before the epipolar search.

With the time maps from the camera and projector generated and rectified, the disparity search is the next step. As explained in 3.2.2 the epipolar lines are horizontal and corresponding lines have the same y-coordinate due to the rectification. So the corresponding pixel in both images or in this case time maps can be found along the horizontal lines with the same y-coordinate.

The basic algorithm for the disparity search works as follows: considering a pixel in the camera time map with a non zero value. For each such pixel the pixel with the closest value, i.e. timestamp, in the horizontal line with the same y-coordinate in the projector time map is searched. These two pixel are considered to be correspondences and their difference in the x-coordinate is the disparity. This disparity value is placed in a new 2D map called disparity map at the x- and y-coordinate of the original camera pixel.

Going step by step, the search starts with the definition of the regions of interest. The goal is to limit

the search only to an area where there are relevant pixel in the respective time maps. The left most, top most, bottom most, and right most pixel with a non-zero value is searched per time map. For the projector time map the values are defined once at generation, since this time map stays the same. For the camera time maps these values are defined for each time map at the beginning of the search. Then a single rectangular region of interest is spanned to contain all pixel with non-zero values to include the left, top, bottom, and right most pixel of both maps.

To accelerate the process, the search is done on a GPU using the `cuda` API from *NVidia*. A disparity map with the same size as the rectified camera/projector time maps is initialized with all zeros. The previously defined region of interest of the camera time map, the projector time map and the empty disparity map are send to the GPU. The search in now done in parallel for each camera pixel that has a non-zero value. During the search for the correspondence to camera pixel (x_c, y_c) with timestamp t_c, all pixels in the corresponding projector time map line at $y_p = y_c$ are iterated upon. The pixel (x_p, y_p) which minimizes the difference of the timestamps

$$\Delta t = |t_p(x_p, y_p) - t_c(x_c, y_c)|$$

is chosen as the corresponding pixel. The disparity d is the difference on the x-axis

$$d = x_p - x_c. \tag{5.2}$$

This value is placed in the disparity map at the same location for the associated camera pixel

$$x_d = x_c \quad \text{and} \quad y_d = y_c \tag{5.3}$$

The GPU only reads from the camera and projector time map in its memory. For each search the GPU writes to a distinct and unique pixel in the disparity map. In addition, as the search is only done for pixels in the camera time map with non-zero values, the disparity map stays zero at pixels where the camera time map has no value as well. These are the areas where the projected light throws shadows due to the shape of the 3D scene, i.e. at depth discontinuities as seen by the projector. To fill these areas, more advanced structured light systems employ more than one projector or camera or both. This is not the case in this thesis and these areas are left with no information. The function which run on the GPU is presented in listing 5.3.

```
1 import numba
2
3 def disp_loop_cam_to_proj(disparity_map, cam_img_rectified, proj_img_rectified):
4   # define a maximum disparity to search in
5   min_disp_search, max_disp_search = 5, 900
```

```
# get the absulte position of the current gpu thread
cam_x, cam_y = numba.cuda.grid(2)
# return if position is outside of current img region of interest
if cam_x > cam_img_rectified.shape[1]-1 or cam_y > cam_img_rectified.shape[0]-1:
return
# initialize minimal values to track for search
min_cost = 1e8
disp = 0
# defince search area and check not to go over image border
proj_x_start = min(x + min_disp_search, cam_img_rectified.shape[1] - 1)
proj_x_end = min(x + max_disp_search, cam_img_rectified.shape[1] - 1)
#check if current pixel has a value
if cam_img_rectified[cam_y, cam_x] > 0:
  # loop over all projector_pixel within search range
  for proj_x in range(proj_x_start, proj_x_end + 1):
    # check if projector pixel has a value
    if proj_img_rectified[cam_y, proj_x] > 0:
      # calculate cost/difference
      cost = abs(cam_img_rectified[cam_y, cam_x] -proj_img_rectified[cam_y,
  proj_x])
      # if cost is smaller than last tracked pixel
      if cost < min_cost:
        min_cost = cost
        disp = proj_x - cam_x

disparity_map[cam_y, cam_x] = disp
```

Listing 5.3: Example of the disparity search function. This function is executed on the GPU for multiple pixel (`cam_x`, `cam_y`) in parallel.

To get the depth values the equation 3.3 solved for Z is used and applied to the disparity value for every non-zero pixel in the disparity map. This depth map is now in to the coordinate system of the rectified camera. In combination with the rectified projection matrix of the camera $P_{\text{rect},1}$ the depth values can also be reprojected to 3D space in this coordinate system giving a point cloud. This can be done with the OpenCV function `reporjectImageTo3D`. This function uses the matrix Q which is also obtained by the function `stereoRectifiy` and which has the form

$$Q = \begin{bmatrix} 1 & 0 & 0 & -x_{0,\text{rect}} \\ 0 & 1 & 0 & -y_{0,\text{rect}} \\ 0 & 0 & 0 & f_{\text{rect}} \\ 0 & 0 & 1/|b| & 0 \end{bmatrix}$$

The function multiplies each pixel with Q to get the point in 3D space

$$\begin{bmatrix} 1 & 0 & 0 & -x_{0,\text{rect}} \\ 1 & 0 & 0 & -y_{0,\text{rect}} \\ 0 & 0 & 0 & f_{\text{rect}} \\ 0 & 0 & 1/|b| & 0 \end{bmatrix} \cdot \begin{bmatrix} x'' \\ y'' \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} x'' - x_{0,\text{rect}} \\ y'' - y_{0,\text{rect}} \\ f_{\text{rect}} \\ d/|b| \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix}$$

and using equation 3.3

$$\begin{bmatrix} x'/w' \\ y'/w' \\ z'/w' \end{bmatrix} = \begin{bmatrix} (x''-x_{0,\text{rect}})\cdot|b|/d \\ (y''-y_{0,\text{rect}})\cdot|b|/d \\ f_{\text{rect}}\cdot|b|/d \end{bmatrix} = \begin{bmatrix} (x''-x_{0,\text{rect}})\cdot Z/f \\ (y''-y_{0,\text{rect}})\cdot Z/f \\ f_{\text{rect}}\cdot|b|/d \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

An example is shown in listing 5.4.

```python
import numpy as np
import cv2

def point_cloud_from disp(disp_map: np.ndarray, Q: np.ndarray, R_inv: np.ndarray)
    -> np.ndarray:
  # reporject the disparity map to 2D map where each pixel contains corresponding
    3D point
  point_set_rect: np.ndarray = cv2.reprojectImageTo3D(disp_map, Q)
  # reshape 2D map to sequential point cloud and add a dimension for matrix
    multiplication
  point_set_rect = point_set_rect.reshape((point_set_rect.shape[0]*point_set_rect.
    shape[1],3))[...,np.newaxis]
  # filter any points out, which had disparity zero and therefore depth infinity
  point_set_rect = point_set_rect[point_set_rect[:,2,0] != np.inf]
  # multiply every point with the inverse of the rectification rotation
  point_set = np.matmul(R_inv, point_set_rect)
  return point_set
```

Listing 5.4: Example of a function to get the point cloud from the disparity map.

The goal is to calculated the 3D information of the scene viewed from the unrectified camera. The unrectified and rectified camera have the same optical center and differ only in rotation $R_{\text{rect},1}$. The inverse of $R_{\text{rect},1}$ can be used to transform the point cloud back to the unrectified coordinate system. Additionally, if the goal is to determine a depth map these points can be projected back into the image plane of the camera using K_1.

5.2.4 Projectors Point of View

In a spatial augmented reality application often a projector projects content onto a scene. The goal in this thesis is to simultaneously measure the depth map with the help of an event camera which in combination with a laser video projector yield a structured light system. The depth information can then directly be used in the content. This gives the augmented reality application the ability to directly and accurately react to changes in the scene.

The 3D information is most useful for the generation of the projected content if it is represented in the projector's coordinate system. In the algorithm for the epipolar search explained in the previous section, the correspondences to a camera pixel are searched in the projector image or in this case the projector time map. As a result the disparity map and the depth map are from the cameras point of view.

The obvious approach to get to the projectors point of view would be to switch the roles during the search. Specifically, beginning with a pixel in the projector time map, searching the correspondence in the camera time map. When directly implement, this approach does not result in a correct disparity map.

As in classical stereo vision, a pixel in one image plane does not necessarily has to have a correspondence in the other image plane. This is the case, if a point in 3D space is visible by one camera and due to the shape of the scene not visible by the second. In a structured light system, this is the case if a ray projected by the projector hits a 3D point in the scene but its reflection is blocked to the camera by another part of the scene. In contrast to the missing pixels in the camera image, this is the case at depth discontinuities as seen by the camera.

Rays outgoing from the projector and reflected into the camera are the only correspondence which can occur. Since the camera is an event camera, events are only created when such a projected ray hits a new pixel. When motion and noise are neglected no other events are created. This means that every pixel which has a value in the camera time map must have a correspondence. The search for a correspondence as implemented in section 5.2 accounts for this since it is only done if the camera time map pixel has a value and if so, a correspondence is always assumed with the projector pixel with the lowest time difference.

In contrast, the projector time map contains every ray that the projector has emitted, which includes the rays blocked to the camera. If the search is done in the projector time map the algorithm assumes a correspondence for every pixel as every pixel has a non-zero value. For pixel where there is no actual correspondence, still the camera pixel with the closest time difference in timestamps is chosen as a correspondence which is wrong.

The assumptions made above allow for a simple and efficient algorithm for the epipolar search so it is beneficial to use it. Since all 3D relationships between camera and projector are known due to calibration a simple way is to transform the cameras coordinate system to the projectors coordinate system after the point cloud has been calculated. This involves a rotation on the point cloud which is a matrix multiplication on every pixel. But this can be done at an earlier stage inside the image plane without additional computation. After a correspondence is found the disparity value is placed at the camera pixel to which the correspondence was found as seen in equation 5.3. The disparity value can instead be placed at the position of the projector pixel (x_p, y_p) which was found during a search. This differs form the previous position only by the disparity.

$$x_d = x_p = x_c - d \quad \text{and} \quad y_d = y_p = y_c \tag{5.4}$$

This is possible since the image planes of camera and projector are rectified. They have the same direction for the z-axis and their translation is perpendicular to the z-axis. A point in 3D space has the same distance Z in the z-axis direction and therefore the same disparity d from the cameras as well as the projectors optical center.

As a remark, the resolution of the projector is notably higher than the time map generated by the camera. As a result multiple pixel of the projector have a correspondence to one camera pixel. The rectified images, in which the search is done, are scaled up and have a higher resolution than both. Due to the upsampling a group of pixels in the rectifed camera time map have the same value. Since the search matches the lowest difference in time stamps, all of these pixels get matched to a single projector pixel and all get placed there. The disparity value at that pixel gets overwritten until the last search matching that pixel is done. This leaves the other projector pixel with possible correspondences untouched and without value. As a result the disparity map for the projector is sparse when compared to the dense disparity map from the cameras point of view. But most pixel from the cameras disparity map are redundant as the underlying time map is upsampled and no information is lost.

5.3 Demonstrations

In this section two simple examples of augmented reality applications are implemented and explained, to apply the depth information simultaneously gather by the structured light system. The first example is a color map applied to the live depth map and projected onto the scene. The other example is a projected image which is automatically transformed to appear without distortion if the screen onto which it is projected is angled.

In these examples, the system projects content other than a white full frame onto the scene. However, for the structured light system to work it is necessary for the event camera to be able to reliably detect the projection. This comes down to some considerations for the content projected. The main point concerns the brightness of the projected content. If an area in the projected content is to dark to reflect enough light to trigger any events, there is no depth information recovered in that area. This also depends on the distance of the projector to the scene. The intensity of the projection gets dimmer as the light gets spread to a bigger area. Darker scene points with a lower bidirectional reflectance towards the camera may also drop of the depth estimation. So for reliable depth estimation to work the projected content should be designed to be as bright as possible.

5.3.1 Depth-based Color Map

This example is a very straight forward application of projecting a colormap of the depth of the scene as seen by the projector directly onto the scene itself. An example for this augmented reality technique is the SARndbox [28] mentioned in section 2.4. The SARndbox uses a Microsoft Kinect for the structured light hardware which works with invisible IR light and a separate projector for the augmented reality content. Its colormap visualizes topographical features and the depth map is also used for gesture detection enabling rich interactivity such as fluid simulation. The example in this thesis works simpler by just projection a live colormap to apply the working principle.

The base for the projected colormap is the depth map as seen by the non-rectified projector. To get this map several steps are necessary starting with the disparity map from the projectors point of view. First, the depth values per pixel are calculated with equation 3.3. Next, according to its Z value, each pixel is reprojected to 3D space in the rectified coordinate system. This point cloud is then rotated into the original projector coordinate system with ${R_{\text{rect},2}}^{-1}$. The final 2D map is a projection of these 3D points into the projector image plane through K_2. Since this implementation is done in python using existing libraries like NumPy and OpenCV the general performance is slower than a purpose build application in a lower level language like C++. Furthermore, except for the epipolar search the calculations are done on the CPU and not in parallel due to pythons global-interpreter-lock.

To enable performance closer to real-time a simpler approach for calculating the depth map is taken. OpenCV offers a very performant way to transform one image into the projection on another image plan with the `remap` function which is also used for the rectification as mentioned in section 5.2.3. Since the maps are calculated beforehand the actual transform comes down to a look up and is very fast.

However, very important to note here is, that this function just projects the pixel value, in this case the disparity or depth in the rectified coordinated system, from its position on image plane into the un-

rectified image plane. This does not take into account how the depth values may change as the camera is rotated. The depth here is defined as the distance from the XZ-plane and thus a rotation around the X- or Y-axis does change the depth. This is not represented in this transformation. Nevertheless, a rotation around the Z-axis does not change the depth.

To mitigate the warping done by the rectification and in this case specifically the rotation around the X- and Y-axis the camera-projector-pair is setup like a rectified pair in the first place. This can be achieved by monitoring the relative rotation and translation during calibration and adjusting the orientation and position accordingly. The relative rotation should be as close to zero as possible, especially in the X- and Y-direction. The translation should only contain an X-component. Being a colormap this type of augmented reality application is forgiving for smaller depth errors which can arise through this simplification.

To conclude, the process to generate the colormap is a follows. The disparity map for the projector is calculated a described in section 5.2.3 and 5.2.4. This disparity map is then transformed to the unrectified projectors image plane introducing some inaccuracies which are mitigated with careful setup and are negligible in this type of application. Lastly, the depth is calculated through the disparity-depth relationship (eq: 3.3). The depth is calculated in the unrectified image, since this image has a lower resolution, which means less computation. As described in section 5.2.4 the disparity image of the projector can be considered sparse and this propagates through to the depth map calculated this way. To project a dense colormap each pixel is dilated with a 7 by 7 kernel. This increases the size of pixel with depth information and effectively decreases the apparent resolution of the projected depth map to match its actual resolution as seen by the camera.

Considering the content projected, the colormap itself contains a range of colors with higher brightness to provide enough light for the structured light system. Further, there may be areas were there is not depth information recovered. Either not enough light was reflected towards the camera or the ray was block entirely by the shape of the scene. In this example these points are colored white to provide the highest chance to be detected in the next frame.

5.3.2 Warping Correction

While the previous example displays the depth information, this example processes this information to alter the content displayed. A projected image gets warped, if the plane which is projected on is angled. For example, if the plane is tilted upwards, the upper part of the image get stretch out and is bigger than the lower part. If there is information about the 3D scene, the original image can be warped to adjust for

this. Additionally, the size of the projected image can be scaled to always appear the same independent on the distance to the projector.

In this example a circle with its center point lying on the optical axis of the projector is projected onto a plane. If the plane is angled in any way, the circle gets warped and if the plane is farther away, the circle gets bigger. In the point cloud of the scene in the unrectified projectors coordinate system this plane can be detected. The Open3D python library provides the function `segmentPlane`. This function takes the point cloud and tries to fit a plane using the ransac algorithm.

In the simple setup of this example the plane is the only object in the point cloud and easily detected. With the description of the plane in 3D space the intersection of the optical axis and the plane is calculated. At this intersection in 3D space a virtual circle of set size is placed and key points on its circumference, i.e. four points which are 90° apart, are calculated. These points are then projected into the projectors image plane. This provides the shape of the circle on the image plane, which, once projected, is a non-warped circle with set size on the receiving plane. The image of the circle can now be warped accordingly.

Chapter 6

Results

The goal of this thesis is to implement a structured light system that can use a laser video projector for depth reconstruction using an event-based camera while projecting arbitrary content. Several datasets have been recorded for evaluation. The aspects which are evaluated here start with the influence of the projector time map and with the general ability to calculate the depth correctly. Further measurements look at the robustness to high ambient illumination and low content brightness. Measurements with specular materials and materials with high inter-reflections are inspired by MC3D [1]. The results point to a noise prevalent in the disparity map and a systematic error to actual depth, which are evaluated as well. Lastly, examples of the implemented demonstrations are shown.

The measurements show mainly the validity of the calculated depth and the underlying implementation as well as prove the ability to project different images at the same time. In favor of real-time processing straightforward and existing algorithms for calculating the depth from the disparity are used. Therefore, no direct comparison of the performance to existing structured light implementations, their processing algorithms, and datasets is done.

6.1 Measurement Methodology

To record measurements for evaluation, the structured light system is setup up as described in chapter 4. The projector and camera are attached to a stereo rig with a baseline of about 5.2 cm. If not specifically mentioned the ambient light is at a typical indoor level and the lens iris, as well as the camera's bias values, are set accordingly. All measurements have been taken in the same place, overlooking the length of a 2 m wide table and onto a white wall. Various scenes were set on the table. For all evaluations concerning the accuracy and robustness of the depth information the point cloud is taken from the camera's perspective as the resulting depth map has already the correct resolution no dilation or other processing

is needed. The disparity maps, point clouds, and resulting depth maps are calculated according to section 5.2. If not otherwise mentioned only a single scan is used, without averaging over multiple scans as this is the intended operating mode in real-time augmented reality applications.

6.2 Projector Time Map

As mentioned in section 5.2.2 the laser beam of the projector does not move with a constant speed over all pixel and does not create a completely linearly increasing time map. To visualize, a plane is scanned so all points should lie on a simple plane. In figure 6.1 the depth map of this plane is visible. On the left is the depth map created with an ideal linear projector time map, whereas on the right is the same scan with a previously calibrated projector time map. For this evaluation multiple scans are averaged upon, since the calibrated projector time map accounts only for static error in the linear projector time map.

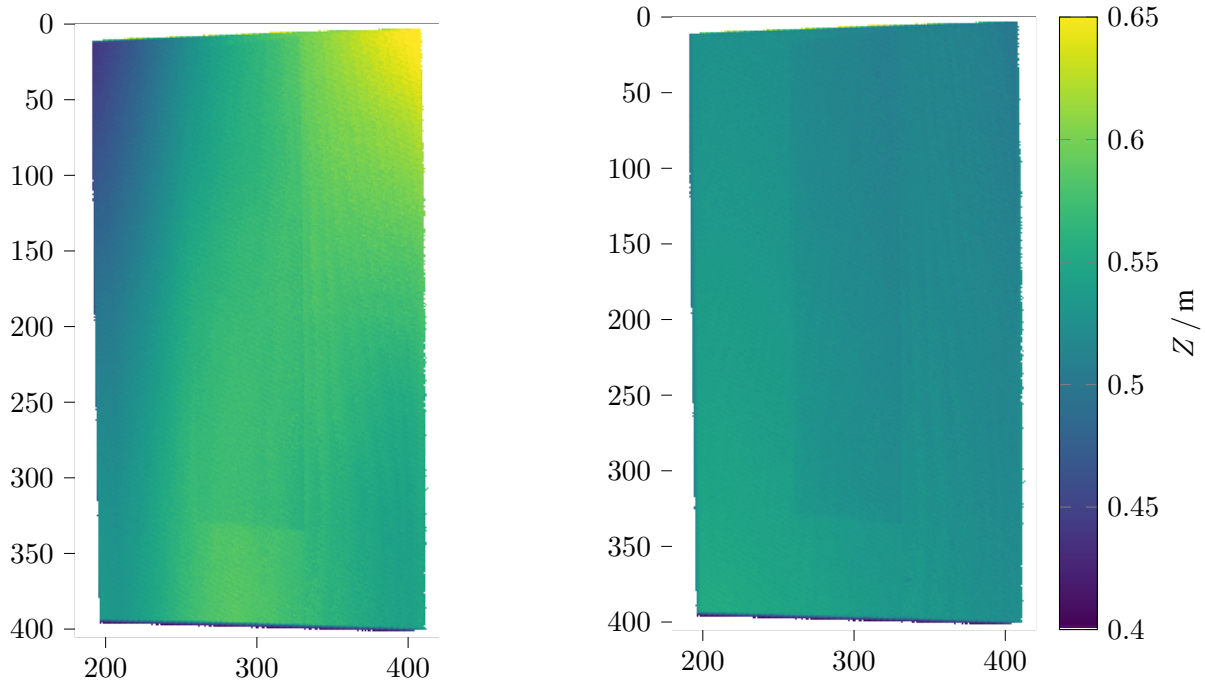


Figure 6.1: Time map of a plane with and without calibrated projector time map. Figure (a) is the depth map without a calibrated projector time map. Strong deviation of the simple plane shape are visible in the upper corners. The depth map in figure (b) is calculated from the same camera time map but using a previously calibrated projector time map. The scale of the colormap is depth in meter.

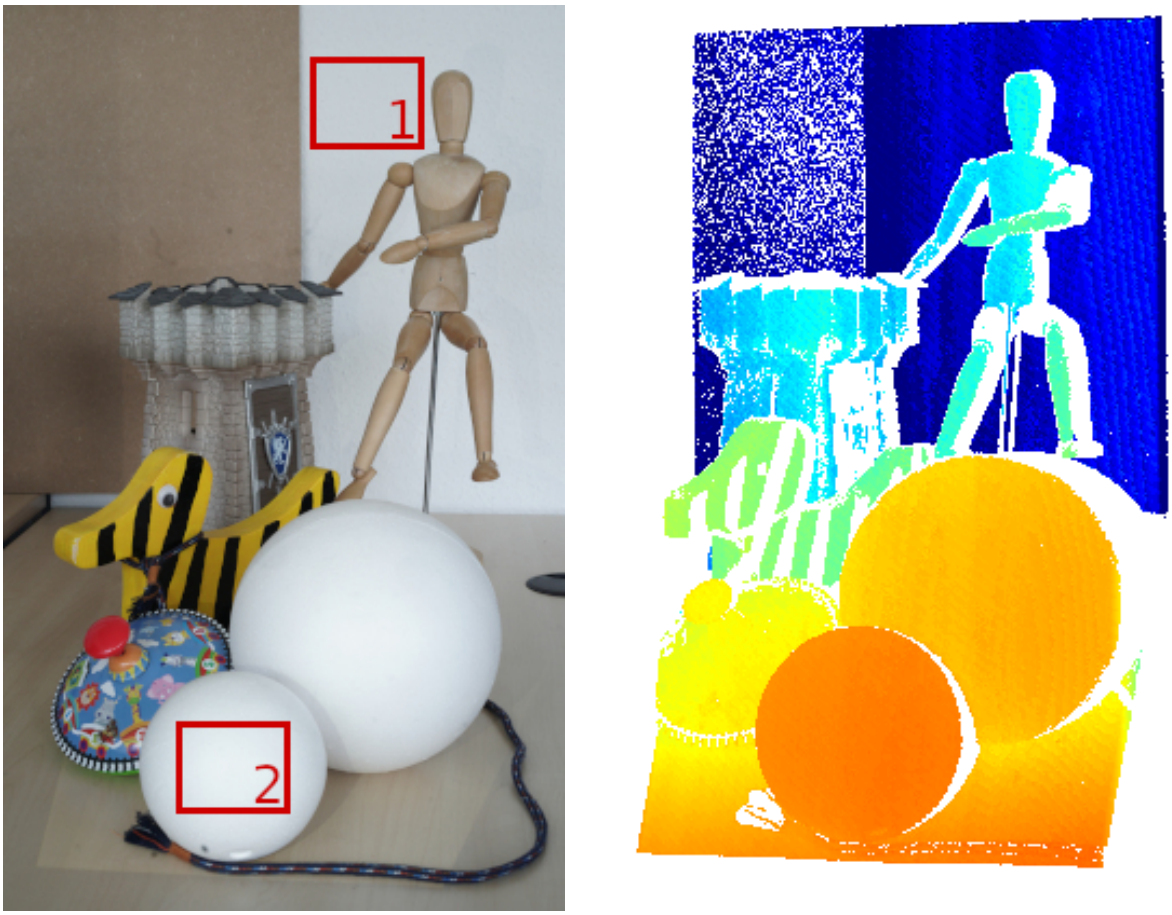
With the attached color bar in mind the depth map in figure 6.1a has strong bends in the upper left and right corners. The upper left corner comes closer and the upper right corner bends away. This

matches the observations in figure 5.4 and 5.5. The projector rotated 90° counterclockwise so the lines are scanned from the bottom to the top. In figure 5.5a is the actual timing of the laser beam at the start of the frame. It is possible to see that, during the end of a line, the difference to the ideal is the biggest. This implies that the biggest error is at the upper edge on the left of the depth map. The same error is at the end of the frame, which is in figure 5.5b. The difference to the ideal time is at the end of a line, but with the opposite sign as at the start of a frame. This is also visible in the depth map, where the biggest error is in the upper right corner and in the other direction. In figure 6.1b the plane in the same setup is scanned. But this time a previously calibrated projector time map is used for the disparity search and these error are not visible.

6.3 Ambient Illumination and Content Level

While projecting arbitrary content, it is important to consider the ability of the event camera to recognize the laser beam. If the pixel projected is too dark, the change in brightness on the corresponding scene point might not be big enough to trigger an event. Different colors are not considered here, as the laser beams of the single colors can be aligned closely to one another manually in the projector. No significant deviation is visible in the projected image and not expected in the depth map. In addition to the content level, the ambient light factors in as well. At a given pixel the threshold to trigger an event scales up with high static illumination at that pixel. Thus, with high ambient illumination a brighter laser is needed to trigger an event. To get an understanding for this an example scene is scanned with varying lightning conditions as well as with different levels of brightness for the projected signal.

Figure 6.2 shows the example scene. The varying light conditions are created by controlling the amount of diffuse sunlight on the scene in addition to artificial light, which hits mostly the white spheres in the front of the scene. The intensity is measured in two areas in the scene with an *URPtek Premium MK350S* light meter. Area 1 on the back wall accounts for the general ambient light, while area 2 is the brightest area in the scene. Eight brightness levels are created. From the brightest to darkest, first the artificial light is dimmed and eventually turned of, than the amount of sunlight is reduced. The brightness levels were recorded while the projector was off. The reference photos are taken with the same exposure. The event cameras exposure and bias value as also not changed.



(a) (b)

Figure 6.2: Scene for the ambient light/content level measurement. Figure (a) shows the example scene and figure (b) its corresponding depth map. The red rectangles denote the areas were the ambient light levels are measured.

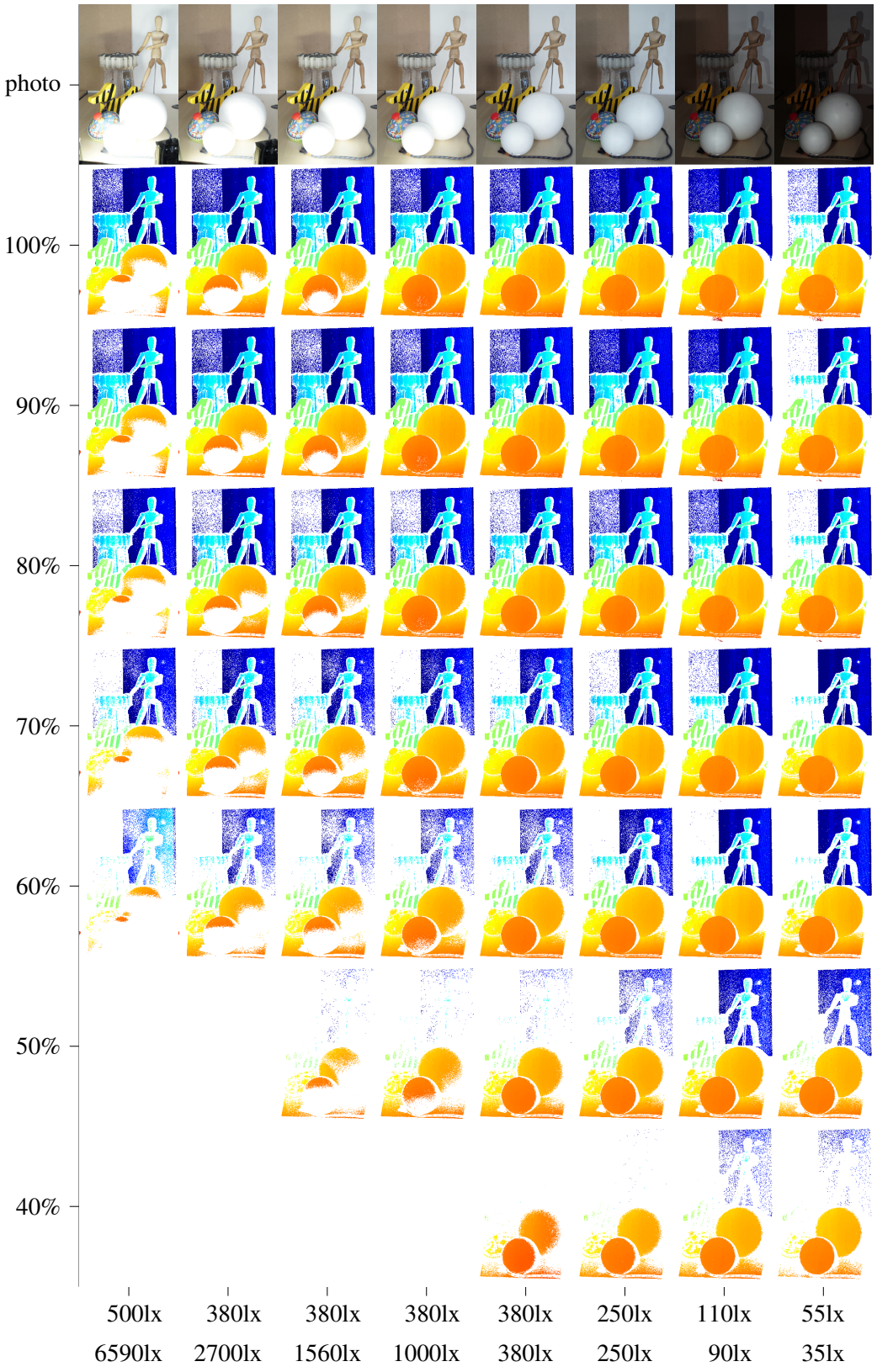


Figure 6.3: Depth maps with various ambient illumination levels and content levels.

The projected signal was controlled by reducing the pixel values of a white image. For 100% all pixel values are 255. Then the pixel values a scaled down linearly in 10% steps to created the darker images.

Every combination of ambient light and content level is displayed in figure 6.3. Below an example photo, the content levels are on the y-axis in decreasing order. On the x-axis are the ambient light levels in decreasing order. The first thing to see is that depth estimation stays consistent over all light levels, as long a reliable trigger can be found. This is expected as high ambient light only masks projected light to trigger an event, not its position or timing. The main aspect which is effected here is the fill rate, i.e. the ratio of how many pixels triggered an event to all pixel. Table 6.1 lists the corresponding fill rates which calculates the ratio of pixels with depth value and to all pixel visible in the presented images.

Ambient	Area 1	500 lx	380 lx	380 lx	380 lx	380 lx	250 lx	110 lx	55 lx
light	Area 2	6590 lx	2700 lx	1560 lx	1000 lx	380 lx	250 lx	90 lx	35 lx
Content	100%	0.444	0.547	0.569	0.619	0.633	0.658	0.658	0.567
level	90%	0.426	0.534	0.554	0.606	0.613	0.65	0.654	0.528
	80%	0.349	0.470	0.486	0.551	0.56	0.599	0.606	0.509
	70%	0.244	0.376	0.386	0.468	0.488	0.537	0.547	0.493
	60%	0.107	0.223	0.228	0.331	0.375	0.46	0.502	0.459
	50%			0.085	0.17	0.234	0.306	0.416	0.347
	40%				0.069	0.153	0.19	0.248	0.202

Table 6.1: Fill rate for various lightning conditions and content level. Here the fill rate is the ratio of pixel with depth value to all pixel.

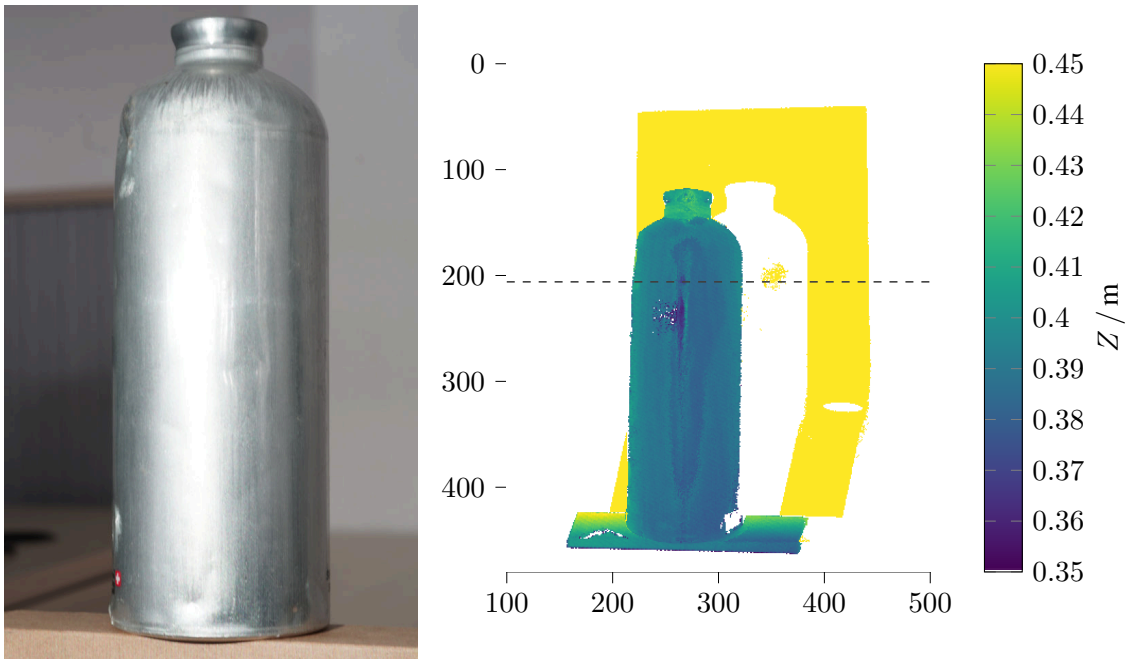
First, the fill rate for a wide range high ambient light, i.e. second row in figure 6.3, shows similar results as MC3D [1]. The ball in the foreground is correctly detected with ambient light as low as 35 lx up to 1000 lx accounting for the high dynamic range of the camera. Secondly, the content level has the expected relation to the fill rate. As the brightness of the projected content goes down, i.e. downwards in the same column, the amount of pixel with a depth value decrease down to a cut off where not enough events where generated for the trigger algorithm to work reliably. The downward trend is worsened by high ambient illumination, hence in high ambient illumination a higher content level is needed. To conclude, the brighter content must be chosen to ensure reliable depth reconstruction and this relationship must be kept in mind while designing an augmented reality with this setup.

In addition, another factor which impacts the fill rate is the distance. The light of the projector spreads over a bigger area and get dimmer. The white back wall drops far earlier from the depth map as the

white spheres in the front. Also the material color and reflectance has an effect. The white spheres are the most consistently detected shapes, whereas the black stripes on the duck never trigger events. The combination of distance and reflectance is visible in darker half of the back wall. Lastly, the fill rate per content level gets better with lower ambient light, except for the last step. As the ambient light almost completely fades the fill rate worsens again. This is most likely a limit of the camera in low light, where it might be hard to establish a reference intensity value to measure the change against.

6.4 Specular Material & Inter-Reflections

In the publication MC3D [1] the advantages of event-based cameras are assessed with different materials which prove challenging for structured light systems. Similar measurements are done here to validate the system.



(a) (b)

Figure 6.4: Photo and depth map of object with highly specular material. The dashed line marks the Y-coordinate of the XZ-plane intersection in figure 6.5.

For the specular material a cylindrical water bottle made of aluminum is used as displayed in figure 6.4. The depth map on the right shows that these materials are still problematic. In the center where the direct reflection of the light source of the projector is visible by the camera erroneous depth values appear. For a closer look, figure 6.5 show the XZ-plane cross section with the object taken from the

point cloud of one scan. The blue dots are the 3D points which lie in the plane. The horizontal lines which emerge in this representation are the result from the depth quantization. The quantization appears due to the discrete nature of the disparity map and the linear relationship from disparity to depth. For easier visualization a sliding window average is used on the depth values, i.e. Z-values of the 3D points. Now it is possible to see that the scanned depth follows the actual shape, which is denoted as a dark red line. As mentioned before the distortions in the middle come from the direct reflection of the projector's light source. These results are similar, albeit a little worse than MC3D.

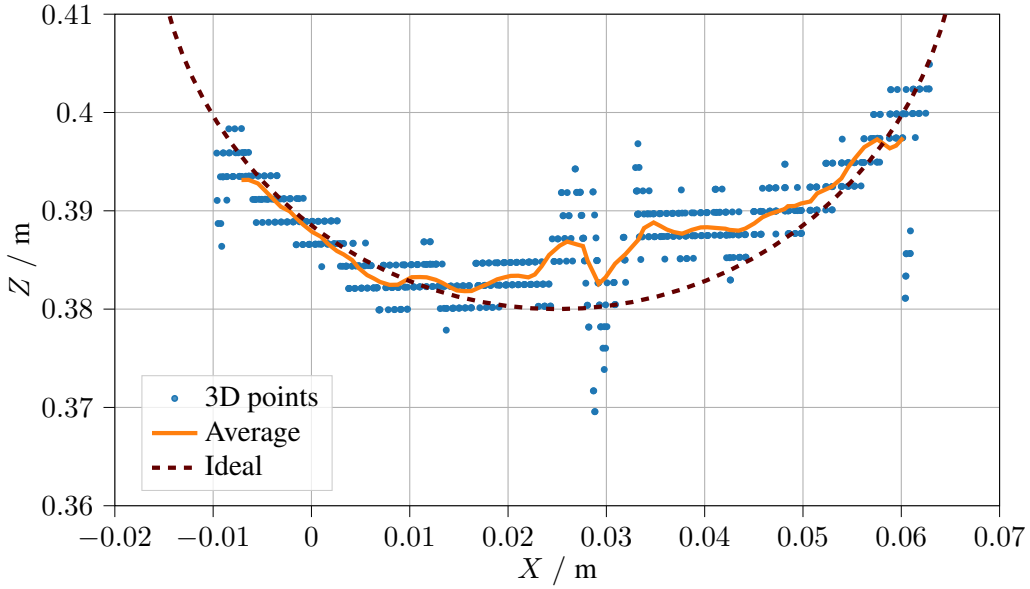
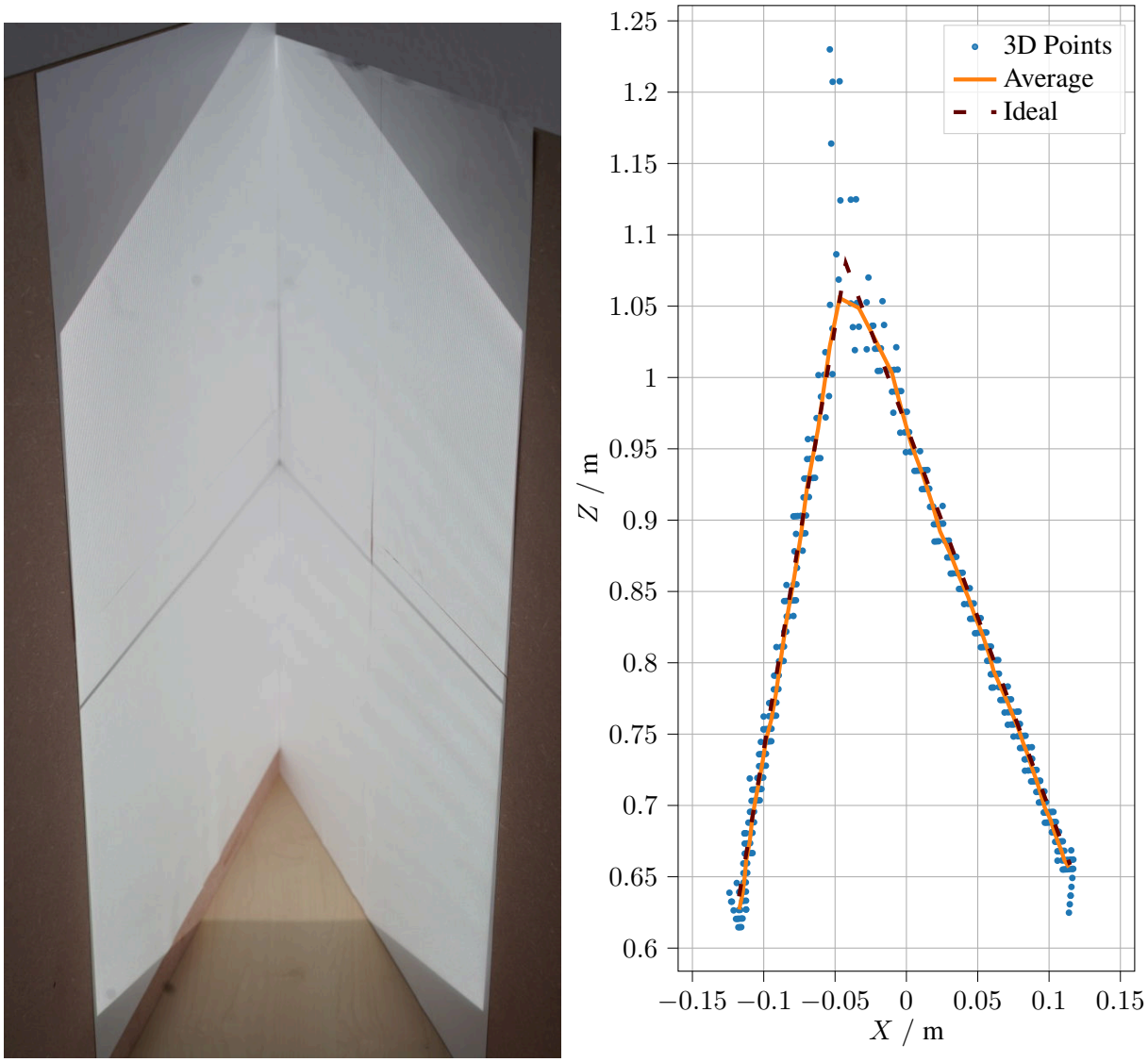


Figure 6.5: Shape of object with highly specular material. The blue points are 3D points in the XZ-plane, the orange line the sliding window average over the depth values, and the dark red line is the actual shape of the object generated according to its physical properties.

Furthermore, a scene with high inter-reflections is setup. Like in MC3D two planes with a white surface are placed with a 30° angle to each other. The reflective surfaces let the light bounce off each other which makes it hard for a conventional structured light system to find the right correspondence. Figure 6.6 shows a photo of the setup on the left and the reconstructed shape on the right. As with the specular material scan, this is the XZ-plane at $Y = 0$ m. Right at the center the reflection is to strong and some events get triggered at the wrong x-coordinate, resulting in false depths. These errors appear through the whole center line as visible in the depth map in figure 6.7. But with an average over the 3D points a shape which is close to the actual shape can be reconstructed. The results are also similar to MC3D's results.



(a) (b)

Figure 6.6: Photo (a) and scan (b) of object with high inter-reflections. The blue points are 3D points in XZ-plane, the orange line the sliding window average over the depth values, and the dark red line is the actual shape of the object generated according to its physical properties.

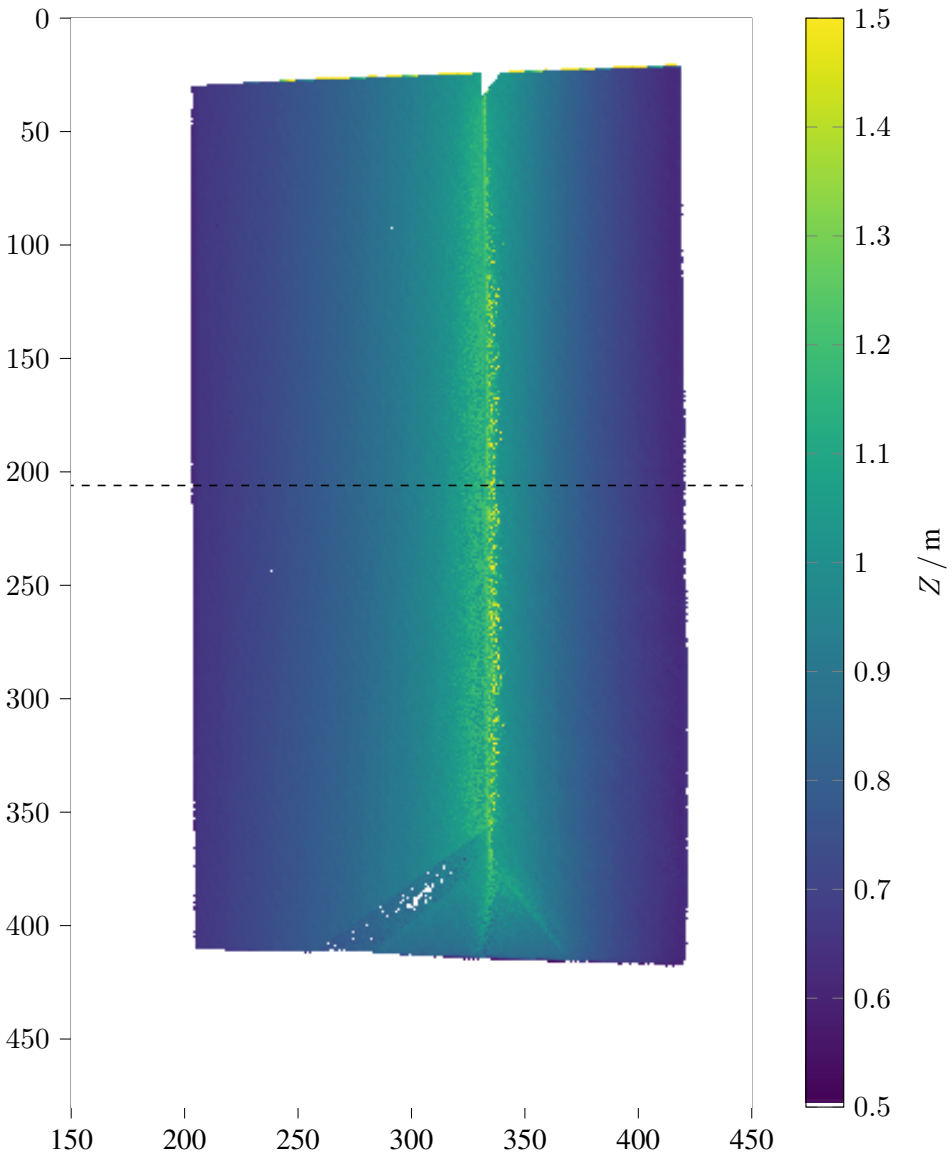
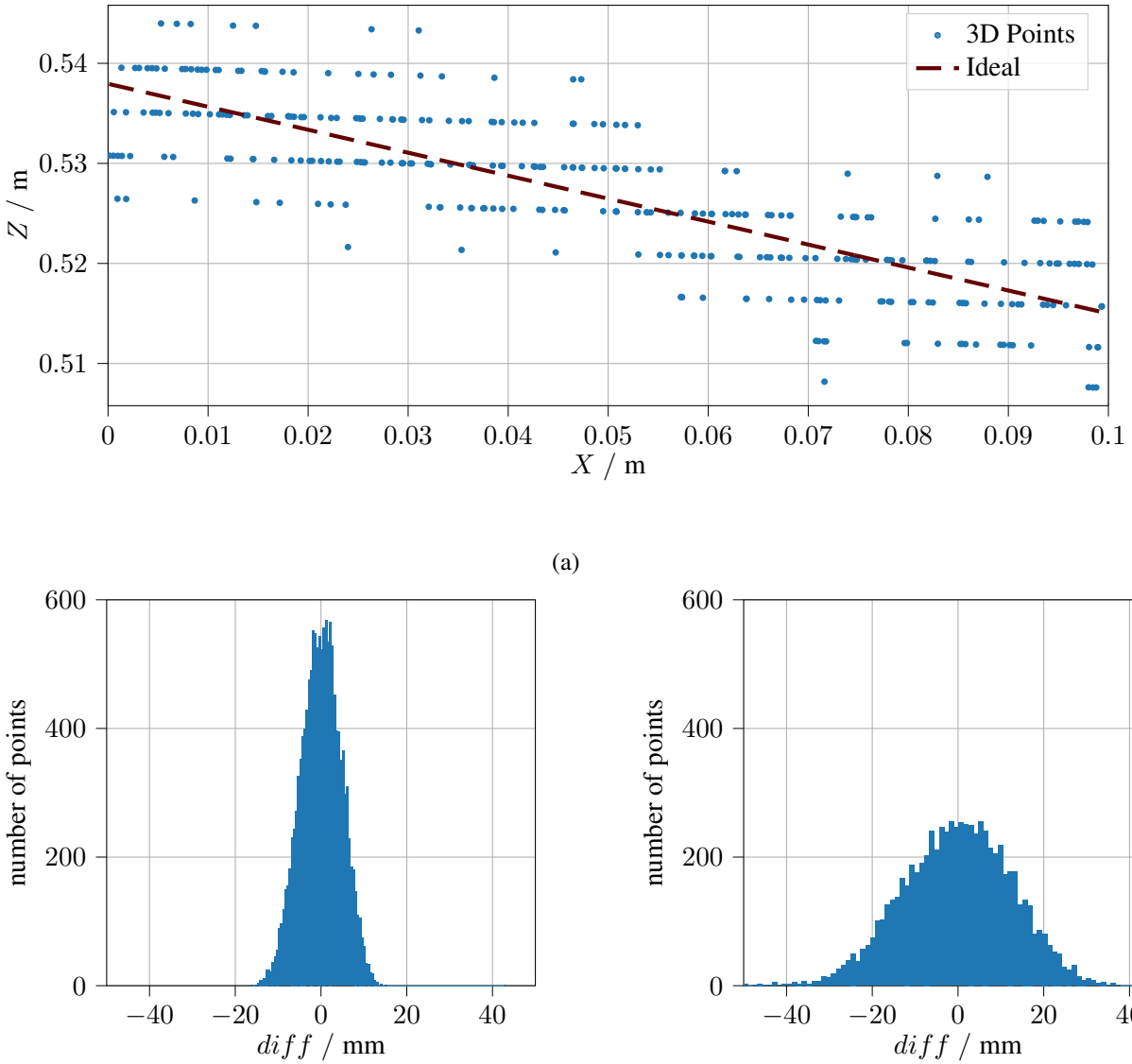


Figure 6.7: Depth map of object with high inter-reflections.

6.5 Noise

In the previous section it possible to see that under non-challenging conditions the depth map follows shapes accurately and with the correct depth. But it is also possible to see that the point cloud is very noisy. In figure 6.8a is the XZ-plane cross section at $Y = 0$ m of the point cloud corresponding to the depth map in figure 6.1b. Here a plane with a slight angle at about 0.5 m distance was scanned. The figure shows an excerpt in the interval $X = [0\text{ m}, 0.1\text{ m}]$. The red line is the plane fitted according to the point cloud.



(a)

(b) mean absolute distance: 0.52 m

(c) mean absolute distance: 0.79 m

Figure 6.8: Error of depth values to fitted plane. (a) shows the XZ-plane cross section of a scanned plane with the 3D points in blue and a fitted plane intersection in dark red. (b) and (c) show histograms of the error of the 3D points to the fitted plane for two different depths of the plane.

It is clear to see that the depth is quantized, which stems from the discrete pixel values of the disparity. The depth is inversely proportional to the disparity and the quantization is coarser at lower disparity and higher depth respectively. But instead of a clear ladder shape, which is expected from a quantized linear signal, the depth values are subject to noise. To quantify this noise the difference in depth to the fitted line is calculated for all points in the interval X = [0 m, 0.1 m] in 41 equidistant cross sections at Y = -0.05 m to Y = 0.05 m. A histogram of the differences is shown in figure 6.8b. The RMS of the

differences is 4.75 mm. According to equation 3.3 and the associate calibration, at the depth of 0.5 m a quantization step is around 4.29 mm. With the same method and the same excerpt the difference to an idealy fitted line is calculated in for a plane which is on average at a distance of 0.79 m, specifically the depth map from figure 6.6b. The respective histogram is presented in figure 6.8c. Here the errors are greater with an RMS of 12,97 mm. The quantization step at this depth is 9.82 mm.

The ratio of the RMS difference Z_{RMS} to the quantization step Z_{step} indicates how strong the noise in disparity pixel is. This ratio and therefore the RMS error in disparity pixel d_{RMS} is 1.1 pixel and 1.3 pixel at 0.5 m and 0.79 m mean distance, respectively. The two values being close suggest that the RMS error for the disparity pixel is independent of the distance and results from various noise sources affecting the disparity search. With this assumption the expected noise on the depth value increases with the depth and can be expressed by

$$\begin{aligned} Z_{\text{RMS}}(Z) &= d_{\text{RMS}} \cdot Z_{\text{step}}(Z) \\ &= d_{\text{RMS}} \cdot \left(\frac{f|b|}{d(Z)} - \frac{f|b|}{d(Z)+1} \right) \\ &= d_{\text{RMS}} \cdot \frac{Z}{f|b|/Z + 1} \end{aligned}$$

Examples for these sources are the timing jitter in projector scan speed, including the jagged pattern visible in figure 4.1b and event timestamp jitter as explained in ESL [3].

One additional observation in the depth maps are bending effects. In figures 6.1b and 6.2b are vertical bending effects on the right side of the color map visible. A little more subtle but prevalent in all depth maps are regular diagonal bending effects over the complete map. Reasons for this may be some periodic modulation on the timing of the projector or timing in the camera or a combination of both. Another reason may be the burst read out mode mentioned before and in ESL of the camera. The shape and direction of the bending can be effected by the transformation of the projector time map during calibration and the rectification.

6.6 Point Cloud Comparison

The strong noise present is very detrimental to a clear point cloud for representing a surface. As the points switch from quantization step to the next it is unclear how the surface of a scanned object is defined. To still be able to process a point cloud generated with this system, the underlying disparity map was heavily filtered. First a pixel wise average over 60 scans is calculated. This breaks the rigid quantization open. Then the disparity map is filtered by an 2D median filter with kernel size (7,7) to

filter out the frequent jumps in depth. Lastly, in the disparity map a sliding window mean is applied on each line with an window length of 12 pixel. As a result the point cloud can represent a surface.

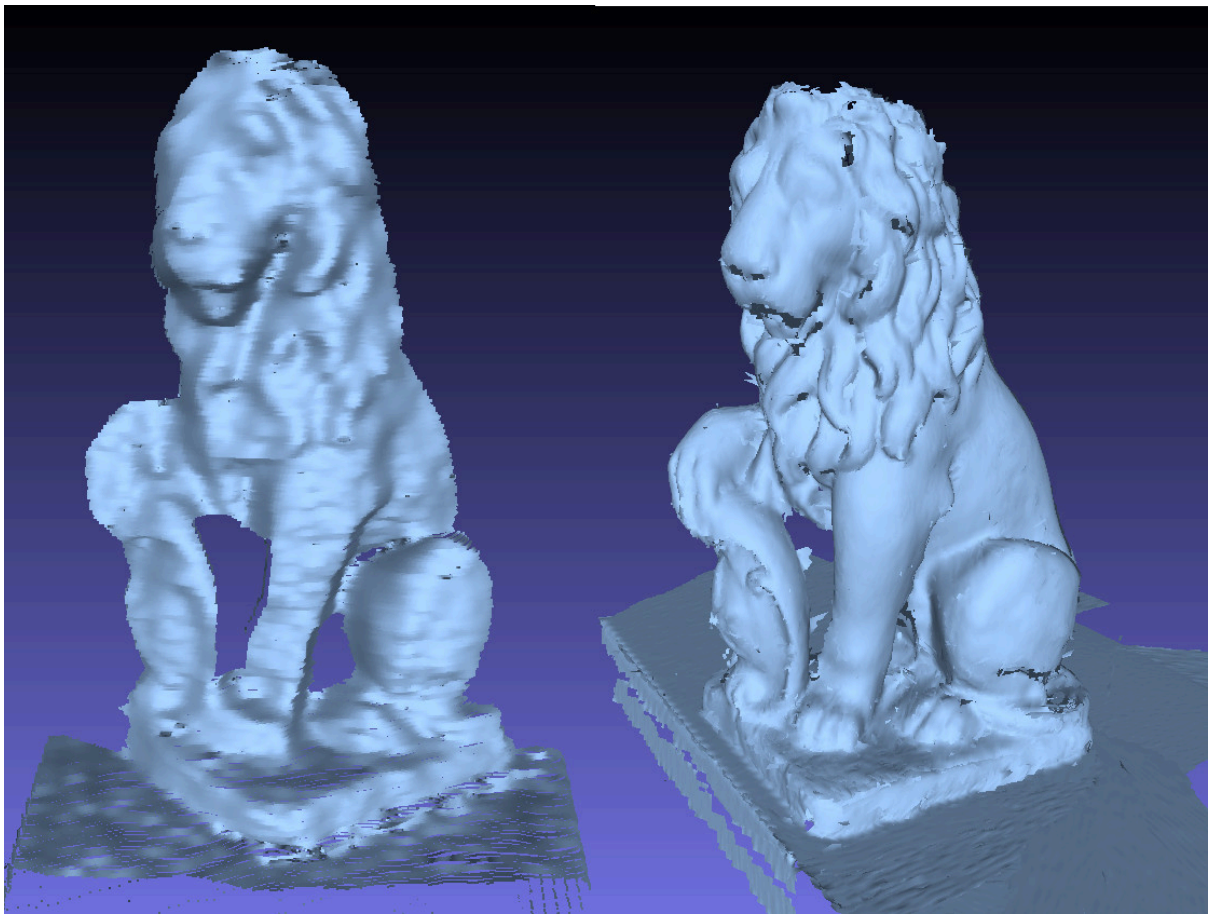


Figure 6.9: Point clouds of a lion statue. The point cloud on the left is a filtered scan generated by the implementation in this thesis. On the right is a scan using dedicated laser scanning hardware.

Figure 6.9 shows a comparison of the point cloud generated using this implementation and a 3D scan of a lion statue created with the *Artec MH-T* lasers scanner from *Artec Ventures*. Although the general shape being correct, the lack of detail in the point cloud on left due to the heavy filtering is obvious. After alignment with the help of the iterative closest point algorithm (ICP) several manual measurements showed that the scan from this implementation is about 4-5% larger than the laser scan. According to the pinhole model and the homogeneous coordinates, the whole surface is scaled up with higher depth values. When the laser scan is regarded as ground truth, this suggest a systematic error in the depth calculation, which most likely stems from inaccuracies in the calibration.

6.7 Demonstrations

The demonstrations are small augmented reality applications that implement the presented structured light system. As they are spatial augmented reality they project onto a physical scene for which no quantitative measurement is possible other than the depth map itself. To convey the working of these applications a qualitative description including photos is done. The depth maps used are from the projector's perspective and are created while the projector is projecting the content.

6.7.1 Depth-based Color Map

The live color map of the depth is the most straight forward application of spatial augmented reality which has 3D knowledge of the scene its projecting on. Although as shown in the SARndbox [28], this idea in combination with a feature rich color map can be used in education and entertainment. Here a color sweep from blue/near to red/far is used to convey the principle. If no depth value was detected for a given pixel because for example occlusion or a too dark projection, that pixel is colored white, to have the best chance to find a depth value in the next frame. Because the resolution of the depth map is lower than the resolution of the projector the pixel with valid depth values are dilated with a (7,7) kernel to fill the image.

The processing is done in python which puts a limit on the processing speed available. With the disparity search moved to the GPU and the calculation of the depth map simplified, the map could be refreshed for every eighth frame scanned by the projector to work in real-time, which comes down to a refresh rate of about 7.5 fps. Events for frames in between were discarded.

In figure 6.10a and 6.10b the color map is projected onto static objects and the boundaries of the color map are set accordingly. Even though the objects are static the depth map is generated from the projected color map and is updated in real-time. As mentioned before, it might be possible to miss the depth information for pixel where the projected color is two dark. This is visible in 6.10a at the upper right corner. Here the red color is to dark and to far away, so pixel are missing the depth information and are colored white.

Figure 6.10c-f shows dynamic scenes. The color map is projected onto hands which move freely in the projection area. The scan speed of 13 ms is fast enough to generate an accurate depth map without artifacts from motion. This speed is one of the main advantages of the event-based camera for structured light, as mentioned in MC3D and ESL. For any movement possible by waving hands in the scanned area the depth map was create.

(a) (b) (c)

(d) (e) (f)

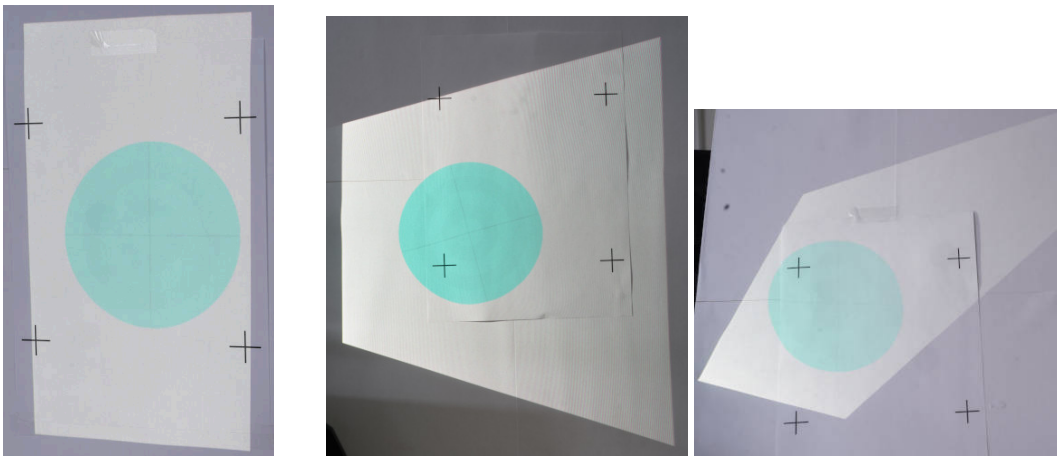
Figure 6.10: Scenes with a live color coded depth map projected onto the themselves. The depth maps and resulting color maps are generated while projecting the color maps. (a) and (b) show static scenes, while (c) - (f) show dynamic scenes.

However, the limiting factor is the refresh rate of 7.5 fps and the inherent delay for the new depth map to be processed and projected. This limits usable depth maps to slower motion if a seamless update is necessary and is the main shortcoming here. Additionally, every depth map is processed out of a single time map on dynamic scenes to stay responsive to changes in the scene. Hence, no averaging over multiple depth maps is done. Thus, dynamic noise is part of the depth maps and visible in the projected color maps. ESL implemented an algorithms to improve the depth map which uses only one depth map and keeps the ability to capture moving objects. But implemented in Python this is to processing in-

tensive to use in a real-time application. Lastly, the size of objects with lower depth is often bigger in the color map than their physical size. As a result the respective color overshoots at the edges of those objects and is projected on the background. This is especially visible at depth discontinuities. This can come from an error in calibration, the mismatch in resolution and dilation as well as noise.

6.7.2 Warping Correction

The second demonstration is the warping of an image to appear correctly if projected onto an angled plane. For this the point cloud from the projector's point of view is used to fit a plane in 3D space using the RANSAC algorithm according to the screen onto which is projected. A virtual circle with fixed radius is placed onto the plane with its center at the intersection of the optical axis of the projector with the plane. This circle is then projected back into the image plane of the projector. The resulting image is the image the projector needs to project to achieve a non-warped circle with set size on a screen with variable distance and angle.



(a) (b) (c)

Figure 6.11: Circle warped to appear undistorted and with fixed size on angled screens. The screens have no angle (a), an angle around the Y-axis (b) and around the X- and Y-axes (c). The crosses act as a reference and span a square with 17 cm side length on the screen. The circle is set to a diameter of 14 cm.

Figure 6.11 shows examples of this. On the left, the screen is mostly perpendicular to the optical axis and none to very little warping is present. Around the circle, the projector projects a white background. In this visualization, this can be used to get a sense of the angle of the screen and the distortion of the projected image. The crosses on the screen act as a reference point and span a square with 17 cm side lenght on the screen. The circle is set to a diameter of 14 cm. The picture in the middle shows a screen

angled around the Y-axis and on the right the screen is angle on both the X- and Y-axes. In all pictures the circle appears circular and is of the same size relative to the reference square.

Since the projector projects white around the circle the point cloud includes the complete area. It is also possible to leave this background black to avoid distraction from the projected image. Now the point cloud only includes points at the projected image. Depending on the size of that image this can be enough to fit an accurate plane. But as the screen moves further away and the projected area increases, the image is scaled down to stay the same physical size in the projection. At some point the point cloud is to small to reliably fit a plane. It is also to note that in this case an external trigger system is necessary. When not the whole projection area is used, not enough events are created to reliably create a trigger signal from the start to the end of one frame.

Due to additional processing the update speed of this implementation is lower than the color map presented before. Here an update every sixteenth frame and therefore a refresh rate of 3.75 fps is possible. But as the screen does not move as quickly as for example hands this does not have a big impact. However, if in further refinement an moving object is tracked to project on, this is to slow.

6.8 Limitations

The results presented before show that the implementation in this thesis works as expected. It can calculate the depth map of a scanned scene while projecting arbitrary content. A few limitations came to light while validating the system.

First, the depth reconstruction works most reliably if the projector outputs as much light as possible. This puts limits on what kind of content might be projected, as it should not be too dark. Here high ambient illumination plays a role as well by further limiting the content brightness. The actual limits also depend on the hardware used and the scenes properties such as reflectiveness and distance to the projector. With the hardware used in this setup, the projected image should not be below 70% off its maximal brightness to give good results on white surfaces in a range of ambient illumination situations. In return, the ambient illumination should not exceed 1000 lx. Moreover, the projected image should fill the full frame in this setup since the trigger algorithm is depending on a constant stream of events. A selective projection and therefore a selective depth map might be desirable in a given augmented reality application but needs an external trigger signal.

Secondly, the disparity map is noisy. There is a need for further processing to filter the noise which

puts limits on the detail of the depth map as well as processing speed. Additionally, averaging over multiple scans or more complex processing like ESL is not possible due to the real-time nature of the application. The calibration method used is prone to propagating errors in the camera calibration to the projector calibration. This can cause systematic errors effecting the calculation.

Finally, the processing speed of this implementation is to slow for meaningful augmented reality applications. The color map projection achieves 7.5 fps while the self correcting circle achieves 3.75 fps. This is mainly limited by the software implementation which was chosen to enable faster prototype development in the context of this thesis. An implementation in a suitable programming language and on dedicated hardware like a GPU or, if integrated, an ASIC or FPGA can achieve much higher processing speeds.

Chapter 7

Conclusion and Outlook

7.1 Summary

In this thesis, a structured light system was implemented using an event-based camera and a video projector system. This solution combines the advantages of events based vision like lower bandwidth and therefore higher scan speeds, independence of constant ambient illumination, and high dynamic range to the laser scan system which excels in light source efficiency, robustness, and accuracy. Moreover, as the event camera follows the movement of the laser beam and not its intensity, the projected content can be chosen arbitrarily as long as it is bright enough. The goal of this thesis was to implement such as system and simultaneously use the projector for spatial augmented reality applications which benefit or rely on 3D information of the scene projected on.

A dedicated calibration application was implemented to account for the event stream as opposed to typical frame-based calibration solutions. With this system, it is possible to calibrate the event camera and the projector as a stereo pair. For the main part, a structured light system was implemented on the basis of the publications MC3D and ESL. Specifically, a trigger detection system, a way to calibrate the reference image of the projector, the disparity search on the GPU, and easily switching from the camera's point of view to the projector's were implemented for this setup. To test the ability to project images other than a white image, two augmented reality use cases were explored. First, projecting a color map of the depth onto the scene, and second, correcting the transformation of a projected image to appear undistorted if projected on an angled screen.

Results show that the system can create a depth map of a scene and truthfully recreate physical shapes and geometry. Projecting images other than a white image is possible but they can not be chosen completely arbitrarily. As the laser beam has to be bright enough to trigger events in the camera the image is

limited to brighter colors for areas where a depth map is supposed to be created. The necessary brightness is also dependet on the ambient illumination of the scene. The depth reconstruction works also relatively well on challenging materials like specular materials and high inter-reflections similar to results of MC3D. The augmented reality applications work as expected. It is possible to project an image while sensing the depth of the scene at the same time.

Although being able to create a depth map of the scene, the depth information suffers from noise in the disparity map. Causes for this are expected to be the non-linear timing of the projector and the jitter therein as well as jitter in the event timing and bending effects through burst read out over multiple lines on the camera sensor. The noise makes detailed processing of the 3D information difficult and can lead to inaccuracies in the images generate for augmented reality. The calibration implementation relies on the projection of the projector image features, as detected in the camera image, back to 3D space. This can propagate errors in the camera calibration to the projector calibration. Small errors in the calibration lead to systematic errors in the depth map which are observed to some degree in this system.

In conclusion, this thesis has shown that it is possible with a structured light system employing an event camera to project images other than white and scan the depth of the scene concurrently. The free choice of the projected image and simultaneous depth map enables augmented reality applications using the same projector. This simplifies the setup compared to systems with separate projector and depth sensing solution and improves the depth sensing compared to other depth sensing solutions with similar speed.

7.2 Future Directions

Most of the limitations mentioned stem from the choice of hardware, e.g. an off-the-shelf consumer product in the case of the projector and software which was chosen for ease of development. Further refinement of this system is possible for all limitations explained before. The noise can be reduced using video laser projectors with a more accurately controlled laser movement and speed. Also, a newer generation of event cameras can have improvements in timing noise as well as resolution. The calibration can be improved upon by integrating the state-off-the-art in event-to-frame conversion and projector-camera calibration techniques as it is done in ESL. The limitations on the brightness level of the content can be alleviated by using a brighter projector which is sensible anyway in professional augment reality installations. The processing speed can be sped up by implementing the system in a lower-level language and on dedicated hardware. A fully integrated product can easily install an external trigger system to be independent of the number of events generated per frame to trigger the camera time map.

Bibliography

[1] N. Matsuda, O. Cossairt, and M. Gupta, "MC3D: Motion Contrast 3D Scanning," in *IEEE International Conference on Computational Photography (ICCP)*, Apr. 2015, pp. 1–10.

[2] J. Chen, T. YAMAMOTO, T. Aoyama, T. Takaki, and I. Ishii, "Real-Time Projection Mapping Using High-Frame-Rate Structured Light 3D Vision," *SICE Journal of Control, Measurement, and System Integration*, vol. 8, pp. 265–272, Jul. 2015.

[3] M. Muglikar, G. Gallego, and D. Scaramuzza, "ESL: Event-based Structured Light," in *International Conference on 3D Vision (3DV)*, Dec. 2021, pp. 1165–1174.

[4] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based Vision: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, Jan. 2022.

[5] R. Szeliski, *Computer Vision: Algorithms and Applications*, ser. Texts in Computer Science. Cham: Springer International Publishing, 2022.

[6] E. Mouaddib, J. Batlle, and J. Salvi, "Recent progress in structured light in order to solve the correspondence problem in stereovision," in *Proceedings of International Conference on Robotics and Automation*, Apr. 1997, pp. 130–136.

[7] B. Pan, Q. Guan, X. Wang, and S. Y. Chen, "Strategies of Real-Time 3D Reconstruction by Structured Light," in *Chinese Conference on Pattern Recognition (CCPR)*, Oct. 2010, pp. 1–5.

[8] M. Gupta, Q. Yin, and S. K. Nayar, "Structured Light in Sunlight," in *IEEE International Conference on Computer Vision*, Dec. 2013, pp. 545–552.

[9] E. Fossum, "CMOS image sensors: electronic camera-on-a-chip," *IEEE Transactions on Electron Devices*, vol. 44, no. 10, pp. 1689–1698, Oct. 1997.

[10] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128x128 120 dB 15 us Latency Asynchronous Temporal Contrast Vision Sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008.

[11] Y. Suh, S. Choi, M. Ito, J. Kim, Y. Lee, J. Seo, H. Jung, D.-H. Yeo, S. Namgung, J. Bong, S. Yoo, S.-H. Shin, D. Kwon, P. Kang, S. Kim, H. Na, K. Hwang, C. Shin, J.-S. Kim, P. K. J. Park, J. Kim, H. Ryu, and Y. Park, "A 1280×960 Dynamic Vision Sensor with a 4.95-um Pixel Pitch and Motion Artifact Minimization," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, Oct. 2020, pp. 1–5.

[12] C. Posch, T. Serrano-Gotarredona, B. Linares-Barranco, and T. Delbruck, "Retinomorphic Event-Based Vision Sensors: Bioinspired Cameras With Spiking Output," *Proceedings of the IEEE*, vol. 102, no. 10, pp. 1470–1484, Oct. 2014.

[13] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143dB dynamic range asynchronous address-event PWM dynamic image sensor with lossless pixel-level video compression," in *IEEE International Solid-State Circuits Conference - (ISSCC)*, Feb. 2010, pp. 259–275.

[14] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240 × 180 130 dB 3 us Latency Global Shutter Spatiotemporal Vision Sensor," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, Oct. 2014.

[15] R. Berner, C. Brandli, M. Yang, S.-C. Liu, and T. Delbruck, "A 240×180 10mW 12us latency sparse-output vision sensor for mobile applications," in *Symposium on VLSI Circuits*, Jun. 2013, pp. C186–C187, iSSN: 2158-5636.

[16] H. Hirschmuller, "Stereo Processing by Semiglobal Matching and Mutual Information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[17] R. Azuma, "A Survey of Augmented Reality," *Presence: Teleoperators and Virtual Environments*, vol. 6, no. 4, pp. 355–385, 1997.

[18] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre, "Recent advances in augmented reality," *IEEE Computer Graphics and Applications*, vol. 21, no. 6, pp. 34–47, Nov. 2001.

[19] H. Ling, "Augmented Reality in Reality," *IEEE MultiMedia*, vol. 24, no. 3, pp. 10–15, 2017.

[20] M. Sereno, X. Wang, L. Besançon, M. J. McGuffin, and T. Isenberg, "Collaborative Work in Augmented Reality: A Survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 6, pp. 2530–2549, Jun. 2022.

[21] X. Zhang, S. Fronz, and N. Navab, "Visual marker detection and decoding in AR systems: a comparative study," in *Proceedings. International Symposium on Mixed and Augmented Reality*, Oct. 2002, pp. 97–106.

[22] R. Raskar, G. Welch, and H. Fuchs, "Spatially Augmented Reality," *Augmented Reality: Placing Artificial Objects in Real Scenes*, pp. 64–71, Nov. 1999.

[23] M. Flagg and J. M. Rehg, "Projector-guided painting," in *Proceedings of the 19th annual ACM symposium on User interface software and technology*, ser. UIST '06. New York, NY, USA: Association for Computing Machinery, Oct. 2006, pp. 235–244.

[24] B. R. Jones, H. Benko, E. Ofek, and A. D. Wilson, "IllumiRoom: peripheral projected illusions for interactive experiences," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '13. New York, NY, USA: Association for Computing Machinery, Apr. 2013, pp. 869–878.

[25] J. Lee, Y. Kim, M.-H. Heo, D. Kim, and B.-S. Shin, "Real-Time Projection-Based Augmented Reality System for Dynamic Objects in the Performing Arts," *Symmetry*, vol. 7, pp. 182–192, Mar. 2015.

[26] M. Sakamoto, T. Shinoda, T. Ishizu, M. Hori, H. Watanabe, A. Takei, and T. Ito, "A Proposal of Interactive Projection Mapping Using Kinect," in *International Conference on Information and Communication Technology Robotics (ICT-ROBOT)*, Sep. 2018, pp. 1–4.

[27] A. Wilson, H. Benko, S. Izadi, and O. Hilliges, "Steerable augmented reality with the beamatron," in *Proceedings of the 25th annual ACM symposium on User interface software and technology*. New York, NY, USA: Association for Computing Machinery, Oct. 2012, pp. 413–422.

[28] O. Kreylos. Sarndbox: Augmented reality sandbox. https://github.com/KeckCAVES/SARndbox and https://www.youtube.com/watch?v=j9JXtTj0mzE, (accessed 1.9.2022).

[29] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.

[30] A. Fusiello, E. Trucco, and A. Verri, "A Compact Algorithm for Rectification of Stereo Pairs," *Machine Vision and Applications*, vol. 12, Oct. 2000.

[31] Microvision. Showwx. (accessed: 31.07.2022). [Online]. Available: https://web.archive.org/web/20110614205539/http://www.microvision.com/showwx/pdfs/showwx_userguide.pdf

[32] Microsoft. Kinect. (accessed: 31.07.2022). [Online]. Available: https://docs.microsoft.com/en-us/archive/msdn-magazine/2012/june/kinect-starting-to-develop-with-kinect

[33] Sony. Mobile projector mpcl1a. (accessed: 31.07.2022). [Online]. Available: https://www.sony.com/electronics/support/televisions-projectors-projectors/mp-cl1a/specifications

[34] Prophesee. Evaluations kits. (accessed: 08.08.2022). [Online]. Available: https://docs.prophesee.ai/stable/hw/evk/gen3.html

[35] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 1280–1286.

[36] Prophesee. Gen3 sensor. (accessed: 08.08.2022). [Online]. Available: https://docs.prophesee.ai/stable/hw/sensors/PPS3MVCD.html

[37] ——. Gen3 sensor biases. (accessed: 08.08.2022). [Online]. Available: https://docs.prophesee.ai/stable/hw/manuals/biases.html

[38] ——. Metavision sdk. (accessed: 22.08.2022). [Online]. Available: https://docs.prophesee.ai/stable/index.html

[39] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "High Speed and High Dynamic Range Video with an Event Camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 6, pp. 1964–1980, Jun. 2021.

[40] D. Moreno and G. Taubin, "Simple, Accurate, and Robust Projector-Camera Calibration," in *Visualization & Transmission 2012 Second International Conference on 3D Imaging, Modeling, Processing*, Oct. 2012, pp. 464–471.

[41] K. Madsen, H. Nielsen, and O. Tingleff, "Methods for non-linear least squares problems," Apr. 2006, (accessed 9.9.2022). [Online]. Available: http://www2.imm.dtu.dk/pubdb/edoc/imm3215.pdf

[42] uzh rpg/ESL. Github: Event-based structured light. (accessed: 24.08.2022). [Online]. Available: https://github.com/uzh-rpg/ESL

List of Figures

2.1 Examples of structured light system 7
2.2 Comparison of frame-based and event-based data output 9

3.1 Epipolar geometry 18
3.2 MC3D model for time-space correspondence of one row 21
3.3 Representation of time maps in ESL paper 23

4.1 Jitter of projector scan 27
4.2 Representation of the physical setup and angles of view 29

5.1 Timestamps of event stream to detect frame 38
5.2 Time map for projector time map calibration 41
5.3 Binary map used to detect projected frame 42
5.4 Flattened calibrated projector time map 43
5.5 Detailed views of flattened calibrated projector time map 43

6.1 Depth map of a plane with and without calibrated projector time map 54
6.2 Scene for the ambient light measurement 56
6.3 Depth maps with various ambient illumination levels and content levels. 57
6.4 Photo (a) and depth map (b) of object with highly specular material. 59
6.5 Shape of object with highly specular material. 60
6.6 Photo and scan of object with high inter-reflections 61
6.7 Depth map of object with high inter-reflections. 62
6.8 Error of depth values to fitted plane 63
6.9 Point clouds of a lion statue 65
6.10 Scenes with a live color coded depth map projected onto the themselves. The depth maps and resulting color maps are generated while projecting the color maps. (a) and (b) show static scenes, while (c) - (f) show dynamic scenes. 67
6.11 Circle warped to appear undistorted on angled screens. 68

List of Tables

6.1 Fill rate for various lightning conditions and content level 58

List of Listings

5.1 Example of trigger detection function 39
5.2 Example of the time map generation function. 40
5.3 Example of the disparity search function 45
5.4 Example for funtction to get the point cloud from the disparity map 47