Pose estimation with event camera

Ahmed Tabia

université PARIS-SACLAY

Pose Estimation With Event Camera

Estimation de la pose avec une caméra évènementielle

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580 Sciences et Technologies de l'Information et de la Communication (STIC) Spécialité de doctorat : Robotique
Graduate School : Science de l'ingénierie et des systèmes. Référent Université d'Évry Val d'Essonne

Thèse préparée dans l'unité de recherche **IBISC (Université Paris-Saclay, Univ Evry)**, sous la direction de **Samia BOUCHAFA-BRUNEAU**, Professeure des Universités, la co-direction de **Fabien BONARDI**, Maître de Conférences.

Thèse soutenue à Paris-Saclay, le 6 septembre 2024, par

Ahmed TABIA

Composition du jury

Membres du jury avec voix délibérative

Samia AINOUZ Professeure des Universités, INSA de Rouen	Présidente
Choubeila MAAOUI Professeure des Universités, Université de Lorraine	Rapporteure & Examinattrice
Christophe SABOURIN Professeur des Universités, Université Paris-Est Créteil	Rapporteur & Examinateur
Michele GOUIFFES Maîtresse de Conférences, HDR, Université Paris-Saclay	Examinatrice

THESE DE DOCTORAT

NNT : 2024UPAST093

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Estimation de la pose avec une caméra évènementielle
Mots clés : Estimation de la pose, 6 degrés de liberté, Apprentissage profond, caméra évènementielle.

Résumé : La pose de la caméra est utilisée pour décrire la position et l'orientation d'une caméra dans un système de coordonnées absolu, en référence à six degrés de liberté. L'estimation de la pose de la caméra est essentielle dans divers domaines d'application, tels que la réalité augmentée, la navigation robotique et les véhicules autonomes. Ces domaines exploitent la pose de la caméra pour des calculs ultérieurs, comme la localisation des objets et la perception de la scène.

Estimer la pose d'une caméra présente des défis dans différents scénarios ; les conditions d'éclairage médiocres, y compris une obscurité ou une luminosité extrêmes, limitent l'efficacité de la plupart des méthodes basées sur des caractéristiques. Ces conditions d'éclairage défavorables entravent la détection et la correspondance précises des caractéristiques, affectant ainsi la précision de l'estimation de la pose de la caméra. Les scènes manquant de textures distinctes compliquent l'extraction de points clés significatifs, tandis que le mouvement rapide entraîne un flou cinétique, nuisant à la qualité de l'image et à la précision de l'estimation de la pose.

La plupart de ces défis rencontrés dans l'estimation de la pose de la caméra sont largement liés à la nature des caméras traditionnelles, qui capturent le monde sous forme d'une série d'images fixes, prises successivement à un rythme rapide. Dans les cas où ces difficultés sont particulièrement prononcées, les caméras événementielles offrent des avantages potentiels.

Les caméras événementielles sont des capteurs bio-inspirés qui imitent le fonctionnement de la rétine humaine, en capturant les changements d'intensité des pixels plutôt que d'enregistrer des images complètes à un taux fixe, comme le font les caméras traditionnelles basées sur des trames.

Cette thèse se concentre sur l'estimation de la pose des caméras événementielles et vise à explorer l'application de méthodes d'apprentissage en profondeur pour la pose et la relocalisation basées sur ces caméras, en tirant parti de leurs propriétés uniques telles que la haute résolution temporelle, la faible latence et la large plage dynamique.

La thèse apporte plusieurs contributions au domaine de l'estimation de la pose de caméra événementielle en utilisant des techniques d'apprentissage profond. Ces contributions peuvent être résumées comme suit :

- La thèse fournit un aperçu complet des informations de base et des travaux connexes, établissant ainsi une base solide et une compréhension contextuelle de l'estimation de la pose de caméra événementielle.
- La thèse explore et développe des approches spécialisées d'apprentissage profond adaptées à l'estimation de la pose de caméra événementielle. Ces techniques exploitent la puissance de l'apprentissage profond pour estimer avec précision la pose de la caméra à l'aide de données événementielles.
- La thèse introduit des méthodes pour projeter les données événementielles en données semblables à des images, facilitant l'application d'approches dédiées d'apprentissage profond. Ce processus de projection permet une utilisation efficace des informations événementielles dans la tâche d'estimation de la pose de la caméra.
- La thèse propose une nouvelle approche qui applique directement des techniques d'apprentissage profond aux données événementielles brutes, les traitant comme un nuage de points plutôt que de les convertir en images. Cette approche exploite l'ensemble des informations capturées par la caméra événementielle et permet un processus d'apprentissage de bout en bout.

Title : Pose Estimation With Event Camera
Keywords : Pose estimation, 6 degrees of freedom, Deep learning, Event-based camera.

Abstract : Camera pose is used to describe the position and orientation of a camera in an absolute coordinate system, with reference to six degrees of freedom. Estimating the camera pose is essential in various application domains, such as augmented reality, robotic navigation, and autonomous vehicles. These fields rely on camera pose for subsequent calculations, such as object localization and scene perception.

Estimating the pose of a camera presents challenges in different scenarios; poor lighting conditions, including extreme darkness or brightness, limit the effectiveness of most feature-based methods. These unfavorable lighting conditions hinder precise feature detection and matching, thereby affecting the accuracy of camera pose estimation. Scenes lacking distinct textures complicate the extraction of meaningful keypoints, while rapid motion leads to motion blur, affecting image quality and pose estimation accuracy.

Most of these challenges encountered in camera pose estimation are largely related to the nature of traditional cameras, which capture the world as a series of static images taken successively at a rapid pace. In cases where these difficulties are particularly pronounced, event-based cameras offer potential advantages.

Event-based cameras are bio-inspired sensors that mimic the functioning of the human retina, capturing changes in pixel intensity rather than recording full images at a fixed rate, as traditional frame-based cameras do.

This thesis focuses on estimating the pose of event-based cameras and aims to explore the application of deep learning methods for pose estimation and relocalization based on these cameras, leveraging their unique properties such as high temporal resolution, low latency, and wide dynamic range.

The thesis makes several contributions to the field of event-based camera pose estimation using deep learning techniques. These contributions can be summarized as follows :

- The thesis provides a comprehensive overview of foundational information and related work, thus establishing a solid foundation and contextual understanding of event-based camera pose estimation.
- The thesis explores and develops specialized deep learning approaches tailored to event-based camera pose estimation. These techniques harness the power of deep learning to accurately estimate camera pose using event data.
- The thesis introduces methods to project event data into image-like data, facilitating the application of dedicated deep learning approaches. This projection process allows for efficient use of event data in the camera pose estimation task.
- The thesis proposes a novel approach that directly applies deep learning techniques to raw event data, treating them as a point cloud rather than converting them into images. This approach leverages the entirety of information captured by the event-based camera and enables an end-to-end learning process.

To my family, whose love, encouragement, and steadfast support have been the driving force and source of inspiration throughout my academic journey…

Acknowledgements

Prior to delving into the scientific and technical aspects presented in this thesis, I would like to take a moment to express my sincere appreciation for the essential individuals who have played an indispensable role in bringing this endeavor to its successful completion. I am deeply grateful to Prof. Samia BOUCHAFA-BRUNEAU, the thesis director, whose guidance and unwavering support have been invaluable throughout the entire process, even before its commencement. Equally significant is the immeasurable assistance provided by Assoc. Prof. Fabien Bonardi, the co-supervisor of this thesis. The trust and freedom bestowed upon me by all my mentors, allowing me to explore new research avenues and expand my horizons, is a favor I will always cherish.

Lastly, I must acknowledge the profound impact my friends have had on me, serving as pillars of strength during times of vulnerability. While it is impractical to individually acknowledge each member of this extensive network due to their sheer number, I hold enduring memories of their acts of kindness. The collective efforts made by all of them on my behalf will forever be etched in my mind.

Résumé en français

La pose de la caméra est utilisée pour décrire la position et l'orientation d'une caméra dans un système de coordonnées absolu, en référence à six degrés de liberté. L'estimation de la pose de la caméra est essentielle dans divers domaines d'application, tels que la réalité augmentée, la navigation robotique et les véhicules autonomes. Ces domaines exploitent la pose de la caméra pour des calculs ultérieurs, comme la localisation des objets et la perception de la scène.

Estimer la pose d'une caméra présente des défis dans différents scénarios ; les conditions d'éclairage médiocres, y compris une obscurité ou une luminosité extrêmes, limitent l'efficacité de la plupart des méthodes basées sur des caractéristiques. Ces conditions d'éclairage défavorables entravent la détection et la correspondance précises des caractéristiques, affectant ainsi la précision de l'estimation de la pose de la caméra. Les scènes manquant de textures distinctes compliquent l'extraction de points clés significatifs, tandis que le mouvement rapide entraîne un flou cinétique, nuisant à la qualité de l'image et à la précision de l'estimation de la pose.

La plupart de ces défis rencontrés dans l'estimation de la pose de la caméra sont largement liés à la nature des caméras traditionnelles, qui capturent le monde sous forme d'une série d'images fixes, prises successivement à un rythme rapide. Dans les cas où ces difficultés sont particulièrement prononcées, les caméras événementielles offrent des avantages potentiels.

Les caméras événementielles sont des capteurs bio-inspirés qui imitent le fonctionnement de la rétine humaine, en capturant les changements d'intensité des pixels plutôt que d'enregistrer des images complètes à un taux fixe, comme le font les caméras traditionnelles basées sur des trames.

Cette thèse se concentre sur l'estimation de la pose des caméras événementielles et vise à explorer l'application de méthodes d'apprentissage en profondeur pour la pose et la relocalisation basées sur ces caméras, en tirant parti de leurs propriétés uniques telles que la haute résolution tem-

porelle, la faible latence et la large plage dynamique.

La thèse apporte plusieurs contributions au domaine de l'estimation de la pose de caméra événementielle en utilisant des techniques d'apprentissage profond. Ces contributions peuvent être résumées comme suit :

- La thèse fournit un aperçu complet des informations de base et des travaux connexes, établissant ainsi une base solide et une compréhension contextuelle de l'estimation de la pose de caméra événementielle.

- La thèse explore et développe des approches spécialisées d'apprentissage profond adaptées à l'estimation de la pose de caméra événementielle. Ces techniques exploitent la puissance de l'apprentissage profond pour estimer avec précision la pose de la caméra à l'aide de données événementielles.

- La thèse introduit des méthodes pour projeter les données événementielles en données semblables à des images, facilitant l'application d'approches dédiées d'apprentissage profond. Ce processus de projection permet une utilisation efficace des informations événementielles dans la tâche d'estimation de la pose de la caméra.

- La thèse propose une nouvelle approche qui applique directement des techniques d'apprentissage profond aux données événementielles brutes, les traitant comme un nuage de points plutôt que de les convertir en images. Cette approche exploite l'ensemble des informations capturées par la caméra événementielle et permet un processus d'apprentissage de bout en bout.

English abstract

Camera pose is used to describe the position and orientation of a camera in an absolute coordinate system, with reference to six degrees of freedom. Estimating the camera pose is essential in various application domains, such as augmented reality, robotic navigation, and autonomous vehicles. These fields rely on camera pose for subsequent calculations, such as object localization and scene perception.

Estimating the pose of a camera presents challenges in different scenarios; poor lighting conditions, including extreme darkness or brightness, limit the effectiveness of most feature-based methods. These unfavorable lighting conditions hinder precise feature detection and matching, thereby affecting the accuracy of camera pose estimation. Scenes lacking distinct textures complicate the extraction of meaningful keypoints, while rapid motion leads to motion blur, affecting image quality and pose estimation accuracy.

Most of these challenges encountered in camera pose estimation are largely related to the nature of traditional cameras, which capture the world as a series of static images taken successively at a rapid pace. In cases where these difficulties are particularly pronounced, event-based cameras offer potential advantages.

Event-based cameras are bio-inspired sensors that mimic the functioning of the human retina, capturing changes in pixel intensity rather than recording full images at a fixed rate, as traditional frame-based cameras do.

This thesis focuses on estimating the pose of event-based cameras and aims to explore the application of deep learning methods for pose estimation and relocalization based on these cameras, leveraging their unique properties such as high temporal resolution, low latency, and wide dynamic range.

The thesis makes several contributions to the field of event-based camera pose estimation using deep learning techniques. These contributions can be summarized as follows:

- The thesis provides a comprehensive overview of foundational information and related work, thus establishing a solid foundation and contextual understanding of event-based camera pose estimation.
- The thesis explores and develops specialized deep learning approaches tailored to event-based camera pose estimation. These techniques harness the power of deep learning to accurately estimate camera pose using event data.
- The thesis introduces methods to project event data into image-like data, facilitating the application of dedicated deep learning approaches. This projection process allows for efficient use of event data in the camera pose estimation task.
- The thesis proposes a novel approach that directly applies deep learning techniques to raw event data, treating them as a point cloud rather than converting them into images. This approach leverages the entirety of information captured by the event-based camera and enables an end-to-end learning process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Camera pose estimation is a foundational task in computer vision and robotics. It also plays a vital role across multiple domains, providing valuable insights into the real world and enabling a wide range of applications.

In robotics and autonomous vehicles, camera pose estimation enables accurate navigation and localization. Augmented reality and virtual reality heavily rely on camera pose estimation to overlay digital information onto the real world or create immersive virtual environments. It is essential for 3D reconstruction, allowing the creation of detailed models from multiple images. Camera pose estimation aids drone navigation in GPS-denied situations, facilitates precise medical procedures, enhances sports and entertainment by tracking athletes or actors, and contributes to geographic information systems by aligning aerial or satellite imagery with maps.

Estimating the pose of a camera, determining its position and orientation in a scene, presents challenges in various scenarios. Poor lighting conditions, including extreme darkness or brightness, limit the application of most of feature based methods. These unfavorable lighting conditions impede the process of detecting and matching features, thereby hindering accurate camera pose estimation. Scenes lacking distinct textures make it difficult to extract meaningful keypoints. Fast motion introduces motion blur, degrading image quality and pose estimation accuracy. Non-rigid object movements, such as swaying plants or flowing cloth, complicate the

process. Wide baselines, where the camera moves significantly between frames, hinder feature matching. Large scene depth and parallax effects distort feature matching, especially in outdoor scenes. Repetitive patterns confuse algorithms, leading to incorrect pose estimation. Occlusions, particularly moving or rapidly changing ones, obstruct the camera's view and affect pose estimation. Lens distortion, common in wide-angle or fisheye lenses, adds complexity. Finally, dynamic scenes with continuous object movement pose additional challenges for camera pose estimation.

Most of the challenges faced in camera pose estimation are largely inherent to the nature of traditional cameras that capture the world as a series of static images, taken one after the other in quick succession. This process involves a mechanical or electronic shutter that opens briefly to let light reach the sensor array, thereby forming an image, and then shuts once more. This way, the visual scene is sampled at discrete time intervals, typically at a constant rate.

However, this prevalent method of visual sensing comes with several other limitations that predominantly contribute to the challenges mentioned earlier: (i) The sampling rate is not adjusted based on the dynamics of the scene. That is, a scene with little to no movement is sampled at the same frequency as a scene with rapid motion. (ii) Motion blur is a possibility due to movement within the scene while the shutter is open. (iii) When exposure is uniform across all pixels, the dynamic range can be restricted. (iv) Lastly, the camera is essentially 'blind' during the periods between consecutive images, leaving gaps in visual information capture.

In spite of these constraints, conventional RGB cameras find extensive applications, notably in photography and videography, video surveillance, object recognition, gaming, and augmented reality. Nonetheless, while they are extensively utilized and prove effective in diverse fields, traditional cameras do face significant challenges and drawbacks. In situations where these difficulties become pronounced, event-based cameras offer potential benefits.

Event-based cameras, also known as neuromorphic or asynchronous cameras, operate on a fundamentally different principle compared to tra-

ditional cameras. Rather than capturing frames at fixed intervals, event-based cameras detect pixel-level changes in brightness asynchronously and transmit these changes as events with precise timestamps.

Event-based Camera Event cameras are bio-inspired sensors that mimic the functioning of the human retina, capturing changes in pixel intensity rather than recording entire frames at a fixed rate as traditional frame-based cameras do Lichtsteiner et al. (2008).

This results in an asynchronous, event-driven data stream, where events are triggered independently for each pixel whenever the intensity changes by a certain threshold Gallego et al. (2020). Due to this unique working principle, event cameras offer several advantages over traditional cameras, including high temporal resolution (in the order of microseconds), low latency, high dynamic range (HDR), and reduced data redundancy Rebecq et al. (2017).

These characteristics make event cameras particularly well-suited for fast and accurate pose estimation and relocalization tasks in dynamic environments. In robotics, for instance, event cameras can provide real-time feedback for high-speed motion control and navigation Mueggler et al. (2017). In augmented reality applications, event cameras can enable accurate and low-latency tracking of objects and scene geometry, improving the user experience and reducing computational requirements Kim et al. (2008). In the context of autonomous vehicles, event cameras can be utilized for robust and efficient localization and mapping in diverse and challenging lighting conditions Zhu et al. (2018).

While a substantial body of literature exists in the domain of traditional camera-based image processing, there is a comparatively limited volume of research focusing on event-based camera methods. Moreover, the exploration of recent techniques like deep learning-based approaches within the context of event-based cameras is even more scant.

The rise of deep learning Deep learning techniques, particularly Convolutional Neural Networks (CNNs), have demonstrated remarkable success in a wide range of computer vision tasks, such as object detection Redmon

et al. (2016), semantic segmentation Long et al. (2015), and image classification Krizhevsky et al. (2017). Deep learning models excel at learning hierarchical features from data, allowing them to capture complex and high-dimensional mappings between inputs and outputs LeCun et al. (2015). The rise of deep learning can be attributed to several interrelated factors that have led to its success across various applications, such as computer vision, natural language processing, and speech recognition. Firstly, the availability of large-scale datasets, like ImageNet for image classification, has provided the necessary data to train complex models effectively. Secondly, advancements in computational power, specifically the development of powerful GPUs and specialized hardware accelerators, have facilitated the training of larger and deeper neural networks, it is possible to harness the potential of these powerful models for pose estimation and relocalization tasks, resulting in improved accuracy, robustness, and efficiency Gallego et al. (2020). Processing event-based data with deep learning models presents, however, unique challenges, as the data is inherently sparse, asynchronous, and non-uniform Gallego et al. (2020). Therefore, novel network architectures and preprocessing techniques must be developed to effectively handle the unique characteristics of event data. Several recent works have demonstrated the feasibility of using deep learning for event camera-based tasks, such as optical flow estimation Zhu et al. (2018), feature extraction Lagorce et al. (2016), and object recognition Gehrig et al. (2019). These studies provide a foundation for further exploration and development of deep learning-based approaches for event camera pose estimation and relocalization tasks.

1.1 Problem Statement and Research Questions

This thesis aims at investigating the application of deep learning methods to event camera-based pose estimation and relocalization tasks, leveraging the unique properties of event cameras, such as high temporal resolution, low latency, and high dynamic range. The primary research questions

addressed in this thesis are:

Why should deep learning be used to process event-based data for pose estimation and relocalization tasks? To answer this question, in this thesis, we will explore various network architectures, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), tailored to handle the unique characteristics of event data, including its asynchronous and sparse nature. This involves the development of novel layers and operations that can efficiently process event data and its corresponding spatial and temporal information. **What are the challenges and limitations associated with applying deep learning techniques to event camera data, and how can they be mitigated?** This research question aims to identify the specific challenges that arise when applying deep learning methods to event data, such as data representation, network architecture design, and training strategies. Possible solutions to these challenges include novel preprocessing techniques that convert event data into a more suitable format for deep learning, the development of network architectures that can better capture the temporal dynamics of event data, and the investigation of training techniques that can improve model convergence and generalization. **How do the proposed deep learning-based methods compare with traditional, non-learning-based approaches for event camera-based pose estimation and relocalization?** To evaluate the performance of the proposed deep learning-based methods, in this thesis we will compare them with existing non-learning-based approaches for event camera pose estimation and relocalization. This involve conducting experiments on benchmark datasets and measuring relevant performance metrics, such as accuracy, robustness, and computational efficiency. Additionally, in this thesis we will analyze the strengths and weaknesses of the proposed methods in various scenarios, such as different lighting conditions, motion speeds, and environmental complexities.

1.2 Contribution and thesis outline

The thesis makes several contributions to the field of event camera pose estimation using deep learning techniques. These contributions can be summarized as follows:

- The thesis provides a comprehensive overview of the background information and related works, establishing a solid foundation and contextual understanding of event camera pose estimation.
- The thesis explores and develops specialized deep learning approaches tailored for event camera pose estimation. These techniques leverage the power of deep learning to accurately estimate the camera's pose using event data.
- The thesis introduces methods to map event data to image-like data, facilitating the application of dedicated deep learning approaches. This mapping process allows for effective utilization of event information in the camera pose estimation task.
- The thesis proposes a novel approach that directly applies deep learning techniques to raw event data, treating it as a point cloud instead of converting it into images. This approach leverages the complete information captured by the event camera and enables an end-to-end learning process.
- The proposed methods and techniques in the thesis aim to achieve highly accurate camera pose estimation results. By effectively leveraging the unique characteristics of event data and employing advanced deep learning models, the thesis contributes to improving the precision of camera pose estimation in event-based vision systems.

The remainder of the manuscript is structured into the following five chapters;

- Chapter 2 offers a comprehensive overview of the background and the related works to the topic. It sets the foundation for the subse-

quent chapters by providing a contextual understanding of the subject matter.

- In Chapter 3, we outline our contributions to event camera pose estimation by leveraging deep learning techniques. Our proposed methods involve mapping event data to image-like data and employing specialized deep learning approaches to accurately estimate the camera's pose. In this chapter, we introduce four model architectures for event-based pose estimation. The first model combines a CNN for feature extraction, an LSTM for capturing spatial dependencies, and a dense fully connected layer for 6-DoF pose estimation. Pretrained weights enhance CNN efficiency in extracting features from event-based images. The second model uses two CNNs with bilinear pooling for improved performance. Our third model includes improved bilinear pooling with matrix function normalization, enabling computations across the entire matrix. Lastly, the fourth model employs a self-attention mechanism with a Transformer encoder, creating a context-aware representation for event patches, incorporating information from both the corresponding event and others in the sequence.

  We conducted multiple experiments using diverse evaluation protocols, and the results exhibit great promise compared to state-of-the-art methods. This highlights the superiority of our proposed approaches.

- Chapter 4 focuses on our novel approach of utilizing deep learning techniques directly on raw event data, treating it as a point cloud, rather than converting the events into images. Our objective is to enable an end-to-end learning process that harnesses the complete information captured by the event camera, ultimately achieving accurate camera pose estimation. We conducted a series of experiments using various evaluation protocols, and the results are promising, pointing towards the extensive potential of our approach.

- Chapter 5 this chapter serves as the conclusion of the thesis, summarizing the research's key points and reflecting on its implications for the field.

Publications

1. PointNet For Real Time Pose Estimation With Event Camera (**In preparation**)
2. Camera Pose Estimation Using Spiking Neural Networks (**In preparation**)
3. 6-DOF Pose Estimation for Event Cameras using a Transformer-based Approach Dicta 2023
4. Tabia, A.; Bonardi, F. and Bouchafa, S. Improved Bilinear Pooling For Real Time Pose Event Camera Relocalisation.ICIAP 2023
5. Tabia, A.; Bonardi, F. and Bouchafa, S. Fully Convolutional Neural Network for Event Camera Pose Estimation. VISIGRAPP 2023
6. A. Tabia, F. Bonardi and S. Bouchafa, Deep Learning For Pose Estimation From Event Camera, 2022 International Conference on Digital Image Computing: Techniques and Applications (DICTA), Sydney, Australia, 2022
7. Tabia, A.; Bonardi, F. and Bouchafa, S. Bilinear Pooling For Event Camera Pose Estimation. ORASIS 2023

Chapter 2

Background and Related Works

Camera pose estimation is a fundamental task in computer vision, with applications spanning augmented reality, robotics, and autonomous systems. The accurate determination of a camera's position and orientation in space is crucial for tasks such as object localization, scene reconstruction, and navigation. Traditional camera pose estimation methods, designed for standard frame-based cameras, have seen significant advancements, yet they face challenges in scenarios with poor lighting, fast motion, or low-textured environments.

In recent years, the emergence of event-based cameras has sparked new avenues for camera pose estimation. Event cameras, inspired by the human retina, operate on a principle of capturing pixel-level intensity changes asynchronously, offering distinct advantages over traditional cameras, such as high temporal resolution, low latency, and wide dynamic range. These unique characteristics make event cameras particularly well-suited for applications where traditional cameras struggle, such as in high-speed motion tracking, low-light conditions, and dynamic environments.

In this chapter, we provide a comprehensive overview of the background and related works to the topic of camera pose estimation, focusing specifically on event camera pose estimation. We begin by discussing the diverse application domains where camera pose estimation plays a pivotal role, including augmented reality, robotic navigation, and autonomous ve-

hicles. Next, we delve into the representation of camera pose and evaluation metrics commonly used to assess the accuracy of pose estimation algorithms. Traditional methods for standard camera pose estimation are then examined, covering geometric methods, feature-based approaches, and template-based techniques. Geometric methods rely on geometric constraints and correspondences between 2D-3D points, while feature-based methods use distinctive image features for pose estimation. Template-based methods match predefined templates to input images to estimate pose. Additionally, we discuss learning-based methods, which leverage machine learning algorithms to learn the mapping from image features to camera poses.

However, the limitations of traditional methods in handling challenging scenarios motivate the exploration of event camera pose estimation. Toward event camera pose estimation, we highlight the advantages of event cameras in addressing issues like fast motion, low-textured scenes, and dynamic lighting conditions. Despite the potential benefits, the field of event camera pose estimation is relatively nascent, with few methods focusing on this specific camera type. This chapter sets the stage for our research by outlining the motivations for using event cameras and the sparse existing literature on event camera analysis using deep learning-based methods.

2.1 Application domains of camera pose estimation

Camera pose estimation with 6 degrees of freedom (6DoF) has a wide range of application domains across various industries and fields. Here are some prominent areas where 6DoF camera pose estimation is utilized:

- Augmented Reality (AR) and Virtual Reality (VR):(see Figure2.1e) In AR and VR applications, accurate camera pose estimation is crucial for seamlessly integrating virtual objects or information into the real-world environment Munoz-Montoya et al. (2018). This enables im-

mersive experiences where virtual elements appear to interact realistically with the physical world.

- Robotics and Automation: Camera pose estimation is essential for robot navigation, manipulation, and interaction with the environment. (see Figure2.1b) Robots can use the estimated camera pose to understand their own position and orientation relative to objects and obstacles, enabling tasks such as pick-and-place operations, assembly, and autonomous exploration Yousif et al. (2015).

- Autonomous Vehicles: Camera pose estimation plays a significant role in the navigation and perception systems of autonomous vehicles. (see Figure2.1c) It helps vehicles understand their position on the road, detect lane markings, recognize traffic signs, and identify other vehicles and pedestrians Scaramuzza and Fraundorfer (2011).

- Industrial Inspection and Quality Control: Camera pose estimation is used in industrial settings for tasks such as inspecting products on an assembly line, verifying correct assembly, detecting defects, and ensuring quality control Zhang et al. (2014).

- Aerial and Satellite Imaging: In aerial and satellite imaging applications, accurate camera pose estimation is important for creating precise maps, performing remote sensing, and monitoring changes in landscapes and urban areas over time Singh et al. (2015).

- Medical Imaging: Camera pose estimation can be used in medical imaging for procedures such as image-guided surgery and diagnostic imaging. It helps align preoperative images with the patient's actual anatomy during surgery Azizian et al. (2014).

- Cultural Heritage Preservation: Camera pose estimation is employed in digitization efforts for preserving cultural heritage sites, artifacts, and artworks in 3D models. It allows accurate capturing of object geometry and texture Di Angelo et al. (2022).

- Architectural and Civil Engineering: In architectural and civil engineering projects, camera pose estimation aids in creating accurate 3D models of buildings, structures, and construction sites. It can assist in design, planning, and monitoring progress Whelan et al. (2016).
- Geographic Information Systems (GIS): Camera pose estimation contributes to GIS applications by enabling the creation of accurate georeferenced maps, 3D models of landscapes, and environmental monitoring Milosavljević et al. (2010).
- Entertainment and Gaming: Camera pose estimation is used in interactive entertainment and gaming applications, where it helps track user movements and gestures for controlling characters or objects within virtual environments Mehta et al. (2017).
- Surveillance and Security: (see Figure2.1a In surveillance systems, camera pose estimation assists in tracking and analyzing the movement of people and objects within monitored areas, enhancing security and situational awareness Patel et al. (2022).
- Human-Computer Interaction: Camera pose estimation is employed in human-computer interaction scenarios, such as gesture recognition and hand tracking, allowing users to interact with computers and devices using natural movements Chen et al. (2020).

These application domains illustrate the broad and versatile impact of 6DoF camera pose estimation across different industries, enabling a wide range of advanced technologies and solutions.

2.2 Pose representation and evaluation metrics

In our exploration of comprehending camera pose, our initial emphasis is placed on delineating the problem domain that involves the components of images, cameras, the representation of pose, and the metrics adopted for assessment. With a given image designated as I_C and emanating from

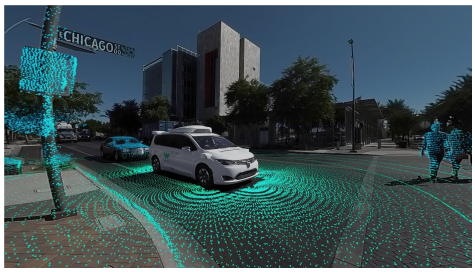
(a) Drones, or general UAVs-Based Object Tracking and 3D Pose Estimation for Smart Surveillance Systems, Zhang et al. (2019)

(b) Boston Dynamics Expands Robot Spot

https://bostondynamics.com/

(c) An autonomous lawn mower developed by Bosch

https://bosch.com/



(d) An autonomous car developed by Waymo

https://waymo.com/

(e) An Augmented Reality display developed by Microsoft

https://microsoft.com/

(f) A Virtual Reality display developed by Oculus

https://oculus.com/

Figure 2.1: Motivation: Examples of commonly found practical applications involving camera pose relocalisation.

a camera designated as C, a sequence of methodologies is enacted upon this image to derive the 6 degrees of freedom (6DoF) coordinates. These coordinates assume the role of a comprehensive depiction, encapsulating both the spatial placement and orientation of the image within the three-dimensional space.

Directing our focus toward the task of visual localization, we delve into the pursuit of determining the pose within a familiar context. This endeavor involves a process wherein a query object image is juxtaposed with a model trained on a dataset containing diverse object images. The result of this comparative analysis culminates in the successful inference of the desired pose.

2.2.1 Camera pose

Camera pose refers to defining both the position and orientation of a camera within a world coordinate system, accounting for six degrees of freedom (6DoF), and is represented using various methods, such as a transformation matrix. These 6DoF are divided into two main groups: translations, encompassing linear, horizontal, and vertical movements, and rotations, which include pitch, yaw, and roll known also as Euler angles. Moreover, camera pose estimation extends to determining the positions of objects within a given scene or scenario as viewed by the camera. In this thesis, we only focus on estimation of the position and the orientation of the camera within the world coordinate system.

In the context of traditional cameras, the inputs to the system originate from camera images, encompassing possibilities like RGB and/or depth images. These images could be singular or in sequences, stemming from stationary or moving cameras. The ultimate outcome sought from the system is the determination of the camera's 6DoF pose. Alongside this, there could exist intermediary stage outputs.

2.2.2 Pose representation

Camera pose estimation refers to predicting both the position and orientation of a camera within a world coordinate system, considering six degrees of freedom (6DoF). The camera pose, p, could be represented under various forms. One typical representation is using a transformation matrix that combines translations and rotations. The 6DoF can be divided into two main groups:

1. Translations (t_x, t_y, t_z): These parameters represent the linear movements of the camera in the horizontal (x), vertical (y), and depth (z) directions, respectively.

2. Rotations ($\theta_x, \theta_y, \theta_z$): These parameters correspond to the rotations of the camera around its axes, including pitch (θ_x), yaw (θ_y), and roll (θ_z).

The camera pose matrix, p, can be defined as follows:

$$p = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.1}$$

Where:

- R is a 3x3 rotation matrix representing the rotation of the camera.
- $\mathbf{t}$ is a 3x1 translation vector containing the camera's position in the world coordinate system.
- $\mathbf{0}$ is a 1x3 zero vector.
- The bottom row is fixed as $[0, 0, 0, 1]$ to maintain homogeneity.

The rotation matrix R can be constructed from the individual pitch, yaw, and roll angles as follows:

$$R = R_z(\theta_z) \cdot R_y(\theta_y) \cdot R_x(\theta_x) \tag{2.2}$$

Where:

$$R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) \\ 0 & \sin(\theta_x) & \cos(\theta_x) \end{bmatrix}$$

$$R_y(\theta_y) = \begin{bmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) \\ 0 & 1 & 0 \\ -\sin(\theta_y) & 0 & \cos(\theta_y) \end{bmatrix}$$

$$R_z(\theta_z) = \begin{bmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The translation vector $\mathbf{t}$ represents the camera's position relative to the world origin:

$$\mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{2.3}$$

Note that this is not the only representation used for expressing changes in position and orientation, encompassing translation and rotation. Concerning orientation, three distinct forms are commonly employed for this purpose: a 3×3 rotation matrix R (as described above), a quaternion comprising four components, and Euler angles (yaw, pitch, roll) given by $(\theta_x, \theta_y, \theta_z)$. These formats can be interchangeably converted to depict the same rotation.

Formally, given Euler angles: $(\theta_x, \theta_y, \theta_z)$, we can compute the quaternion in two steps:

1. Calculate the half angles:

$$\begin{aligned} c_x &= \cos\left(\frac{\theta_x}{2}\right) \\ c_y &= \cos\left(\frac{\theta_y}{2}\right) \\ c_z &= \cos\left(\frac{\theta_z}{2}\right) \\ s_x &= \sin\left(\frac{\theta_x}{2}\right) \\ s_y &= \sin\left(\frac{\theta_y}{2}\right) \\ s_z &= \sin\left(\frac{\theta_z}{2}\right) \end{aligned} \tag{2.4}$$

2. Calculate the quaternion components:

$$\begin{aligned} W &= c_x \cdot c_y \cdot c_z + s_x \cdot s_y \cdot s_z \\ P &= s_x \cdot c_y \cdot c_z - c_x \cdot s_y \cdot s_z \\ Q &= c_x \cdot s_y \cdot c_z + s_x \cdot c_y \cdot s_z \\ R &= c_x \cdot c_y \cdot s_z - s_x \cdot s_y \cdot c_z \end{aligned} \tag{2.5}$$

After calculating these values, we get the quaternion $q = [W, P, Q, R]$ representing the orientation in quaternion form.

Typically, the representation of position and orientation is realized separately. In this thesis, the focus is solely on estimating the position (t_x, t_y, t_z) and the orientation represented as ($\theta_x, \theta_y, \theta_z$) or ($q = [W, P, Q, R]$) of the camera within the world coordinate system. We denote the ground truth camera pose vector by p and the estimated pose vector $\hat{p}$. $\hat{p}$ are constructed by combining the elements of translation and rotation.

This estimation is crucial for various applications, such as augmented reality, robotic navigation, and scene reconstruction, where knowing the precise camera pose enables accurate localization of objects within the scene.

2.2.3 Evaluation metrics

The assessment strategies for localization tasks undergo adaptation based on the specific metrics under consideration and the particular methods employed for localization. In the context of evaluating the efficacy of camera pose estimation methods, a critical step involves juxtaposing the computed pose derived from the estimation technique against the authentic ground truth pose. This comparative analysis serves as a gauge for determining the proximity of the estimated outcome to the actual ground truth.

In datasets where ground truth poses are readily available, the precision of a given method's poses is evaluated by quantifying the disparity between the estimated and actual ground truth poses. Within the realm of direct localization methodologies, two primary error metrics hold prominence: the absolute pose error (APE) and the relative pose error (RPE).

APE serves as a particularly fitting measure for assessing the performance of visual Simultaneous Localization and Mapping (SLAM) systems. It provides an effective means of gauging how accurately the estimated poses align with the true ground truth poses.

In contrast, RPE finds its utility in evaluating the drift experienced by a visual odometry system. This metric is well-suited for quantifying drift over specific time intervals, such as drift per second. It offers valuable insights into the cumulative deviations that may occur over continuous motion sequences.

Absolute pose error (APE) becomes particularly relevant when the algorithm's input consists of a single image. APE is quantified through the amalgamation of two distinct components: absolute position error and orientation error.

For the determination of position error, the metric involves computing the Euclidean distance, measured in meters, between the estimated position (represented as $\hat{X}$ and the actual ground truth position (denoted as X). This measurement effectively captures the spatial disparity between the estimated and true positions.

$$APE = \sqrt{(\hat{x}_{\text{est}} - x_{\text{gt}})^2 + (\hat{y}_{\text{est}} - y_{\text{gt}})^2 + (\hat{z}_{\text{est}} - z_{\text{gt}})^2} + \theta_{\text{error}}. \tag{2.6}$$

Where: $(\hat{x}_{\text{est}}, \hat{y}_{\text{est}}, \hat{z}_{\text{est}})$ are the estimated position coordinates. $(x_{\text{gt}}, y_{\text{gt}}, z_{\text{gt}})$ are the ground truth position coordinates. θ_{error} is the orientation error, usually represented in terms of an angular difference (e.g., quaternion or Euler angles).

In essence, APE encapsulates both the position and orientation discrepancies, thereby furnishing a comprehensive evaluation of the accuracy in estimating the pose from a single image.

Relative Pose Error (RPE) is particularly relevant when the algorithm is provided with image pairs in the form of a time-series derived from sequential images. Much like Absolute Pose Error, RPE is evaluated through the amalgamation of relative position error and orientation error.

For the assessment of relative position error, the metric involves calculating the speed of Euclidean distance shift, measured in meters per second (m/s), between the estimated relative position (denoted as $\hat{x}_{\text{rel}}$) and the actual ground truth relative position (x_{rel}). This measurement effectively captures the rate of change in the relative position.

Simultaneously, orientation error is quantified by determining the minimum angle deviation rate in degrees per second (degree/s) between the estimated relative quaternion ($\hat{q}_{\text{rel}}$) and the ground truth relative orientation (q_{rel}). This assessment employs the quaternion representation to encapsulate the angular discrepancy in orientation.

Fixed Thresholds Error is an alternative metric utilized for assessing the performance of indirect camera pose methods. This metric quantifies the proportion of images that fall within predefined error thresholds concerning both position and orientation. In essence, it measures the percentage of images that achieve accurate registration based on specific localization criteria.

For instance, this metric may evaluate the localization performance by considering the proportion of images whose estimated poses are within a

certain distance threshold (X meters) and angular threshold (Y degrees) of their respective ground truth poses. It provides a clear and practical evaluation of the algorithm's effectiveness in achieving accurate pose estimation within predefined tolerance levels.

Median and Average Error

To rigorously evaluate the performance of pose estimation algorithms, the provided code outlines a comprehensive criterion. The assessment focuses on two key metrics: position error and orientation error protocols that reported in Nguyen et al. (2019) and used in PoseNet Kendall et al. (2015) and Bayesian PoseNet Kendall and Cipolla (2016).

- **Position Error:** The spatial accuracy of the estimated pose is gauged by comparing the predicted 3D position to the true position. Mathematically, if $P_{\text{pred}} = [x_{\text{pred}}, y_{\text{pred}}, z_{\text{pred}}]$ is the predicted position and $P_{\text{true}} = [x_{\text{true}}, y_{\text{true}}, z_{\text{true}}]$ is the true position, the position error, E_{position}, is computed using the Euclidean distance formula:

$$E_{\text{position}} = \sqrt{(x_{\text{pred}} - x_{\text{true}})^2 + (y_{\text{pred}} - y_{\text{true}})^2 + (z_{\text{pred}} - z_{\text{true}})^2} \quad (2.7)$$

- **Orientation Error:** The orientation, represented by quaternions, sheds light on the rotational accuracy. Before any computation, the quaternions are normalized to ensure they correspond to valid rotations. If q_{pred} and q_{true} are the predicted and true normalized quaternions, respectively, the dot product of these two quaternions gives a measure of their alignment. The orientation error, θ, in degrees, is derived from:

$$d = |q_{\text{pred}} \cdot q_{\text{true}}| \quad (2.8)$$

$$\theta = 2 \times \arccos(d) \times \frac{180}{\pi}$$

  Here, θ provides a measure of the angular difference between the predicted and true orientations.

- **Median Position Error** $= \text{median}(E_{\text{position}_1}, E_{\textbf{position}_2}, \ldots, E_{\text{position}_n})$

- **Median Orientation Error** $= \text{median}(\theta_1, \theta_2, \ldots, \theta_n)$
- **Average Position Error** $= \frac{1}{n} \sum_{i=1}^{n} E_{\text{position}_i}$
- **Average Orientation Error** $= \frac{1}{n} \sum_{i=1}^{n} \theta_i$

Where:

- E_{position_i} is the position error for the i-th sample.
- θ_i is the orientation error for the i-th sample.
- n is the total number of samples.

As quantitative evaluation, we choose to calculate the median and average error of the predicted pose in position and orientation. The Euclidean distance is used to compare the predicted position to the groundtruth, and the anticipated orientation is normalized to unit length before being compared to the groundtruth. For location and orientation, the median and average error are recorded in m and deg(°), respectively.

2.3 Traditional camera pose estimation

2.3.1 Geometric Methods

In the realm of computer vision, geometric methods stand out as a pivotal set of techniques grounded in mathematical and geometric principles, tasked with deducing an object's pose its spatial positioning and orientation within a three-dimensional space. Esteemed for their theoretical integrity and direct interpretability, these methods address several core issues, among which the Perspective-n-Point (PnP) Fischler and Bolles (1981) challenge is prominent, a problem extensively explored by Fischler and Bolles in their seminal 1981 work.

The essence of the PnP problem lies in determining a camera's pose with the knowledge of n three-dimensional points in the world and their

corresponding two-dimensional projections on the camera's imaging sensor. Here, the term n signifies the quantity of point correspondences bridging the three-dimensional reality and the two-dimensional image plane Figure 2.3. In essence, if the positions of specific points in a three-dimensional space are known and their appearances on a two-dimensional image are identifiable, solving the PnP problem entails pinpointing the camera's precise location in three-dimensional space to capture the points in the observed manner on the image plane.

Among the spectrum of solutions to this quandary, the Random Sample Consensus (RANSAC) method proposed by Fischler and Bolles emerges as notably distinguished. RANSAC, an iterative approach, serves to estimate the parameters of a mathematical model from a dataset potentially laden with outliers. This process involves the repetitive selection of a random data subset, model fitting to this subset, and subsequent exclusion of ill-fitting data points. Consequently, the method yields a model that accurately represents the majority of the data, unaffected by the presence of outliers. Geometric methodologies, especially those tackling the PnP problem, are celebrated for their established mathematical properties and comprehensive understanding. Nonetheless, they often demand meticulous calibration and exhibit sensitivity to data noise and outliers, underscoring the necessity for robust techniques like RANSAC. Furthermore, these approaches typically assume a rigid transformation between the three-dimensional points and their two-dimensional image projections, an assumption that may not universally hold in practical scenarios. While traditional methods may falter in complex environments such as dynamic scenes or those featuring non-rigid deformations emerging strategies, notably deep learning, have demonstrated significant promise in overcoming these challenges.

To illuminate the mathematical underpinnings of the PnP dilemma, consider the following formulation: Given a set of n corresponding points, where $\mathbf{P}_i$ represents a point in three-dimensional space and $\mathbf{p}_i$ its projection on the two-dimensional image, the objective is to ascertain the rota-

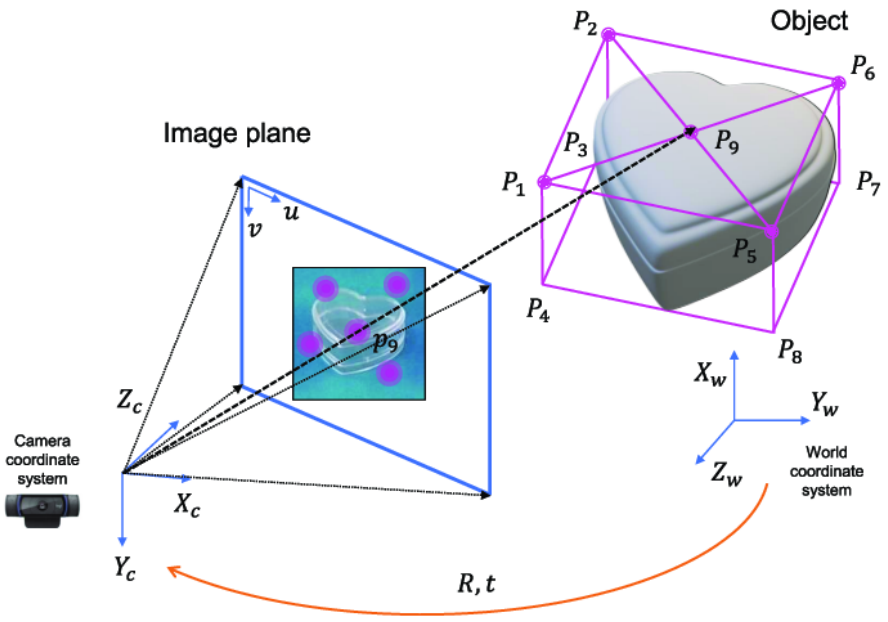


Figure 2.2: 6D pose estimation using Perspective-n-Point. The network outputs nine keypoints in the image space Byambaa et al. (2022).

tion $\mathbf{R}$ and translation $\mathbf{t}$ that minimize the re-projection error:

$$\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{n} \|\mathbf{p}_i - \text{proj}(\mathbf{R}\mathbf{P}_i + \mathbf{t})\|^2 \tag{2.9}$$

Here, proj denotes the projection operation from three dimensions to two, typically encompassing intrinsic camera parameters. This optimization challenge seeks to identify the pose (rotation and translation) that optimally aligns the three-dimensional points with their two-dimensional projections across the image plane, thereby resolving the PnP conundrum.

2.3.2 Feature-based Methods

Feature-based methods are an important category of techniques in the computer vision domain, widely used for pose estimation tasks. The central idea behind these methods is to detect distinctive features or keypoints in an image, describe these features using robust descriptors, and then

use the matching features across different images to infer the pose Lowe (2004).

One of the most popular feature descriptors is the Scale-Invariant Feature Transform (SIFT), introduced by Lowe in 2004. The SIFT algorithm can be summarized in four main steps: scale-space peak selection, keypoint localization, orientation assignment, and keypoint descriptor. Mathematically figure 2.3, the scale-space peak selection is performed using the difference-of-Gaussians function that identifies potential interest points that are invariant to scale and orientation:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{2.10}$$

where $L(x, y, \sigma)$ is the scale-space function, and $D(x, y, \sigma)$ represents the difference-of-Gaussians. This step is followed by keypoint localization, where keypoints are refined to achieve higher accuracy. The orientation assignment ensures that each keypoint has a consistent orientation based on local image properties, enhancing invariance to image rotation.

The final step involves creating a keypoint descriptor from the local image gradient directions, ensuring robustness to affine distortion and changes in illumination.

Another widely used descriptor is Speeded Up Robust Features (SURF), which provides a faster alternative to SIFT. SURF utilizes integral images for efficient image convolutions and employs a blob detector based on the Hessian matrix for interest point detection:

$$H(x, y, \sigma) = \begin{bmatrix} L_{xx}(x, y, \sigma) & L_{xy}(x, y, \sigma) \\ L_{xy}(x, y, \sigma) & L_{yy}(x, y, \sigma) \end{bmatrix} \tag{2.11}$$

where $L_{xx}(x, y, \sigma)$, $L_{xy}(x, y, \sigma)$, and $L_{yy}(x, y, \sigma)$ are the second-order derivatives of the Gaussian-blurred image, which constitute the Hessian matrix $H(x, y, \sigma)$ at scale σ.

The Oriented FAST and Rotated BRIEF (ORB) algorithm offers a robust, binary descriptor that is both faster and less memory-intensive than SIFT and SURF. ORB combines the FAST keypoint detector and the BRIEF de-

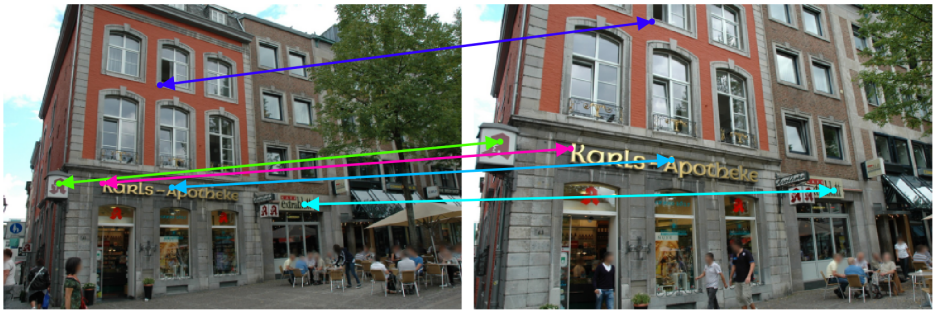


Figure 2.3: Keypoint matching
Given a pair of partially covisible images, the goal of keypoint matching is to find a set of 2D-to-2D matches across both images. The accuracy of such correspondences is critical for downstream computer vision tasks like visual localization. Here showen a few putative correspondences using SIFT (Lowe (2004). SIFT detections are matched using a mutual nearest-neighbour algorithm on their respective SIFT descriptors and filtered using a ratio test

scriptor with modifications for orientation invariance and noise resistance.

After feature extraction and description, the subsequent stage involves establishing correspondences between features in different images by matching descriptors based on their Euclidean distance in the descriptor space. This process can utilize various strategies, including brute-force matching or FLANN-based matching for efficiency.

With established feature correspondences, these can be utilized to estimate the relative pose between images, applying techniques like the PnP problem, Essential Matrix, or Homography, depending on the specific requirements of the scene.

In summary, feature-based methods offer a robust framework for pose estimation in environments rich in distinctive features. However, challenges arise in situations with repetitive patterns, textureless surfaces, or severe occlusions. Moreover, traditional descriptors such as SIFT, SURF, and ORB may struggle with significant lighting condition variations. Recent developments in learning-based approaches, particularly deep learning, are making strides towards overcoming these limitations by directly learning more invariant and discriminative features from data.

2.3.3 Template-based Methods

Template-based methods for pose estimation fundamentally rely on predefined templates of the subject or scene whose pose needs to be deduced. These templates can be either 2D or 3D, representing a standard or known perspective of the subject Hinterstoisser et al. (2013). The primary idea behind these techniques is to establish a correlation between the observed subject or scene and the predefined template, generating an estimate of the subject's pose. The estimation's quality heavily depends on the precision and representativeness of the templates. There are several ways to match the observed object to the template:

Edge Matching

This technique involves aligning the edges in the observed subject's image with the template's edges. Edge detection algorithms such as Sobel, Canny, or Laplacian are typically employed to detect an object's edges in an image. Once the edges are identified, a similarity measure, usually through a distance function, is calculated between the detected edges and those in the template Canny (1986).

Sobel Operator

The Sobel operator is used for edge detection by computing the gradient magnitude of an image. It applies two 3x3 kernels, one estimating the gradient in the x-direction (G_x) and the other in the y-direction (G_y). The gradient magnitude (G) at each pixel is calculated as:

$$G = \sqrt{G_x^2 + G_y^2} \tag{2.12}$$

Canny Edge Detector

The Canny edge detector 2.4, introduced by John Canny in 1986, is a multistage algorithm designed to detect a wide range of edges in images. It includes the following steps:

1. Noise Reduction: A Gaussian filter is applied to smooth the image and reduce noise.

2. Gradient Calculation: The gradient magnitude and direction are calculated to identify the edge strength and orientation.

3. Non-maximum Suppression: Thins edges to 1-pixel lines by suppressing all except the maximum gradient magnitudes along the edge direction.

4. Double Thresholding: Determines potential edges by applying two thresholds, a low and a high one.

5. Edge Tracking by Hysteresis: Finalizes the detection of edges by suppressing weak edges not connected to strong edges.

The gradient magnitude (G) and direction (θ) are given by:

$$G = \sqrt{G_x^2 + G_y^2} \tag{2.13}$$

$$\theta = \arctan\left(\frac{G_y}{G_x}\right) \tag{2.14}$$

where G_x and G_y are the gradients in the x and y directions, respectively.

Similarity Measure

After edge detection, the next step is to calculate a similarity measure between the detected edges in the observed image and those in the template. This often involves a distance function, such as the Euclidean distance for simplicity, or more complex measures that can account for geometric transformations. The similarity measure enables the alignment of the observed image's edges with the template, facilitating accurate pose estimation.

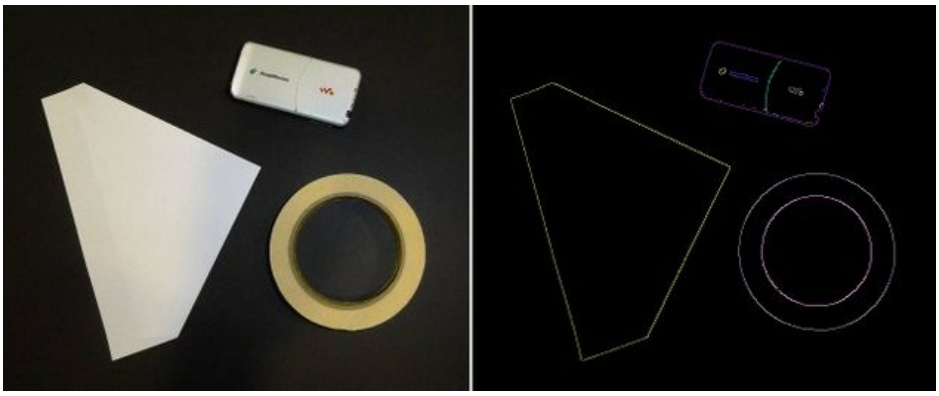


Figure 2.4: Edge or contour detection
is a basic computer vision problem. The Canny edge detector is a popular algorithm for detecting edges in an image which uses hystersis thresholding.

Distance Function

The distance between two sets of edges, A from the observed image and B from the template, can be quantified as:

$$D(A, B) = \sum_{a \in A} \min_{b \in B} \|a - b\| \tag{2.15}$$

This function calculates the minimum distance from each edge point in set A to the closest edge point in set B, summing these minimum distances to give a measure of how well the edges from the observed image align with those of the template.

Edge Matching is a foundational technique in computer vision for pose estimation, leveraging the precision of edge detection algorithms like the Canny edge detector and employing similarity measures to align detected edges with template edges. This method's effectiveness hinges on the robust detection of edges and the accurate calculation of similarities, enabling precise estimation of the pose of objects within images.

Gradient Matching

In gradient matching, the gradients – defined as the rates of intensity changes in pixel values – of the observed image and the template are compared. This method is predicated on the principle that object surfaces sharing the same orientation ought to exhibit similar gradients. Formally, if $I_o(x, y)$ and $I_t(x, y)$ represent the intensity values of the observed image and the template at coordinates (x, y), respectively, and $\nabla I_o(x, y)$ and $\nabla I_t(x, y)$ denote their gradients, then the similarity measure S between the two can be defined as:

$$S = \sum_{x,y} \cos(\theta_o(x, y) - \theta_t(x, y)) \tag{2.16}$$

where $\theta_o(x, y) = \arctan\left(\frac{\partial I_o/\partial y}{\partial I_o/\partial x}\right)$ and $\theta_t(x, y) = \arctan\left(\frac{\partial I_t/\partial y}{\partial I_t/\partial x}\right)$ are the orientations of the gradients at (x, y) in the observed image and the template, respectively. The gradients, $\nabla I_o(x, y)$ and $\nabla I_t(x, y)$, are computed using operators such as Sobel or Prewitt.

This approach is notably robust to changes in illumination, as it focuses on the relative changes in pixel intensities rather than their absolute values Hinterstoisser et al. (2011) 2.5. The effectiveness of gradient matching is thus significantly enhanced in scenarios where illumination conditions are variable or difficult to control.

Feature Matching

This technique involves the identification and matching of local features, such as corners, blobs, and ridges, detected in the observed image with those present in the template. The process leverages commonly used feature detectors including Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), and Oriented FAST and Rotated BRIEF (ORB), among others. The mathematical foundation of feature matching can be exemplified by the descriptor matching process used in SIFT, which is defined as follows:

Figure 2.5: A method can detect texture-less 3D objects in real-time under different poses over heavily cluttered background using gradient orientation.Hinterstoisser et al. (2011)

Given a set of keypoints and their descriptors in both the observed image I_o and the template I_t, the goal is to find pairs of keypoints (k_o, k_t) such that their descriptors $D(k_o)$ and $D(k_t)$ minimize the Euclidean distance between them, under certain threshold to ensure robustness against noise and minor variations:

$$\min_{k_o,k_t} ||D(k_o) - D(k_t)||^2, \quad \text{subject to} \quad ||D(k_o) - D(k_t)||^2 < T \tag{2.17}$$

where T is a predefined threshold value.

Feature matching offers significant robustness against variations in scale, rotation, and affine transformations, making it highly effective for pose estimation tasks Lowe (2004) figure 2.6.

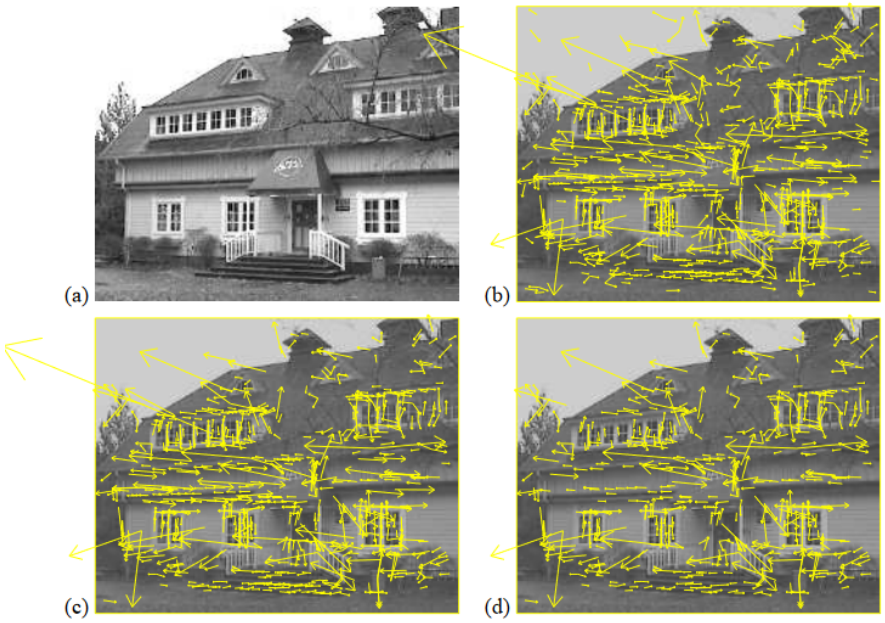


Figure 2.6: The stages of keypoint selection.
(a) The 233x189 pixel original image. (b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 keypoints remain. (d) The final 536 keypoints that remain following an additional threshold on ratio of principal curvatures..Lowe (2004)

Advantages and Limitations: A primary advantage of template-based methods, including feature matching, is their relative simplicity in implementation and conceptual understanding. Moreover, when the templates are accurately designed and closely match the object of interest, these methods can achieve high levels of accuracy. However, challenges arise in scenarios involving substantial variations in view, scale, or illumination, as well as in the presence of occlusion or changes in the object's shape or appearance. These factors can significantly impact the effectiveness of template-based pose estimation methods.

In recent years, some advancements have been made to improve the robustness and flexibility of template-based methods. For instance, learning-based methods have been proposed to learn a more flexible template rep-

resentation. Such methods can handle more variations in appearance and pose than traditional template-based methods.

2.3.4 Learning-based Methods

Learning-based methods for pose estimation, especially those employing machine learning techniques, have significantly evolved over the last decade. While traditional algorithms like Support Vector Machines (SVM) and Random Forests have been utilized in this domain, the advent of deep learning has brought about a substantial transformation.

Support Vector Machines (SVMs)

Support Vector Machines (SVMs) represent a class of supervised learning models used for classification and regression tasks Cortes and Vapnik (1995). In the context of pose estimation, SVMs are utilized to differentiate between various poses by training on labeled data. The core principle behind SVMs is to transform the input data into a higher-dimensional feature space and then identify an optimal hyperplane that separates the data points into different classes, corresponding to different poses in this scenario. The mathematical formulation of an SVM that aims to find this hyperplane can be described as follows:

Given a training dataset of instance-label pairs $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{-1, 1\}$, the SVM solves the following optimization problem:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{m}\xi_i \tag{2.18}$$

subject to

$$y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, m, \tag{2.19}$$

where $\mathbf{w}$ is the normal vector to the hyperplane, b is the bias term, $\phi(\mathbf{x}_i)$ maps $\mathbf{x}_i$ into a higher-dimensional space, C is the penalty parameter, and

ξ_i are slack variables allowing for misclassification of difficult or noisy examples.

SVMs have been effectively applied in numerous pose estimation tasks, demonstrating their capability to handle complex classification problems within this field Smola and Schölkopf (2004). One of the strengths of SVMs is their ability to find a decision boundary that maximizes the margin between different classes in high-dimensional feature spaces. This is particularly useful in pose estimation, where the pose classes may not be linearly separable in the original feature space. The use of kernel functions in SVMs allows for the mapping of input data to a higher-dimensional space where a linear separator can be found, thereby improving classification performance

Random Forests

Random Forests are an ensemble learning technique for classification, regression, and other tasks, which operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (or mean prediction of the trees in the case of regression) Cutler et al. (2012). For pose estimation, random forests aggregate the decisions of multiple decision trees trained on various subsets of the dataset to predict the pose class.

The general approach for using Random Forests in pose estimation involves training multiple decision trees on randomly selected subsets of the training data and features. The prediction of an unknown pose is made by aggregating the predictions of all the trees. This method can be formalized as follows.

Given a set of training data with features and pose labels, a random forest builds N decision trees. Each tree T_i is trained on a random subset of the data and features. For a new sample, each tree provides a pose estimate, and the final pose prediction is obtained by averaging (regression) or majority voting (classification) across all trees.

Random Forests have shown promising results in pose estimation, ben-

efiting from their ability to model complex decision boundaries and resist overfitting by averaging multiple trees' predictions Girshick et al. (2011).

In recent years, the shift towards deep learning-based methods LeCun et al. (2015) has been evident, largely due to the superior performance of these methods in many computer vision tasks, including pose estimation. Deep learning methods like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been extensively used for pose estimation. These techniques leverage the power of large amounts of training data and deep hierarchical feature learning to produce highly accurate and robust pose estimates Litjens et al. (2017).

Deep Learning-based Methods

In recent years, the advent of deep learning technologies has revolutionized the approach to pose estimation. Among these, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) stand out as particularly effective for this task. These methods excel in learning complex mappings directly from data, leading to state-of-the-art performance across a wide range of pose estimation challenges. A notable implementation of deep learning in pose estimation is PoseNet, a method that employs a CNN to directly regress the six degrees of freedom (6-DOF) pose from a single RGB image Kendall et al. (2015).

The mathematical foundation of CNNs, which are pivotal in processing spatial data, can be succinctly represented by the convolution operation integral to their function:

$$F_{ij}^{l} = \sigma\left(\sum_{m}\sum_{n} W_{mn}^{l} \cdot X_{(i+m)(j+n)}^{l-1} + B^{l}\right) \tag{2.20}$$

where F_{ij}^{l} denotes the feature map at layer l, W_{mn}^{l} represents the weights of the convolutional kernel, $X_{(i+m)(j+n)}^{l-1}$ is the input from the previous layer, B^{l} is the bias, and σ is a non-linear activation function.

RNNs, on the other hand, are tailored for sequential data, making them suitable for video-based pose estimation. Their defining characteristic is

the feedback loops allowing information to persist, described by:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{2.21}$$

where h_t is the hidden state at time t, x_t is the input at time t, W_{xh} and W_{hh} are the weights, b_h is the bias, and σ is the activation function.

While deep learning methods have demonstrated unparalleled accuracy in pose estimation, the choice between CNNs, RNNs, and hybrid models like PoseNet often depends on the specific application's requirements, including the nature of the input data (single images vs. image sequences) and the desired precision and robustness of the pose estimation.

Each method presents its own set of advantages and limitations. For instance, CNNs are highly effective for spatial analysis in static images, whereas RNNs offer advantages in temporal sequence analysis, crucial for understanding motion in video data. The integration of geometric considerations in models like PoseNet exemplifies the ongoing evolution of deep learning approaches, blending traditional computer vision techniques with the latest in neural network architectures to push the boundaries of what is achievable in pose estimation tasks.

2.4 Toward event camera pose estimation

Frame-based cameras have their limitations, many of which stem from their inherent design principles. To start, these cameras operate on a fixed dynamic range, typically around 60 dB for most consumer cameras. This limited range can significantly impact the camera's ability to capture detailed images in scenes with stark contrast or varied lighting conditions, like a shadowed area next to a brightly lit window Xiao et al. (2002). Moreover, traditional frame-based cameras work at a predetermined frame rate. While this is acceptable in most scenarios, it can be inefficient when dealing with scenes where little changes over time, leading to unnecessary data and power consumption. Additionally, quick, substantial changes that occur between frames can be missed, hindering the capture of critical informa-

tion. Fast-moving objects can create what is known as motion blur when captured at low frame rates. This blurring effect can impair the performance of visual recognition and tracking algorithms, which rely on sharp, distinct images to function effectively. Figure 2.7 provides an example of this limitation. In low-light conditions, conventional cameras often struggle to capture clear images. Increasing the sensor's sensitivity to address this can unfortunately also increase the amount of noise in the image, thereby decreasing the overall image quality. Energy efficiency is another challenge for conventional cameras. While they are relatively low-power compared to other sensors, they can still consume a significant amount of a robot's power resources, especially in applications where power efficiency is crucial, like space probes or remote monitoring devices. Processing overhead is another concern. Because they capture the entire scene at each frame, frame-based cameras generate a large volume of data. This requires significant computational resources for processing, and can be a challenge for on-board robotic systems where computational power is typically limited.

Finally, conventional cameras are also subject to latency issues. There can be a delay from the time a scene is captured to the time the processed information is available, which can be problematic in applications that require real-time responses, such as autonomous vehicles or drones. Given these limitations, it's essential for researchers to carefully consider the choice of vision sensors for each specific robotic application. These limitations also serve as motivation for the continuous exploration and development of alternative sensing technologies and processing techniques.

2.4.1 Motivation

The birthplace of event cameras lies within the realm of neuromorphic engineering, a field whose ambitious goal is to understand brain function and replicate it on a microchip Gallego et al. (2020). Drawing inspiration from biological retinas, which have been refined through millions of years of

Figure 2.7: Limitations of conventional frame-based cameras from Bardow (2019)

natural selection, event cameras aspire to capitalize on benefits such as minimal power usage, efficient data transmission, and rapid motion detection.

They represent a revolutionary departure from traditional frame-based cameras, potentially addressing issues like inadequate temporal resolution, restricted dynamic range (over/underexposure), and motion blur figure 2.7. At the heart of these cameras is a "smart pixel" that springs into action in response to changes in light intensity, while remaining dormant otherwise. This is akin to the transient ganglion cells found in the retina of a biological eye figure 2.8. Each pixel operates in an asynchronous and independent manner, culminating in a camera that perceives changes in the environment while remaining 'blind' to static scenes. This, in essence, is a dynamic vision sensor.

2.4.2 Event Cameras

Event cameras, inspired by biological vision, capture the dynamic visual information of the world in a fundamentally different way compared to traditional cameras figure 2.8. Instead of outputting a sequence of images at fixed intervals, event cameras continuously record changes in the scene with a high temporal resolution, resulting in an asynchronous stream of events.

Each pixel in an event camera operates independently and only reacts to changes in brightness, thus naturally suppressing redundant information. The pixel stores a reference brightness level and continually com-

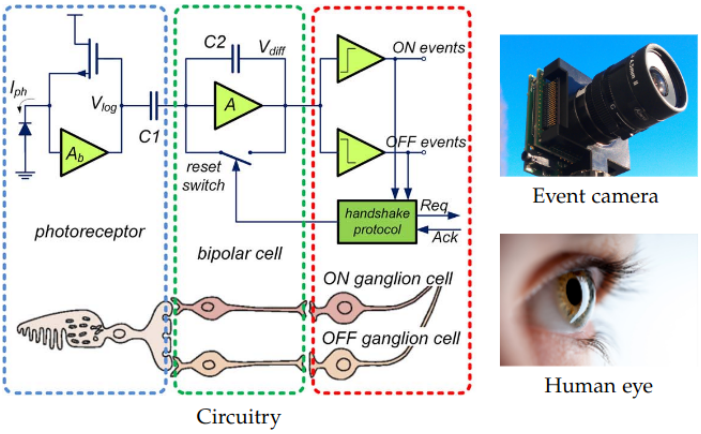


Figure 2.8: Circuitry of an event camera pixel (top) is directly inspired by biological retinas. Adapted from Posch et al. (2014)

pares it with the current level. If the difference exceeds a certain threshold, an event is triggered. This event, encapsulating the x, y coordinates of the pixel, the polarity (increase or decrease) of the brightness change, and a precise timestamp, is immediately transmitted off-chip Lichtsteiner et al. (2008). The data output by event cameras is fundamentally different from traditional video. It is not a series of frames but a temporal stream of asynchronous events, with each event encoding a brightness change at a particular pixel. The data is sparse in space and time because events are only triggered at the pixels where changes occur, and only when these changes occur figure 2.9.

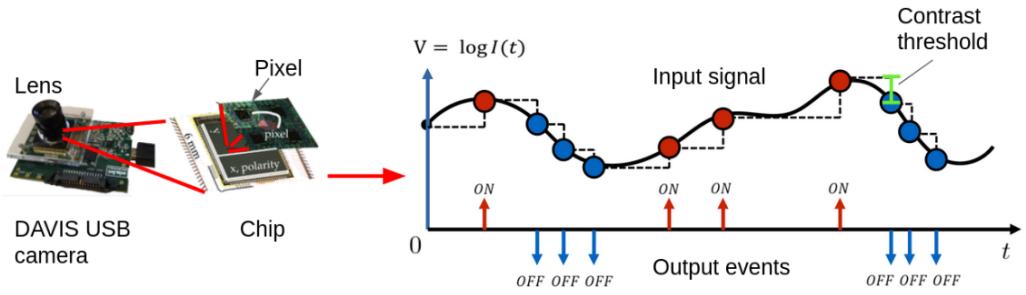


Figure 2.9: An event camera pixel

Event cameras offer several benefits over their traditional counterparts.

They have a high dynamic range (up to 140 dB compared to 60 dB for conventional cameras), which makes them effective in high-contrast environments. They also have a low latency (on the order of microseconds) and low power consumption since only the changing pixels need to report their data Gallego et al. (2020). These attributes make event cameras a promising technology for various applications, including high-speed robotics, autonomous driving, and augmented/virtual reality. Event cameras primarily capture dynamic elements within a scene, representing this information via events that are plotted in a 3-dimensional space-time construct, as seen in figure 2.10. A practical instance would be a rotating disc, which generates a spiral sequence of events within this space-time framework. Additionally, objects in motion incite events at their peripheries.

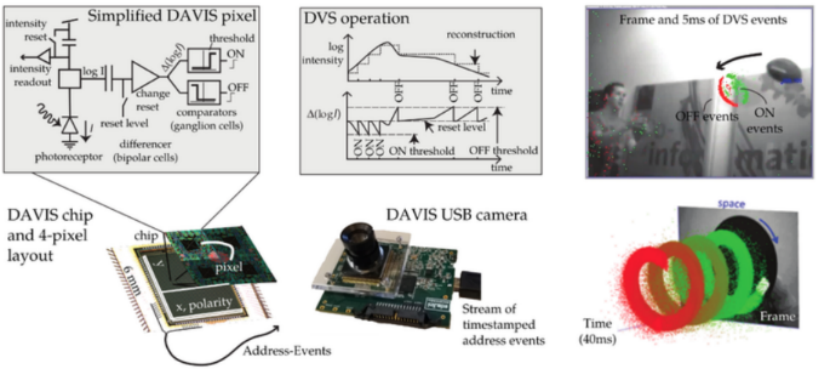


Figure 2.10: The simplified circuitry of a single DAVIS pixel as described by Brandli et al. Brandli et al. (2014), along with the process of ON/OFF event generation given a log intensity signal, is depicted. The bottom illustration shows the DAVIS chip and a DAVIS USB camera. On the right, events are superimposed on an intensity image, with green representing ON events and red indicating OFF events. Additionally, a 3D event point cloud is displayed. This visualization is sourced from Gallego et al. Gallego et al. (2020).

One crucial aspect of event cameras is their inability to register stationary or unchanging parts of a scene. This characteristic partially motivates the incorporation of a conventional APS camera into the DAVIS device to supplement the event data. To be more precise, event cameras fail to detect unchanging scenes, including moving objects that lack texture. How-

ever, these cameras can record static scenes if triggered by certain stimuli, such as a strobe light. Event cameras exhibit several nonidealities and characteristic noise, as characterized by Lichtsteiner et al. Lichtsteiner et al. (2008), Brandli et al. Brandli et al. (2014), and Delbruck et al. Delbruck et al.. These include:

(i) Mismatch of contrast threshold between pixels: The effective contrast threshold has been found to fluctuate spatially across pixels and temporally for a given pixel Lichtsteiner et al. (2008) Brandli et al. (2014). Consequently, direct integration of the event signal is deemed inefficient without noise suppression, necessitating online calibration to offset temporal variations in contrast threshold. A typical event camera has a minimum contrast threshold of approximately 11% change in log-intensity Brandli et al. (2014). However, more sensitive cameras can achieve lower thresholds - as low as 3.5% for ON events and 1% for OFF events Linares-Barranco et al. (2019).

(ii) Bandwidth limitations: The bandwidth of event cameras increases monotonically with light intensity, registering about 3 kHz in bright conditions and 300 Hz at 1000x lower intensity Gallego et al. (2020). High bandwidth sensors such as the Samsung DVS-Gen4 Son et al. and Prophesee Gen 4 CD Finateu et al. (2020) have maximum bandwidths exceeding 1 billion events per second.

(iii) Hot pixels: These refer to pixels that emit many spurious events in rapid succession, either continuously, randomly, or in overreaction to changes in illumination, consuming significant bandwidth. A refractory period following an event firing at each pixel can mitigate the effect of hot pixels and alleviate bus congestion. However, this introduces tracking errors for fast brightness changes by suppressing events that 'should' have fired.

(iv) Random background noise events: Event cameras, such as DAVIS that share circuitry to generate image frames, can generate noise events in approximately 0.25% of pixels under uniform, unchanging illumination during every frame acquisition. This is attributable to unwanted parasitic coupling between APS (frame) and DVS (event) pixel circuitry Brandli et al.

(2014). Random background noise events, typically uncorrelated, result from leakage in the reset transistor in DVS pixel circuitry and can be filtered out by discarding spatio-temporally isolated events Delbruck et al..

However, these advantages also come with unique challenges. The asynchronous, event-based data is not directly compatible with most existing computer vision algorithms, which are designed to work with frames. Therefore, new algorithms and methods are needed to process and interpret event camera data effectively.

2.5 Pose estimation using event cameras

The field of pose estimation using event cameras is rich, innovative, and continues to evolve, bringing together facets of neuroscience, computer vision, robotics, and machine learning. The bio-inspired nature of event cameras presents both challenges and opportunities in developing effective algorithms for pose estimation.

2.5.1 Event-Based Visual Odometry (EVO)

EVO, proposed by Reinbacher et al. Munda et al. (2018), is a method for estimating camera motion from event data. The approach operates on an accumulated event image and employs an image alignment technique for motion estimation. While the method showed promising results, it has difficulty with complex motions due to its reliance on an intensity-like image, which may be affected by noise. EVO works by accumulating events over a certain period of time to generate what is called an "event image." This is similar to a standard intensity image but instead encodes changes in brightness at each pixel location. These event images can then be aligned using image alignment techniques to estimate the camera's motion

2.5.2 Probabilistic Event-based Motion Tracking

This method, proposed by Kim et al. Kim et al. (2008), employs a probabilistic framework to estimate 6-DOF motion from event data. The work demonstrated a fast and accurate motion estimation performance, which is valuable in real-time applications such as robotics. This method builds on the inherent qualities of event cameras their high temporal resolution and ability to respond to changes in the scene to estimate motion. The essence of the approach lies in modeling the motion estimation problem as a Bayesian inference task. Given the data produced by the event camera, this probabilistic method attempts to estimate the most likely motion that would lead to the observed data. Specifically, Kim et al. introduced a novel event likelihood function, which they use to formulate an optimization problem. By solving this optimization problem, the method provides an estimate of the camera's motion. Notably, this approach considers the uncertainty associated with the measurements, as characterized by the probabilistic framework. This enables it to deliver robust performance, even under noisy conditions

2.5.3 Event-based Pose Tracking using a DAVIS (Dynamic and Active-pixel Vision Sensor)

In the work by Kueng et al. Kueng et al. (2016), the researchers used a DAVIS sensor, which can output both events and standard frames, to perform pose tracking. Their method combined information from both the events and frames to achieve accurate pose estimation. The DAVIS sensor is unique in its capacity to output both event data and standard image frames. The core idea of their method is to exploit the complementary nature of these two types of data. Traditional frames offer holistic scene information at fixed time intervals, while event data provides high temporal resolution updates on pixel-level brightness changes. By combining the data from both streams, their method aims to overcome the limitations of either type of data alone and achieve accurate pose estimation. Kueng

et al. design a pipeline for this process, which includes event preprocessing, feature extraction from both event data and frames, data fusion, and finally pose estimation. The events and frames are combined in a probabilistic filter, where the events are used for fast updates and the frames for slower, but more globally accurate corrections.

2.5.4 Deep Learning-based Methods

Recently, Deep learning provides a way to learn complex mappings from data, making it well-suited for processing the high-dimensional spatio-temporal event data. One notable contribution in this space is by Nguyen et al. Nguyen et al. (2017), who presented a real- time pose estimation method using event cameras and deep learning LeCun et al. (2015). Their work harnesses the power of LSTM networks to process event data and estimate 6-DOF pose. The network takes the event data and learns to map it directly to pose information, thus bypassing the need for hand-crafted features or explicit event-to-image conversion. Their results demonstrated the effectiveness of LSTM networks in this application and paved the way for further research into deep learning-based event camera pose estimation. Each of these methods brings unique approaches to handling the challenges of event-based pose estimation, and they collectively push forward the state-of- the-art in the field. Event-Based Visual Odometry (EVO), EVO employs image alignment techniques on an accumulated event image. However, it struggles with complex motions due to its reliance on intensity-like images, which can be heavily influenced by noise. Its performance may also degrade when the scene has rapid or drastic brightness changes that can affect the quality of the generated event images. Probabilistic Event-based Motion Tracking, While this method demonstrates a fast and accurate motion estimation, the probabilistic nature of the model requires it to deal with uncertainty in measurements. In noisy environments or during rapid movements, the model's performance could be adversely affected due to the increased uncertainty in event data. Event-based Pose Tracking using a DAVIS: This method leverages the advantages of both event

data and standard frames from a DAVIS sensor. However, the data fusion step may be computationally expensive. Moreover, this method requires a DAVIS sensor, which limits its applicability to other types of event cameras that only output event data. Deep Learning-based Methods, While Nguyen et al.'s work showed the potential of deep learning for pose estimation with event cameras, it heavily relies on LSTM networks, which are computationally intensive and may not be feasible for real-time applications.The method achieves promising results in camera pose estimation, but requires an expensive training time due to the retraining of the full network model with the LSTM layer. Considering these limitations, Chapter 3 will introduce a new approach that specifically addresses some of these issues. For instance, the data representation could be designed to be robust to noise and rapid brightness changes. It could also introduce a more efficient way of combining event data and standard frames, or it could propose a lightweight deep learning architecture that can be trained with limited data and still deliver real-time performance.

2.5.5 Spiking Neural Networks (SNNs)

Advancements in computer vision technologies are increasingly being driven by Spiking Neural Networks (SNNs) and event cameras. These technologies have shown significant progress in various fields, including optical flow estimation, image classification, object detection, and pose estimation. This discussion explores the applications, advantages, and recent developments in this area, highlighting the unique capabilities of SNNs and their synergy with event-based sensing.

Optical flow estimation is a pivotal application in computer vision that involves analyzing and estimating the motion of objects within a scene. This task poses substantial challenges due to the dynamic and complex nature of real-world visual environments. SNNs stand out in this domain due to their event-based computational paradigm. Unlike conventional neural networks that process continuous data streams, SNNs are designed to process discrete events, making them inherently suited for capturing the

temporal dynamics of visual scenes.

- **Efficient Temporal Information Processing:** SNNs excel at managing temporal data, a crucial aspect of optical flow estimation. By concentrating on temporal changes, SNNs can track the movement and direction of objects with higher accuracy than traditional methods.
- **Energy Efficiency:** SNNs offer superior energy efficiency compared to traditional neural networks. This trait is especially beneficial for real-time applications where minimizing power consumption is paramount, such as in wearable devices, mobile phones, and embedded systems.
- **Biologically Inspired Design:** The architecture of SNNs is inspired by the information processing mechanisms of biological neural systems. This biologically plausible approach facilitates a more intuitive and effective method for dealing with the intricate dynamics of optical flow.
- **Real-time Processing:** The capability of SNNs to process information in real time is critical for applications necessitating instant feedback, such as autonomous driving and interactive robotics.

Recent advancements in the field have demonstrated the potential of SNNs for revolutionizing computer vision. Researchers have explored various architectures and learning algorithms to optimize the performance of SNNs in tasks requiring complex temporal dynamics and low-latency processing. For instance, the work by Lagorce et al. (2017) introduces a hierarchical temporally aware network architecture that significantly improves the robustness and efficiency of event-based vision systems. Moreover, the integration of SNNs with event cameras, which capture changes in light intensity at each pixel independently, offers a promising direction for developing highly responsive and efficient vision systems. Event cameras, such as those described by Gallego et al. (2019), provide high temporal resolution and low latency, characteristics that are perfectly complemented by the event-driven nature of SNNs.

In summary, the synergy between SNNs and event-based sensing technologies holds the promise of creating vision systems that are more efficient, faster velocity figure 2.11, and capable of operating in more challenging environments than ever before. The continuous research and development in this area suggest a bright future for applications leveraging these innovative technologies.

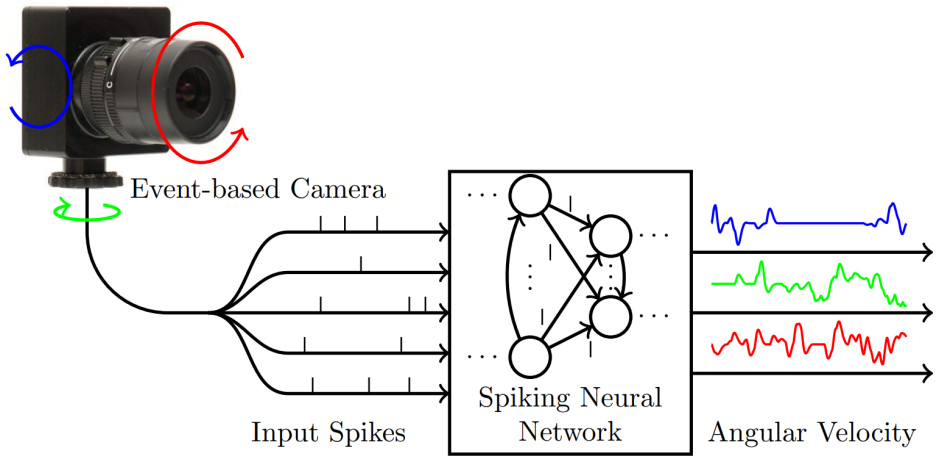


Figure 2.11: Processing pipeline for event-based angular velocity regression using a spiking neural network.Gehrig et al. (2020)

2.6 Conclusion

In conclusion, this chapter has provided an in-depth exploration of camera pose estimation techniques, focusing initially on methodologies designed for conventional cameras and then delving into the emerging domain of event-based camera pose estimation. While a significant amount of research has been conducted in the domain of conventional RGB camera pose estimation, spanning from traditional handcrafted feature approaches to more contemporary learning-based methods, the field of event-based camera pose estimation remains relatively unexplored, with only a limited body of work available, particularly in the domain of deep learning applications. The subsequent chapters of this work will present our con-

tributions to advancing event camera pose estimation through the utilization of deep learning techniques. This research seeks to bridge the gap between the well-established conventional camera pose estimation methods and the evolving domain of event-based approaches, potentially paving the way for new advancements and insights in this dynamic field.

Chapter 3

Camera pose estimation from event images

Event cameras, unlike traditional frame-based cameras, record pixel intensity changes as a stream of discrete events. This unique characteristic presents both opportunities and challenges for image processing. In this chapter, we introduce a novel approach to processing images captured by event cameras, focusing on the specific task of 6-Degrees of Freedom (6-DoF) pose estimation. Our method aims to harness the high temporal resolution and low latency of event cameras by converting the event stream into interpretable "event images" through spatial and temporal event binning (see Figure 3.1). We then explore the utilization of deep learning models to process these event images, proposing four different model architectures.

Our first proposed model architecture consists of a sequence of interconnected components: a Convolutional Neural Network (CNN), a Long Short-Term Memory (LSTM) layer, and a dense fully connected layer. The CNN extracts features from preprocessed event-based images, leveraging pretrained weights for efficiency. The LSTM captures spatial dependencies within these features, crucial for pose regression, and the final output of the LSTM, processed by a dense fully connected layer, provides detailed 6-DoF pose estimation, integrating CNN feature extraction and LSTM spatial

learning.

The second proposed model employs two separate CNNs (A and B) to extract features from event images, followed by bilinear pooling to capture local pairwise feature interactions. This approach, surpassing single CNN capabilities, integrates features from both CNNs for improved performance in pose estimation.

Our third proposed model architecture introduces improved bilinear pooling, leveraging deep learning and event camera characteristics. It includes matrix function normalization post-pooling, specifically utilizing the matrix logarithm and matrix power function for fractional positive values, notably $p = 1/2$ for matrix square root. These matrix functions require computations dependent on the entire matrix, with techniques like Newton iterations or Singular Value Decomposition (SVD) for effective computation.

The fourth proposed architecture is based on a self-attention mechanism to capture global dependencies and interactions among event patches. In this model, a feature map obtained from a CNN is passed to a Transformer encoder, which generates a context-aware representation for each event patch. This representation considers not only the information of the corresponding event but also information from other events in the sequence.

Throughout this chapter, we aim to demonstrate the potential of combining event cameras with deep learning techniques for efficient and effective image processing under challenging conditions. Our proposed methods represent significant advancements in the field, opening up avenues for various real-world applications such as robotics, autonomous navigation, and augmented reality.

By the end of this chapter, readers will gain insights into innovative approaches to leverage event-based images for precise and reliable 6-DoF pose estimation, with each proposed model offering unique strengths and capabilities. Experimental evaluations on real-world event camera datasets will be presented to validate the performance and effectiveness of these architectures.

3.1 From events to event images

In the context of event-based cameras, a fundamental difference arises when compared to traditional frame-based cameras. Rather than capturing a complete image at fixed intervals, event cameras record individual events that are triggered by changes in pixel brightness at the local level.

A key part of our work is focused on solving the pose relocalization problem. This involves determining the pose of the camera at a given point in time after it has been moved from its original position.

Our primary step is to process the event stream and convert it into what we term an "event image" (see Figure 3.1). The event image, denoted as I, is a matrix of dimensions h x w, where h represents the height and w denotes the width of the image.

Each event in the stream, denoted as e, can be thought of as a tuple or ordered pair represented by:

$$e =< e_t, (e_x, e_y)e_p >, \tag{3.1}$$

In this tuple, e_t denotes the timestamp at which the event occurred, while e_x and e_y specify the x and y coordinates of the pixel where the change in brightness was detected, and e_p represents the polarity of the event, indicating whether the change was an increase or decrease in brightness. This structure allows each event to capture and convey rich, temporally precise information about the dynamic changes happening in the scene.

Formally the conversion of an event stream into an event image is performed by aggregating events within a specified time window. The process involves initializing the image I with zeros, setting a time window T, and iterating through each event e in the event stream. If the event's timestamp e_t falls within the current time window, the intensity at the event's spatial location (e_x, e_y) in the image is increased by the event's polarity e_p according to equation 3.1.

Once we have transformed the event stream into an event image, it

serves as a foundation for subsequent stages of processing and analysis, including the application of feature extraction, matching algorithms, and pose estimation methodologies. This transformation step facilitates the usage of methodologies initially designed for traditional images, thus bridging the gap between event-based and frame-based vision systems.

The figure 3.1 showcases examples of event images generated prior to the preprocessing stage. From these representations, we can observe the influence of the parameter 'n' discussed in the work of Gallego et al. Gallego and Scaramuzza (2017). This parameter proves to be of significant importance as it directly affects the quality of the event images.

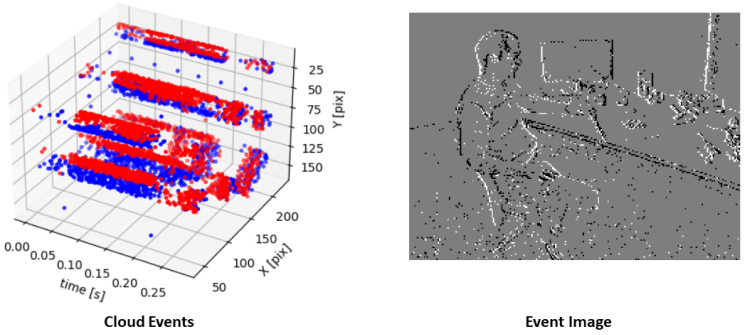


Figure 3.1: Image preprocessing from point cloud events to event image

High-quality event images are pivotal in our process as they form the input for training our CNN. The performance of the CNN, and consequently the accuracy of our camera relocalization estimation, depends heavily on these inputs. This, careful consideration and fine-tuning of this parameter n is necessary to optimize the transformation from event streams to images, ultimately enhancing the pose estimation results.

$$I(e_x, e_y) = \begin{cases} 0, & \text{if } e_\rho = -1 \\ 1, & \text{if } e_\rho = 1 \\ 0.5, & \text{otherwise} \end{cases}$$

Algorithm 1 Event Stream to Event Image Conversion (with Polarity-based Intensity)

Require: Event Stream $E = \{e_1, e_2, ..., e_n\}$
Ensure: Event Image I

Initialize an empty image I of size $h \times w$ (according to your camera resolution), initialized with zeros
Initialize a timestamp T to track the current time window for event aggregation
for each event $e = (e_t, (e_x, e_y), e_\rho)$ in E **do**
    if e_t is within the current time window **then**
        if $e_\rho = 1$ **then** ▷ Event is ON
            Set intensity at location (e_x, e_y) in I to 1
        else if $e_\rho = -1$ **then** ▷ Event is OFF
            Set intensity at location (e_x, e_y) in I to 0
        else ▷ Otherwise (other polarities)
            Set intensity at location (e_x, e_y) in I to 0.5
        end if
    else
        Store the current image I for processing or output
        Reset I to zeros
        Update T to the timestamp of the new time window
        if $e_\rho = 1$ **then** ▷ Event is ON
            Set intensity at location (e_x, e_y) in I to 1
        else if $e_\rho = -1$ **then** ▷ Event is OFF
            Set intensity at location (e_x, e_y) in I to 0
        else ▷ Otherwise (other polarities)
            Set intensity at location (e_x, e_y) in I to 0.5
        end if
    end if
end for

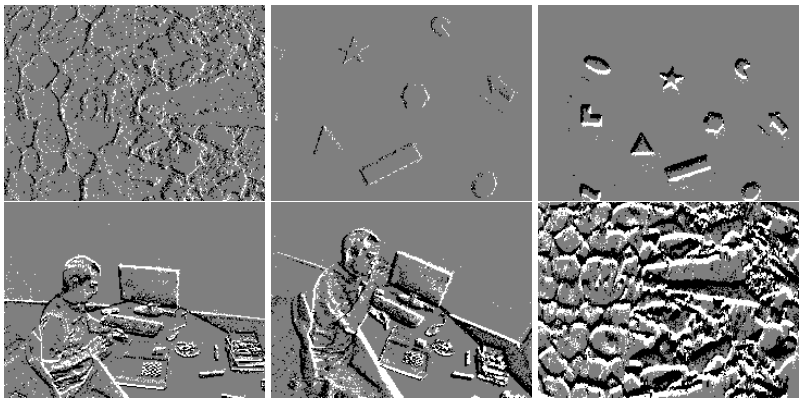


Figure 3.2: Event stream to event images

Wrapping up, our suggested approach to event-based image processing using event cameras unveils a new pathway for efficient and streamlined processing of visual data figure 3.2. The distinct advantages of event cameras, such as their high temporal resolution and minimal latency, are capitalized upon in our method. This allows us to transform the event stream into comprehensible event images that are compatible with standard image processing techniques.

In the second phase of our process, we adjust the dimensions of the event image. The image is scaled up to a standard size of 224×224 pixels while maintaining the original aspect ratio of the image.

Furthermore, we advocate for the utilization of several deep learning models tailored to process event images for the explicit purpose of 6-DoF pose estimation. In the subsequent section focused on the network model, we will delve deeper into the architecture and training procedures of the models utilized in our approach.

3.2 The proposed network models

In our work, we introduced four distinct model architectures tailored for event-based pose estimation. Each model takes event images as input and produces an estimated camera pose as its output.

1. The first proposed model architecture consists of a sequence of interconnected components: a Convolutional Neural Network (CNN), a Long Short-Term Memory (LSTM) layer, and a dense fully connected layer. The CNN extracts features from preprocessed event-based images, leveraging pretrained weights for efficiency. The LSTM captures spatial dependencies within these features, crucial for pose regression, by updating gates and cell states. The final output of the LSTM, processed by a dense fully connected layer, provides detailed 6-DoF pose estimation, integrating CNN feature extraction and LSTM spatial learning.

2. The second proposed model employs two separate CNNs (A and B) to extract features, followed by bilinear pooling to capture local pairwise feature interactions. This approach, surpassing single CNN capabilities, integrates features from both CNNs for improved performance in pose estimation.

3. The third proposed model architecture is based on improved bilinear pooling, leveraging deep learning and event camera characteristics. It introduces matrix function normalization post-pooling, specifically utilizing the matrix logarithm and matrix power function for fractional positive values, notably $p = 1/2$ for matrix square root. These matrix functions require computations dependent on the entire matrix, with techniques like Newton iterations or SVD for effective computation.

4. The fourth proposed architecture is based on a self-attention mechanism to capture global dependencies and interactions among these patches. In this architecture, we pass a feature map obtained from a CNN model to a Transformer encoder. The Transformer encoder generates a context-aware representation for each event patch, where each feature not only captures the information of the corresponding event but also considers the information from other events in the sequence.

The strategic choice of transitioning from one model to another reflects a progression towards more sophisticated and capable architectures. Each subsequent model builds upon the strengths of the previous one, aiming to address the challenges of event-based pose estimation by leveraging advancements in deep learning and neural network architectures.

The initial model architecture (CNN-LSTM-Dense Model) starts with a Convolutional Neural Network (CNN) for feature extraction, a Long Short-Term Memory (LSTM) layer to capture spatial dependencies crucial for pose regression, and a dense fully connected layer for detailed 6-DoF pose estimation. This model is a foundational step, leveraging the CNN's efficiency in extracting features from preprocessed event-based images. The LSTM then enhances the model's ability to capture spatial relationships within these features, improving pose estimation accuracy. The sequential flow from CNN to LSTM to dense layer demonstrates a progression from basic feature extraction to spatial learning and detailed pose estimation.

The second model (Dual-CNN with Bilinear Pooling) introduces a more complex approach with two separate CNNs for feature extraction, followed by bilinear pooling to capture local pairwise feature interactions. This dual-CNN architecture surpasses the capabilities of a single CNN by integrating features from both networks, allowing for a richer representation of the event images. The strategic choice to move to dual-CNN with bilinear pooling aims to further enhance the model's ability to extract and combine features for improved performance in pose estimation, building upon the foundation of the CNN-LSTM-dense model.

The third model (Improved Bilinear Pooling) architecture takes a step further by introducing improved bilinear pooling. This model leverages deep learning and event camera characteristics, incorporating matrix function normalization post-pooling. This normalization enables the model to capture more nuanced and complex relationships among features extracted from event images. The transition to improved bilinear pooling demonstrates a strategic choice to refine feature interactions and computations, aiming for higher accuracy in pose estimation.

Finally, the fourth model (Transformer-Based Model with Self-Attention)

architecture introduces a Transformer-based approach with a self-attention mechanism. This model represents a significant departure from traditional CNN-based approaches. Here, a feature map from a CNN is passed through a Transformer encoder, which generates a context-aware representation for each event patch. This representation considers not only the information from the corresponding event but also information from other events in the sequence, capturing global dependencies. The shift to a Transformer-based model with self-attention reflects a strategic choice to explore more advanced architectures capable of modeling long-range dependencies and interactions among event patches, potentially leading to improved pose estimation accuracy.

A detailed description of each proposed model architecture is provided in the following sections.

3.2.1 First architecture: CNN-LSTM-Dense Model

Our first proposed approach for precise pose estimation consists of three interconnected components arranged in a sequential manner: a Convolutional Neural Network (CNN), a Long Short-Term Memory (LSTM) layer, and a dense fully connected layer. Each component serves a distinct purpose and collaborates with the others to achieve the ultimate goal of accurate pose estimation.

- **Convolutional Neural Network** The initial stage of our approach involves the utilisation of a CNN, which plays a crucial role in extracting relevant features from preprocessed event-based images. CNNs have demonstrated exceptional capabilities in computer vision tasks, thanks to their ability to learn hierarchical representations, progressing from basic features like edges and colors to more complex patterns and shapes LeCun et al. (2015). However, training CNNs from scratch can be computationally demanding and prone to overfitting due to the vast number of parameters. To address these challenges, we incorporate pretrained weights from well-established models such as ResNet18 He et al. (2016), ResNet50 He et al. (2016), GoogleNet

Szegedy et al. (2015), VGG16 Simonyan and Zisserman (2014), VGG19 Raja et al. (2021), MobileNet Howard et al. (2017), Inception Szegedy et al. (2016), and EfficientNet Tan and Le (2019). These models have been trained on extensive datasets and possess the ability to extract robust and discriminative features. If X is the input image, then the k-th feature map F_k at a certain layer could be represented as:

$$F_k = \text{Activation}(W_k * X + b_k) \tag{3.2}$$

where W_k are the weights of the k-th filter, b_k is the bias term, and $*$ denotes the convolution operation. Activation is a function that introduces non-linearity such as Relu or Sigmoid.

The feature extraction power of CNNs relies on various hyperparameters including the chosen architecture, number of layers, filter sizes, and the optimization algorithm, among others. In our study, we systematically evaluated a range of well-known backbones to select the best-performing model. This thorough evaluation allowed us to identify the architecture that most effectively extracts features from event-based images, optimizing our model's performance for the task of camera pose estimation.

- **Long Short-Term Memory (LSTM) layer:** The Long Short-Term Memory (LSTM) network, introduced by S. Hochreiter and J. Schmidhuber in their seminal 1997 paper Hochreiter and Schmidhuber (1997), represents a significant advancement in the field of recurrent neural networks (RNNs). LSTMs were designed to overcome the limitations of traditional RNNs, particularly the challenges associated with learning long-term dependencies.

  In our CNN-LSTM-Dense Model for camera pose estimation, following the feature extraction by the CNN, an LSTM layer is introduced to capture spatial dependencies within these extracted features. LSTMs, unlike traditional Recurrent Neural Networks (RNNs), can learn and retain long-term dependencies, making them suitable for tasks where

past information significantly influences future outputs. In our method, the LSTM layer receives the feature vectors from the CNN as sequential input and processes them accordingly. This sequential processing enables the model to aggregate and compress the extracted features, focusing solely on the most relevant information for pose regression.

LSTMs are designed to handle sequences of data and are capable of learning long-term dependencies. An LSTM unit has a cell state c_t and three gates (input i_t, forget f_t, and output o_t) that control the flow of information. These gates are updated at each time step t as follows:

$$i_t = \sigma(W_{xi}v_t + W_{hi}h_{t-1} + b_i) \quad \text{(input gate)} \tag{3.3}$$

$$f_t = \sigma(W_{xf}v_t + W_{hf}h_{t-1} + b_f) \quad \text{(forget gate)} \tag{3.4}$$

$$o_t = \sigma(W_{xo}v_t + W_{ho}h_{t-1} + b_o) \quad \text{(output gate)} \tag{3.5}$$

$$g_t = \tanh(W_{xg}v_t + W_{hg}h_{t-1} + b_g) \quad \text{(cell input activation)} \tag{3.6}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad \text{(cell state update)} \tag{3.7}$$

$$h_t = o_t \odot \tanh(c_t) \quad \text{(hidden state/output)} \tag{3.8}$$

where σ is the sigmoid activation function, $\tanh$ is the hyperbolic tangent function, and $\odot$ denotes element-wise multiplication. As the LSTM processes each feature vector in sequence, it updates its internal state. This state acts as a memory that retains relevant information and discards irrelevant details, effectively compressing the information.

The final output of the LSTM layer, which can be the last hidden state h_T, or a combination of all hidden states, encapsulates the aggregated information of the entire sequence, focusing on the most relevant features for the subsequent task, such as pose regression.

- **Dense Fully Connected Layer:** The final component of our architecture is a dense fully connected layer comprising seven neurons. This layer receives the output from the LSTM layer, which has been

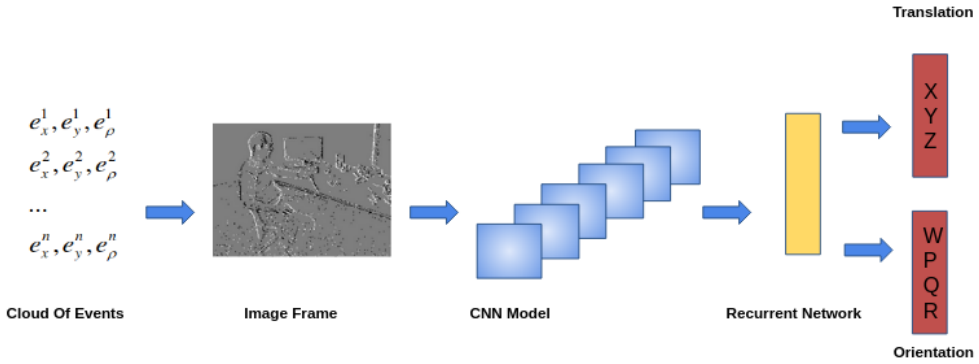


Figure 3.3: An overview of our 6DOF pose relocalization method for event cameras.
We first create an event image from stream of events. Then we extract features from the created event image, the feature vector is then given to a single layer reccurent network. Finally, a fully connected layer of seven neutrons is used to regress the camera pose vector.

condensed to retain only the most important information. Each neuron in this layer is responsible for estimating different camera coordinates, resulting in a detailed 6-DoF pose estimation.

Our proposed CNN-LSTM-Dense model seamlessly integrates these three components CNN for feature extraction, LSTM for learning spatial dependencies, and a fully connected layer for pose estimation figure 3.3. By employing this model, we aim to provide an efficient and accurate solution for pose estimation using event-based images.

Why opt for a single layer instead of two for LSTM and CNN? The decision to use a single layer for LSTM and CNN depends on a trade-off between computational complexity and performance. Each additional layer adds computational cost and complexity to the model, requiring more training data and time. Furthermore, increasing model depth does not always guarantee improved performance and can lead to overfitting if not carefully managed. Sagheer and Kotb (2019) For CNNs, a single layer can suffice if pretrained models such as ResNet or VGG already provide effective feature representations. These models are already deep and capable of

extracting hierarchical features. Regarding LSTMs, a single layer might be adequate if the temporal dependencies within the data are not excessively complex. Adding more LSTM layers involves stacking additional LSTMs on top of each other, which helps when dealing with more intricate sequential patterns. However, it significantly increases model complexity and the risk of overfitting. In conclusion, although it may seem that adding more layers always leads to improved performance, this is not necessarily the case. It is crucial to consider the specific problem, data complexity, available computational resources, and the risk of overfitting when designing the architecture of a deep learning model. We give more details about our model performance in the experimental section.

3.2.2 Second architecture: Dual-CNN with Bilinear Pooling

Bilinear pooling, a method frequently applied in computer vision for integrating features from varied sources or layers, is predominantly utilized in classification scenarios but can also be adapted for regression tasks. To employ bilinear pooling for regression, we start by selecting an appropriate base model for our pose estimation task, we used a CNN model, which serves to extract features from the input data. This base model is crucial as it lays the foundation for feature extraction that will be further processed using bilinear pooling in the context of the camera pose regression.

Bilinear pooling combines features from two different sources. In our case of camera pose regression, we use the feature maps from a chosen base model as both sources. Compute the outer product between the feature maps from source 1 and source 2. This is done by flattening both feature maps and then taking their outer product. To handle the high-dimensional tensor generated by the outer product in bilinear pooling, a dimensionality reduction step is essential. In our work this has been achieved by performing a pooling operation, i.e. max or average pooling, across each tensor dimension. This step is crucial for isolating the most significant information. Following the pooling process, the next action is to

transform the pooled tensor into a one-dimensional vector through flattening. This creates a consolidated feature vector suitable for subsequent stages in the model. Repeat this process for all the samples in your dataset. Concatenate all the resulting vectors to form a single feature vector for each sample.

More precisely, in our work, we begin by preprocessing raw event data acquired from an event camera. This process involves converting the data into event images, where each pixel corresponds to the polarity of an event, indicating a brightness change at a specific pixel. To ensure compatibility with popular convolutional neural networks (CNNs), these event images are resized to a standard size, typically 224×224 pixels.

Two separate CNNs, denoted as A and B, are utilized to extract features from the event images. These CNNs produce output feature maps represented by matrices U and V, with dimensions $m \times d$ and $n \times d$ respectively, where n and m are the number of kernels in the output layers of networks A and B Lin et al. (2015). The employed CNNs, A and B, are pretrained models such as VGG16 Simonyan and Zisserman (2014) or MobileNetV2 Sandler et al. (2018), and each independently processes the event images, resulting in two sets of feature maps.

Following feature extraction, we apply the technique of bilinear pooling. This step involves taking the outer product of the feature maps from CNNs A and B, capturing local pairwise feature interactions. The result is a new feature matrix that combines the strengths of the feature maps from both CNNs, enabling the incorporation of higher-order, complex feature interactions and surpassing the capabilities of a single CNN.

The final stage of our approach involves pose estimation. The pooled feature matrix is fed into a fully connected layer with a linear activation function, which produces an estimated pose represented by a 7-dimensional vector encompassing both position (3D) (X, Y, Z) and orientation (4D quaternion) (W, P, Q, R). This method aims to enhance the accuracy and comprehensiveness of pose estimation by capturing intricate feature interactions beyond what can be achieved using a single CNN, thereby providing a more precise and detailed understanding of event-based images.

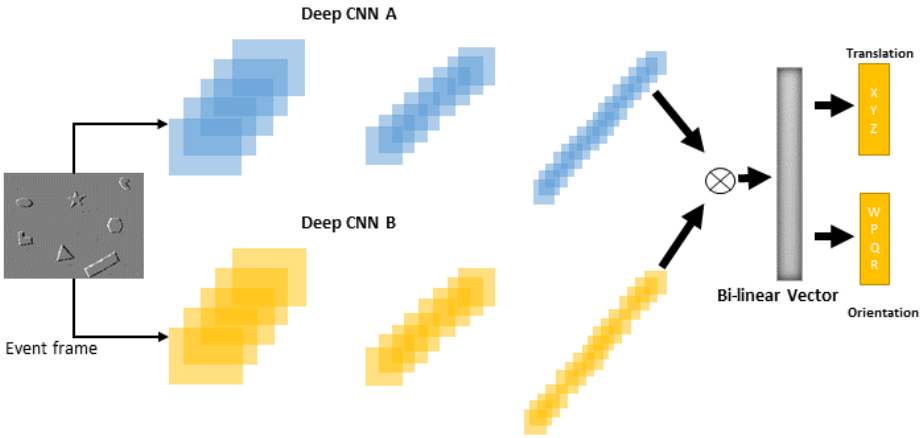


Figure 3.4: An overview of our 6DOF pose relocalization method for event cameras.
We first create an event image from stream of events. After we extract features from the created event image using a Bilinear pooling. Then the feature vector is then given to a fully connected layer of seven neurons is used to regress the camera pose vector.

We put forth an innovative attention-based fully convolutional neural network for the task of pose estimation. Our distinctive attention mechanism enables the model to concentrate on the motion-relevant areas in images, thereby enhancing its accuracy.

We start by gathering a sequence of events using an event camera. These raw events are processed and transformed into a set of event images, following the methodology outlined by Nguyen et al. (2019). The details of image preprocessing are elaborated in Section3.1.

These features are then subjected to bilinear pooling. Bilinear pooling is a method used to combine features that encapsulates second-order statistics. Both network A VGG16 Simonyan and Zisserman (2014) and B MobileNets Howard et al. (2017) outputs feature maps represented respectively by the matrix V of dimensionality $n \times d$, and the matrix U of size $m \times d$. Here, n and m are the number of kernels in the output layers of the networks A and B, respectively. The dimensionality of each filter is d; it is obtained by flattening the 2-dimensional feature map, i.e., the output

image that has undergone several kernel convolutions and pooling transformations. The bilinear pooling operation is then defined as:

$$X = UV^T, U \in \mathbb{R}^{m*d}, V \in \mathbb{R}^{n*d}, X \in \mathbb{R}^{m*n} \tag{3.9}$$

Finally, this bilinear pooling vector is used for pose estimation, thus capitalizing on the feature-rich information it contains, as explained in Lin et al.Lin et al. (2015).

3.2.3 Third architecture: Improved Bilinear pooling

Bilinear pooling is a technique used to capture second-order like explain in section 3.2.2 statistics between feature activations from two different feature maps. In the context of pose relocalization with event cameras, the event image is fed into a convolutional neural network (CNN) to obtain feature maps. These feature maps represent the learned visual features extracted from the event image. To perform bilinear pooling, pairs of feature maps are chosen, and the outer product is computed between the corresponding activations. The outer product combines the activations of each pair, resulting in a matrix that captures the pairwise interactions between the features. This process is repeated for all possible pairs of feature maps, generating a set of bilinear pooled features.

The bilinear pooled features figure 3.4 contain rich information about the relationships between different visual features in the event image. This is beneficial for capturing complex spatial dependencies and improving the discriminative power of the extracted features. After obtaining the bilinear pooled features, L2 normalization layers are applied to the feature vectors. L2 normalization normalizes the feature vectors to have a unit Euclidean length. This normalization step helps to make the features more invariant to changes in scale and increases the robustness of the pose estimation process. Moving on to the pose regression step, the normalized feature vectors obtained from the previous step are fed into a fully connected layer with seven neurons. This fully connected layer acts as a regression layer,

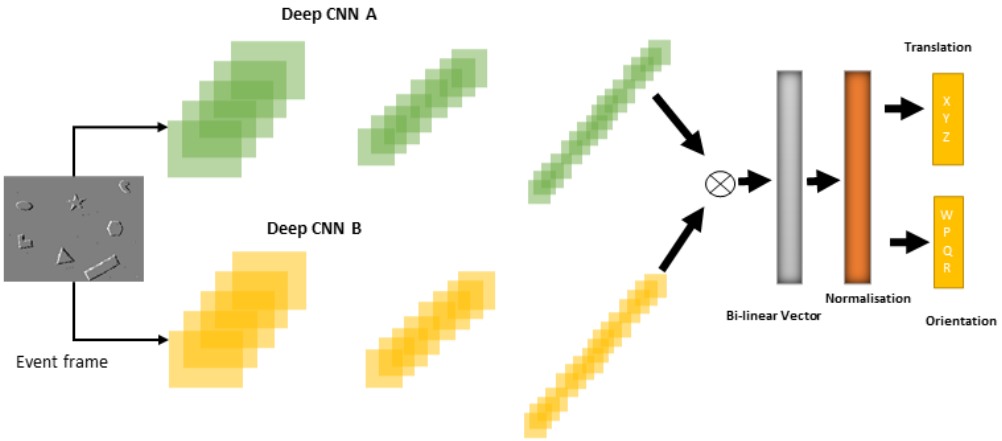


Figure 3.5: 6DOF pose relocalization method for event cameras consists of three steps. First, we create an event image from the event stream. Next, we extract features from the image using Bilinear pooling and l2 normalization layers added after the bilinear pooling of CNN. Finally, we use a fully connected layer of 7 neurons to regress the camera's pose vector from the feature vector.

mapping the feature representations to the camera's pose vector.

The pose vector typically consists of six degrees of freedom (6DOF), representing the camera's position and orientation in 3D space. The fully connected layer performs regression by learning the mapping between the features and the corresponding camera poses. During training, annotated pose labels are provided to supervise the learning process, allowing the network to learn to estimate the camera's pose accurately.

By combining the event image creation, feature extraction with bilinear pooling and L2 normalization, and pose regression with a fully connected layer, this method aims to achieve real-time pose relocalization with event cameras. The process leverages the power of deep learning and the unique characteristics of event cameras to estimate the camera's pose from the event stream efficiently. An improved approach introduces matrix function normalization after the pooling step (as shown in figure 3.5). Specifically, the matrix logarithm ($log(A)$) proposed in the Second-Order Pooling (O2P) Carreira et al. (2012) scheme and the matrix power function (A^p) for

fractional positive values of $0 < p < 1$ are considered. Notably, the scenario where $p = 1/2$ is of special interest because it equates to taking the square root of a matrix. This entails finding a matrix Z such that $ZZ = A$. Unlike element-wise transformations, matrix functions require computations that depend on the entire matrix. Techniques such as Newton iterations or Singular Value Decomposition (SVD) can be employed to perform these matrix function computations effectively.

Our third model utilizes bilinear pooling to capture second-order statistics between features, and matrix function normalization is applied to the resulting covariance matrix. This normalization step improves the network's performance, and matrix functions like the matrix logarithm and matrix power functions are utilized to improve feature representations.

3.2.4 Fourth architecture: Transformer-Based Model with Self-Attention

In contrast to CNNs that revolutionized image processing by convolving local, translationally invariant filters over the input, Vision Transformers (ViTs) brought about a significant shift in the field. The central tenet of ViTs is to perceive an image not as a 2D matrix of pixels, but as a sequence of patches, each holding its own spatial information. This allows transformers to leverage their self-attention mechanism to capture global dependencies and interactions among these patches Dosovitskiy et al. (2020).

The initial step is to partition an image into non-overlapping patches of a fixed size (P x P), effectively translating a 2D image of size H x W into a sequence of flattened 1D patches. Each patch is then linearly transformed to D dimensions, creating a sequence of token embeddings $Z_0 \in \mathbb{R}^{(N+1)\times D}$. Here, N is the total number of patches ($N = \frac{H}{P} \times \frac{W}{P}$), D is the dimension of the token embeddings, and '+1' accounts for an additional learnable 'class' token that's appended to the sequence for classification tasks Vaswani et al. (2017). Mathematically, it's represented as:

$$Z_0 = [z_{class}; E(x) + E_{pos}], \tag{3.10}$$

where $z_{class} \in \mathbb{R}^D$ is the class token, $E(x) \in \mathbb{R}^{N \times P}$ is the linear transformation of the patches, and $E_{pos} \in \mathbb{R}^{N \times P}$ is the positional encoding.

Our proposed model harnesses the advantages of Vision Transformers for the estimation of camera poses from event images. The architecture is composed of an image patch embedding layer, a transformer encoder layer, and a final multi-layer perceptron for pose estimation.

In the initial image patch embedding layer, the 2D patches are transformed into 1D embeddings through a convolution operation where both the kernel and stride size are equivalent to the patch size. The positional information, which is crucial in the context of a transformer, is preserved by adding positional embeddings to these patch embeddings.

These initial embeddings Z_0 are then propagated through a transformer encoder. The core of the transformer encoder is the self-attention mechanism which permits the model to calculate the relevance of a patch with respect to all other patches in the image, thereby capturing a global context. This is achieved using the scaled dot-product attention mechanism where the input comprises queries Q, keys K, and values V. The output is computed as $Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}}) * V$, where 'd' is the dimensionality of the queries and keys, and softmax is the softmax function applied across the keys Kingma and Ba (2014). The encoder outputs a more refined sequence of embeddings Z_l.

In the final stage, the output of the transformer (i.e., the embeddings corresponding to the class token) is funneled through a multi-layer perceptron to estimate the pose. The MLP essentially works as a function $F : \mathbb{R}^D \rightarrow \mathbb{R}^6$, which maps the D-dimensional embeddings to a 6-dimensional vector representing the estimated 6-DoF pose.

In mathematical terms, given Z_l, the pose $\hat{y}$ is given by:

$$\hat{y} = MLP(Z_l), \tag{3.11}$$

where Z_l denotes the class token embedding from the final layer.

The overall process thus forms a pipeline where the input image is transformed into a sequence of patch embeddings, which after multiple

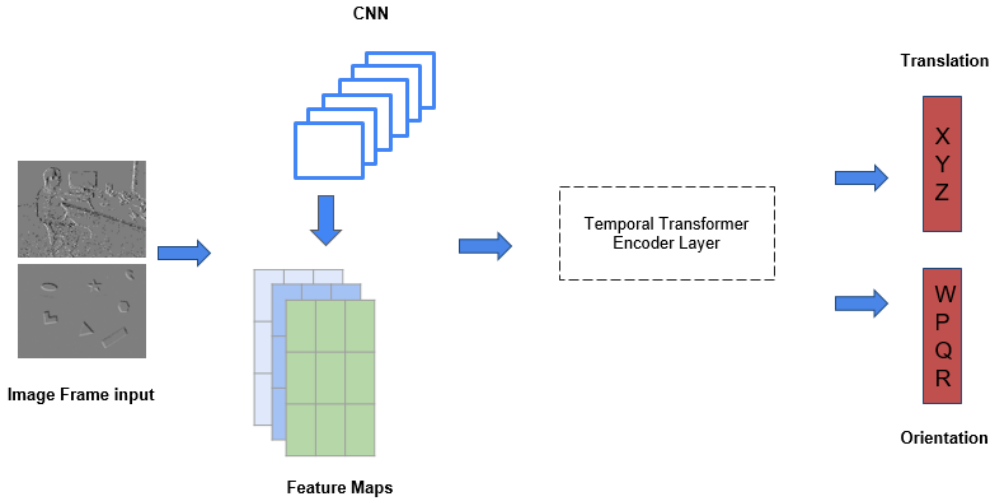


Figure 3.6: An overview of our 6DOF pose relocalization method for event cameras .
We first create an event image from stream of events. Then we extract features from the created event image.Once these event images are generated, they are fed into our feature extraction model, which is built on a ResNet50 architecture pre-trained on ImageNet. This model extracts high-level spatial features from the event images, providing a robust representation of the input data. Then the output is passed to a custom transformer encoder.

stages of self-attention and transformations, results in an estimate of the 6-DoF camera pose.

3.3 Experimental results

The proposed camera pose estimation methods were thoroughly evaluated using several datasets, which encompass both synthetic and real-world scenarios. This section describes the datasets that were utilized for method validation, the training environment configuration, and the detailed experimental results that substantiate the efficacy of our approach.

3.3.1 Dataset

We employed a variety of datasets to ensure a comprehensive assessment of the proposed methods: (1) simulated data and (2) real data.

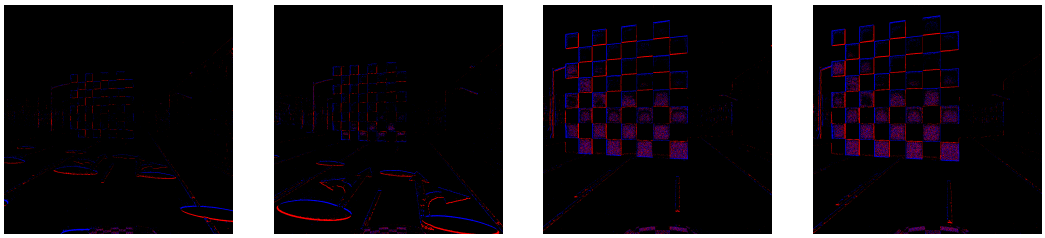


Figure 3.7: Dataset from CARLA Dosovitskiy et al. (2017)

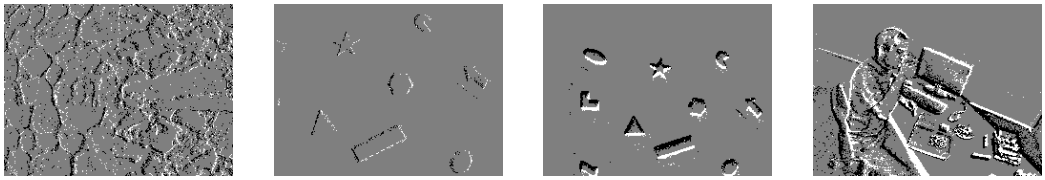


Figure 3.8: Real Event Camera Dataset After Pre-processing 3.1 Mueggler et al. (2017)

Figure 3.9: Both Real And Similator Datasets

Synthetic dataset

We synthesised a dataset from CARLA Dosovitskiy et al. (2017) 10000 event frame for both training and testing. Figure 3.7 provides a glimpse into the simulated images used for evaluation. This dataset is characterized by its variety, featuring scenes collected in two distinct environments - Clear Town and Dynamic Town. These virtual towns, complete with nuanced architectural elements and a range of atmospheric conditions, serve as excellent platforms to test the robustness of our model in a controlled but diverse environment.

Real dataset

The synthetic data is complemented by the use of real-world data. For this, we utilized the event camera dataset introduced in Mueggler et al.

(2017) 3.8. This carefully curated dataset comprises a collection of scenes captured by a DAVIS camera. The camera, designed to mimic the human eye's ability to respond to changes in the environment, is versatile enough to capture both indoor and outdoor environments. Thus, it provides an excellent resource for the task at hand.

What sets this dataset apart is the precision of its ground truth data. The camera poses, vital to our task, were collected from a state-of-the-art motion-capture system. This system can record data with sub-millimeter precision at an impressive rate of 200Hz. This ensures that the ground truth data is highly accurate and can be a reliable basis for comparison with the poses estimated by our model.

For the evaluation process using the real dataset, we adopted the protocol used in Nguyen et al. (2019). This protocol includes two distinct types of splits: a random split and a sequential or novel split.

1. In the **random split**, we select 70% of the event images for training and the remaining 30% for testing. This selection is random and does not consider the temporal order of the events. In this split, we used 6 sequences (*shapes rotation, box translation, shapes translation, dynamic 6dof, hdr poster, poster translation*) for this experiment. These sequences are selected to cover different camera motions and scene properties.

2. In the **sequential (novel) split**, we select the first 70% of each event for training and the rest, 30%, for testing. This implies that the training and testing data are temporally separate but originate from the same scene, essentially creating two independent sequences. By following the timestamps suggested by . Nguyen et al. (2019), where the training sequence spans from timestamp t_0 to t_{70}, and the testing sequence from timestamp t_{71} to t_{100}, we maintain a clear demarcation between training and testing sequences. This process effectively trains our model on a large portion of the sequence and then tests it on the remaining unseen section. In this manner, it puts our model's predictive capabilities to the test, giving us a clear picture of its per-

formance on novel data. We use three sequences from the shapes scene (*shapes rotation, shapes translation, shapes 6dof*) in this novel split experiment to compare the results when different camera motions are used.

3.3.2 Training Environment

In order to evaluate the effectiveness and robustness of our proposed methods, we have undertaken several experiments. These experiments aim not only to test the performance of our method but also to place our results in context by comparing them with those obtained from deep learning architectures.

To carry out these experiments, we employed PyTorch, the open-source machine learning library, as suggested by Paszke et al Paszke et al. (2019). This tool offers a wide range of utilities and functionalities that are ideally suited for our purpose. It provides a platform that not only facilitates the implementation of our proposed method but also allows for easy replication of the results. The computational platform on which our experiments were conducted consists of an Intel(R) Xeon(R) CPU clocked at 2.00GHz, CPU memory of 24GB, and a single Tesla T4 GPU. This high-performance platform was chosen to ensure that our experiments could be carried out seamlessly and without computational bottlenecks.

In terms of training the neural networks, we adhered to a rigorous training regime. The networks were trained for a significant period - a total of 500 epochs - to ensure that the models had ample opportunity to learn and capture the underlying patterns within the data. We used a learning rate of $2 \exp -3$, which strikes a balance between speedy convergence and model stability. To keep our model robust and prevent overfitting, we utilized a combination of momentum-decay and weight decay. The momentum-decay, set to $4 \exp -3$, helped us accelerate our gradient vectors in the right directions and thus led to faster convergence. Simultaneously, the weight decay, set to 0, was used as a regularization technique. Overall, the described setup and approach were designed to ensure a comprehensive

evaluation of the proposed method, provide comparable results with state-of-the-art architectures, and foster an environment conducive to in-depth exploration and learning.

3.3.3 Experimental Results

As quantitative evaluation, we choose to calculate the median and average error 2.2.3 of the predicted pose in position and orientation. The Euclidean distance is used to compare the predicted position to the groundtruth, and the anticipated orientation is normalized to unit length before being compared to the groundtruth.

We compare our results under the same protocols reported in Nguyen et al. (2019) also presented in Section 3.3.1 and used in PoseNet Kendall et al. (2015), Bayesian PoseNet Kendall and Cipolla (2016).

For location and orientation, the median and average error are recorded in m and deg(°), respectively.

Results from the CNN-LSTM-Dense Model

We start by evaluating the CNN-LSTM-Dense Model, our first proposed architecture. The results are given in Table 3.1. In this table, we compare different CNN architectures combined with LSTM when tested on the synthetic CARLA dataset 3.7. The best-performing architecture in this experiment, as indicated by the bold values, is ResNet18 combined with LSTM, showing the lowest median and average errors.

Table 3.2) compares the same set of architectures but tested on a real dataset 3.8 referred to as shapes_6dof. Again, ResNet18 combined with LSTM shows superior performance compared to the other architectures.

Architecture	Median Error	Average Error
VGG16 + LSTM Nguyen et al. (2019)	0.81 (m) 5.35 (°)	0.88 (m) 6.29 (°)
VGG19 + LSTM	0.52 (m) 6.29 (°)	0.62 (m) 7.96 (°)
GoogleNet + LSTM	0.52 (m) 6.29 (°)	0.62 (m) 7.96 (°)
MOBILENET + LSTM	0.44 (m) 3.36 (°)	0.53 (m) 4.48 (°)
INCEPTION + LSTM	0.47 (m) 3.86 (°)	0.61 (m) 4.88 (°)
EFFICIENTNET + LSTM	0.52 (m) 5.28 (°)	0.65 (m) 4.36 (°)
ResNet50 + LSTM	0.34 (m) 3.21 (°)	0.43 (m) 4.32 (°)
ResNet18 + LSTM	**0.21** (m) **2.91** (°)	**0.33** (m) **3.82** (°)

Table 3.1: Comparison between different architecture. The experiments are conducted on simulated dataset from CARLA Dosovitskiy et al. (2017)

Architecture	Median Error	Average Error
VGG16 + LSTM	0.108 (m) 5.23 (°)	0.125 (m) 6.21 (°)
VGG19 + LSTM	0.131 (m) 5.621 (°)	0.123 (m) 6.215 (°)
MOBILENET + LSTM	0.100 (m) 5.267 (°)	0.116 (m) 6.742 (°)
INCEPTION + LSTM	0.153 (m) 5.685 (°)	0.135 (m) 6.684 (°)
EFFICIENTNET + LSTM	0.125 (m) 5.921 (°)	0.124 (m) 5.736 (°)
GoogleNet + LSTM	0.102 (m) 5.245 (°)	0.111 (m) 6.548 (°)
ResNet50 + LSTM	0.092 (m) 4.513 (°)	0.119 (m) 5.981 (°)
ResNet18 + LSTM	**0.071** (m) **4.215** (°)	**0.091** (m) **5.601** (°)

Table 3.2: Comparison between different model architectures. The experiments are conducted on Real dataset Mueggler et al. (2017) shapes_6dof

The results highlighted in these tables are critical for understanding the efficacy of different CNN backbones used in our CNN-LSTM-Dense Model in the task of camera pose estimation. These results help in identifying which combinations of CNNs and LSTM yield the most accurate pose estimations for both synthetic and real-world data, guiding the choice of the CNN backbone for similar tasks in the future.

In table 3.1, when using the carla similator dataset (Figure 3.7), we compared different CNN backbones for our CNN-LSTM-Dense Model performance. The lower the error values, the better the model's performance.

In analyzing these results, several observations emerge regarding the performance of different CNN backbones. The **VGG16 + LSTM** configuration shows a median error of 0.81 meters and 5.35 degrees, with an average error of 0.88 meters and 6.29 degrees. Slightly better results are achieved with the **VGG19 + LSTM** configuration, with a median error of 0.52 meters and 6.29 degrees, along with an average error of 0.62 meters and 7.96 degrees. Notably, the **ResNet18 + LSTM** model stands out as the best performer in this comparison, demonstrating a median error of 0.21 meters and 2.91 degrees, with an average error of 0.33 meters and 3.82 degrees. The **GoogleNet + LSTM** model shows similar results to VGG19 + LSTM, with a median error of 0.52 meters and 6.29 degrees, and an average error of 0.62 meters and 7.96 degrees. The **MobileNet + LSTM** configuration achieves a median error of 0.44 meters and 3.36 degrees, with an average error of 0.53 meters and 4.48 degrees. Meanwhile, the **Inception + LSTM** model yields a median error of 0.47 meters and 3.86 degrees, along with an average error of 0.61 meters and 4.88 degrees. The **EfficientNet + LSTM** model presents a median error of 0.52 meters and 5.28 degrees, with an average error of 0.65 meters and 4.36 degrees. Lastly, the **ResNet50 + LSTM** configuration shows a median error of 0.34 meters and 3.21 degrees, along with an average error of 0.43 meters and 4.32 degrees. These results highlight the varying performance of the model on the synthetic dataset, with ResNet18 + LSTM showing the lowest errors, indicating its superior accuracy in estimating both position and orientation.

In this table 3.2, a comparison is made between different model archi-

tectures based on their performance on the Real dataset Mueggler et al. (2017) using the shapes_6dof evaluation metric. In comparing the results of the different model CNN backbones, several insights emerge. The **VGG16 + LSTM** backbone achieves a median error of 0.108 meters and 5.23 degrees, with an average error of 0.125 meters and 6.21 degrees. Similarly, the **VGG19 + LSTM** model yields a median error of 0.131 meters and 5.621 degrees, along with an average error of 0.123 meters and 6.215 degrees. The **MobileNet + LSTM** model stands out with a median error of 0.100 meters and 5.267 degrees, and an average error of 0.116 meters and 6.742 degrees, demonstrating good performance. In contrast, the **Inception + LSTM** model presents a median error of 0.153 meters and 5.685 degrees, with an average error of 0.135 meters and 6.684 degrees. The **EfficientNet + LSTM** configuration showcases a median error of 0.125 meters and 5.921 degrees, along with an average error of 0.124 meters and 5.736 degrees. The **GoogleNet + LSTM** model delivers a median error of 0.102 meters and 5.245 degrees, with an average error of 0.111 meters and 6.548 degrees. Notably, the **ResNet50 + LSTM** model offers good accuracy with a median error of 0.092 meters and 4.513 degrees, and an average error of 0.119 meters and 5.981 degrees. However, the standout performer in this comparison is the **ResNet18 + LSTM** configuration, achieving the lowest errors with a median error of 0.071 meters and 4.215 degrees, and an average error of 0.091 meters and 5.601 degrees. This model demonstrates superior accuracy in predicting both position and orientation, making it a compelling choice for event-based pose estimation tasks.

From this two tables3.1 3.2 , we can see that the ResNet18 + LSTM architecture outperforms the other two in terms of both median and average errors. It achieves the lowest errors overall, indicating better accuracy in predicting the camera position and orientation.

Table 3.3 and Table 3.4 report the performance of our best CNN-LSTM-Dense Model compared to the state-of-the-art methods using random split and novel split protocols, respectively. Our CNN-LSTM-Dense Model demonstrates significant improvements in pose estimation accuracy compared to state-of-the-art techniques. Under **the random split protocol**1, as shown

	PoseNet		**Bayesian PoseNet**		**SP-LSTM**		**Ours**	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.109m, 7.388°	0.137m, 8.812°	0.142m, 9.557°	0.164m, 11.312°	0.025m, 2.256°	0.028m, 2.946°	0.012m, 1.652°	0.015m, 1.975°
shapes translation	0.238m, 6.001°	0.252m, 7.519°	0.264m, 6.235°	0.269m, 7.585°	0.035m, 2.117°	0.039m, 2.809°	0.020m, 1.471°	0.023m, 1.973°
box translation	0.193m, 6.977°	0.212m, 8.184°	0.190m, 6.636°	0.213m, 7.995°	0.036m, 2.195°	0.042m, 2.486°	0.013m, 0.873°	0.016m, 0.965°
dynamic 6dof	0.297m, 9.332°	0.298m, 11.242°	0.296m, 8.963°	0.293m, 11.069°	0.031m, 2.047°	0.036m, 2.576°	0.016m, 1.662°	0.017m, 1.974°
hdr poster	0.282m, 8.513°	0.296m, 10.919°	0.290m, 8.710°	0.308m, 11.293°	0.051m, 3.354°	0.060m, 4.220°	0.033m, 2.421°	0.038m, 3.223°
poster translation	0.266m, 6.516°	0.282m, 8.066°	0.264m, 5.459°	0.274m, 7.232°	0.036m, 2.074°	0.041m, 2.564°	0.020m, 1.468°	0.024m, 1.855°
Average	0.231m, 7.455°	0.246m, 9.124°	0.241m, 7.593°	0.254m, 9.414°	0.036m, 2.341°	0.041m, 2.934°	**0.019**m, **1.591**°	**0.022**m, **1,994**°

Table 3.3: Comparison between results from our First architecture (CNN-LSTM-Dense Model in Section 3.2.1) and results from PoseNet Kendall et al. (2015), Bayesian PoseNet Kendall and Cipolla (2016) and SP-LSTM Nguyen et al. (2019).

The evaluation is performed using the random split protocol. Rows indicate the 6 event sequences of the real dataset (Section 3.3.1). The results are given in term of Median and Average Error.

	PoseNet		**Bayesian PoseNet**		**SP-LSTM**		**Ours**	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.201m, 12.499°	0.214m, 13.993°	0.164m, 12.188°	0.191m, 14.213°	0.045m, 5.017°	0.049m, 11.414°	0.036m, 2.113°	0.040m, 4.101°
shapes translation	0.198m, 6.969°	0.222m, 8.866°	0.213m, 7.441°	0.228m, 10.142°	0.072m, 4.496°	0.081m, 5.336°	0.061m, 3.372°	0.069m 4.550°
shapes 6dof	0.320m, 13.733°	0.330m, 18.801°	0.326m, 13.296°	0.329m, 18.594°	0.078m, 5.524°	0.095m, 9.532°	0.071m, 4.215°	0.091m, 5.602°
Average	0.240m, 11.067°	0.255m, 13.887°	0.234m, 10.975°	0.249m, 14.316°	0.065m, 5.012°	0.075m, 8.761°	**0.056**m, **3.233**°	**0.066**m, **4.417**°

Table 3.4: Comparison between our First model architecture (Section 3.2.1) results and the results from PoseNet Kendall et al. (2015), Bayesian PoseNet Kendall and Cipolla (2016) and SP-LSTM Nguyen et al. (2019).

The evaluation is performed using the novel split protocol. Rows indicate the 3 event sequences of the real dataset of the novel split protocol (Section 3.3.1).

	PoseNet		Bayesian PoseNet		SP-LSTM		Ours	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.109m, 7.388°	0.137m, 8.812°	0.142m, 9.557°	0.164m, 11.312°	0.025m, 2.256°	0.028m, 2.946°	0.018m, 1.753°	0.020m, 2.551°
shapes translation	0.238m, 6.001°	0.252m, 7.519°	0.264m, 6.235°	0.269m, 7.585°	0.035m, 2.117°	0.039m, 2.809°	0.033m, 2.211°	0.036m, 2.717°
box translation	0.193m, 6.977°	0.212m, 8.184°	0.190m, 6.636°	0.213m, 7.995°	0.036m, 2.195°	0.042m, 2.486°	0.029m, 1.507°	0.032m, 1.693°
dynamic 6dof	0.297m, 9.332°	0.298m, 11.242°	0.296m, 8.963°	0.293m, 11.069°	0.031m, 2.047°	0.036m, 2.576°	0.027m, 1.802°	0.029m, 2.394°
hdr poster	0.282m, 8.513°	0.296m, 10.919°	0.290m, 8.710°	0.308m, 11.293°	0.051m, 3.354°	0.060m, 4.220°	0.040m, 2.937°	0.051m, 3.783°
poster translation	0.266m, 6.516°	0.282m, 8.066°	0.264m, 5.459°	0.274m, 7.232°	0.036m, 2.074°	0.041m, 2.564°	0.036m, 2.045°	0.039m, 2.315°
Average	0.231m, 7.455°	0.246m, 9.124°	0.241m, 7.593°	0.254m, 9.414°	0.036m, 2.341°	0.041m, 2.934°	**0.030m, 2.204°**	**0.034m, 2.708°**

Table 3.5: Comparison between the results from our Second architecture based on bilinear pooling 3.2.2 and the results of PoseNet Kendall et al. (2015), Bayesian PoseNet Kendall and Cipolla (2016) and SP-LSTM Nguyen et al. (2019).

The evaluation is performed using the random split protocol.

in Table 3.3, our method outperforms existing methods across all tasks. Notably, in the **shapes rotation** and **dynamic 6dof** scenarios, our method reduces the median error to 0.012m and 0.016m, respectively, with corresponding average errors also being the lowest among the compared methods. This improvement is indicative of our method's robustness to rotational variations and dynamic conditions.

The novel split protocol1, outlined in Table 3.4, further validates the robustness of our approach. While the difficulty of the tasks increases, our method consistently maintains lower median and average errors. For instance, in the **shapes 6dof** task, our method achieves a median error of 0.071m, which is a significant reduction from the PoseNet's 0.320m median error. This highlights the capability of our approach to generalize well to novel data distributions. Across both protocols, our method demonstrates superior accuracy in both median and average error metrics. The results indicate that our method not only excels in capturing the translational aspects of pose estimation but also significantly improves the rotational accuracy, which is often more challenging to predict.

The average error reduction in the **dynamic 6dof** task, for example, from 9.124° to 1.994° in the random split, and from 13.887° to 4.417° in the novel split, showcases our method's proficiency in handling complex motion patterns. Moreover, the consistent performance across different test conditions underscores the method's adaptability and reliability.

	PoseNet		Bayesian PoseNet		SP-LSTM		Ours	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.201m, 12.499°	0.214m, 13.993°	0.164m, 12.188°	0.191m, 14.213°	0.045m, 5.017°	0.049m, 11.414°	0.050, 3.681°	0.053m, 6.823°
shapes translation	0.198m, 6.969°	0.222m, 8.866°	0.213m, 7.441°	0.228m, 10.142°	0.072m, 4.496°	0.081m, 5.336°	0.062m, 4.554°	0.068m, 5.854°
shapes 6dof	0.320m, 13.733°	0.330m, 18.801°	0.326m, 13.296°	0.329m, 18.594°	0.078m, 5.524°	0.095m, 9.532°	0.071m, 5.787°	0.091m, 7.550°
Average	0.240m, 11.067°	0.255m, 13.887°	0.234m, 10.975°	0.249m, 14.316°	0.065m, 5.012°	0.075m, 8.761°	**0.061m, 3.448°**	**0.070m, 6.742°**

Table 3.6: Comparison between the results from our Second architecture based on bilinear pooling 3.2.2 and the results of PoseNet Kendall et al. (2015), Bayesian PoseNet Kendall and Cipolla (2016) and SP-LSTM Nguyen et al. (2019).

The evaluation is performed using the novel split protocol.

Results from the Dual-CNN with Bilinear Pooling

Now we move to our bilinear pooling approach 3.2.2 heralds significant progress in the domain of pose estimation, showcasing considerable enhancements over contemporary state-of-the-art methodologies. This section presents the empirical outcomes using random split and novel split protocols, accentuating the efficacy of our bilinear pooling technique.

The random split evaluation, given in Table 3.5, illustrates our method's superior performance compared to existing models. Our approach notably excels in handling various pose estimation challenges, such as **shapes rotation** and **dynamic 6dof**, where it significantly lowers the median error to 0.018m and 0.027m, respectively. These results not only underscore the precision of our technique but also its adeptness in managing rotational dynamics and motion complexities.

Our bilinear pooling method surpasses the state-of-the-art methods across all evaluated tasks in the random split protocol. It achieves the lowest average errors, highlighting its capacity for precise pose estimation. The significant reduction in errors, such as in the **dynamic 6dof** task where the average error is minimized to 2.394°, illustrates our method's proficiency in accurately capturing complex motions.

The consistent outperformance against methods like PoseNet, Bayesian PoseNet, and SP-LSTM not only attests to the robustness of our approach but also its effectiveness in diverse conditions, ranging from static shapes to dynamic environments.

Through meticulous evaluation, our bilinear pooling approach sets a

new benchmark in the realm of pose estimation, demonstrating substantial reductions in error rates and consistent accuracy across diverse testing scenarios. This progress underscores the innovative design of our method and its potential to significantly enhance the precision and reliability of real-time 6DOF pose estimation technologies. Comparable to the achievements of other methods in the field, our approach not only showcases the ability to accurately estimate poses in real-time scenarios but also highlights the efficacy of bilinear pooling in achieving these improvements. This advancement reflects the proposed methods capacity to push the boundaries of current pose estimation capabilities.

Results from the improved Bilinear pooling

We present the results from our third architecture, the improved bilinear pooling model discussed in Section 3.2.3, shown in Table 3.7. This table compares our performance with PoseNet Kendall et al. (2015) and SP-LSTM Nguyen et al. (2019) using the random split protocol. Notably, the improved bilinear pooling architecture exhibits enhanced results compared to the previous model with dual-CNN and bilinear pooling. However, these results are still slightly below the performance of our first model, the CNN-LSTM-Dense Model. This can be attributed to the meticulous search and optimization process employed for the first model, which explored various CNN backbones and LSTM configurations. In contrast, the focus here was specifically on improving the dual-CNN with bilinear pooling. It is worth noting that despite this, our model continues to outperform state-of-the-art methods, indicating its effectiveness in event-based pose estimation tasks.

Results from the Transformer-Based Model with Self-Attention

The fourth model, the Transformer-Based Model with Self-Attention architecture, marks a significant departure from traditional CNN-based approaches. This model introduces a Transformer-based approach with a

	PoseNet		SP-LSTM		Ours	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.109m, 7.388°	0.137m, 8.812°	0.025m, 2.256°	0.028m, 2.946°	0.014m, 1.952°	0.017m, 2.144°
shapes translation	0.238m, 6.001°	0.252m, 7.519°	0.035m, 2.117°	0.039m, 2.809°	0.020m, 1.396°	0.027m, 1.835°
box translation	0.193m, 6.977°	0.212m, 8.184°	0.036m, 2.195°	0.042m, 2.486°	0.024m, 1.132°	0.026m, 1,219°
dynamic 6dof	0.297m, 9.332°	0.298m, 11.242°	0.031m, 2.047°	0.036m, 2.576°	0.025m, 1.730°	0.028m, 2.267°
hdr poster	0.282m, 8.513°	0.296m, 10.919°	0.051m, 3.354°	0.060m, 4.220°	0.038m, 2.154°	0.047m, 2,686°
poster translation	0.266m, 6.516°	0.282m, 8.066°	0.036m, 2.074°	0.041m, 2.564°	0.024m, 1.557°	0.032m, 1.959°
Average	0.231m, 7.455°	0.246m, 9.124°	0.036m, 2.341°	0.041m, 2.934°	**0.024m, 1.653°**	**0.029m, 1.951°**

Table 3.7: Comparison between results from our third architecture (improved bilinear pooling) Section 3.2.3 and the results of PoseNet Kendall et al. (2015) and SP-LSTM Nguyen et al. (2019). The evaluation is performed using the random split protocol.

	PoseNet		SP-LSTM		Ours	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.201m, 12.499°	0.214m, 13.993°	0.045m, 5.017°	0.049m, 11.414°	0.029m, 3.281°	0.038m, 5.713°
shapes translation	0.198m, 6.969°	0.222m, 8.866°	0.072m, 4.496°	0.081m, 5.336°	0,060m, 4,033°	0,067m,4,943°
shapes 6dof	0.320m, 13.733°	0.330m, 18.801°	0.078m, 5.524°	0.095m, 9.532°	0.068m, 4.886°	0.081m, 6.653°
Average	0.240m, 11.067°	0.255m, 13.887°	0.065m, 5.012°	0.075m, 8.761°	**0.052m, 4.066°**	**0.062m, 5,769°**

Table 3.8: Comparison between results from our third architecture (improved bilinear pooling) 3.2.3 and the results of PoseNet Kendall et al. (2015) and SP-LSTM Nguyen et al. (2019). The evaluation is performed using the novel split protocol.

self-attention mechanism, where a feature map from a CNN is passed through a Transformer encoder. This encoder generates a context-aware representation for each event patch, considering information not only from the corresponding event but also from others in the sequence, thereby capturing global dependencies. This strategic shift toward a Transformer-based model with self-attention reflects our intent to explore more advanced architectures capable of modeling long-range dependencies and interactions among event patches, potentially leading to improved pose estimation accuracy. Tables 3.9 and 3.10 present the results from our fourth model using the random split protocol and the novel split protocol, respectively. The Transformer-Based Model with Self-Attention compares favorably to different state-of-the-art methods, particularly for camera orientation estimation, although it exhibits slightly lower performance than our CNN-LSTM-Dense model. We attribute this to the limited quantity of training data available for training the transformer-based model. These models

	PoseNet		Bayesian PoseNet		SP-LSTM		Ours	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.109m, 7.388°	0.137m, 8.812°	0.142m, 9.557°	0.164m, 11.312°	0.025m, 2.256°	0.028m, 2.946°	0.020m, 1.893°	0.025m, 2.591°
shapes translation	0.238m, 6.001°	0.252m, 7.519°	0.264m, 6.235°	0.269m, 7.585°	0.035m, 2.117°	0.039m, 2.809°	0.035m, 2.321°	0.032m, 2.523°
box translation	0.193m, 6.977°	0.212m, 8.184°	0.190m, 6.636°	0.213m, 7.995°	0.036m, 2.195°	0.042m, 2.486°	0.032m, 1.977°	0.041m, 1.825°
dynamic 6dof	0.297m, 9.332°	0.298m, 11.242°	0.296m, 8.963°	0.293m, 11.069°	0.031m, 2.047°	0.036m, 2.576°	0.031m, 1.912°	0.042m, 2.594°
hdr poster	0.282m, 8.513°	0.296m, 10.919°	0.290m, 8.710°	0.308m, 11.293°	0.051m, 3.354°	0.060m, 4.220°	0.042m, 2.945°	0.059m, 3.883°
poster translation	0.266m, 6.516°	0.282m, 8.066°	0.264m, 5.459°	0.274m, 7.232°	0.036m, 2.074°	0.041m, 2.564°	0.038m, 2.145°	0.040m, 2.415°
Average	0.231m, 7.455°	0.246m, 9.124°	0.241m, 7.593°	0.254m, 9.414°	0.036m, 2.341°	0.041m, 2.934°	**0,031m, 2.198°**	**0.039m, 2.638°**

Table 3.9: Comparison between results from our fourth architecture (transforms-based) 3.2.4 and the results of PoseNet Kendall et al. (2015), Bayesian PoseNet Kendall and Cipolla (2016) and SP-LSTM Nguyen et al. (2019). The evaluation is performed using the random split protocol.

	PoseNet		Bayesian PoseNet		SP-LSTM		Ours	
	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error	Median Error	Average Error
shapes rotation	0.201m, 12.499°	0.214m, 13.993°	0.164m, 12.188°	0.191m, 14.213°	0.045m, 5.017°	0.049m, 11.414°	0.049, 3.581°	0.048m, 6.901°
shapes translation	0.198m, 6.969°	0.222m, 8.866°	0.213m, 7.441°	0.228m, 10.142°	0.072m, 4.496°	0.081m, 5.336°	0.064m, 4.685°	0.071m, 5.384°
shapes 6dof	0.320m, 13.733°	0.330m, 18.801°	0.326m, 13.296°	0.329m, 18.594°	0.078m, 5.524°	0.095m, 9.532°	0.072m, 5.875°	0.093m, 7.652°
Average	0.240m, 11.067°	0.255m, 13.887°	0.234m, 10.975°	0.249m, 14.316°	0.065m, 5.012°	0.075m, 8.761°	**0.061m, 4.713°**	**0.070m, 6.645°**

Table 3.10: Comparison between results from our fourth architecture (transforms-based) 3.2.4 and the results of PoseNet Kendall et al. (2015), Bayesian PoseNet Kendall and Cipolla (2016) and SP-LSTM Nguyen et al. (2019).

The evaluation is performed using the novel split protocol.

are known to be data hungry, and we anticipate that higher performance could be achieved with a larger training dataset. It's worth noting that our bilinear and improved bilinear pooling methods could potentially be combined with the transformer-based methods, which might further enhance the model's performance. However, we halted our search for the best model architecture configuration as we have already achieved superior performance compared to the state-of-the-art. Our current focus is now shifting towards the analysis of raw event data rather than event images.

3.4 Conclusion

This chapter introduced a pioneering approach to processing images captured by event cameras, leveraging their unique ability to record pixel intensity changes as events rather than frames. The method presented exploits the high temporal resolution and low latency of event cameras through

Methods	Median error		Average error	
	Metre (m)	Degree (°)	Metre (m)	Degree (°)
Random split				
CNN-LSTM	0.019	1.591	0.022	1.994
Bilinear pooling	0.03	2.204	0.034	2.708
Improved BP	0.024	1.653	0.029	1.951
Transformers	0.031	2.198	0.039	2.638
Novel split				
CNN-LSTM	0.056	3.233	0.066	4.417
Bilinear pooling	0.061	3.448	0.07	6.742
Improved BP	0.052	4.066	0.062	5.769
Transformers	0.061	4.713	0.070	6.645

Figure 3.10: A comparison of the proposed method performance.

an innovative technique that transforms the event stream into interpretable "event images" using spatial and temporal binning.

Furthermore, we introduced the utilization of deep learning models to process these event images, presenting four distinct model architectures: (1) CNN-LSTM-Dense Model, (2) Dual-CNN with Bilinear Pooling, (3) Improved Bilinear Pooling, and (4) Transformer-Based Model with Self-Attention. A series of experiments were conducted on various synthetic and real datasets, demonstrating the effectiveness and superiority of the proposed models over state-of-the-art methods. The proposed models that performs the best over all proposed ones in the CNN-LSTM-Dense Model. We believe this is because we conducted an intensive investigation to identify the optimal CNN architecture for extracting relevant features. The LSTM serves as a spatial LSTM that combines the features extracted by the CNN. This approach also enhances the global features, as the LSTM captures redundancy and correlations between features.

Building upon these foundations, the next phase of our exploration will

focus on a compelling alternative approach. In the upcoming chapter, we will shift our attention to the direct application of deep learning methods on raw event data, bypassing the intermediate step of converting events into event images.

Chapter 4

Pose Estimation From raw event data

4.1 Introduction

In this chapter, our primary focus is on an in-depth exploration of deep learning architectures that possess a unique capability: the ability to reason about raw, unprocessed events that are represented as point clouds. Traditional convolutional architectures have been highly effective in handling input data with a well-defined and regular structure, such as images arranged in grids or 3D voxel representations. These regular formats facilitate weight sharing and optimizations within convolutional kernels, leading to efficient feature extraction.

However, when dealing with events – which are essentially discrete occurrences in time and space – the data lacks this inherent regularity. As a result, the conventional approach often involves a preprocessing step, where the event data is transformed into a more regular format like 3D voxel grids or collections of images. This transformation allows the use of existing deep neural network architectures, which are designed to operate on structured input formats. As discussed in the previous chapter, this preprocessing step involves converting the event data into a compatible representation prior to inputting it into the neural network.

Unfortunately, this conversion process comes with certain drawbacks. One significant concern is the potential expansion of data volume. Converting event data into voxel grids or image collections can lead to a substantial increase in the amount of data being processed, which may impact computational efficiency and memory requirements.

Moreover, this transformation introduces a risk of introducing quantization artifacts. Quantization involves discretizing continuous data into a finite number of levels, which can lead to loss of information and fidelity in the original data. In the context of event data, such quantization artifacts can obscure the natural invariances and nuanced patterns that exist within the raw event representation. This, in turn, might hinder the performance of the deep learning model and limit its ability to capture the true essence of the data.

Therefore, the central theme of this chapter is to propose an innovative and alternative approach to handling event data. Specifically, our focus shifts towards utilizing point clouds as the input representation for deep learning architectures. Point clouds, which are collections of individual data points each carrying information about an event's location and possibly other attributes, provide a more direct and unadulterated representation of the event data. By directly working with point clouds, we aim to bypass the need for the data transformation steps that can lead to unnecessary data volume expansion and quantization artifacts.

In essence, this chapter delves into the exploration of deep learning methodologies that can effectively operate on raw event data in the form of point clouds. By embracing the inherent characteristics and irregularities of event data, we aspire to develop architectures that can capture the true essence of these unprocessed occurrences, thereby potentially leading to more accurate and meaningful insights from the data. Through our investigation, we aim to demonstrate the advantages and challenges of utilizing point clouds as an alternative input representation for deep learning models, ultimately contributing to the advancement of understanding and leveraging event data in various applications.

4.2 Problem Statement

Traditional cameras, as presented in the previous chapters, capture images by sampling the intensity of light at fixed time intervals, resulting in a sequence of frames. On the other hand, event cameras operate based on the principle of temporal contrast, where they detect changes in pixel intensity (brightness) asynchronously and independently. This means that they only transmit information when there is a significant change in the scene, such as motion or lighting changes. Each of these events is characterized by its pixel location (x, y), a timestamp t, and a polarity (indicating whether the change is a brightness increase or decrease).

The proposed approach introduces a deep learning framework designed specifically to work with data captured by event cameras. The key innovation is that instead of representing each event as a separate entity, it suggests treating each event as a 3D point (x, y, t). In this representation, x and y still correspond to the pixel coordinates, while t represents the timestamp of the event.

By treating events as 3D points, the framework can take advantage of existing deep learning architectures that are designed to process 3D point sets. This is a departure from traditional approaches that often involve converting event data into image sequences or applying specialized event-based processing algorithms.

4.3 Network architecture

Our network draws inspiration from the PointNet architecture Qi et al. (2017) initially designed for point cloud classification. However, a key distinction lies in our approach, as we leverage a valuable attribute inherent to our event point cloud data: its temporal order.

4.3.1 PointNet architecture

PointNet Qi et al. (2017) is a deep learning architecture designed for processing and understanding point cloud data, which is commonly used in 3D data applications like computer vision and robotics Jiang et al. (2020).

1. PointNet takes a set of points $\{x_1, x_2, ..., x_n\}$, each represented by its 3D coordinates, as input. These points are typically sampled from a 3D scene.

2. The first step in PointNet is to embed each point into a higher-dimensional space to capture local and global features. This is achieved through a shared multi-layer perceptron (MLP) applied independently to each point:

   $$f_{\text{MLP}} : \mathbb{R}^3 \rightarrow \mathbb{R}^k$$

   The output of this MLP is a new feature vector $\{f_{\text{MLP}}(x_1), f_{\text{MLP}}(x_2), ..., f_{\text{MLP}}(x_n)\}$ for each point.

3. To capture global features from the entire point set, PointNet aggregates information from all the point embeddings. This is done by taking the max or average over all the point embeddings:

   $$g = \text{MAX/AVERAGE}\ \{f_{\text{MLP}}(x_1), f_{\text{MLP}}(x_2), ..., f_{\text{MLP}}(x_n)\}$$

   The resulting global feature g summarizes the entire point cloud.

4. PointNet then applies another MLP to each point's embedding to further refine the features:

   $$f'_{\text{MLP}} : \mathbb{R}^k \rightarrow \mathbb{R}^{d'}$$

   $$\{f'_{\text{MLP}}(f_{\text{MLP}}(x_1)), f'_{\text{MLP}}(f_{\text{MLP}}(x_2)), ..., f'_{\text{MLP}}(f_{\text{MLP}}(x_n))\}$$

5. The refined point-wise features are aggregated to form a global fea-

ture vector $\hat{g}$:

$$\hat{g} = \text{MAX/AVERAGE}\,\{f'_{\text{MLP}}(f_{\text{MLP}}(x_1)), f'_{\text{MLP}}(f_{\text{MLP}}(x_2)), ..., f'_{\text{MLP}}(f_{\text{MLP}}(x_n))\}$$

This global feature $\hat{g}$ captures the higher-level representation of the entire point cloud.

6. Finally, the global feature $\hat{g}$ is fed into a fully connected layer for classification or regression:

$$y = \text{FC}(\hat{g})$$

Where y represents the predicted class probabilities (for classification) or continuous outputs (for regression).

PointNet distinguishes itself by directly handling point cloud data, deviating from models that first convert points into images. The shared multilayer perceptron (MLP) f_{MLP} within the PointNet architecture embeds the coordinates of each 3D point into a higher-dimensional feature space, forming the foundation for detailed feature extraction within the point cloud. Through subsequent global feature aggregation steps, whether through MAX or AVERAGE operations, PointNet synthesizes these point embeddings, capturing overarching trends across the entirety of the point cloud. The point-wise feature learning MLP f'_{MLP} then refines these representations, iteratively enhancing context-specific details. Following this refinement, another round of global feature aggregation amalgamates the refined point-wise features into a coherent global feature vector $\hat{g}$. Finally, the classification/regression head translates these enriched feature representations into actionable outputs, showcasing PointNet's comprehensive approach from initial transformation to final output. This process underscores the significant computational prowess of PointNet in effectively processing and interpreting 3D point cloud data.

4.3.2 Adapting PointNet for event camera pose estimation

Adapting the PointNet architecture to work with processed event camera data involves modifying the network architecture to handle the unique characteristics of event data, its temporal nature and sparsity.

Temporal Integration: Event data is inherently temporal, capturing changes in the scene over time. To incorporate this temporal aspect, we considered using recurrent neural network (RNN) layers such as Long Short-Term Memory (LSTM) after the PointNet layers. These recurrent layers can capture the sequential dependencies in the event frames and help the network understand the motion and dynamics in the scene.

PointNet Layers: The core of your architecture will still be based on PointNet layers. However, since our data now consists of event points with three coordinates (x, y, t), we need to modify the input channels of the network accordingly. Instead of using traditional 3D coordinates (x, y, z), we used (x, y, t) as the coordinates for each point. We represent the point cloud as a 3D tensor of shape (num_points, 3), where each row corresponds to a point's (x, y, t) coordinates.

PointNet is designed to work with unordered point sets, but event data arrives in a stream with timestamps. To account for this, the adaptation might involve grouping events that occur within a short time interval into "event frames." Each event frame captures a snapshot of the scene's changes during that time window. These event frames can then be treated as point sets, and the initial PointNet layers process them to extract local features.

After the initial PointNet processing, the extracted features from each event frame is fed into the LSTM layers. The LSTM units can then capture the temporal patterns and dependencies between event frames. This allows the network to learn the underlying motion patterns and dynamics present in the event data, which is crucial for our camera pose estimation task.

Predicting Poses and loss functions: The final layers of our adapted architecture is responsible for predicting the camera poses. We used fully connected layers to map the extracted features to pose parameters. We have defined an appropriate loss function that captures the pose estimation task. This involves a combination of translation and rotation losses. We used the mean squared error (MSE) for translation and quaternion loss for rotation.

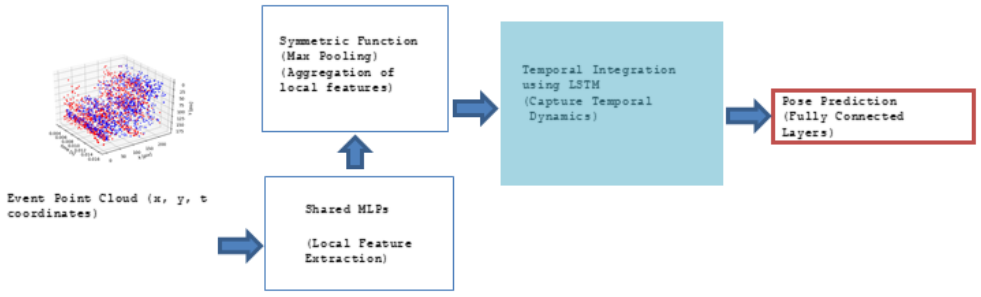


Figure 4.1: An advanced neural network architecture designed for pose estimation from raw event data.
At its core is the Event Point Cloud Representation that inputs 3D tensors representing spatial and temporal event data. The network employs Shared MLPs for extracting local features from each event point, followed by Max Pooling for feature aggregation. It then integrates temporal dynamics using LSTM layers, culminating in a Pose Prediction module that outputs the estimated camera pose. This architecture uniquely combines PointNet's spatial processing with LSTM's temporal modeling

4.4 Experiments and results

The evaluation of our event camera pose estimation method using PointNet on the hdr_poster dataset, as depicted in Figure 4.2, showcases compelling results. When examining the median error of translation, our method emerges as the frontrunner, surpassing the performance of state-of-the-art methods. This heightened precision is particularly noteworthy, with our

method exhibiting approximately one centimeter higher accuracy in camera translation compared to existing approaches. This improved precision is crucial in applications where accurate spatial positioning is paramount, such as augmented reality and robotic navigation.

Moving on to Figure 4.3, which illustrates the median error of rotation, we observe a similar trend. Our method consistently outperforms other state-of-the-art methods, demonstrating its robustness and accuracy in estimating rotational poses. This aspect is crucial for applications requiring precise orientation determination, such as 3D scene reconstruction and object manipulation tasks.

Furthermore, Figure 4.4 and Figure 4.5 provides insights into the mean and deviation of the translation/orientation error between the ground truth and the predicted translation/orientation. Here again, our method shines, showcasing lower mean errors and tighter error distributions compared to competing methods. This indicates that our approach not only achieves higher accuracy in individual predictions but also maintains a more consistent performance across different translation and orientation estimations.

These results highlight the effectiveness and superiority of our proposed event camera pose estimation method using PointNet. By leveraging the capabilities of PointNet and its ability to directly process point cloud data, we have achieved remarkable accuracy in both translation and rotation estimation. These findings bode well for real-world applications where precise and reliable camera pose estimation is paramount for tasks ranging from robotics to augmented reality experiences.

4.5 Conclusion

The proliferation of event cameras has ushered in an exciting frontier in computer vision. Unlike traditional cameras, which operate on periodic sampling, event cameras respond to dynamic changes in the environment. This results in data that is not just high in temporal resolution but also intrinsically adaptive to the scene's dynamics.

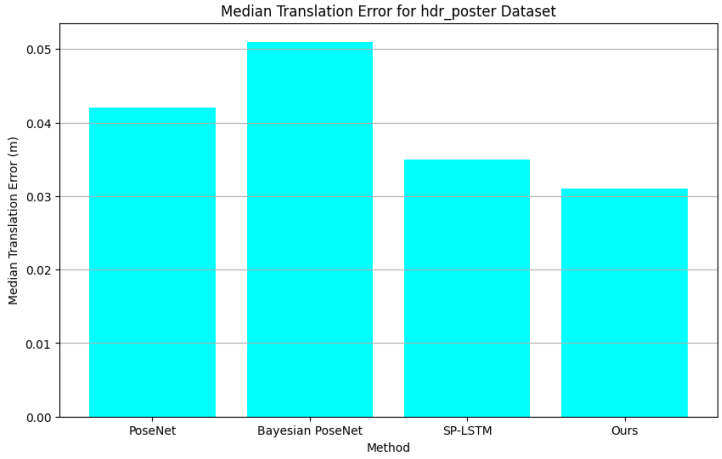


Figure 4.2: Median error of translation for the hdr_poster dataset.

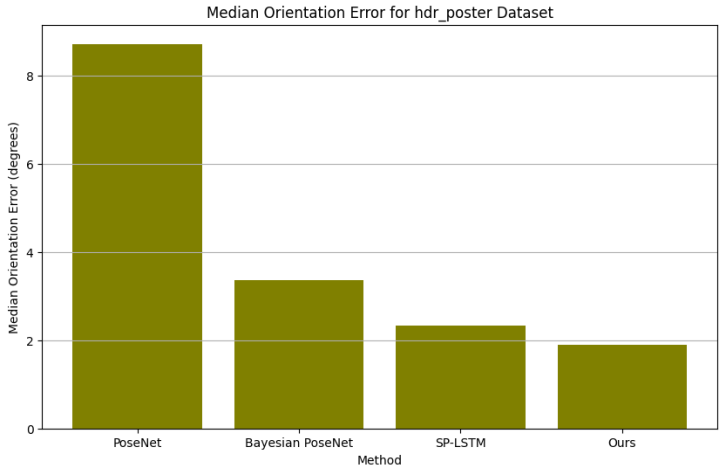


Figure 4.3: Median error of rotation for the hdr_poster dataset.

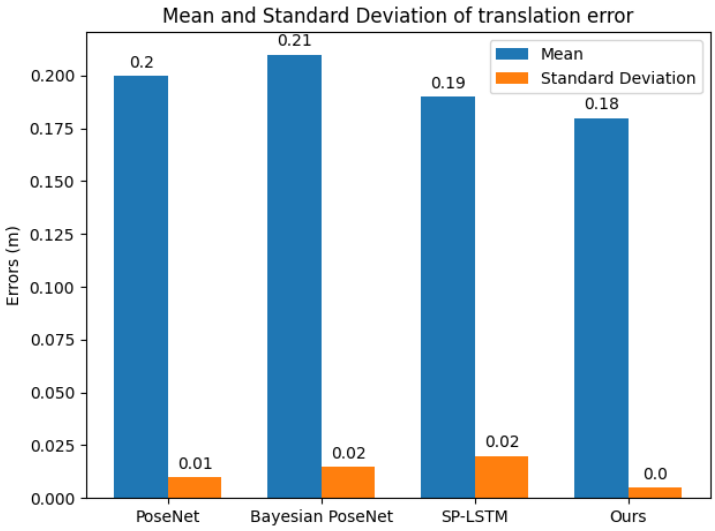


Figure 4.4: Distribution of translation error.

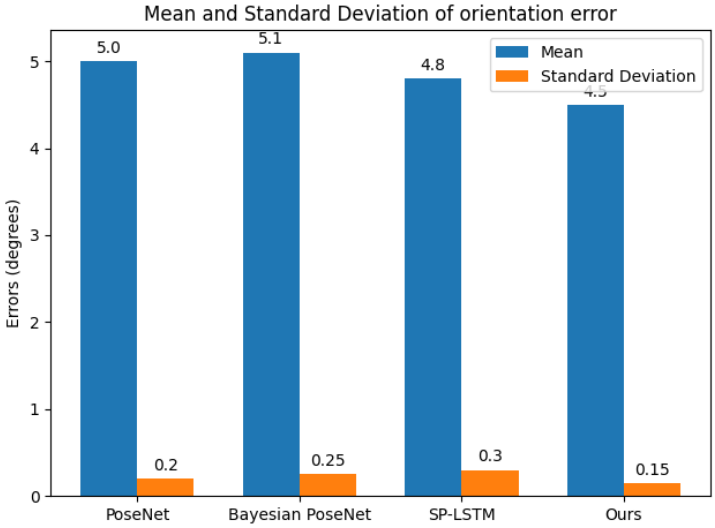


Figure 4.5: Distribution of rotation error.

However, this novel data format also presents fresh challenges. Conventional neural architectures, groomed to devour structured, grid-like data, are caught off-foot by the erratic and asynchronous nature of event data. This is where the allure of preprocessing comes in. But, as with any translation, converting raw events into more digestible formats risks loss of nuances and introduces complexities that could be avoided.

The beauty of our approach lies in its simplicity and directness. By adapting PointNet, we are acknowledging and preserving the inherent nature of event data. We are not trying to retrofit the data into pre-existing paradigms but are reshaping our tools to understand this new kind of information. The spatial intricacies captured by PointNet are beautifully complemented by the temporal modeling prowess of LSTMs, creating a harmony between space and time.

What this endeavor underscores is the flexibility and potential of deep learning. Neural networks, often criticized for being black boxes, here showcase their capability to evolve, adapt, and cater to unconventional data sources. It's a testament to the philosophy that rather than forcing data to fit our models, we can, and should, mold our models to respect the essence of the data.

In conclusion, the interplay between event cameras and deep learning is still in its nascent stages. Yet, the early results are promising and indicative of the vast possibilities ahead. We envision a future where event-based data processing becomes the norm rather than the exception, powering applications in robotics, augmented reality, and beyond. Our work is just a stepping stone in this expansive journey, and we eagerly invite the community to join us in this exploration

Chapter 5

Conclusion and Future Work

In this thesis, we have made significant contributions to the field of event camera pose estimation using deep learning techniques. Our work has aimed to push the boundaries of accuracy and efficiency in camera pose estimation by leveraging the unique characteristics of event data. The key contributions of this thesis can be summarized as follows:

Firstly, we provided a thorough overview of the background information and related works, laying a solid foundation and contextual understanding of event camera pose estimation. This served as the starting point for our exploration into novel deep learning approaches tailored for event camera pose estimation.

Secondly, we introduced methods to map event data to image-like data, enabling the application of dedicated deep learning approaches. This mapping process was crucial in effectively utilizing the rich event information for camera pose estimation.

Thirdly, we proposed a groundbreaking approach that directly applies deep learning techniques to raw event data, treating it as a point cloud rather than converting it into images. This innovative method allowed for an end-to-end learning process, harnessing the complete information captured by the event camera.

Our main focus was the development and evaluation of four distinct model architectures for event-based pose estimation. The first model com-

bined a Convolutional Neural Network (CNN) for feature extraction, a Long Short-Term Memory (LSTM) for capturing spatial dependencies, and a dense fully connected layer for 6-DoF pose estimation. The second model utilized two CNNs with bilinear pooling for improved performance, while the third model introduced improved bilinear pooling with matrix function normalization. Lastly, the fourth model employed a self-attention mechanism with a Transformer encoder, generating a context-aware representation for event patches.

We conducted a series of experiments using diverse evaluation protocols, and the results were promising, showcasing the superiority of our proposed approaches compared to state-of-the-art methods. Our models exhibited impressive accuracy and efficiency in estimating camera poses from event data.

In Chapter 4, we delved into our novel approach of directly applying deep learning techniques to raw event data, treating it as a point cloud. This approach aimed to unlock the full potential of event data, enabling accurate camera pose estimation through an end-to-end learning process. Our experiments yielded promising results, indicating the vast potential of this approach.

This thesis has significantly advanced the field of event camera pose estimation by introducing novel deep learning approaches and model architectures. By effectively harnessing the unique characteristics of event data and employing state-of-the-art deep learning techniques, we have achieved remarkable accuracy and efficiency in camera pose estimation. The contributions of this thesis pave the way for further advancements in event-based vision systems, offering new avenues for research and development in this exciting field.

Future Work

While our research has achieved significant milestones in event camera pose estimation, the field offers numerous avenues for future exploration and enhancement:

Hybrid Architectures: Combining the strengths of PointNet with other neural network architectures presents an exciting opportunity. Hybrid models could potentially leverage the robust feature learning of PointNet alongside the spatial hierarchies of CNNs or the sequential memory of LSTM networks, resulting in a more potent model with enhanced performance.

Real-time Applications: Having established the efficacy of our model, the next step is its deployment in real-time applications. Industries such as augmented reality (AR), virtual reality (VR), robotics, and drone navigation could greatly benefit from a fast and accurate camera pose estimation system. Integrating our model into these applications could pave the way for innovative solutions in these domains.

Extended Datasets: To further bolster the robustness and generalization capabilities of our model, training on larger and more diverse datasets is crucial. These datasets could encompass various environmental conditions, lighting scenarios, and camera types, ensuring that the model can effectively handle a wide range of real-world scenarios.

Model Compression and Optimization: With the rise of edge computing and mobile platforms, there is a growing demand for lightweight models that can deliver accurate results without requiring substantial computational resources. Exploring techniques for model compression and optimization could make our pose estimation model more accessible and efficient for deployment on resource-constrained devices.

Incorporating Uncertainty: An important aspect of model reliability is the ability to quantify uncertainty. By incorporating Bayesian Neural Networks or other uncertainty quantification methods, our model can not only provide predictions but also offer insights into the confidence or uncertainty associated with those predictions. This could be invaluable in critical applications where understanding the model's level of certainty is essential for decision-making.

The future of event camera pose estimation holds exciting possibilities. By exploring these avenues for improvement, we can continue to advance the field, creating more robust, efficient, and reliable models that can be seamlessly integrated into a wide range of applications.

Bibliography

M. Azizian, M. Khoshnam, N. Najmaei, and R. V. Patel. Visual servoing in medical robotics: a survey. part i: endoscopic and direct vision imaging–techniques and applications. *The international journal of medical robotics and computer assisted surgery*, 10(3):263–274, 2014.

P. A. Bardow. *Estimating general motion and intensity from event cameras*. PhD thesis, Imperial College London, 2019.

C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck. A 240 $\times$ 180 130 db 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 49(10):2333–2341, 2014.

M. Byambaa, G. Koutaki, and L. Choimaa. 6d pose estimation of transparent object from single rgb image for robotic manipulation. *IEEE Access*, 10:114897–114906, 2022.

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

J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VII 12*, pages 430–443. Springer, 2012.

W. Chen, C. Yu, C. Tu, Z. Lyu, J. Tang, S. Ou, Y. Fu, and Z. Xue. A survey on hand pose estimation with wearable sensors and computer-vision-based methods. *Sensors*, 20(4):1074, 2020.

C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20: 273–297, 1995.

A. Cutler, D. R. Cutler, and J. R. Stevens. Random forests. *Ensemble machine learning: Methods and applications*, pages 157–175, 2012.

T. Delbruck, Y. Hu, and Z. He. V2e: From video frames to realistic dvs event camera streams. arxiv 2020. *arXiv preprint arXiv:2006.07722*.

L. Di Angelo, P. Di Stefano, and E. Guardiani. A review of computer-based methods for classification and reconstruction of 3d high-density scanned archaeological pottery. *Journal of Cultural Heritage*, 56:10–24, 2022.

A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

T. Finateu, A. Niwa, D. Matolin, K. Tsuchimoto, A. Mascheroni, E. Reynaud, P. Mostafalu, F. Brady, L. Chotard, F. LeGoff, et al. 5.10 a 1280$\times$ 720 back-illuminated stacked temporal contrast event-based vision sensor with 4.86 μm pixels, 1.066 geps readout, programmable event-rate controller and compressive data-formatting pipeline. In *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, pages 112–114. IEEE, 2020.

M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

G. Gallego and D. Scaramuzza. Accurate angular velocity estimation with an event camera. *IEEE Robotics and Automation Letters*, 2(2):632–639, 2017.

G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, et al. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020.

D. Gehrig, A. Loquercio, K. G. Derpanis, and D. Scaramuzza. End-to-end learning of representations for asynchronous event-based data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5633–5643, 2019.

M. Gehrig, S. B. Shrestha, D. Mouritzen, and D. Scaramuzza. Event-based angular velocity regression with spiking networks. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4195–4202. IEEE, 2020.

R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *2011 International Conference on Computer Vision*, pages 415–422. IEEE, 2011.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE transactions on pattern analysis and machine intelligence*, 34(5): 876–888, 2011.

S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of

texture-less 3d objects in heavily cluttered scenes. In *Computer Vision–ACCV 2012: 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part I 11*, pages 548–562. Springer, 2013.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and Pattern recognition*, pages 4867–4876, 2020.

A. Kendall and R. Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *2016 IEEE international conference on Robotics and Automation (ICRA)*, pages 4762–4769. IEEE, 2016.

A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. J. Davison. Simultaneous mosaicing and tracking with an event camera. *J. Solid State Circ*, 43:566–576, 2008.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6): 84–90, 2017.

B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza. Low-latency visual odometry using event-based feature tracks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 16–23. IEEE, 2016.

X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman. Hots: a hierarchy of event-based time-surfaces for pattern recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1346–1359, 2016.

X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman. Hots: A hierarchy of event-based time-surfaces for pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7):1346–1359, 2017.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

P. Lichtsteiner, C. Posch, and T. Delbruck. A 128×128 $120\,db$ $15\,\mu s$ latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576, 2008.

T.-Y. Lin, A. RoyChowdhury, and S. Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015.

A. Linares-Barranco, F. Perez-Pena, D. P. Moeys, F. Gomez-Rodriguez, G. Jimenez-Moreno, S.-C. Liu, and T. Delbruck. Low latency event-based filtering and feature extraction for dynamic vision sensors in real-time fpga applications. *IEEE Access*, 7:134926–134942, 2019.

G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.

D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *Acm transactions on graphics (tog)*, 36 (4):1–14, 2017.

A. Milosavljević, A. Dimitrijević, and D. Rančić. Gis-augmented video surveillance. *International Journal of Geographical Information Science*, 24(9): 1415–1433, 2010.

E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, 36(2):142–149, 2017.

G. Munda, C. Reinbacher, and T. Pock. Real-time intensity-image reconstruction for event cameras using manifold regularisation. *International Journal of Computer Vision*, 126:1381–1393, 2018.

F. Munoz-Montoya, M.-C. Juan, M. Mendez-Lopez, and C. Fidalgo. Augmented reality based on slam to assess spatial short-term memory. *IEEE Access*, 7:2453–2466, 2018.

A. Nguyen, T.-T. Do, D. G. Caldwell, and N. G. Tsagarakis. Real-time pose estimation for event cameras with stacked spatial lstm networks. *arXiv preprint arXiv:1708.09011*, 3, 2017.

A. Nguyen, T.-T. Do, D. G. Caldwell, and N. G. Tsagarakis. Real-time 6dof pose relocalization for event cameras with stacked spatial lstm networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai,

and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

D. Patel, H. Sanghvi, N. K. Jadav, R. Gupta, S. Tanwar, B. C. Florea, D. D. Taralunga, A. Altameem, T. Altameem, and R. Sharma. Blockcrime: Blockchain and deep learning-based collaborative intelligence framework to detect malicious activities for public safety. *Mathematics*, 10(17): 3195, 2022.

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*, 102(10):1470–1484, 2014.

C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

J. Raja, P. Shanmugam, and R. Pitchai. An automated early detection of glaucoma using support vector machine based visual geometry group 19 (vgg-19) convolutional neural network. *Wireless Personal Communications*, 118:523–534, 2021.

H. Rebecq, T. Horstschaefer, and D. Scaramuzza. Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization. 2017.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

A. Sagheer and M. Kotb. Unsupervised pre-training of a deep lstm-based stacked autoencoder for multivariate time series forecasting problems. *Scientific reports*, 9(1):19038, 2019.

M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

M. K. Singh, Snehmani, R. Gupta, A. Bhardwaj, P. Joshi, and A. Ganju. High resolution dem generation for complex snow covered indian himalayan region using ads80 aerial push-broom camera: a first time attempt. *Arabian Journal of Geosciences*, 8:1403–1414, 2015.

A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14:199–222, 2004.

B. Son, Y. Suh, S. Kim, H. Jung, J. Kim, C. Shin, K. Park, K. Lee, J. Park, J. Woo, et al. A 640$\times$ 480 dynamic vision sensor with a 9 um pixel and 300 meps address-event representation. In *Proceedings of the IEEE International Conference on Solid-State Circuits*.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger. Elasticfusion: Real-time dense slam and light source estimation. *The International Journal of Robotics Research*, 35(14):1697–1716, 2016.

F. Xiao, J. M. DiCarlo, P. B. Catrysse, and B. A. Wandell. High dynamic range imaging of natural scenes. In *Color and imaging conference*, volume 2002, pages 337–342. Society for Imaging Science and Technology, 2002.

K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad. An overview to visual odometry and visual slam: Applications to mobile robotics. *Intelligent Industrial Systems*, 1(4):289–311, 2015.

B. Zhang, W. Huang, J. Li, C. Zhao, S. Fan, J. Wu, and C. Liu. Principles, developments and applications of computer vision for external quality inspection of fruits and vegetables: A review. *Food Research International*, 62:326–343, 2014.

H. Zhang, G. Wang, Z. Lei, and J.-N. Hwang. Eye in the sky: Drone-based object tracking and 3d localization. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 899–907, 2019.

A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 3(3):2032–2039, 2018.