Unifying Motion Segmentation, Estimation, and Tracking for Complex Dynamic Scenes

Kevin M. Judd

Somerville College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity 2019

Acknowledgements

First and foremost, I would like to thank my supervisors, Dr. Jonathan Gammell and Professor Paul Newman. Paul has been a great source of guidance and encouragement in navigating both the research and bureaucracy of academia. Jon has been incredibly generous in sharing his time and experience to support me in my work and to help me become a better researcher. He also patiently edited and reviewed my work, and never shied from a healthy debate over a comma. I endeavor to reflect his dedication in my own life, and I look forward to seeing the Estimation, Search, and Planning group grow and flourish under his leadership.

I would like to thank my examiners, Professor Jonathan Kelly and Professor Victor Adrian Prisacariu, for taking the time to evaluate my thesis. I am also grateful to the many teachers, professors, and mentors throughout my life who, official or not, have helped me learn to be the person I am today. Many of the people in my life who have influenced me the most may have had no idea they were doing anything noteworthy for me to begin with.

To my colleagues at the Oxford Robotics Institute, as well as Wendy Adams and my fellow students in the Autonomous Intelligent Machines and Systems CDT, it was a pleasure to work and learn alongside you. I hope our research continues to propel us forward and leads to us working together again. I would also like to thank the team at Prophesee, who have enabled my exploration of event cameras by providing their equipment and technical support.

To my fellow graduate members of Somerville College, thank you for keeping me sane and helping me to balance my work with a healthy dose of laughter and a few good pints. Thank you Allison, for always pushing me to be the best that I can be and for reminding me what really matters; and thank you Fraser, for being a steadfast friend to me since I arrived here.

Finally, I want to thank my family, whose unending love and support have carried me through the long hours of work. My parents, Bob and Aileen, have always pushed me to indulge my curiosity and pursue my interests, and everything I have achieved is in part due to their influence. I'm grateful to them and to my sister, Katie, for always answering my late-night calls, thanks to a sizable time difference, and to my grandmothers, aunts, uncles, and cousins for never letting me forget that my research is cool but I should come home more often. I aspire to be half as great as they think I am.

Abstract

Visual navigation is an critical task in mobile robotics. To navigate through an area, a robot must understand *where* it is, *what* is around it, and *how* to get to its goal. Integral to each of these questions is the task of motion analysis. Estimating the egomotion of a sensor is a well-studied problem, but only recently have similar questions about the *other* dynamic objects in the scene been addressed. Understanding the static structure of an environment is crucial to navigating through it, but understanding the motions of other dynamic objects is crucial to doing so *safely*.

Previous work has developed techniques to estimate the motion of a moving camera in a largely static environment and to segment or track motions in a dynamic scene using known camera motions. It is more challenging to estimate the unknown motions of the camera and the dynamic scene simultaneously, and this thesis focuses on addressing this multimotion estimation problem (MEP).

The estimation of *third-party* dynamic motions is much more difficult than estimating the sensor egomotion. The static assumption that applies to the background of a scene has no analogue for these dynamic objects, whose observed motions comprise both their arbitrary real-world motions and the camera egomotion.

This thesis introduces Multimotion Visual Odometry (MVO), a novel multimotion estimation pipeline that incorporates motion segmentation and tracking techniques into the traditional visual odometry pipeline in order to estimate the full $SE(3)$ trajectory of *every* motion in the scene, including the egomotion. MVO segments and estimates all motions *simultaneously*, treating them all equivalently until the segmentation converges, after which the egomotion can be determined and used to calculate all other motions in a geocentric frame.

Highly dynamic scenes also tend to exhibit significant occlusions, which make accurate motion estimation and object tracking even more challenging. A physically founded continuous motion prior introduced to the MVO pipeline to extrapolate temporarily occluded motions and reacquire them when they become unoccluded. This *motion closure* procedure maintains trajectory consistency and allows the pipeline to estimate and track multiple $SE(3)$ motions, even in the presence of occlusion.

The estimation accuracy of MVO is evaluated quantitatively and qualitatively using real-world data from stereo RGB and event cameras in several highly dynamic multimotion scenes. Much of this data was published in the Oxford Multimotion Dataset (OMD), which was designed to explore the MEP and serve as a scaffold for the development and evaluation of new multimotion estimation techniques.

Contents

List of Figures

List of Notation

General Notation

a, A	:	Lower- and upper-case variables or functions in this font are scalar valued.
$\mathscr{A}$	:	Upper-case variables or functions in this font are set or sequence valued.
$\mathbf{a}$	:	Lower-case variables or functions in this font are vector valued.
$\mathbf{A}$	:	Upper-case variables or functions in this font are matrix valued.
$\dot{\mathbf{a}}$	:	Variables with a dot are time derivatives.
$\bar{\mathbf{a}}$	:	Variables with an overbar are mean quantities.
$\check{\mathbf{a}}$	:	Variables with a check are extrapolated or interpolated quantities.

Specific Symbols

$\mathbf{1}$	:	The identity matrix.
$\mathbf{0}$	:	The zero matrix.
ℓ	:	A motion label.
$\mathscr{L}$	:	The set of motion labels.
$\mathscr{P}$	:	The set of tracklets.
$\mathcal{N}$	:	The neighborhood graph.
K	:	The number of frames in a batch estimation window.

Geometric Symbols

$SO(n)$	:	The special orthogonal group of n-dimensional rotations.
$\mathfrak{so}(n)$	:	The Lie algebra associated with $SO(n)$.
$SE(n)$	:	The special Euclidean group of rigid n-dimensional transforms.
$\mathfrak{se}(n)$	:	The Lie algebra associated with $SE(n)$.
$\mathbf{u}^a$	:	The homogeneous vector representing the 2D image space point a of the form $\begin{bmatrix} u & v & 1 \end{bmatrix}^T$
$\boldsymbol{u}^a$	:	The homogeneous vector representing the 3D image space point a of the form $\begin{bmatrix} u & v & d & 1 \end{bmatrix}^T$
$\underrightarrow{\boldsymbol{\mathcal{F}}}_A$	:	The 3D reference frame C.
$\underrightarrow{p}^a$	:	The 3D world point a.
$\mathbf{p}_C^{ab}$	:	The vector from point b to point a, expressed in $\underrightarrow{\boldsymbol{\mathcal{F}}}_C$.

$\boldsymbol{p}_C^{ab}$	:	The homogeneous vector from point b to point a, expressed in $\underrightarrow{\mathcal{F}}_C$.
$\mathcal{p}_C^a$	:	The sequence of homogeneous vectors representing measurements of the 3D world point a, expressed in $\underrightarrow{\mathcal{F}}_C$ (this is often referred to as a *tracklet*).
$\mathbf{C}_{BA}$	:	The 3×3 rotation matrix that transforms vectors from frame $\underrightarrow{\mathcal{F}}_A$ to $\underrightarrow{\mathcal{F}}_B$, such that $\mathbf{p}_B^{cB} = \mathbf{C}_{BA}\mathbf{p}_A^{cB}$.
$\boldsymbol{\phi}_{BA}$	:	The $\mathbb{R}^{3\times1}$ representation of a member of $\mathfrak{so}(3)$ corresponding to $\mathbf{C}_{BA}$.
$\mathbf{T}_{BA}$	:	The 4×4 transform matrix in $SE(3)$ that transforms homogeneous points from frame $\underrightarrow{\mathcal{F}}_A$ to $\underrightarrow{\mathcal{F}}_B$, such that $\boldsymbol{p}_A^{cA} = \mathbf{T}_{AB}\boldsymbol{p}_B^{cB}$.
$\boldsymbol{\xi}_{BA}$	:	The $\mathbb{R}^{6\times1}$ representation of a member of $\mathfrak{se}(3)$ corresponding to $\mathbf{T}_{BA}$.
$\boldsymbol{\rho}_{BA}$	:	The $\mathbb{R}^{3\times1}$ representation of the translational component of $\boldsymbol{\xi}_{BA}$.
$\boldsymbol{\varpi}_C$	:	The 6×1 vector in $\mathfrak{se}(3)$ representing the velocity of $\underrightarrow{\mathcal{F}}_C$.
$\boldsymbol{\mathcal{T}}_{BA}$	:	The 6×6 adjoint of $\mathbf{T}_{BA}$.
$\mathcal{T}_C$	:	A sequence of $SE(3)$ transforms representing the motion of $\underrightarrow{\mathcal{F}}_C$.
${}^{\ell}\mathcal{T}_C$	:	The ℓ-th estimate of the motion of $\underrightarrow{\mathcal{F}}_C$.
$\mathcal{S}_C(t)$	:	The continuous-time trajectory of $\underrightarrow{\mathcal{F}}_C$, consisting of the continuous-time transforms, $\mathbf{T}_C(t)$, and velocities, $\boldsymbol{\varpi}_C(t)$.
$\mathcal{S}_{C_k}$	:	The discrete trajectory state corresponding to $\mathcal{S}_C(t_k)$ and consisting of $\mathbf{T}_{C_kC_1}$ and $\boldsymbol{\varpi}_{C_k}$.

Operators

$(\cdot)^{\wedge}$	:	Transforms a vector in $\mathbb{R}^3$ into a 3×3 skew-symmetric member of $\mathfrak{so}(3)$ and transforms a vector in $\mathbb{R}^6$ into a 4×4 member of $\mathfrak{se}(3)$.
$(\cdot)^{\vee}$	:	The inverse of $(\cdot)^{\wedge}$.
$\lvert\cdot\rvert$	:	The cardinality of a set or the determinant of a matrix.
$\lVert\cdot\rVert$	:	The L2 vector norm.
$(\cdot)^{\times}$	:	The cross-product or skew-symmetric matrix of a 3×1 vector.
$(\cdot)^{\curlywedge}$	:	Transforms a vector in $\mathbb{R}^6$ into a 6×6 matrix analogous to $(\cdot)^{\wedge}$ for $SE(3)$ adjoints.

1 Introduction

The field of robotics is accelerating rapidly as autonomous robotic platforms become more pervasive in society. The ubiquity of applications for mobile robotic platforms is matched by the variety of their environments. From indoor household and warehouse floors (Fig. 1.1a) to motorways (Fig. 1.1b) to the air (Fig. 1.1c) to outer space (Fig. 1.1d), autonomous robots are expected to interact with and navigate through a myriad of environments and challenges. Many platforms and algorithms are highly specialized to perform in a small subset of these environments, but the requirement for robust autonomous navigation tools will only increase as robotic platforms become more broadly adopted into new areas of society.

Autonomous vehicles have historically been widely used in largely static, well-structured environments, such as households and warehouses. These domains allow for several simplifying assumptions about the constancy of the environment. It may also be appropriate to assume the vehicle's kinematics constrain it to simple planar motions, but these assumptions break down as the complexity of the vehicle and the environment increase. As the applications of autonomous vehicles are expanded into more varied, dynamic domains, the need for robust motion analysis becomes more critical. These environments increasingly require autonomous vehicles to interact with, or at least navigate around, other dynamic objects, such as humans, human-driven vehicles, or other independent autonomous vehicles.

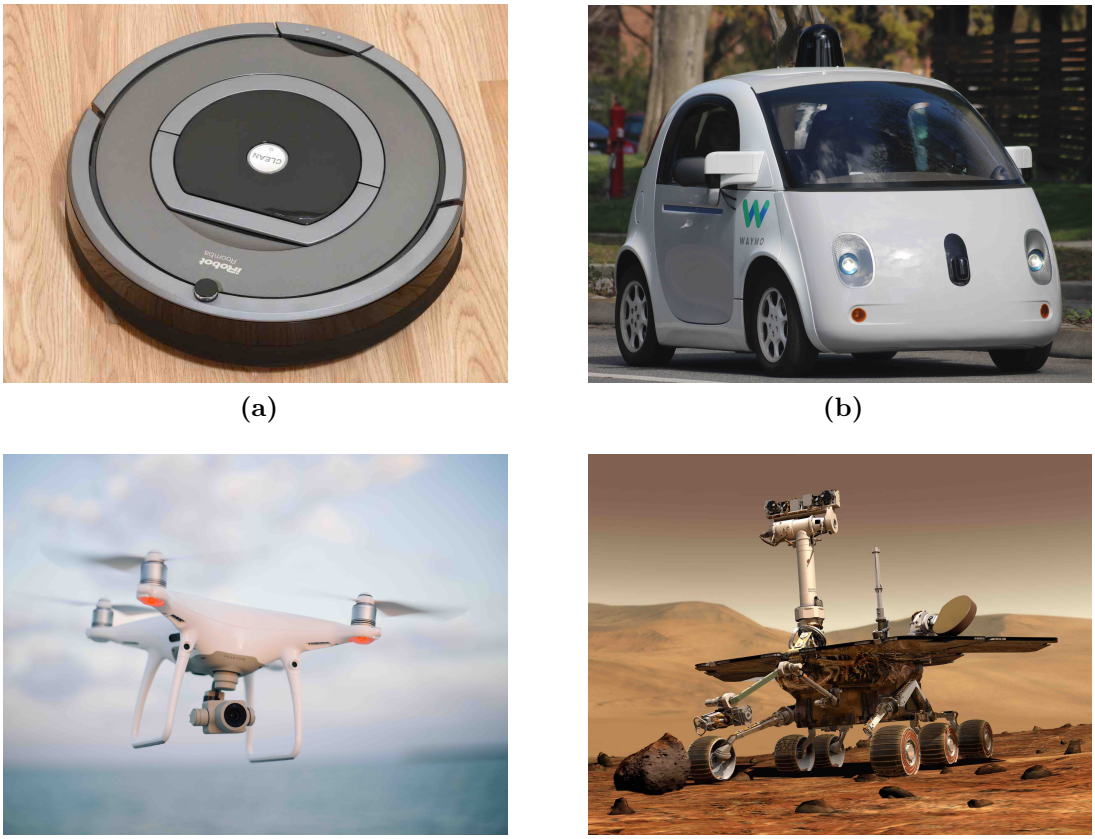


Figure 1.1: Examples of various autonomous systems. An iRobot autonomous vacuum (a, Wikimedia Commons 2011), a Waymo driverless car (b, Wikimedia Commons 2017), a DJI Phantom quadcopter unmanned aerial vehicle (c, Wikimedia Commons 2018), and the Mars Exploration Rover (d, Wikimedia Commons 2003).

In order to safely navigate through an environment and interact with other agents within it, an autonomous platform must be able to observe and understand its surroundings. Just as humans use a mixture of senses to interact with their world, autonomous agents employ a variety of sensor modalities to observe and understand their surroundings. Each of these modalities is unique in both the type of information available and the manner in which that information is observed. One of the most commonly employed modalities for robotic navigation is that of sight, and robotic vision constitutes a broad body of research and development.

Robotic visual navigation is traditionally concerned with answering the questions of *where am I?*, i.e., the current location of the robot, *what is around me?*. i.e., the map of the environment, and *how do I get where I want to go?*, i.e., the best set of

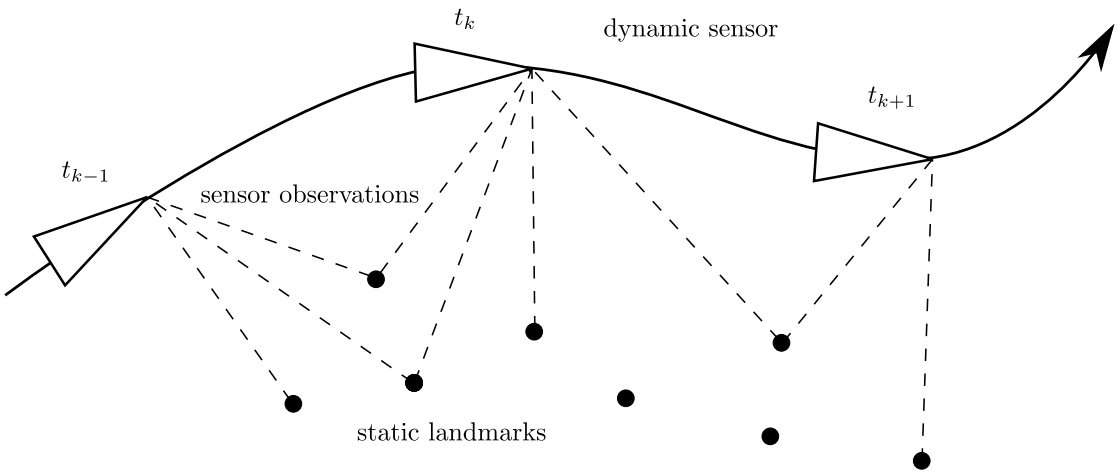


Figure 1.2: An illustration of the traditional motion estimation problem. A dynamic sensor platform (triangular body) travels through an environment containing several static landmark objects or points (dark circles) over a series of timesteps, $\ldots, t_{k-1}, t_k, t_{k+1}, \ldots$. The sensor takes measurements of the landmarks from its position at each timestep (dashed lines) and those measurements are used to determine the trajectory of the platform (solid line) relative to the static environment.

actions to take. Localization and mapping within a static environment have been the focus of work addressing the simultaneous localization and mapping (SLAM) problem. Only recently have similar questions about the *other* dynamic objects in the scene been addressed. Understanding the static structure of an environment is crucial to navigating through it, but understanding the motions of other dynamic objects in the environment is crucial to doing so *safely*.

Estimating the motion of a sensor relative to a static environment, i.e., its *egomotion*, is relatively straightforward (Fig. 1.2), and Moravec (1980) first introduced a pipeline for doing so with a camera in a process that is known as visual odometry (VO). VO is a crucial piece of almost all autonomous camera platforms, and it has analogues in other sensor modalities, such as radar and lidar. This estimation task is significantly more complicated when the scene contains other dynamic objects, and the static regions must be accurately isolated from any dynamic noise to preserve the consistency of the estimation. This segmentation is itself an important focus of visual navigation research (Nistér et al. 2004), but much less research has focused on also analyzing the dynamic regions of the scene that the segmentation rejects.

The estimation of these *third-party* dynamic motions is much more difficult than estimating the sensor egomotion. The static assumption that applies to the background of a scene has no analogue for these dynamic objects, whose observed motions comprise both their arbitrary real-world motions and the camera egomotion. This multimotion estimation problem (MEP) is often simplified by constraining motions according to kinematic assumptions, or by first isolating and estimating the sensor egomotion and then compensating for it while estimating the remaining third-party motions in the scene. These techniques can be successful in specific applications, but few generalized approaches have been proposed to address the full MEP.

This thesis explores the MEP by extending traditional egomotion-estimation techniques to simultaneously estimate the trajectories of all motions within a scene. It introduces Multimotion Visual Odometry (MVO), a novel multimotion estimation pipeline that incorporates multimotion segmentation and tracking techniques into the traditional VO pipeline in order to estimate the egomotion of the sensor, as well as the third-party motions in the scene. The MVO pipeline is a sparse, feature-based approach that casts the MEP as a multilabeling problem and employs multimodel-fitting techniques to estimate the full $SE(3)$ trajectory of every motion in the scene, including the egomotion. The pipeline is shown to be able to segment and estimate multiple motions in complex dynamic scenes, as well as real-world scenarios, even in the presence of temporary occlusions. The stereo pipeline represents several fundamental contributions:

- Simultaneous motion segmentation and estimation of *every* motion in the scene using low-level feature points;
- Full $SE(3)$ trajectory estimation of each motion in the scene using only a rigid-body assumption;
- Motion-based tracking, extrapolation, and interpolation of occluded motions;
- Quantitative evaluation of multimotion estimation techniques on complex multimotion scenes with ground truth, as well as qualitative evaluation on real-world multimotion scenarios.

Chapter 2 introduces several fundamental concepts and requisite notation that are integral to the work in this thesis. The chapter covers topics relating to projective geometry and camera models, multiview geometry and sparse feature processing, and $SE(3)$ notation and rigid-body kinematics. The following chapters build directly on these concepts and draw heavily from the core relations defined in it, but an informed reader can safely move directly to Chapter 3.

Chapter 3 introduces and defines the MEP. The general form of the problem is introduced first, along with the difficulties of estimating $SE(3)$ motions as compared to $SE(2)$ or $\mathbb{R}^n$ motions. The problem is further complicated by factors such as occlusions and collisions that are common in highly dynamic environments. The rest of the chapter is devoted to exploring current approaches to both single- and multimotion estimation and contextualizing this thesis within the broader field of research.

Chapter 4 elaborates on the MEP and its constituent challenges in the context of the Oxford Multimotion Dataset (OMD). The dataset was designed as a scaffold for developing and evaluating multimotion estimation techniques. Key aspects of the problem, such as estimating rotational motions and tracking through occlusion, are isolated and discussed. This work first appeared as Judd and Gammell (2019a) in IEEE Robotics and Automation Letters (RA-L) and was presented at the 2019 IEEE International Conference on Robotics and Automation (ICRA).

Chapter 5 addresses the MEP by introducing the MVO pipeline. MVO applies multimodel-fitting techniques to the traditional VO pipeline by casting the MEP as a multilabeling problem. Following the structure of the standard stereo VO pipeline, MVO replaces the egomotion estimator with a robust multimotion estimator. Sparse, 3D feature tracklets are decomposed into independent rigid motions and the $SE(3)$ trajectories of all of these motions, including the egomotion of the camera, are estimated simultaneously. The estimation accuracy of the MVO pipeline is quantitatively evaluated on highly dynamic scenes from the OMD. This work first appeared as Judd et al. (2018a) at the 2018 Joint Industry and Robotics

CDTs Symposium (JIRCS) and was expanded in Judd et al. (2018b) at the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Chapter 6 explores the challenges posed by occlusion in multimotion estimation. The limitations of the original MVO framework are addressed by exploiting a physically founded motion prior to estimate multiple motions through occlusion. This prior is used to extrapolate previously observed motion estimates until the object becomes visible again. Extrapolated estimates are used in *motion closure* to recover tracking when objects reappear in the predicted location. This occlusion-robust MVO pipeline estimates the full $SE(3)$ trajectory of every motion in the scene through both direct and indirect occlusions. The estimation accuracy of the occlusion-robust MVO pipeline is quantitatively evaluated on scenes from the OMD that feature significant occlusion. This work first appeared in Judd and Gammell (2019b) at the Long-term Human Motion Planning Workshop at ICRA 2019 and was expanded in Judd and Gammell (2020) and submitted to ICRA 2020.

Chapter 7 extends these multimotion estimation techniques to other sensor types, specifically event cameras. Event cameras are asynchronous sensors that measure pixel-wise changes in brightness, rather than the image frames of traditional cameras. These sensors present interesting new opportunities and challenges for multimotion estimation techniques, as they inherently measure dynamic changes within a scene but upend the traditional, synchronous data format most visual navigation pipelines are designed for. The estimation accuracy of the event-based MVO pipeline is qualitatively evaluated on real-world scenes with multiple motions. Though calibration and feature tracking pose unique challenges in event data, MVO is shown to be capable of addressing the MEP in real-world scenarios using this new sensor type.

2
Background

Contents

This chapter presents several fundamental topics that are integral to the work in this thesis while also introducing the requisite notation used throughout the rest of the thesis. Though these concepts are fundamental to the work presented in later chapters, an informed reader can safely move directly to Chapter 3. Section 2.1 introduces projective geometry and homogeneous coordinates. Sections 2.2 and 2.3 summarize the concepts surrounding rigid-body motion estimation. Section 2.4 describes camera models and how they project 3D world points onto 2D image planes, and Section 2.5 discusses how multiple images from different views can be related. Section 2.6 illustrates how the camera projection can be inverted to estimate 3D world points from multiple 2D image observations. Section 2.7 explains the various tools used to detect salient image points and to match them across multiple images.

2.1 Projective Geometry

A point in world space, $\underset{\rightarrow}{p}^{a}$, can be represented as a 3D Euclidean vector, $\mathbf{p}_C^{aC} = \begin{bmatrix} x^a & y^a & z^a \end{bmatrix}^T$, by defining it relative to some world origin frame, $\underset{\rightarrow}{\boldsymbol{\mathcal{F}}}_C$. This origin is arbitrary, and is usually defined in relation to the observer, e.g., the initial position of a sensor. In visual navigation, cameras are commonly used to observe these 3D points by projecting them onto a 2D image plane. Projective geometry is concerned with this type of projective transform. See Hartley and Zisserman (2003) for more detail.

Euclidean points can be given a *homogeneous* representation in projective space, which is an extension of Euclidean space that provides for a set of points that exist at infinity. The homogeneous representations of the Euclidean point, $\mathbf{p}_C^{aC}$, is defined as, $\boldsymbol{p}_C^{aC} = \begin{bmatrix} kx^a & ky^a & kz^a & k \end{bmatrix}^T$, where k represents a scaling factor. Homogeneous points are considered equivalent if they differ by a common multiple. Points at infinity are represented as $\boldsymbol{p}_C^{\infty C} = \begin{bmatrix} kx^\infty & ky^\infty & kz^\infty & 0 \end{bmatrix}^T$. A 2D image point, $\mathbf{u}^a = \begin{bmatrix} u^a & v^a \end{bmatrix}^T$, also has the homogeneous form, $\boldsymbol{u}^a = \begin{bmatrix} ku^a & kv^a & k \end{bmatrix}^T$. For simplicity, homogeneous points are usually normalized such that k is equal to one.

2.2 Rigid-Body Kinematics

Rigid-body kinematics, specifically in three dimensions, describe the motions of most solid objects throughout the world. The rigidity constraint requires that the distance between any pair of points on the body remains constant over time, i.e., it is not deformable. While the world contains many deformable objects, it is reasonable to model it as, for the most part, piecewise rigid.

These rigid-body kinematics can be defined as curves within the special Euclidean group, $SE\,(3)$, which forms a Lie group with the corresponding Lie algebra, $\mathfrak{se}\,(3)$. The members of $SE\,(3)$ can be represented as 4×4 transform matrices, $\mathbf{T}$, with six degrees of freedom, three translational and three rotational. For example, we can describe the motion of the sensor frame, $\underset{\rightarrow}{\boldsymbol{\mathcal{F}}}_C$, between two time points as the transform,

$$\mathbf{T}_{C_{k+1}C_k} \coloneqq \begin{bmatrix} \mathbf{C}_{C_{k+1}C_k} & \mathbf{p}_{C_{k+1}}^{C_kC_{k+1}} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{2.1}$$

which describes the motion between frames $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_k}$ and $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_{k+1}}$. The rotation between the frames, $\mathbf{C}_{C_{k+1}C_k}$, is a 3×3 orthogonal matrix and a member of the special orthogonal group, $SO\,(3)$, that defines 3D rotations. Being orthogonal, it has the useful property, $\mathbf{C}_{C_{k+1}C_k}^{-1} = \mathbf{C}_{C_{k+1}C_k}^{T} = \mathbf{C}_{C_kC_{k+1}}$. The vector $\mathbf{p}_{C_{k+1}}^{C_kC_{k+1}}$ defines the translation from $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_{k+1}}$ to $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_k}$ measured in $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_{k+1}}$ such that

$$\mathbf{p}_{C_{k+1}}^{C_kC_{k+1}} = -\mathbf{C}_{C_{k+1}C_k}\mathbf{p}_{C_k}^{C_{k+1}C_k},$$

and alternatively,

$$\mathbf{p}_{C_k}^{C_{k+1}C_k} = -\mathbf{C}_{C_{k+1}C_k}^{T}\mathbf{p}_{C_{k+1}}^{C_kC_{k+1}}. \tag{2.2}$$

A homogeneous point, $\boldsymbol{p}_C^{aC}$, expressed relative to the coordinate frame, $\underrightarrow{\boldsymbol{\mathcal{F}}}_C$, can be expressed in frame, $\underrightarrow{\boldsymbol{\mathcal{F}}}_A$, using the transform, $\mathbf{T}_{AC}$, such that

$$\boldsymbol{p}_A^{aA} = \mathbf{T}_{AC}\boldsymbol{p}_C^{aC}.$$

See Barfoot (2017) for more detail.

These 3D rigid kinematics have analogous forms in two dimensions forming the group $SE\,(2)$ and the algebra $\mathfrak{se}\,(2)$. The members of $SE\,(2)$ are 3×3 transform matrices with three degrees of freedom, two translational and one rotational. Many autonomous navigation applications can be reasonably constrained to $SE\,(2)$, which greatly simplifies the estimation challenge, but this thesis focuses on the more general $SE\,(3)$ motion space.

2.3 Lie Groups, Jacobians, and Adjoints

The $SO\,(3)$ and $SE\,(3)$ Lie groups and their associated algebras have several important properties that are relevant to motion estimation. Each algebra represents the tangent space of its corresponding Lie group, making them essential in linearizing $SE\,(3)$ and $SO\,(3)$ estimation functions.

The 3D rotation group, $SO\,(3)$, is related to its algebra, $\mathfrak{so}\,(3)$, through the exponential map,

$$\mathbf{C}_{BA} = \exp\left(\boldsymbol{\phi}_{BA}^{\wedge}\right),$$

where $(\cdot)^\wedge$ transforms $\boldsymbol{\phi}_{BA} \in \mathbb{R}^3$ into a member of $\mathfrak{so}(3)$, and is equivalent to the skew-symmetric operator, $(\cdot)^\times$,

$$\boldsymbol{\phi}_{BA}^\wedge = \begin{bmatrix} x \\ y \\ z \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \in \mathfrak{so}(3). \tag{2.3}$$

Likewise, the $SE(3)$ group of rigid-body transforms is related to the corresponding Lie algebra, $\mathfrak{se}(3)$, through a similar mapping,

$$\mathbf{T}_{BA} = \exp\left(\boldsymbol{\xi}_{BA}^\wedge\right) = \begin{bmatrix} \mathbf{C}_{BA} & \mathbf{p}_B^{AB} \\ \mathbf{0}^T & 1 \end{bmatrix},$$

where $(\cdot)^\wedge$ is overloaded to transform $\boldsymbol{\xi}_{BA} \in \mathbb{R}^6$ into a member of $\mathfrak{se}(3)$,

$$\boldsymbol{\xi}_{BA}^\wedge = \begin{bmatrix} \boldsymbol{\rho}_{BA} \\ \boldsymbol{\phi}_{BA} \end{bmatrix}^\wedge = \begin{bmatrix} \boldsymbol{\phi}_{BA}^\wedge & \boldsymbol{\rho}_{BA} \\ \mathbf{0}^T & 0 \end{bmatrix} \in \mathfrak{se}(3), \tag{2.4}$$

often referred to as a *twist*. The overloading of the $(\cdot)^\wedge$ operator reflects its functional consistency of mapping from column vector representations to Lie algebra members. This thesis maintains an intentional semantic distinction between $(\cdot)^\wedge$ for 3D vectors and $(\cdot)^\times$; this deliberately distinguishes between the skew-symmetric matrix corresponding to a member of $\mathfrak{so}(3)$ and the matrix corresponding to a vector cross product.

The inverse of the exponential map is the matrix logarithm, such that,

$$\ln\left(\mathbf{C}_{BA}\right)^\vee = \boldsymbol{\phi}_{BA}$$

and

$$\ln\left(\mathbf{T}_{BA}\right)^\vee = \boldsymbol{\xi}_{BA},$$

where $(\cdot)^\vee$ is the inverse transform to $(\cdot)^\wedge$. The sub- and superscript notation used to define the relevant frames for these transforms is explicit but cumbersome, so it is often dropped in situations where the notation is unambiguous.

The translation in the $\mathfrak{se}(3)$ algebra, $\boldsymbol{\rho}_{BA}$, is related to the $SE(3)$ translation, $\mathbf{p}_B^{AB}$, through the $SO(3)$ Jacobian,

$$\mathbf{p}_B^{AB} = \mathbf{J}\left(\boldsymbol{\phi}_{BA}\right)\boldsymbol{\rho}_{BA},$$

where

$$\begin{aligned}\mathbf{J}(\boldsymbol{\phi}) &\coloneqq \int_0^1 \mathbf{C}^\alpha d\alpha = \sum_{n=0}^{\infty} \frac{1}{(n+1)!} (\boldsymbol{\phi}^\wedge)^n \\ &= \frac{\sin\phi}{\phi}\mathbf{1} + \left(1 - \frac{\sin\phi}{\phi}\right)\mathbf{a}\mathbf{a}^T - \frac{1-\cos\phi}{\phi}\mathbf{a}^\wedge \\ &\approx \mathbf{1} + \frac{1}{2}\boldsymbol{\phi}^\wedge\end{aligned}$$

is defined using the axis-angle representation of the rotation vector, $\phi = \|\boldsymbol{\phi}\|$ and $\mathbf{a} = \boldsymbol{\phi}/\phi$. The inverse of this Jacobian is given by

$$\begin{aligned}\mathbf{J}(\boldsymbol{\phi})^{-1} &= \frac{\phi}{2}\cot\frac{\phi}{2}\mathbf{1} + \left(1 - \frac{\phi}{2}\cot\frac{\phi}{2}\right)\mathbf{a}\mathbf{a}^T - \frac{\phi}{2}\mathbf{a}^\wedge \\ &\approx \mathbf{1} - \frac{1}{2}\boldsymbol{\phi}^\wedge.\end{aligned}$$

The $SE(3)$ Jacobian is defined as

$$\begin{aligned}\boldsymbol{\mathcal{J}}(\boldsymbol{\xi}) &\coloneqq \int_0^1 \boldsymbol{\mathcal{T}}^\alpha d\alpha = \sum_{n=0}^{\infty} \frac{1}{(n+1)!} (\boldsymbol{\xi}^\curlywedge)^n \\ &\approx \mathbf{1} + \frac{1}{2}\boldsymbol{\xi}^\curlywedge,\end{aligned} \tag{2.5}$$

where the operator $(\cdot)^\curlywedge$ is analogous to $(\cdot)^\wedge$,

$$\boldsymbol{\xi}^\curlywedge = \begin{bmatrix}\boldsymbol{\rho} \\ \boldsymbol{\phi}\end{bmatrix}^\curlywedge = \begin{bmatrix}\boldsymbol{\phi}^\wedge & \boldsymbol{\rho}^\times \\ \mathbf{0} & \boldsymbol{\phi}^\wedge\end{bmatrix}, \tag{2.6}$$

and has the inverse operator, $(\cdot)^\curlyvee$. The adjoint transform, $\boldsymbol{\mathcal{T}}_{BA}$, relates two twists in different reference frames,

$$\boldsymbol{\mathcal{T}}_{BA} = \exp(\boldsymbol{\xi}_{BA}{}^\curlywedge) = \begin{bmatrix}\mathbf{C}_{BA} & (\mathbf{J}(\boldsymbol{\phi}_{BA})\boldsymbol{\rho}_{BA})^\wedge \mathbf{C}_{BA} \\ \mathbf{0} & \mathbf{C}_{BA}\end{bmatrix}, \tag{2.7}$$

and the inverse Jacobian is approximated as

$$\boldsymbol{\mathcal{J}}(\boldsymbol{\xi})^{-1} \approx \mathbf{1} - \frac{1}{2}\boldsymbol{\xi}^\curlywedge.$$

These first-order approximations are appropriate for small rotations and twists. See Barfoot (2017) for more detail.

2.4 Camera Models

A camera sensor maps 3D world points on a 2D image plane, such that

$$\mathbf{u}_C^a = \mathbf{P}_C \boldsymbol{p}_C^{aC} = \boldsymbol{\pi}\left(\boldsymbol{p}_C^{aC}\right), \tag{2.8}$$

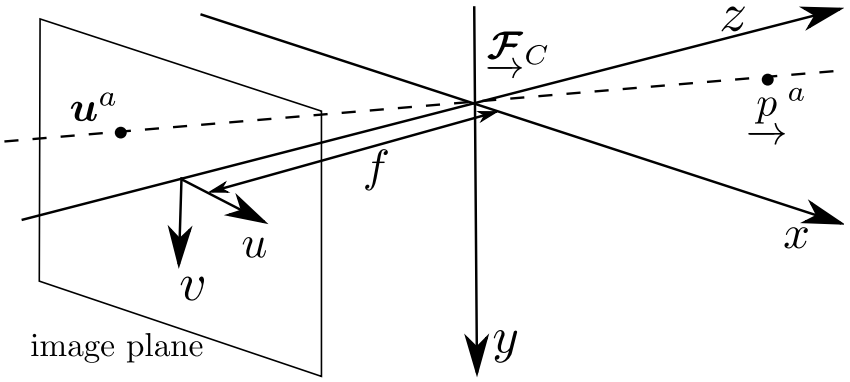


Figure 2.1: An illustration of the pinhole camera model projecting the world point, $\underrightarrow{p}^a$, onto the image point, $\boldsymbol{u}^a$. The image plane is placed at a distance, f, called the focal length, from the camera center, $\underrightarrow{\mathcal{F}}_C$. Physical cameras place the image plane behind the camera center, but the image plane can also be modeled in front of the camera center, which is geometrically equivalent. Camera frames are traditionally oriented with the x-axis to the right, the y-axis down, and the z-axis forward, corresponding to the optical axis. The depth ambiguity of a single camera view is illustrated by the fact that any world point on the dashed line is projected onto the same image point.

where $\mathbf{P}_C$ is the 3×4 projection matrix for camera C, and $\boldsymbol{\pi}(\cdot)$ is a function that applies this projection. The form of the projective transform matrix depends on the type of camera model used to approximate the projection.

The most common type of camera model is the *pinhole* model, which projects all visible points onto a single point known as the *camera center* (Fig. 2.1). In this model, the 3D homogeneous world point, $\boldsymbol{p}_C^{aC} = \begin{bmatrix} x^a & y^a & z^a & 1 \end{bmatrix}^T$, is mapped to the 2D image point, $\mathbf{u}_C^a$, determined by the intersection of the line joining both $\underrightarrow{p}^a$ and the camera center, C, with the image plane (Fig. 2.1). The line defined by $\mathbf{P}_C$ (Fig. 2.1, dashed) projects an infinite number of points at different depths in the 3D world space onto the same 2D image point. This ambiguity is a severe limitation of monocular camera systems.

The position of the camera center relative to the image plane forms the distinction between the two major classes of camera models: finite, which this thesis uses, and infinite. Finite models place the camera center at some finite distance from the image plane, the *focal length*, f; whereas in affine cameras, this length is infinite. The finite projection matrix is comprised of several parts,

$$\mathbf{P}_{finite.} = \mathbf{K}\mathbf{P}_{can}\mathbf{T}_{CW}. \tag{2.9}$$

Here, $\mathbf{K}$ is the intrinsic calibration matrix of the camera,

$$\mathbf{K} = \begin{bmatrix} m_u f & 0 & u_0 \\ 0 & m_v f & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

where f is the camera focal length, m_u and m_v are scale factors, and u_0 and v_0 are the principal point pixel coordinates. For square camera sensors, the two scale factors are equal to one, but sensors with rectangular pixels introduce unequal stretching, often represented as $f_u = m_u f$ and $f_v = m_v f$. The principle point, or image center, is the intersection of the optical axis of the camera and the image plane. The principal point coordinates define the translation between the camera and image coordinate frames, while the extrinsic transformation from the world frame to the camera frame is defined by $\mathbf{T}_{CW}$. The camera frame is often defined with the x-axis to the right, the y-axis down, and the z-axis forward, so $\mathbf{C}_{CW}$ often reflects this rotation. The canonical projection matrix,

$$\mathbf{P}_{can} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

captures the perspective effect, where distant objects appear smaller than nearby ones, and forms the core difference between finite and infinite camera models.

Infinite, or *affine*, camera models are motivated by the effect of moving the camera center backward to infinity, resulting in an infinite focal length. The result is that the canonical projection of the pinhole camera model is stretched into an orthogonal projection,

$$\mathbf{P}_{orth} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

This projection can be used in (2.9) in place of $\mathbf{P}_{can}$ to form the affine camera projection matrix $\mathbf{P}_{infinite}$. The main distinction is that the finite model involves a nonlinear transformation, whereas the affine camera transform is linear, simply dropping the z-coordinate of the 3D point in its projection onto the 2D image plane, i.e., the 3D world point, $\boldsymbol{p}_C^{aC} = \begin{bmatrix} x^a & y^a & z^a & 1 \end{bmatrix}^T$, is mapped to the 2D image point, $\mathbf{u}_C^a = \mathbf{K} \begin{bmatrix} x^a & y^a & 1 \end{bmatrix}^T$. This is distinct from the canonical projection, which

projects the point onto $\mathbf{u}_C^a = \mathbf{K}\begin{bmatrix} x^a & y^a & z^a \end{bmatrix}^T = \mathbf{K}\begin{bmatrix} \frac{x^a}{z^a} & \frac{y^a}{z^a} & 1 \end{bmatrix}^T$, clearly enforcing the perspective effects caused by variations in depth, z^a.

An interesting property of the orthogonal projection is that parallel world lines are preserved in the image plane. The affine camera also has no principal point, so the calibration matrix, $\mathbf{K}$, is simplified with both u_0 and v_0 being equal to zero. This all makes the affine camera model significantly simpler, but it introduces perspective errors that are especially severe in scenes with a wide field of view or with large depth of field. Some segmentation algorithms depend on this simpler projection, meaning their performance is usually limited to smaller scenes. See Hartley and Zisserman (2003) for more details.

These models assume that the camera projection is linear, i.e., that the world point, image point, and camera center are all colinear. In reality, a variety of factors, such as radial lens distortion, can corrupt this relation and introduce significant model error. It is therefore necessary to undistort an image before applying a linear camera model. The calibration and undistortion processes involved are beyond the scope of this thesis. See Zhang (2000) for more details.

2.5 Multiple Views

As shown in Section 2.4, a single camera observation is ambiguous in depth. This can be mitigated by using multiple views of the scene from different positions, either by employing multiple cameras or by moving a single camera and observing the scene from different positions. Two images of the same planar surface are related by a *homography*. The 3×3 homography matrix, $\mathbf{H}$, satisfies the constraint

$$\mathbf{u}^{a'} = \mathbf{H}\mathbf{u}^a, \tag{2.10}$$

where $\mathbf{u}^a$ and $\mathbf{u}^{a'}$ are corresponding image observations of the same point, $\underrightarrow{p}^a$, in two different camera views. A homography can be linearly calculated from four or more image point correspondences. A single homography is only valid for a given plane in the scene, and different parts of the scene will be related by different homographies between the same two image views.

The essential matrix, $\mathbf{E}$, is a generalization of this homography, relating points of arbitrary structure between two calibrated camera views (Longuet-Higgins 1981). The matrix embodies the *epipolar* geometry relating two views and is independent of scene structure, depending only on the camera's intrinsic parameters and relative pose. The two points in (2.10) and the essential matrix describing the two image views satisfy the constraint,

$$\left(\mathbf{u}^{a'}\right)^T \mathbf{E}\mathbf{u}^a = \mathbf{0}.$$

This relation is valid for all corresponding points in the two image views, not just planar ones. The essential matrix can be calculated linearly from eight image point correspondences or nonlinearly from five (Nistér 2004). The geometric interpretation of this constraint is that there is a plane defined by the two camera centers and the world point, $\underrightarrow{p}^a$. This plane intersects each image plane, and that intersection defines the *epipolar* lines on which $\mathbf{u}^a$ and $\mathbf{u}^{a'}$ must lie. Algebraically, this line is defined by $\mathbf{E}\mathbf{u}^a$.

The fundamental matrix, $\mathbf{F}$, further generalizes the essential matrix (Faugeras 1992; Hartley et al. 1992). It satisfies the same constraint,

$$\left(\mathbf{u}^{a'}\right)^T \mathbf{F}\mathbf{u}^a = \mathbf{0},$$

but does not require the views to be calibrated. The fundamental matrix can be linearly calculated from eight or more image point correspondences, or nonlinearly from seven. The fundamental and essential matrices are related through the camera intrinsics,

$$\mathbf{E} = \left(\mathbf{K}'\right)^T \mathbf{F}\mathbf{K},$$

where $\mathbf{K}$ and $\mathbf{K}'$ are the camera intrinsics for each view.

Determining the epipolar geometry relating two camera views from a set of point correspondences is a popular method for determining the motion of a camera between two frames, $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_k}$ and $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_{k+1}}$. From the essential matrix relating the two

camera frames, $\mathbf{E}_{C_{k+1}C_k}$, the rotation, $\mathbf{C}_{C_kC_{k+1}}$, and translation, $\mathbf{p}_{C_k}^{C_{k+1}C_k}$, can be calculated up to a scale factor, α, using singular value decomposition,

$$\begin{aligned}
\mathbf{E}_{C_{k+1}C_k} &= \alpha \left(\mathbf{p}_{C_k}^{C_{k+1}C_k}\right)^{\times} \mathbf{C}_{C_{k+1}C_k} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \\
\mathbf{C}_{C_{k+1}C_k} &= \mathbf{U}\left(\pm\mathbf{W}^T\right)\mathbf{V}^T, \\
\alpha\mathbf{p}_{C_k}^{C_{k+1}C_k} &= \mathbf{U}\left(\pm\mathbf{W}^T\right)\mathbf{U}^T, \\
\mathbf{W} &= \begin{bmatrix} 0 & \pm 1 & 0 \\ \mp 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix},
\end{aligned}$$

where $(\cdot)^{\times}$ is the skew-symmetric cross-product operator in (2.3). This decomposition is not unique, so the ambiguity is resolved with cheirality constraints where triangulated points must be in front of the camera in both views to be observed. See Hartley and Zisserman (2003) for more detail.

2.6 3D Triangulation

Triangulation is the process of determining the 3D position of a point in space via two or more images. As explained in Section 2.4, a single 2D image observation is ambiguous in depth; however, a second observation of the same point from a different position fully constrains the 3D world point. This triangulation can be performed using a stereo pair of cameras (Fig. 2.2), or with a single camera at consecutive positions.

Stereo camera setups are commonly employed in robotics because they introduce a static *baseline* between the two cameras that can be calculated offline and used to directly estimate the 3D structure of a scene. This baseline transform between the left and right camera in a stereo pair, $\mathbf{T}_{C_lC_r}$, has an analogous form for monocular cameras that represents the motion between two frames, but this motion can be arbitrary, making it difficult to directly estimate depth.

Under perfect conditions, with zero sensor noise, well-calibrated cameras, and known relative transforms, triangulation is trivial as it amounts to finding the intersection of two epipolar lines (Fig. 2.2, solid). Unfortunately, measurement, calibration, and data-association errors can corrupt the image measurements,

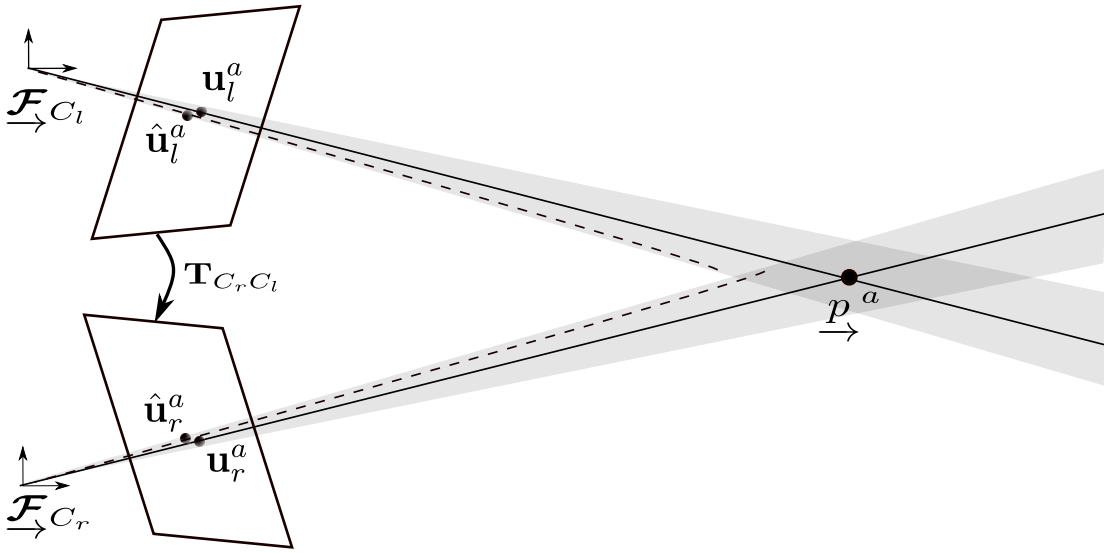


Figure 2.2: An illustration of the process of triangulating a 3D world point, $\underrightarrow{p}^a$, from two image correspondences. The true image points, $\mathbf{u}_l^a$ and $\mathbf{u}_r^a$, satisfy the epipolar constraints and are the intersection of each image plane with the solid lines connecting $\underrightarrow{p}^a$ and the camera centers, $\underrightarrow{\mathcal{F}}_{C_l}$ and $\underrightarrow{\mathcal{F}}_{C_r}$. The observed points, $\hat{\mathbf{u}}_l^a$ and $\hat{\mathbf{u}}_r^a$, are corrupted by noise and therefore do not satisfy the epipolar constraints, i.e., their epipolar lines (dashed) are skew and do not intersect. The shaded regions represent the uncertainty envelopes caused by this sensor noise, and the region where they overlap represents the uncertainty in the triangulated position of $\underrightarrow{p}^a$. The transform, $\mathbf{T}_{C_rC_l}$, represents the baseline transform in a stereo pair, or the motion of the camera in a monocular system.

resulting in non-epipolar (i.e., skew) lines that do not intersect in space (Fig. 2.2, dashed). In these situations the 3D position of p must be estimated through some approximate triangulation algorithm (Hartley and Zisserman 2003).

In a well-calibrated stereo setup, the baseline transform, $\mathbf{T}_{C_lC_r}$, is known and can be used to rectify the images such that their image planes are coplanar. Generally, the rectified image centers are also horizontally aligned, i.e., there is zero vertical disparity. This means that the horizontal epipolar lines of the two cameras are aligned, so points observed in the left image will have the same vertical position in the right image. A well-calibrated, horizontally aligned and symmetric stereo rig has camera intrinsics, $\mathbf{K}_{C_l} = \mathbf{K}_{C_r}$, and a baseline transform,

$$\mathbf{T}_{C_lC_r} = \begin{bmatrix} \mathbf{1} & \begin{bmatrix} b & 0 & 0 \end{bmatrix}^T \\ \mathbf{0}^T & 1 \end{bmatrix},$$

where b is the horizontal distance between the two cameras. Given a pair of associated image points in each view, $\mathbf{u}_l^a$ and $\mathbf{u}_r^a$, the 3D position of the point,

$\underrightarrow{p}^{a}$, relative to the left camera can be calculated as

$$\mathbf{p}_{C_l}^{aC_l} = \frac{1}{d^a} \begin{bmatrix} (u_l^a - u_0)\, b \\ (v_l^a - v_0)\, b \\ fb \\ 1 \end{bmatrix},$$

where $d^a = u_l^a - u_r^a$ is the *disparity* between the image points. This process of triangulating a 3D world point using image points is referred to as *back-projection*. The inverse process of *forward*-projecting a 3D world point onto its corresponding image coordinates often represents the two 2D image points as a single 3D image point using the disparity, such that

$$\boldsymbol{u}_C^a = \begin{bmatrix} \mathbf{u}_{C_l}^a \\ d^a \\ 1 \end{bmatrix} = \mathbf{s}\left(\mathbf{p}_{C_l}^{aC_l}\right) = \begin{bmatrix} \frac{fx^a}{z^a} - u_0 \\ \frac{fy^a}{z^a} - v_0 \\ \frac{fb}{z^a} \\ 1 \end{bmatrix}, \tag{2.11}$$

where $\mathbf{s}(\cdot)$ is the nonlinear perspective camera projection function. The corresponding image point in the right frame is given by

$$\mathbf{u}_{C_r}^a = \mathbf{u}_{C_l}^a - \begin{bmatrix} d^a \\ 0 \\ 0 \end{bmatrix}.$$

The uncertainty in the image point manifests in a conic uncertainty envelope over its back-projected ray (light shaded area, Fig. 2.2). The uncertainty from each camera is compounded, resulting in an asymmetric uncertainty envelope that worsens as the distance from the cameras increases (dark shaded area, Fig. 2.2). The growth of this uncertainty is quadratic with the depth of the point relative to the baseline distance, and it is a major limitation in the applicability of short-baseline stereo setups, especially to outdoor scenes. Longer baseline camera rigs can resolve depths more accurately, but it is often difficult to find and maintain a fixed calibration. In addition to this image noise, accurate triangulation requires the ability to correctly associate observations of the same real-world point between two different views, which is difficult in complex scenes.

2.7 Feature Detection and Matching

In order to avoid the computational load of processing every pixel in an image, it is common to focus on sparse image *features*. Feature detection is the process of identifying salient image points with the goal of selecting uniquely recognizable points within the observed scene. Similarly, feature matching involves associating these features across multiple image frames.

There are several ways to define an "interesting" image point, and most approaches do so in relation to the image gradient. Features can be edges, which have large gradients in a single direction; corners, which have large gradients in two or more directions; or blobs, which are local image extrema. A common way to calculate these gradients is by applying a filter of a certain size at every pixel position in the image and selecting the strongest responses. The size of this filter in relation to the resolution of the image is an important consideration, and it is common to also apply the filter to a downsampled image to detect a variety of image features. A diverse set of algorithms has been proposed to detect such features boasting various advantages or intended applications (e.g., HARRIS, Harris and Stephens 1988; SIFT, Lowe 1999; FAST, Rosten and Drummond 2006; SURF, Bay et al. 2006). This thesis uses the detection method of Geiger et al. (2011), which uses 5×5 blob and corner detection filters.

After the feature locations are found, discriminative feature *descriptors* are extracted from those locations. Descriptors are designed to encapsulate the local image texture at a point both uniquely and compactly, and many descriptors make use of the local gradient and polarity information found in the detection process. Some feature descriptors are scale or rotational invariant, making them robust to significant changes in the scene or camera position (e.g., SIFT, Lowe 1999; SURF, Bay et al. 2006); however, they can be computationally costly compared to binary descriptors, which sample local intensity values near the feature detection (e.g., BRIEF, Calonder et al. 2010; BRISK, Leutenegger et al. 2011; FREAK, Alahi et al. 2012). For this reason, binary features are often preferred in real-time motion estimation pipelines, where the camera and object motion is smooth. ORB

features build upon FAST corners and BRIEF features to introduce a level of scale and rotational invariance while maintaining the effeciency of binary descriptors (Rublee et al. 2011). More recently, learning-based detectors and descriptors have been introduced that outperform these traditional techniques, but tend to have slower run-time speeds compared to binary techniques (e.g., VGG, Simonyan et al. 2014; MatchNet, Han et al. 2015; LIFT, Yi et al. 2016). This thesis uses the descriptors described by Geiger et al. (2011), which creates a binary descriptor by sampling 16 horizontal and vertical edge-detection filter responses from designated neighboring pixel locations.

Once these features are detected in and extracted from multiple images, they can be compared to determine which features correspond to the same world point. Binary features can be compared using simple and efficient similarity metrics such as the sum of absolute/squared distances (SAD/SSD) or the Hamming distance; whereas, more complex features require more computation to compare. This thesis matches features using the SAD metric (Geiger et al. 2011).

Exhaustively comparing every pair of features in two images is often unnecessary if certain assumptions about the camera positions and the scene are valid. For example, in the case of small camera and scene motion between views, a window can be used to only compare features in one image that are within a given image distance from the feature in the other image. Likewise, a variety of factors can be used to determine if a given similarity measure between two features constitutes a match. Beyond simple thresholds on the similarity metric, matches can also be required to be unique, unambiguous, and symmetric. A unique match means a feature can only have one corresponding matching feature. Ambiguity measures require a feature match to be better than the second-best feature match by some threshold ratio. Symmetric matches require matched features to be each other's best match. The symmetric constraint can be extended to stereo setups where matches are required to close a "circle" of symmetric matches between the current left and right stereo images, the current right and previous right images, the previous right and previous left stereo images, and the previous left and the current left images.

In a well-calibrated stereo camera setup, the task of matching features between the two cameras is significantly simpler than the more general *temporal* matching between consecutive frames. As shown in Section 2.6, a well-calibrated stereo setup aligns the epipolar lines of the two cameras, so points observed in the left image will have the same vertical position in the right image. This greatly simplifies the matching process, as only points along these horizontal lines need to be compared. Furthermore, the cameras are separated by a static horizontal baseline, so a point in the right image must be observed to the left of where it is observed in the left image; therefore, it is only necessary to search in one direction on the epipolar line. These relations are similar for a vertically aligned stereo pipeline, but such a setup is much less common in practice. For these reasons, this thesis primarily focuses on well-calibrated horizontal stereo camera rigs with known intrinsic and extrinsic parameters. The process by which these calibration parameters are determined is beyond the scope of this thesis. See Zhang (2000) for more details.

3
Literature Review

Contents

This chapter introduces the MEP and illustrates its multifaceted nature, requiring accurate motion estimation, segmentation, and tracking. The MEP is then contextualized among the broader field of robotics research by exploring each of

these sub-problems. Each aspect is explored individually before exploring them in the context of the full MEP.

Section 3.1 illustrates the MEP and describes the various challenges involved in addressing it. Autonomous navigation in dynamic environments involves estimating the pose of a moving observer, i.e., the camera, relative to the static background in the presence of one or more independent, third-party motions. Each of these motions is generated by one or more objects in the scene, and the MEP is concerned with estimating the $SE(3)$ trajectories of those motions, as well as that of the camera. This involves both *estimation*, i.e., calculating the motion of a set of points, and *segmentation*, i.e., clustering points according to their movement between observations. This creates a *chicken-and-egg* problem, where segmenting a scene into independent motions requires knowledge of those motions, and estimating the constituent motions in a scene requires knowledge of its segmentation. When the scene is composed of many independent dynamic bodies, most of the simplifying assumptions used to initially decompose the scene are no longer valid, so the MEP presents a challenging, but integral, problem in autonomous navigation.

Section 3.2 introduces the various types of sensors used in visual navigation and estimation. Each of these sensors presents a unique set of advantages and challenges relevant to addressing the MEP. While active scanning sensors, such as lidar, can create accurate 3D point clouds illustrating the geometry of a scene, their cost and sparsity limit these benefits. In contrast, cameras are inexpensive sensors that provide dense observations of a portion of the scene, but a single camera view is ambiguous in depth. Addressing and overcoming these sensor limitations is the focus of several broad fields of robotics and computer vision research.

Section 3.3 discusses VO, a ubiquitous approach for single-motion segmentation and estimation. VO is commonly used in robotics to estimate the motion of an autonomous platform relative to its static environment through a sequence of images. From this single-motion perspective, the section explores motion estimation techniques, such as Kalman filters and bundle adjustment. The core assumption in most VO pipelines is that the dominant motion in a scene is that of the

static background. While this is often the case in many autonomous navigation applications, this assumption is not readily extensible to multimotion estimation.

Section 3.4 focuses on segmenting a dynamic scene into its constituent motions. Image segmentation is a rich field of research in computer vision, and many motion segmentation approaches extend these techniques with temporal information. Many of these techniques can be roughly, though not necessarily uniquely, categorized based on their focus, e.g., energy functionals, flow fields, statistical sampling , matrix factorization, or spectral clustering. Some approaches even combine techniques from several of these categories, but it is common to apply simplifying assumptions that limit the applicability of an approach to the general MEP.

Section 3.5 explores motion tracking, specifically in the context of tracking through occlusion. Tracking a single object involves detecting it in an image and estimating its position in subsequent observations. This task is made much more complicated when extended to multiple motions, as the challenge of associating current and past observations is difficult when multiple similar objects are interacting. These complex dynamic scenes also tend to include significant occlusion, so tracking multiple objects with incomplete observations is the focus of a significant body of research. Many of those techniques are also relevant to addressing the MEP.

Section 3.6 explores existing multimotion estimation approaches that attempt to address the full MEP. The majority of the techniques described in Sections 3.3 to 3.5 are focused on addressing a specific aspect of the MEP; however, several approaches are shown to be capable of addressing most or all of the MEP under certain conditions. This section describes those efforts that best address the full MEP, as well as the remaining areas of necessary work.

3.1 The Multimotion Estimation Problem

The fundamental challenge in the MEP is to determine the individual trajectory of each object in the scene, including the static objects. This requires simultaneously segmenting the observed points in a scene into independent rigid objects and estimating the $SE(3)$ trajectories along which those objects move.

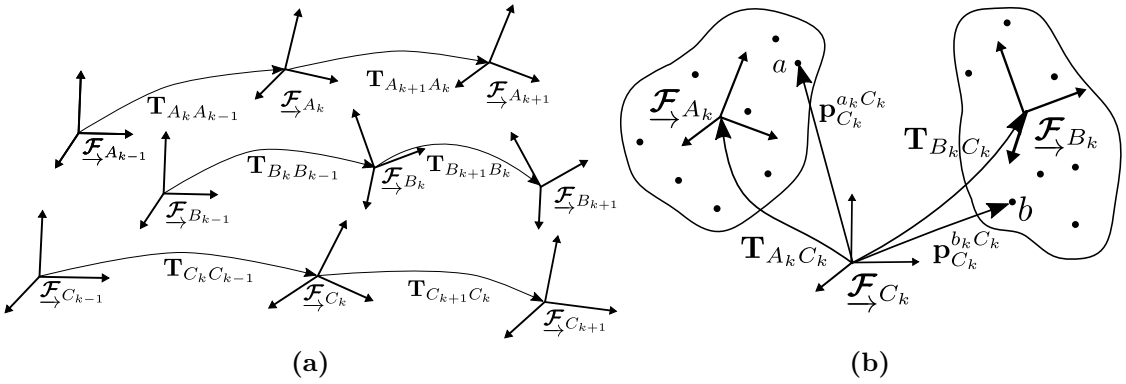


Figure 3.1: Illustrations of the MEP showing the motion of frames through time, (a), and the relative point observations, (b). A series of two independent third-party motions, $\underrightarrow{\boldsymbol{\mathcal{F}}}_A$ and $\underrightarrow{\boldsymbol{\mathcal{F}}}_B$, are observed by a moving camera, $\underrightarrow{\boldsymbol{\mathcal{F}}}_C$, through feature measurements on the objects, $\mathbf{p}_{C_k}^{a_k C_k}$ and $\mathbf{p}_{C_k}^{b_k C_k}$. Solving the problem requires simultaneously segmenting and estimating the set of measurements.

Dynamic environments consist of the static background, a moving observer, i.e., the camera, and one or more independent, third-party motions. Each motion, ℓ, is generated by one or more objects in the scene. This makes the MEP distinct from the multiobject tracking (MOT) problem, because the focus is on determining the *motions* present in the scene, rather than the *objects* that generate them.

The MOT problem is the subject of an extensive body of research in robotics, whereas the MEP has received much less attention. It is often worthwhile to favor tracking objects over motions, but this requires the generation of some form of models to represent the various objects in an environment. This thesis contends that for successful autonomous navigation, it is more important to understand *how* things are moving in the environment than to understand *what* they are.

At each discrete time, k, a motion, ℓ, is represented as a coordinate frame, $\underrightarrow{\boldsymbol{\mathcal{F}}}_{\ell_k}$, and related to a privileged initial pose through an $SE\,(3)$ transform, $\mathbf{T}_{\ell_k \ell_1}$ (Fig. 3.1a). A sequence of these transforms over a set of K frames constitutes the trajectory of the motion, $\mathscr{T}_\ell \coloneqq (\mathbf{T}_{\ell_k, \ell_1})_{k=1 \ldots K}$. These *global* transforms are defined relative to the privileged initial frame, $\underrightarrow{\boldsymbol{\mathcal{F}}}_{\ell_1}$, whereas *relative* transforms are usually defined relative to the previous frame, e.g., $\mathbf{T}_{\ell_k \ell_{k-1}}$. Global and relative

transforms are related such that

$$\mathbf{T}_{\ell_k \ell_1} = \mathbf{T}_{\ell_k \ell_{k-1}} \mathbf{T}_{\ell_{k-1} \ell_1}.$$

The set of these motion models describing a scene, $\mathscr{L}$, must adapt to the scene as those motions evolve over time.

Likewise, a sequence of observations of a point, j, by a moving camera, C, over multiple frames forms a *tracklet*, $\mathscr{p}_C^j \coloneqq \left(\boldsymbol{p}_{C_k}^{j_k C_k}\right)_{k=1\ldots K}$, where C_k refers to the observing camera frame at time k (Fig. 3.1b). The set of tracklets visible in a scene, $\mathscr{P} \coloneqq \left\{\mathscr{p}_C^j\right\}_{j=1\ldots P}$, can be segmented according to its constituent bulk motions.

This MEP is further complicated by full and partial occlusions, which are common in highly dynamic environments. Occlusions represent any lack of direct observations of parts of a scene. Direct occlusions are caused when an object partially or fully obscures another or leaves the field of view of the sensor. Likewise, self occlusions occur when an object's own motion partially occludes previously visible portions of itself. Occlusions can also be caused indirectly by sensor limitations or algorithmic failure, such as when motion blur or lighting changes corrupt feature matching or object detection. In these situations, the ability to indirectly estimate the motion of an object through extrapolation and interpolation is critical to addressing the MEP.

3.2 Sensor Archetypes

There is a wide variety of sensors available to autonomous agents for visually perceiving the world in which they act. Passive sensors, such as most cameras and microphones, record information emitted by external objects, e.g., the sun, or reflected by other surfaces in the environment. Conversely, active sensors, such as radar and lidar scanning systems, emit a signal and perceive the world around them through the reflections of that signal.

Each sensor archetype has a unique set of advantages and disadvantages relevant to the MEP, and some approaches fuse several different sensor modalities into a single technique (Vidal et al. 2018). Acoustic sensors provide omnidirectional

sensing, but it is still very difficult to accurately localize a sound source and acoustic localization is very rarely employed for autonomous navigation (Valin et al. 2003; Wang et al. 2004). On the other hand, cameras are relatively cheap sensors that provide dense 2D observations within a fixed field of view, and they have become near-ubiquitous in the field of autonomous navigation.

The fundamental function of a camera is to map observations of the 3D world onto a 2D image plane (Section 2.4), whereas ranging and depth sensors directly measure the structure of the 3D world. The observations from a single passive camera are ambiguous in depth, but multiple camera views can be used to estimate 3D world points (Section 2.6). A single camera can also be used in concert with an inertial measurement unit (IMU) in visual-inertial odometry (VIO) to resolve scale ambiguities (Mourikis and Roumeliotis 2007), but it is difficult to extend this scale-resolution to other motions in the scene.

Active scanning sensors, e.g., lidar and radar, emit a fixed-width signal at a particular bearing and wait for the reflected response, decoding the received signal to determine the depth of the reflecting surfaces. The target bearing is changed and this process is repeated to create a map of the surfaces in the environment. While these scanning sensors tend to rotate through different observation bearings very rapidly, the temporal offsets and rolling updates in the observations can make motion estimation much more complicated (Anderson and Barfoot 2015).

Many active cameras, e.g., RGB-D devices, use time-of-flight technology, which involves emitting a pulse much like lidar systems. Other active cameras use a fixed projection pattern and infer the scene geometry from the observed distortion in the pattern. These sensors do not include any mechanical scanning capabilities, meaning the entire depth image is collected at once, but within a limited field of view. There are also fixed lidar systems that measure depth in a fixed plane, and one or more of these sensors can be used on a mobile platform as a *pushbroom* system to create a depth map as the platform moves.

Active scanning sensors can provide very accurate depth information over a much greater distance and wider field of view than cameras, but they are expensive

in terms of both cost and power requirements. Additionally, most scanning sensors, do not provide the same density of information as cameras. For these reasons passive cameras remain commonplace in robotic vision systems.

Most passive cameras focus on observing the same spectrum of visible light that humans perceive. Specialized hardware, such as lens filters, can be used to observe different types of light, but these tend to be expensive and targeted toward very specific applications. Most cameras used for visual estimation have a global shutter, meaning the entire image is collected simultaneously, as compared to rolling shutter cameras, which record regions of the image in a scanning manner. Rolling shutter cameras introduce similar temporal complications to active scanning sensors, which makes estimation much more challenging (Hedborg et al. 2012; Oth et al. 2013). Improvements in camera and data transfer technology have caused camera resolutions and frame rates to steadily increase over time. This means that more information can be collected at a given time, and stronger assumptions can be made about the changes between frames, e.g., constant, small motion.

A relatively nascent type of camera technology is that of *event* cameras. Rather than simultaneously collecting the entire 2D image at a given frequency, an event camera records an asynchronous stream of brightness changes at each pixel. This type of camera directly measures the *dynamism* of the scene at very high frequencies and in a much sparser information stream than traditional cameras. The asynchronous event stream does not have the same structure as traditional image frames, and a new class of algorithms must be constructed for this type of sensor.

The multimotion approaches introduced in this thesis operate on sparse 3D tracklet points and are largely agnostic to the sensor that generates those tracklets. Chapters 5 and 6 focus on stereo RGB pipelines, and Chapter 7 introduces a stereo event camera pipeline. Each sensor archetype presents a unique set of challenges for addressing the MEP.

3.3 Motion Estimation

The most common approach for autonomously navigating through an environment based on visual observations is VO. VO is the process of estimating the motion of a camera relative to its static environment, i.e., its *egomotion*, and has a long history in robotics with its roots in the motion-estimation pipeline presented by Moravec (1980). This thesis introduced a mobile robot platform that could move through its environment, occasionally stopping and recording multiple images from different viewpoints at each location. These images are used as a form of multiview stereo camera, and feature points in the images are matched along epipolar lines and back-projected into the 3D scene. The 3D points are used to calculate the rigid body transformation between two observation points.

This motion-estimation pipeline represented the first stereo VO system, and subsequent work has focused on improving its robustness to noise and extending the approach to monocular cameras. All of these techniques focus on estimating the rigid-body transform describing the motion of the camera platform between two poses.

At a high level, a VO pipeline processes an incoming stream of images and outputs the egomotion trajectory describing the motion of the camera between those frames (Fig. 3.2). Input images are first undistorted according to the intrinsic camera parameters if they are known. In stereo VO pipelines, the images are also rectified using the extrinsic parameters (Section 2.4). Sparse VO techniques then detect salient feature points in the image and match them across images (Section 2.7). Monocular techniques only match features temporally, while stereo techniques also match features across stereo frames. Dense and direct VO techniques skip this step; dense techniques use pixels across the entire image and direct techniques use the pixel values themselves, rather than a back-projected representation. Finally, the egomotion is robustly estimated, usually by first rejecting any outliers from the set of observations.

The following sections explore the approaches to this estimation pipeline. Section 3.3.1 describes the various methods of estimating the motion between two frames, and Section 3.3.2 extends this estimation to multiple frames. Section 3.3.3

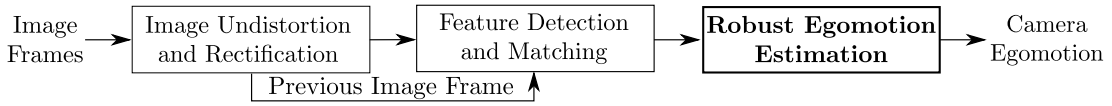


Figure 3.2: An illustration of the traditional stereo VO pipeline. The pipeline operates on RGB stereo image pairs, rectifies them using known camera intrinsic and extrinsic parameters, and matches salient image points across stereo pairs and temporally across consecutive stereo frames. It then isolates the static portions of the scene and estimates the motion of the camera from these static points.

explores different types of motion models and how they can be used to constrain the estimator to improve the accuracy of the estimation. Section 3.3.4 explains how outliers can be rejected to improve the robustness of the estimation.

3.3.1 Frame-to-Frame $SE(3)$ Estimation

Fundamentally, VO is concerned with estimating the $SE(3)$ transform describing the motion of a reference frame over consecutive time steps, k and $k+1$, relative to observations taken at those times. The reference frame usually relates to the sensor, $\underrightarrow{\mathcal{F}}_C$, and the observations are assumed to be measurements of static points in the environment, $\mathscr{P} = \left\{ \mathscr{p}_C^1, ..., \mathscr{p}_C^M \right\}$. Each transform can be estimated from these points using 3D world or 2D image coordinates and can be calculated from a sparse set of points or dense image patches.

3D-to-3D Estimation

A simple way to estimate this transform is using a set of 3 or more 3D points observed in each frame. The goal is to find the transform, $\hat{\mathbf{T}}_{C_{k+1}C_k}$, such that the geometric error between the points observed in the current frame, $\boldsymbol{p}_{C_{k+1}}^{jC_{k+1}}$, and the transformed points from the previous frame, $\boldsymbol{p}_{C_k}^{jC_k}$, is minimized, i.e.,

$$\underset{\hat{\mathbf{T}}_{C_{k+1}C_k}}{\arg\min} \sum_{j=1}^{m} \left\| \boldsymbol{p}_{C_{k+1}}^{jC_{k+1}} - \hat{\mathbf{T}}_{C_{k+1}C_k} \boldsymbol{p}_{C_k}^{jC_k} \right\|, \tag{3.1}$$

where $m \leq M$ is the number of observations used in the estimation. Horn (1987) shows that the rotation and translation that comprise this transform can be estimated independently such that

$$\underset{\left(\hat{\mathbf{C}}_{C_{k+1}C_k}, \hat{\mathbf{p}}_{C_k}^{C_{k+1}C_k}\right)}{\arg\min} \sum_{j=1}^{m} \left\| \mathbf{p}_{C_{k+1}}^{jC_{k+1}} - \hat{\mathbf{C}}_{C_{k+1}C_k} \mathbf{p}_{C_k}^{jC_k} - \hat{\mathbf{p}}_{C_{k+1}}^{C_{k+1}C_k} \right\|.$$

If the rotation is known, the translation can be calculated using the centroids in each frame,

$$\hat{\mathbf{p}}_{C_{k+1}}^{C_{k+1}C_k} = \bar{\mathbf{p}}_{C_{k+1}} - \mathbf{C}_{C_{k+1}C_k}\bar{\mathbf{p}}_{C_k},$$

where

$$\bar{\mathbf{p}}_{C_k} = \frac{1}{m}\sum_{j=1}^{m}\mathbf{p}_{C_k}^{jC_k}$$

Determining the rotation involves solving Wahba's problem (Wahba 1965),

$$\underset{\hat{\mathbf{C}}_{C_{k+1}C_k}}{\arg\min}\frac{1}{2}\sum_{j=1}^{m}w_j\left\|\left(\mathbf{p}_{C_{k+1}}^{jC_{k+1}} - \bar{\mathbf{p}}_{C_{k+1}}\right) - \hat{\mathbf{C}}_{C_{k+1}C_k}\left(\mathbf{p}_{C_k}^{jC_k} - \bar{\mathbf{p}}_{C_k}\right)\right\|,$$

where w_j is a weighting term for each pair of observations, which can be used to prioritize more certain observations. The solution can be found robustly using quaternions and eigendecomposition (Keat 1977) or singular value decomposition (Markley 1988). The singular value decomposition approach involves defining and decomposing the attitude profile matrix, $\mathbf{B}_{C_{k+1},C_k}$, such that

$$\begin{aligned}
\mathbf{B}_{C_{k+1},C_k} &= \sum_{j=1}^{m} w_j\left(\mathbf{p}_{C_k}^{jC_k} - \bar{\mathbf{p}}_{C_k}\right)\left(\mathbf{p}_{C_{k+1}}^{jC_{k+1}} - \bar{\mathbf{p}}_{C_{k+1}}\right)^T \\
\mathbf{B}_{C_{k+1},C_k} &= \mathbf{U}_{C_{k+1},C_k}\mathbf{\Sigma}_{C_{k+1},C_k}\mathbf{V}_{C_{k+1},C_k}^T \\
\hat{\mathbf{C}}_{C_{k+1},C_k} &= \mathbf{U}_{C_{k+1},C_k}\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & |\mathbf{U}_{C_{k+1},C_k})||\mathbf{V}_{C_{k+1},C_k}|\end{bmatrix}\mathbf{V}_{C_{k+1},C_k}^T.
\end{aligned}$$

This solution, combined with the centroid translation, comprises the least-squares estimate of the transform, $\hat{\mathbf{T}}_{C_{k+1}C_k}$.

This is a straightforward way to estimate the relative transforms between poses, but requires accurate data association of points between frames, as well as accurate 3D observations. Various forms of measurement noise can corrupt this estimation, so Milella and Siegwart (2006) further refine the least-squares solution by applying the iterative closest point (ICP) algorithm (Besl and McKay 1992). ICP is commonly used in registering lidar scans or depth images, and proceeds by iteratively assigning point associations based on a simple distance metric and minimizing (3.1) until the estimate converges. This approach is more robust to point mismatches, but it requires a reasonable initialization and is susceptible to strong outliers so a robust outlier rejection filter is also required (Section 3.3.4).

2D-to-2D Estimation

The approaches described above require 3D observations at both k and $k+1$. This is straightforward for stereo cameras and other 3D sensors, but the triangulation can be an additional source of error and is complicated for monocular cameras. Instead, the motion can be estimated through the essential matrix, $\mathbf{E}$, which relates 2D image points at k and $k+1$ up to a scale factor. As shown in Section 2.5, the translation and rotation between the two views can be found through singular value decomposition.

Longuet-Higgins (1981) presents an 8-point algorithm for estimating the essential matrix that involves solving a system of linear equations. Each pair of image points, $\boldsymbol{u}_k^j$ and $\boldsymbol{u}_{k+1}^j$, generates the set of constraints,

$$\mathbf{Y}^j \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{21} & E_{22} & E_{23} & E_{31} & E_{32} & E_{33} \end{bmatrix}^T = \mathbf{0},$$

where $\mathbf{Y}^j = \begin{bmatrix} u_k^j u_{k+1}^j & v_k^j u_{k+1}^j & u_{k+1}^j & u_k^j v_{k+1}^j & v_k^j v_{k+1}^j & v_{k+1}^j & u_k^j & v_k^j & 1 \end{bmatrix}$ and E_{ij} are the row-major elements of $\mathbf{E}$. The essential matrix can then be calculated by stacking the constraints from each pair of points as rows of $\mathbf{Y}$ and solving the system,

$$\mathbf{Y} \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{21} & E_{22} & E_{23} & E_{31} & E_{32} & E_{33} \end{bmatrix}^T = \mathbf{0}.$$

This algorithm requires knowledge of the camera calibration and that the points not be coplanar. Though the 3×3 essential matrix has nine elements, it only has five degrees of freedom, and Kruppa (1913) first showed that it could be estimated from just five points; though it wasn't until much later that a practical five-point solution was presented (Nistér 2004). This approach is valid for coplanar points, but still requires known camera calibrations.

Hartley (1995) shows that by normalizing the image points the 8-point algorithm can also be used to estimate the fundamental matrix, $\mathbf{F}$, for a pair of uncalibrated cameras; however, extracting the rotation and translation from the fundamental matrix still requires known calibration. Assuming the camera parameters do not change between views, this calibration can be found through self-calibration, which relies on successful image point tracking and a static scene (Faugeras 1992).

3D-to-2D Estimation

It is possible to bridge the gap between 3D-to-3D and 2D-to-2D estimation by projecting 3D points at k into the image at $k+1$. It has been shown that this approach achieves better results compared to 3D-to-3D estimation because it avoids the extra triangulation error in the second frame and measures estimation error directly via the observations.

The goal is to minimize the reprojection error in image space,

$$\underset{\mathbf{T}_{C_{k+1}C_k}}{\arg\min} \sum_{j=1}^{m} \left\| \mathbf{u}_{k+1}^{j} - \boldsymbol{\pi}\left(\mathbf{T}_{C_{k+1}C_k} \mathbf{p}_{C_k}^{jC_k}\right) \right\|,$$

where $\boldsymbol{\pi}(\cdot)$ applies the nonlinear perspective camera projection (Section 2.4). This problem is often referred to as the perspective-n-point (PnP) problem, where n refers to the number of required constraining points. Fischler and Bolles (1981) show that P3P is the minimal form of the problem, but many approaches use more points for robustness. The approach can also be used with monocular cameras but requires alternating frames to estimate depth and motion independently. This means points must be tracked for at least three frames.

Dense and Direct Estimation

The approaches above rely on an intermediate representation of the image data in the form of sparse features. This formulation is efficient and relatively invariant to measurement variations caused by lighting and exposure changes; however, it ignores a large amount of image data and can cause aliasing problems in regions with repetitive texture. As computer processing power has improved over time, image processing techniques have begun to consider the entire image at once. These *dense* approaches can enforce smoothness throughout the observed scene and infer structure in poorly textured regions, but they struggle with variations in lighting.

Dense approaches often employ an indirect representation in the form of geometric depth or optical flow. Optical flow is the pixel-wise 2D velocity of each pixel in the image, and can be calculated using variational methods that enforce brightness constancy over time and local smoothness in image space (Horn

and Schunck 1981). Valgaerts et al. (2012) explore how dense flow methods can be used to calculate the fundamental matrix. Stühmer et al. (2010) introduce a similar approach for estimating the geometric depth in a scene jointly with the camera motion. The indirect representations employed by these methods still introduce potential sources of error, but can overcome some of the texture limitations of sparse, indirect methods at the cost of extra computation.

Direct approaches forgo these intermediate representations and compare pixel observations in each image directly. They usually apply a smooth geometric prior directly on the image itself, exploiting smoothness in the image to enforce smoothness in the reconstructed scene. Newcombe et al. (2011) directly sample multiple different inverse depth values for each pixel and determine the photometric errors associated with that depth. The depth map and associated camera poses are estimated by minimizing this error along with a smoothing regularizer that embodies the geometric prior. This process is computationally expensive, so semi-dense methods only reconstruct informative portions of the scene, such as those with large image gradients (Engel et al. 2013). There are also sparse, direct VO formulations that use sparse image patches in the scene to estimate the camera trajectory (Engel et al. 2017). These direct techniques fundamentally rely on a geometric prior that assumes a static scene and small camera motions, so it is difficult to extend them to multimotion estimation.

Learning-based techniques have also been shown to be capable of regressing pose information from image sequences. For example, Kendall et al. (2015) use structure-from-motion estimates to train a network, PoseNet, that can accurately localize camera poses in difficult lighting conditions where traditional feature-based methods fail. Likewise, Zhou et al. (2017) demonstrate a network capable of generating both depth and pose estimates from monocular camera sequences, constraining the estimated depth with the motion estimates and vice versa. These techniques can be trained in an unupervised manner, leveraging existing localization and odometry techniques to generate training data; however, these end-to-end techniques are not readily applicable to addressing the MEP.

The approaches in this thesis use sparse, 3D-to-3D estimation techniques and measure residuals in 3D image space. This leverages the directness and simplicity of 3D estimation with the sensor fidelity of image-space residuals.

3.3.2 Batch $SE(3)$ Estimation

The approaches in Section 3.3.1 only leverage a single pair of frames in the estimation and have no method for recovering from errors and accumulated drift. Instead, batch estimation approaches use several, or all if possible, frames of measurements to estimate a chain of transforms between several or all of the poses. A common batch-estimation approach is bundle adjustment, which uses a window of K frames and minimizes the reprojection error associating the estimated transforms and the measured 3D point observations in each of those frames (Triggs et al. 1999). When K is less than the total number of frames, it is known as *sliding-window* estimation, which is common in online motion estimation pipelines. These windowed techniques form stronger constraints on the trajectory than in frame-to-frame estimation because points tend to be tracked over multiple frames.

Other multiframe estimation techniques such as *loop closure* rely on the ability to recognize when the camera has returned to a previously visited location (Newman and Ho 2005). Once this loop is closed, the estimated poses at both times the landmark was observed can be linked to form a looped constraint on the entire trajectory between those points. The drift can then be corrected and smoothed throughout the trajectory. This technique is predicated on the assumption that the landmark has not moved since the last time it was seen. Assuming the world is mostly static is useful for improving the accuracy of egomotion estimation, but it is not immediately applicable to the MEP for third-party motions. It is therefore very difficult to correct for accumulated drift in the estimation of other motions in the scene.

The static-background assumption is also integral to the SLAM problem. The SLAM and VO problems are both concerned with estimating the pose of a dynamic sensor, but SLAM additionally aims to generate a global map of the environment.

In highly dynamic environments, this map can quickly be corrupted when dynamic objects are added to the map, so most approaches simply focus on identifying and ignoring these dynamic objects (Hahnel et al. 2003). Other techniques focus on classifying objects as static or dynamic and independently maintaining a consistent static map and tracking those objects (Wang et al. 2007). Because SLAM approaches focus on generating a globally consistent map of the environment, they are not well-suited for addressing the MEP.

3.3.3 Motion Models

The rigid-motion assumption reduces the complex space of motion trajectories to $SE(3)$. While this assumption maintains fidelity with many dynamic motions in the world, many approaches use *a priori* information about an application domain to further constrain the estimated motions. For example, the motion of an aerial platform may exhibit a broad range of $SE(3)$ motions, but a warehouse robot may only move in $SE(2)$ or even $\mathbb{R}^2$. By constraining the motion trajectory according to the expected motions, the accuracy and efficiency of the estimator can be greatly improved.

Some approaches are only concerned with estimating the motion of an object in image space. This reduces the trajectory space to $SE(2)$ or $\mathbb{R}^2$. This type of model is commonly employed in motion segmentation (Tomasi and Kanade 1990) or tracking (Shi and Tomasi 1993; Yilmaz et al. 2006) algorithms, and tends to rely on the affine camera model (Section 2.4).

Real-world motions can also be constrained to $SE(2)$ (Ortin and Montiel 2001), and some approaches detect the ground plane and constrain dynamic objects to move upon it (Mitzel et al. 2010; Lenz et al. 2011; Byeon et al. 2018). Sabzevari and Scaramuzza (2016) constrain third-party motions to $SE(2)$, and use vehicular kinematics to further constrain the camera egomotion as locally circular. Some approaches define complex, high-dimensional models for applications such as human tracking (Wu and Nevatia 2007; Shu et al. 2012; Yang and Nevatia 2012). These

specialized motion constraints improve the accuracy of the estimation in particular applications, but cannot be applied to the general MEP.

In addition to the trajectory space constraints, models can be defined discretely or continuously. Discrete models represent a trajectory as a sparse set of states, which is well-suited for synchronized sensors such as globally shuttered cameras. Continuous models smoothly represent a trajectory at all times using an assumption, or *prior*, about the motion of objects. These models incorporate a smooth prior directly into the representation, which is preferable for scanning or rolling-shutter sensors. A discrete representation is still required to model the motion in accordance with discrete sensor measurements, such as a camera frame rate, but this prior can also be exploited to intelligently interpolate between those measurements or when an object is occluded.

An example of these formulations can be seen in the enforcement of a constant-velocity prior. In a discrete sense, this prior can be embodied in a cost that penalizes distance between consecutive velocity vectors (Breitenstein et al. 2009; Milan et al. 2013). In a continuous model, this prior is embodied by directly modeling the acceleration as zero-mean, white-noise Gaussian process (Anderson and Barfoot 2015). This type of prior can analogously enforce a constant-acceleration assumption both discretely (Kuo and Nevatia 2011) and continuously (Tang et al. 2019).

Motions can also be defined in different frames, and simple motions in one frame may become complex when expressed in another. Two bodies, each moving according to some known prior relative to some static reference frame, do not exhibit the same type of motion relative to *each other*. A model that is expressed egocentrically may be appropriate for estimating the egomotion of a camera relative to its static environment but not relative to other dynamic objects. It is therefore often necessary to express models in an inertial frame. For this purpose, and throughout this thesis, an Earth-attached, or *geocentric*, frame is approximated as an inertial frame.

3.3.4 Outlier Rejection

While the estimators above tend to be robust to small amounts of noise in the observation data, gross outliers can greatly affect performance. Least-squares estimators in particular are susceptible to these outliers, so an important stage in any estimation pipeline involves mitigating or altogether removing them. Simple consistency (Moravec 1980) and circular matching (Geiger et al. 2011) checks can be used to reject obvious matching outliers, but complex scene geometry, texture, and camera motion, as well as sensor noise and blur, can lead to significantly more outliers than these techniques can handle. Sim and Hartley (2006) explicitly fit outliers using the L_∞ norm to iteratively find the largest residual in the data and remove it, which has good results in cases with few, strong outliers. Various innovations in M-estimators introduce loss functions, such as the Huber loss (Huber 1964), that mitigate the effect of gross outliers, but they are still not robust to high percentages of them (Torr 1998).

Fischler and Bolles (1981) present an iterative algorithm for robustly estimating model parameters in the presence of noise. RANdom SAmple Consensus (RANSAC) repeatedly samples a minimal number of data points, m, to estimate a model hypothesis and evaluates that hypothesis based on how many data residuals lie within some threshold. The hypothesis with the greatest number of inliers after n iterations is reestimated using its entire inlier set and selected as the estimate.

RANSAC is a powerful algorithm due to its simplicity and efficiency that provides relatively strong statistical guarantees about the solution. The number of iterations required to produce a consistent solution, i.e., one sampled exclusively from inliers, with probability P is given by

$$n = \frac{log(1-P)}{log(1-r^m)},$$

where r is the inlier ratio of the model support to the rest of the data. Unfortunately, in cases where the signal to noise ratio is very low, the inlier ratio becomes small and the number of required iterations becomes impractical.

RANSAC has other shortcomings. Most notably, it is fundamentally greedy in the number of model inliers, meaning a model that weakly fits many points within the given threshold is more appealing than one that strongly fits slightly fewer points. Torr and Zisserman (2000) introduce Maximum Likelihood Estimation SAmple Consensus (MLESAC) to address this by maximizing the quality of the model support, not just its quantity.

RANSAC also has no concept of the quality or spatial distribution of points within the subspace. This means the sampling process can be made much more efficient by applying priors based on match quality or spatial distribution (Tordoff and Murray 2005). When estimating the egomotion, it is often useful to sample distant points from across the image, whereas the inverse is often beneficial for estimating multiple motions.

The performance of RANSAC can also be improved by reducing the size of the sample set, m. Algorithmic improvements, such as the five-point algorithm (Nistér 2004) compared to the eight-point algorithm (Longuet-Higgins 1981), can result in large reductions in the required number of iterations. Likewise, estimating a model from 3D correspondences only requires 3 points. Another way to reduce the model size involves applying model constraints (Section 3.3.3). Ortin and Montiel (2001) show that $SE\,(2)$ motion can be parametrized by two points, and Scaramuzza (2011) locally parametrizes the egomotion of a vehicle-mounted camera as circular, reducing the model size to 1. These constraints significantly improve the accuracy and efficiency of the estimation, but limit the applicability of the approaches.

RANSAC fundamentally finds the dominant model in a set of data, so RANSAC-based VO approaches tend to assume the majority of the observed scene is static. The egomotion can therefore be estimated by isolating the largest motion in a scene, with all other motions classified as outliers and "dynamic noise." While this assumption works well in applications where most of the scene truly is static, it fails in highly dynamic environments where every independent motion is considered noise, i.e., the "signal-to-other-signal" ratio is low. In these multimotion scenarios, more robust segmentation is required.

3.4 Motion Segmentation

Motion segmentation is the task of clustering, or labeling, points according to their motion between frames. These points can be pixels in an image or salient feature points tracked over time, and this distinction separates the field into dense and sparse approaches, respectively. Sparse approaches tend to be more efficient compared to dense techniques, but inherently discard some information collected by the sensor, so they can also suffer in terms of performance. Most motion segmentation approaches are based on energy functionals (Section 3.4.1), flow fields (Section 3.4.2), statistical sampling (Section 3.4.3) , matrix factorization (Section 3.4.4), or spectral clustering (Section 3.4.5).

3.4.1 Energy Functionals

A simple, yet effective, approach to motion segmentation is image subtraction, wherein the pixel-wise change in intensity between consecutive frames is calculated (Piccardi 2004). The resulting temporal mask can be used to segment dynamic objects from the static background, but this approach is very sensitive to image noise and lighting changes. Furthermore, while it can be effective for static cameras, it is difficult to extend to dynamic camera scenarios, as it requires accurate motion compensation.

More nuanced segmentation techniques define a more expressive cost functional that is minimized over the entire image. This functional is defined over a graph structure $\mathcal{N}$, where each vertex is a pixel or sparse feature point and the edges either define a pixelwise lattice over the image or a sparse neighbor graph. This energy generally takes the basic form,

$$E\left(\mathscr{L}\right) = \underbrace{\sum_{\boldsymbol{p}^i \epsilon \mathscr{P}} \rho\left(\boldsymbol{p}^i, \ell\left(\boldsymbol{p}^i\right)\right)}_{Residual} + \underbrace{\sum_{(\boldsymbol{p}^i, \boldsymbol{p}^j) \epsilon \mathcal{N}} V\left(\ell\left(\boldsymbol{p}^i\right), \ell\left(\boldsymbol{p}^j\right)\right)}_{Smoothness}, \tag{3.2}$$

where $\mathscr{L}$ is the current set of motion models, $\ell\left(\boldsymbol{p}^i\right)$ gives the model assigned to point $\boldsymbol{p}^i$, the unary residual function, ρ, evaluates how well that model applies to $\boldsymbol{p}^i$, and the binary smoothness function, V, penalizes neighbors in the graph that have

different assigned models. The data term is summed over the entire set of points, $\mathscr{P}$, and the smoothness term is summed over all edges in the graph, $\mathcal{N}$. This cost can be minimized using graph cuts (Greig et al. 1989) or approximate techniques, such as alpha-expansion (Nieuwenhuis et al. 2013), and has been shown to be very effective for binary foreground segmentation, even with a moving camera (Rother et al. 2004).

This formulation is commonly used in image segmentation (Boykov and Jolly 2001), but it can also be used for motion segmentation by extending the data and smoothness costs with temporal information. Isack and Boykov (2012) introduce Propose Expand and Re-estimate Labels (PEARL), a geometric model-fitting algorithm that uses α-expansion and model-refitting to iteratively estimate both model parameters and segmentations, addressing many of the issues found in most greedy subspace clustering techniques. PEARL extends the cost in (3.2) with a complexity term,

$$E\left(\mathscr{L}\right) = \underbrace{\sum_{p^i \epsilon \mathscr{P}} \rho\left(p^i, \ell\left(p^i\right)\right)}_{Residual} + \underbrace{\sum_{(p^i, p^j) \epsilon \mathcal{N}} V\left(\ell\left(p^i\right), \ell\left(p^j\right)\right)}_{Smoothness} + \underbrace{\sum_{\ell \epsilon \mathscr{L}} \gamma_\ell \psi_\ell}_{Complexity},$$

where γ_ℓ is the cost of using the model ℓ, and ψ_ℓ determines if the model was assigned to any points. This cost penalizes the use of each model, favoring compactness over complexity in a solution; i.e., a *minimum description length*.

The PEARL formulation is shown to be capable of frame-to-frame motion segmentation using fundamental matrices, as well as multiframe estimation using affine motion constraints (Isack and Boykov 2012). Roussos et al. (2012) use PEARL as part of an expectation maximization method that estimates both depth maps and motion segmentation. Amayo et al. (2018) extend the PEARL formulation using a convex relaxation algorithm (CORAL), which relaxes the discrete labeling of PEARL, allowing a point's label to be defined on the continuous range $[0, 1]$. This "soft" labeling leads to a primal-dual formulation of the minimization that involves point-wise calculations, meaning it is well suited for parallelization on a GPU.

Energy-based techniques are capable of segmenting complex scenes by employing expressive cost functions but tend to be computationally expensive. It is also possible

for these techniques to find incorrect segmentations due to local minima in the cost function caused by ambiguities in the observations. This thesis leverages the innovation of CORAL, which achieves good performance on general-purpose GPU devices.

3.4.2 Flow Fields

A large class of techniques based on similar energy formulations to those in Section 3.4.1 focuses on segmenting motions via flow fields. Optical flow refers to the pixel-wise velocities in an image arising from the relative motions between a camera and its environment (Horn and Schunck 1981). These velocities are the projection of 3D motions onto the 2D image plane and form a vector field. The flow field can be segmented along any discontinuities, which occur at the boundaries of each independent moving object, as well as depth discontinuities on a single rigid body. This flow is calculated by enforcing a brightness constancy constraint, where an image pixel, $\mathbf{u}^a$, at time t is displaced by some image distance, $\Delta\mathbf{u}^a$, at time Δt, such that

$$\mathbf{I}\left(\mathbf{u}^a + \Delta\mathbf{u}^a, t + \Delta t\right) = \mathbf{I}\left(\mathbf{u}^a, t\right),$$

where $\mathbf{I}\left(\mathbf{u}^a, t\right)$ is the image intensity at the pixel, $\mathbf{u}^a$, at time t. The calculation of this displacement is the concern of a large body of research (Fortun et al. 2015), including recent learning-based techniques (Ilg et al. 2017). The 3D extension of optical flow, scene flow, is defined similarly as a pixel-wise 3D translational vector for every point in the scene, and is usually estimated as the optical flow in multiple monocular views (Vedula et al. 1999) or via colored point clouds from RGB-D sensors (Menze and Geiger 2015).

These pixel-wise flow fields are purely translational, which can be locally accurate for small motions, but they do not appropriately model the true complexity of object motions, especially for rotations. Instead, most approaches employ a piece-wise affine motion model for the scene, which is algebraically simple and has many useful properties. Ochs et al. (2014) demonstrate the ability to segment long-term motions using optical flow by processing image sequences in batches. The frame-to-frame optical flow fields are concatenated to generate long pixel-wise flow trajectories

and occlusions are detected by comparing the flow backward in time, i.e., current image to previous image flow. These trajectories are segmented into objects based on their long-term motion. Rather than segmenting an externally estimated flow field, Memin and Perez (1998) couple the estimation of the piecewise-smooth flow field with the estimation of the affine object motions in the scene. These affine motion models can describe the image-space motion observed by the camera, but they do not accurately reflect the motion of objects in 3D space, especially if there are significant perspective effects or out-of-plane rotations.

Scene flow lends itself better to segmentation based on full $SE(3)$ motion models because it is naturally 3D. Wedel et al. (2009) calculate dense scene flow and segments dynamic objects from the static background using binary graph cuts, but this requires separately estimating their motion. Menze and Geiger (2015) model the world as being piecewise planar and simultaneously calculates both scene flow and the homographies defining the motion of those planes, but models those motions as full $SE(3)$ trajectories. Quiroga et al. (2014) define an even more over-parametrized framework in which the rigid body motions that induce the observed scene flow are calculated directly as $SE(3)$ transforms. Scene flow can also be segmented in a sparse framework by tracking sparse 3D points over time and calculating their 3D translational velocities (Lenz et al. 2011). While it can be useful for motion segmentation, the generation of the intermediate flow field representation of a scene is generally unnecessary for addressing the MEP.

Another approach is to use an iterative refinement technique such as k-means clustering to decompose a scene into its constituent motions. k-means iteratively assigns points to clusters based on their similarity with the cluster mean, then reestimating those means based on the points assigned to that cluster. Kottke and Sun (1994) use this approach to individually segment pixels in two consecutive images based on spatial locality and image intensity. The clusters are then compared between the two images based on their positions, intensities, and shape to determine the underlying object motions in the scene. Wang and Adelson (1994) use k-means clustering to segment optical flow in order to decompose a scene into motion

layers. This layered representation allows for reconstruction of occluded regions by accumulating data over many frames. In this situation, objects with the same motion at different depths, e.g., static parts of the scene, are geometrically segmented because they have significantly different image motions. These techniques require k to be known beforehand. Kim and Kim (2003) show how, for static camera applications, change detection techniques can be used to determine the number of motion clusters; however, determining this number is much more difficult in highly dynamic scenes with a moving camera.

3.4.3 Statistical Sampling

Sampling methods focus on statistical, rather than algebraic or geometric, properties of the data. As mentioned in Section 3.3.4, the quintessential statistical model-fitting algorithm is RANSAC (Fischler and Bolles 1981). Building upon the RANSAC model-fitting framework, Torr (1998) presents a two-view method for fitting multiple motion models using RANSAC by fitting a model to the largest motion, and then recursively applying RANSAC to the outliers to that model. Recursion continues until a predetermined number of models has been found, or the dominant model becomes improbably small and is likely due to noise. RANSAC variants (Torr and Zisserman 2000; Tordoff and Murray 2005) can be used to improve the efficiency and robustness of this *sequential* framework, and it is common to constrain the models according to the given application in order to improve performance at the expense of generality (Ortin and Montiel 2001; Scaramuzza 2011). This approach is simple and elegant, but RANSAC's greediness can still lead to poor segmentation, and its recursive nature makes it hard to recover from errors.

In contrast to these recursive techniques, Schindler et al. (2006) present a model clustering technique which initially generates a large number of candidate models by sampling various regions in the scene. Realizing that many of them would be redundant and many others would poorly fit the data, models are iteratively merged to improve the solution. Models are determined to be redundant if their inlier sets are largely overlapping, and the models with the largest nonoverlapping inlier sets are

taken as the constituent scene motions. Similarly, Sabzevari and Scaramuzza (2016) iteratively generate multiple motion hypothesis by sampling from 2D monocular correspondences and reassign points to the models that best describe their motion.

These sampling techniques are simple and efficient, but they can struggle in highly dynamic environments. This is because it can be hard to sample consistent motion hypotheses when each independent motion is considered noise to every other motion, i.e., the "signal-to-other-signal"' ratio is low. Likewise, partial and full occlusions can significantly impact the models estimated from the visible observations, and sampling from multiple frames can quickly become complicated.

3.4.4 Matrix Factorization

Another class of segmentation approaches focuses on algebraically decomposing feature tracks into the objects and motions within a scene. Tomasi and Kanade (1990) first introduced the technique for segmentation in 2D and it was extended to 3D by Tomasi and Kanade (1992). It proceeds by forming a matrix, $\mathbf{W} \in \mathbb{R}^{2K \times M}$, by stacking horizontal and vertical coordinates of each feature point observed in each frame. Here, K is the number of observation frames and M is the number of points observed. Under an affine camera model, this data matrix can be factorized using single value decomposition,

$$\mathbf{W} = \begin{bmatrix} u_1^1 & \cdots & u_1^M \\ \vdots & \ddots & \vdots \\ u_K^1 & \cdots & u_K^M \\ v_1^1 & \cdots & v_1^M \\ \vdots & \ddots & \vdots \\ v_K^1 & \cdots & v_K^M \end{bmatrix} = \underbrace{\mathbf{U}\mathbf{\Sigma}^{\frac{1}{2}}}_{\mathbf{M}} \underbrace{\mathbf{\Sigma}^{\frac{1}{2}}\mathbf{V}^T}_{\mathbf{S}},$$

where the motion matrix, $\mathbf{M} \in \mathbb{R}^{2K \times 4}$, describes the trajectories of the points, and the shape matrix, $\mathbf{S} \in \mathbb{R}^{4 \times M}$, describes the object itself. The decomposition is not unique, so accurately determining both $\mathbf{M}$ and $\mathbf{S}$, involves applying rotational (i.e. orthonormality) and translational constraints to $\mathbf{M}$.

This technique is elegant in its simplicity but limited in its application. It relies on the affine camera model and only estimates the motion and structure

of a single object. Additionally, it requires points to be tracked for the entire estimation window, which is uncommon in practice, and dealing with incomplete observations involves a fragile sequence of submatrix factorizations (Tomasi and Kanade 1992). Despite these limitations, it is the basis for a number of factorization methods that attempt to address these issues.

Extending this formulation to multiple motions is nontrivial, because it is difficult to know which features, i.e., rows in $\mathbf{W}$, belong to each object. Boult and Brown (1991) describe a clustering method that defines an initial motion using four features and recursively adds new features or spawns new motions depending on whether they are linearly dependent with the existing model(s). Gear (1994) presents a similar approach that uses matrix row reduction to achieve similar results. These approaches leverage the affine model to approximate motion rigidity with linear dependence, but they are very sensitive to noise and it is not guaranteed that any initial set of features forms a consistent motion.

Costeira and Kanade (1998) attempt to address some of the limitations of these approaches by introducing a shape interaction matrix,

$$\mathbf{Q} = \mathbf{V}\mathbf{V}^T, \tag{3.3}$$

whose elements indicate whether a pair of features belong to the same object. This matrix can be rearranged into block-diagonal form,

$$\mathbf{Q}^* = \underbrace{\begin{bmatrix} \mathbf{M}_1^1 & \cdots & \mathbf{M}_1^L \\ \vdots & \ddots & \vdots \\ \mathbf{M}_K^1 & \cdots & \mathbf{M}_K^L \end{bmatrix}}_{2K \times 4L} \underbrace{\begin{bmatrix} \mathbf{S}^1 & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{S}^L \end{bmatrix}}_{4L \times M},$$

where L is the number of motions in the scene. In $\mathbf{Q}^*$, point trajectories belonging to the same object also belong to the same block, so each object's submatrix can be factored independently. The off-diagonal elements of this permuted matrix are zero in the noise-free case, and a threshold can be used in the presence of noise. Unfortunately, it is difficult to determine the noise threshold and measurement noise propagates in complex ways through the factorization and affects the segmentation.

Several approaches have been presented to mitigate the affect of noise. Ichimura (1999) seeks to select the most informative features available and determine the permutation from that subset. Wu et al. (2001) and Kanatani (2001) detail bottom-up processes that initially over-segment the data and then iteratively merge subspaces if they differ by some metric or information theoretic threshold. These approaches still require that points be successfully tracked throughout the entire window, which is a significant limitation in real-world dynamic scenes. Gruber and Weiss (2004) overcome this limitation via an expectation-maximization framework that individually models the uncertainty of each feature observation, which also allows for the inclusion of priors concerning the uncertainty of the measurements.

These approaches all fundamentally depend on an orthogonal camera projection assumption, and therefore they fail under any significant perspective effects (Section 2.4). Some progress has been made in extending the approach to perspective views, such as Han and Kanade (1999), which iteratively factors the shape and motion of an object under a weak-perspective model that eventually converges to the full perspective model, but the approach is still very sensitive to noise. Despite these efforts, factorization approaches are difficult to extend to the full MEP.

3.4.5 Spectral Clustering

Affinity-based methods embed features in graph structures and employ the field of *spectral graph theory* to analyze the matrix representations of those graphs. These approaches first define an adjacency matrix, $\mathbf{A}$, such that each element, A_{ij}, expresses the connectivity or similarity of the i-th and j-th features. Next, a graph Laplacian, $\mathbf{L}$, whose definition varies throughout the literature, is calculated. The adjacency matrix itself is often analyzed directly, $\mathbf{L} = \mathbf{A}$, and other common graph Laplacians are the unnormalized, $\mathbf{L} = \mathbf{D} - \mathbf{A}$, symmetric normalized, $\mathbf{L} = \mathbf{1} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, and random walk normalized, $\mathbf{L} = \mathbf{1} - \mathbf{D}^{-1}\mathbf{A}$, versions where, $\mathbf{D}$ is a diagonal matrix with diagonal elements $D_{ii} = \sum_{j=1}^{m} A_{ij}$ (Chung and Graham 1997; Ng et al. 2002; Shi and Malik 1997).

The graph can be segmented into k clusters by calculating the first k eigenvectors of $\mathbf{L}$ and combining them as columns of the matrix $\mathbf{U}$. The rows of $\mathbf{U}$ can then be clustered using k-means clustering, correlating directly to the clusters in the original data (von Luxburg 2007).

While the choice of $\mathbf{L}$ separates certain methods, they primarily differ in the affinity measure they use to define $\mathbf{A}$. For example, a basic distance-based affinity measure is very popular (Ochs et al. 2014), but it is not uncommon for neighboring points to have different motions or distant points to have the same motion. Shi and Malik (1997) use spectral clustering for motion segmentation using optical flow and *motion profiles* as affinity measures. Recognizing that exact optical flow can be difficult to calculate accurately, motion profiles instead relate the probability distribution over potential flow values at each pixel. Using this as a similarity feature yields better performance for image regions where the optical flow is ambiguous. Inoue and Urahama (2001) use the absolute value of the shape interaction matrix defined in (3.3) as the affinity matrix, but still is not very robust to noise.

Yan and Pollefeys (2006) assume that nearby points are likely to have the same motion. They proceed by estimating an affine motion for each sparse feature point based on its nearest neighbors. An affinity matrix is then defined based on the similarity between these affine motions and segmented through the normalized Laplacian. These clustering techniques require *a priori* knowledge of the number of motion clusters to find, and it is difficult to define an affinity measure that properly encompasses the relations between points belonging to the same subspace.

Both Mateus and Horaud (2007) and Lenz et al. (2011) define affinity functions for segmenting sparse scene flow. Mateus and Horaud (2007) encode the rigid-motion constraint into the similarity measure through the variance in the Euclidean distance between two points over time. Points are then clustered using the process described above. Lenz et al. (2011) define a similarity based on the Mahalanobis distance between the flows of two neighboring points. Graph edges between points with low similarities are truncated, and the resulting subgraphs are refined using heuristics relating to object size and position relative to the ground-plane. Because they are

based on scene flow, which fundamentally calculates translational motion, these methods do not perform well for motions with large rotations. Spectral clustering methods are limited by the fidelity of their affinity function to the rigid-motion constraint. Likewise, their sensitivity to noise hinders their ability to address the MEP on their own; however, the exploration of these affinity functions is also relevant to other graph-based techniques.

3.5 Motion Tracking

At its core, motion tracking involves the estimation of the pose of one or more objects in a scene given a set of current and past observations. Usually stated in terms of Bayesian inference, the problem becomes one of estimating the posterior probability of a given state using the past states and the current observations. The objects to be tracked can either be low-level point features (Tomasi and Kanade 1991) or higher-level object representations such as image regions or object contours (Yilmaz et al. 2006).

3.5.1 Single Object Tracking

Tracking a single object can often be complicated by various factors that alter the observed appearance of the object, such as changes in lighting or scale. The observed appearance of an object can also vary significantly due to out-of-plane rotations, which cause previously observed portions of the object to be occluded by newly revealed portions. Successfully tracking an object requires accurately modeling both its appearance and its motion, which can each be highly volatile.

Appearance-based models attempt to describe an object based on its color, texture, and shape. The object itself can be represented by its centroid or a bounding region, or by more complex contour- or segment-based models. Centroid and bounding-region representations poorly approximate the full target pose, so objects are usually tracked in image or Cartesian space using simple motion models. Contour- and segment-based models can represent the complex pose of an object, but are prone to failure in environments with multiple objects and

partial occlusion. A more detailed review of object tracking representations can be found in Yilmaz et al. (2006).

Initializing an object model generally requires either user interaction or accurate motion detection and segmentation (Section 3.4). Many techniques predefine object models and limit the scope of their tracking to those object classes. The design considerations involved in choosing how to model a target object vary widely among different applications, and tracking techniques tend to be highly specialized. The limitations inherent to relying on basic object representations restrict the ability of general tracking techniques to estimate the full $SE(3)$ pose of each object.

Once an object has been detected in a given image and a representation has been defined, the tracking task involves locating that same object in subsequent images. This procedure of defining a model, detecting it in the next image, and associating multiple detections as a trajectory is often referred to as *tracking by detection*. A naïve tracking approach would be to exhaustively search each image for the region that best matches this model. By predicting the location of the object in the next image, this search can be limited to a particular section of the image. Doing so requires accurately modeling the motion of the object.

Motion models describe how an object's pose changes over time, and they are used to predict the next target state, given the current and past states. Most tracking techniques focus on tracking targets within the image plane, so simple affine transformation models are often used to describe the motion. A common motion constraint is constant linear velocity, which attempts to enforce smooth target trajectories. As discussed in Section 3.3.3, this is a reasonable approximation of real-world motion, but tracking techniques often enforce it in 2D image space, which does not accurately reflect real-world motion due to perspective effects.

A large body of motion detection and tracking research is focused on surveillance applications from a stationary camera platform (Kim et al. 2010), but tracking is made significantly more difficult when the camera itself is moving. Some of these static-camera techniques rely on the same background modeling and segmentation approaches detailed in Section 3.4.1. Simple, linear motion models also poorly

describe the observed motion of a dynamic object with constant velocity observed by a dynamic camera with constant velocity (Section 3.3.3). The static-camera limitations of these approaches means they are not extensible to the MEP for dynamic cameras.

Kalman (Kalman 1960) and particle (Del Moral 1997) filters use simple motion models to recursively predict the location of the target and update the current state based on current observations. The Kalman filter was commonly used in radar tracking (Reid 1979) before Broida and Chellappa (1986) used a Kalman filter to track the $SE\,(2)$ motion of an object in image space. Kalman filters assume that the object state has a Gaussian distribution, and does not perform well in situations where noise distorts this model. In these situations, particle filters can achieve better tracking results (Blake and Isard 1997). Tracking multiple objects with these filters requires accurate data association techniques, and their recursive nature means it is difficult to recover from association errors.

Another approach is to adapt the current state representation to match the object in the next image using an optimization procedure. This approach is useful for tracking complex, deformable objects using contour- or segment-based models, but often fails in environments with multiple objects and partial occlusion. Addressing the MEP requires the ability to track multiple independent objects from a moving camera in the presence of significant occlusion.

3.5.2 Multiobject Tracking (MOT)

The challenge of object tracking is made significantly more complex when expanded to MOT, and most techniques follow the tracking-by-detection paradigm. Given an appearance model of a target and a method for detecting potential target locations in a new image, the tracking problem reduces to the challenge of accurately associating present and past detections.

A major difficulty in extending filter-based single object tracking techniques to MOT is this data association because it is difficult for a recursive technique to recover from erroneous associations. A common approach is to use the Hungarian

algorithm (Kuhn 1955), a combinatorial assignment algorithm, to assign detections in the current frame to tracklets currently being tracked based on geometric and appearance-based costs (Stauffer 2003; Wu and Nevatia 2007; Geiger et al. 2014) Breitenstein et al. (2009) approximate this approach by greedily associating the best matches between detections. These techniques are dependent on defining accurate similarity metrics between detections, and object appearances can be highly volatile in dynamic scenes.

Reid (1979) introduces *multiple hypothesis tracking*, which uses Kalman filters to track multiple objects, but delays firmly associating observations by spawning multiple hypotheses that can reasonably explain a trajectory. This method quickly becomes computationally expensive, and successive works have focused on limiting the growth of the hypothesis tree (Cox and Hingorani 1996; Zhang et al. 2008). Khan et al. (2004) use a particle filter to track multiple targets and define a Markov random field structure that explicitly models the interactions between targets. This improves the tracking performance for complex scenes with significant interactions, but the recursive filter cannot recover from association failures.

Rather than assign a filter-based tracker to each target, energy-based techniques incorporate object appearance, motion, and interaction models in a cost functional describing the entire scene. The functional is defined over a graph where vertices represent detections and edges represent transitions between frames, and it is minimized using flow-based techniques (Zhang et al. 2008) or continuous energy minimization (Milan et al. 2013). This means all object tracks are estimated simultaneously and information from recent observations can be used to correct errors in past associations caused by detection failure or occlusion.

Interaction and exclusion models can be included in the cost to capture the complex behavioral relationships between nearby objects to predict and prevent collisions. Helbing and Molnar (1995) present a *social force* model for predicting the behavior of pedestrians, and Hu et al. (2008) model bulk crowd-based motion patterns, but defining accurate behavioral models remains a very difficult problem. Byeon et al. (2018) define a highly specialized model for tracking humans in $\mathbb{R}^3$

within a fixed volume that includes 3D reconstruction, object interactions, and volume entrance and exit costs. The approach uses multiple static cameras and processes detections offline in multiframe batches that means occlusions can be detected and explained after-the-fact; however, online tracking requires the ability to actively detect and recover from occlusions as they occur. These specialized approaches are dependent on defining representative cost functionals and do not generalize well to the full MEP.

3.5.3 Tracking Through Occlusion

Accurate data association is considerably more difficult in highly dynamic environments with significant occlusion. As explained in Section 3.1, both direct and indirect occlusions can greatly complicate the tracking problem. Direct occlusions can be predicted by modeling object overlaps (Mitzel et al. 2010) or using scene understanding (Kaucic et al. 2005), which can help to avoid misassociated detections. Yang et al. (2011) present a learning-based conditional random field model that directly considers the interdependence of observed motions, especially in the presence of occlusion. These prediction methods can be used for direct occlusions, but indirect occlusions are more difficult to predict.

Even partial occlusions are challenging for appearance-based techniques because they change the observed shape of the occluded object. Feature-based techniques track targets through partial occlusions when a sufficient number of feature points can be tracked (Sugimura et al. 2009), but grouping features into distinct objects is difficult if their bulk motion is similar. Other techniques define specific, part-based appearance models to infer the position of the entire object from the portions that are visible. Hu et al. (2012) define a block-based appearance model that subdivides an object into regions, and occlusions can be understood based on which blocks are unsuccessfully tracked. Likewise, humans can be tracked through significant occlusions by using models that individually track each body part (Wu and Nevatia 2007; Shu et al. 2012).

Full occlusions are often overcome by using motion priors to extrapolate trajectories in the absence of direct observations. Zhang et al. (2008) generate occlusion hypotheses that are explicitly incorporated into their flow minimization. This *hypothesize-and-test* paradigm works well in the presence of short or partial occlusions but fails under long occlusions as there is no information available to prune hypotheses. Ryoo and Aggarwal (2008) avoid this impractical growth with their *observe-and-explain* paradigm, which avoids hypothesizing occluded motions until an unoccluded detection is observed near the source of occlusion. Likewise, Mitzel et al. (2010) extrapolate unobserved target trajectories for a set number of frames to allow for reassociation when the target becomes unoccluded. The applicability of these occlusion models is limited by the effectiveness of the target detectors and the fidelity of their motion models to the object motions in a scene.

3.5.4 Learning-Based Tracking

Recently, advancements in deep learning have lead to several novel approaches to object tracking with neural networks. Convolutional neural networks (CNNs) are particularly useful networks for object detection and are currently considered the state of the art (Ren et al. 2015; Liu et al. 2016; Redmon and Farhadi 2017). Given accurate and repeatable object detections, single object tracking becomes trivial, but MOT remains difficult in highly dynamic environments due to occlusions and object interactions.

CNN-based MOT trackers generally rely on traditional affinity-based association techniques, such as the Hungarian algorithm, to compare some representation of the object detection generated by the CNN, but these representations do not include the temporal information required for continuous tracking over time in complex dynamic environments. In contrast, recurrent neural network (RNN) architectures have feedback mechanisms that perform well for time-series estimation, which is particularly important for motion analysis in MOT. For example, Milan et al. (2017) proposed a tracker that used previous object states to learn a motion model that predicted the future location of the object, which was then refined using the

object detections available at that time. Many other network architectures and variations have been applied to the MOT problem; see Ciaparrone et al. (2019) for a more comprehensive review of learning-based tracking techniques.

These learning-based techniques have consistently performed well on multiobject tracking benchmarks (e.g., MOT16, Milan et al. 2016; MOT19, Dendorfer et al. 2019; KITTI, Geiger et al. 2012), but because these networks are fundamentally tracking-by-detection systems or specially trained end-to-end tracking networks, they are not readily applicable to addressing the MEP. This means the classical frameworks and algorithms described in the previous sections are more relevant.

3.6 Multimotion Estimation Techniques

Sections 3.3 to 3.5 each give a foundational review of the various estimation, segmentation, and tracking techniques relevant to the MEP. While they are not meant to be comprehensive reviews of the entirety of each field, the techniques described in each section can be applied in concert to address the MEP. Some approaches come close to addressing the full MEP, but few address it in its entirety.

Qiu et al. (2019) and Eckenhoff et al. (2019) attempt to address the MEP using VIO techniques. These techniques use monocular cameras, meaning their measurements are underconstrained and a separate scale parameter must be estimated using the IMU. This scale parameter is valid for the egomotion of the camera, but estimating the scale for third-party motions is difficult and leads to several degenerate cases, such as when the object and camera motions are colinear. The performance of these techniques is impressive given the limitations of the sensors, but they are not broadly applicable to the general MEP.

Wang et al. (2007) extend the traditional SLAM formulation to include MOT in their Simultaneous Localization, Mapping, and Moving Object Tracking (SLAMMOT) framework. The 3D SLAM state, which includes the $SE\,(3)$ pose and velocity of the platform and the position of the static landmark points that constitute the map, is extended to include the range, bearing, and linear velocity of the tracked objects in the scene. The state is estimated and updated using an extended Kalman filter, and

dynamic objects are detected by recognizing inconsistencies in the static map caused by their motion. The original SLAMMOT framework was demonstrated using laser ranging sensors and was later extended to a vision-based framework for both monocular and stereo cameras (Lin and Wang 2010). The monocular framework suffers in certain underobserved cases, but the stereo formulation overcomes this. While SLAMMOT can simultaneously estimate the motion of the sensor and track multiple objects, it only estimates the linear velocity of the third-party objects. Likewise, SLAMMOT requires accurate data association to avoid corrupting the static map, which is difficult in highly dynamic environments.

Lenz et al. (2011) use sparse scene flow to detect and track multiple dynamic objects from a dynamic, vehicle-mounted camera. The approach operates on stereo image pairs and clusters sparse points based on the Mahalanobis distance between the flows of two neighboring points. Graph edges between points with low similarities are truncated, and the resulting subgraphs form clusters that are used to track 3D object volumes through $\mathbb{R}^3$ space. The approach does not limit itself to tracking specific classes of objects, but it does require objects to be of limited size and in contact with the ground plane. The technique is founded on clustering sparse feature points, so it is able to track objects through partial occlusions if enough feature points are available. Scene flow fundamentally calculates translational motion, rather than rotational, so the approach would not perform well for rotating bodies.

Quiroga et al. (2014) extend the translational formulation of scene flow by modeling the underlying motions as full $SE\,(3)$ motions. Similarly, Menze and Geiger (2015) model the world as piecewise planar *superpixels* and simultaneously calculates both scene flow and the homographies defining the motion of the planes, which are modeled as full $SE\,(3)$ motions. This formulation complicates the MEP by introducing unnecessary constraints to estimate the pixel-wise velocity of every point in the scene. Each of these techniques also relies on accurate RGB-D sensing and an independent egomotion estimation pipeline, which can lead to estimation errors in highly dynamic scenes.

Jaimez et al. (2017) use an $SE(3)$ formulation of scene flow to segment RGB-D images into their constituent $SE(3)$ motions. The scene is first clustered according to the spatial proximity of pixels in the depth image. This improves efficiency, but spatial locality alone is not a strong indicator of rigidity. The dominant scene flow of the entire scene is then calculated by minimizing the photometric and geometric residuals caused by transforming the cluster by the estimated transform and warping the 3D points into the image via the depth map and camera calibration. In order to identify the static regions of the scene, clusters with low residuals for this dominant motion are identified as the background. The segmentation is made more robust by posing it as an energy minimization that encodes the likelihood a cluster belongs to the background as a continuous variable. The energy incorporates a residual fidelity term, a spatial smoothness term, a temporal smoothness term that penalizes switching from background to foreground, and an additional term that biases distant points toward the background. This formulation is appropriate for indoor scenes, where the depth of field is limited, but in outdoor scenes, the static background and dynamic foreground objects can each span a variety of depths. The $SE(3)$ motion of the camera is reestimated from these background clusters, and the motions of all other nonbackground clusters are also estimated. This approach addresses the estimation and segmentation parts of the MEP and estimates full $SE(3)$ motions in the scene, but it prioritizes dense scene flow rather than motion estimation and relies on accurate RGB-D sensing.

Roussos et al. (2012) simultaneously and densely estimate depth maps and segment motions from a monocular camera while also performing dense object tracking. The technique defines a cost function involving photometric consistency, geometric smoothness in depth, spatial smoothness in image space, and a minimum description length term that promotes a compact solution. The three parameters to the cost, the depth map, object labeling, and rigid transformations, are alternately minimized while the other two are held fixed. The depth map is estimated with a variational approach similar to Newcombe et al. (2011) and the object labeling is found with PEARL (Isack and Boykov 2012). The $SE(3)$ transformations are

estimated for each labeled region using the depth map to minimize the dense photometric error. The segmentation and transformations must be initialized: the segmentation is first found by calculating and segmenting the optical flow of the scene; the trajectories are found by applying an off-the-shelf structure from motion algorithm to the segmented regions. The approach proceeds in an offline, batch manner and its additional focus on dense reconstruction, combined with its initialization requirements, make it ill-suited for MEP applications.

Other techniques use RGB-D sensors to segment and estimate motions while also performing suitable tracking for object reconstruction. Rünz and Agapito (2017) use an RGB-D camera to segment and track targets while simultaneously fusing 3D object models. The technique combines motion segmentation with object instance segmentation, which relies on predefined class-based object detectors. The $SE\,(3)$ pose of each object is tracked using photometric error and geometric ICP, and the segmentation is found using these active models in an energy-minimization framework based on the tracking residual and a smoothness term. The segmentation is refined by merging labels with similar transforms and suppressing small labels and disconnected regions within a label via a connected graph walk. If a connected region of outliers is sufficiently large, a new object model is spawned for future tracking. Labeled points are then used to update the current 3D reconstruction of each object model. Objects that are occluded are treated as inactive, but their geometric models are maintained in case they reenter the scene. Runz et al. (2018) extend this work to real-time processing by improving the efficiency of the semantic segmentation, and Xu et al. (2019) define a similar system using a volumetric representation, rather than surface normals. These techniques represent significant progress in addressing the MEP, with the added ability to fuse 3D models of the tracked objects; but they are reliant on high-quality, dense depth sensing and they are limited in the number of active models they can reasonably process.

3.7 Summary

The approaches described above represent significant progress toward addressing the MEP, but estimating multiple $SE(3)$ motions in complex dynamic scenes remains a difficult task. The majority of research has focused on a subset of the MEP, with few techniques demonstrating the ability to segment, estimate, and track the full $SE(3)$ motions of multiple objects in the scene. The few techniques that do address the full problem rely on dense RGB-D data or significant initialization procedures. Furthermore, only Qiu et al. (2019) quantitatively measured the $SE(3)$ error in the trajectory estimates for the third-party motions in the scene.

This thesis introduces MVO, a stereo-based approach to the full MEP. MVO draws on several of the concepts introduced in this chapter in order to estimate the full $SE(3)$ trajectory of every motion in the scene, as well as track those motions through potential occlusions. The $SE(3)$ trajectory for a group of tracklets is found by estimating the frame-to-frame transform via their back-projected positions in world space, as described in Section 3.3.1. These trajectories are assigned to points to segment the scene according to a cost functional similar to those described in Section 3.4.1, and the graph structure draws from the affinity-based segmentation techniques described in Section 3.4.5. The segmentation is then used to refine the trajectory estimates, treating each segmented region as a single motion segmentation problem and using the sampling methods described in Sections 3.3.4 and 3.4.3. Once the segmentation converges, the trajectories are reestimated using the full-batch bundle adjustment techniques described in Section 3.3.2. The motions are tracked over time using simple MOT data association techniques (Section 3.5.2) and they are extrapolated forward in the case of occlusions until they can be reobserved (Section 3.5.3). These stages combine to address each aspect of the MEP, and the following chapters explain them in more detail.

4

Decomposing the Multimotion Estimation Problem

Contents

The general MEP described in Chapter 3 comprises several important challenges. The ability to estimate the egomotion of a camera relative to its environment is paramount to autonomous navigation, but accurate motion estimation becomes more difficult as the camera motion becomes more complex. This task can be simplified in certain applications by constraining the egomotion to simpler motion spaces, such as $SE(2)$, but it is still challenging when significant portions of the scene are dynamic. Likewise, other objects in a scene often exhibit complex motions but many approaches constrain them to $SE(2)$ or $\mathbb{R}^3$. These simplifications are appropriate in limited applications, but multimotion estimation in complex, dynamic

scenes is often further complicated by significant occlusions. Addressing the full MEP requires the ability to estimate multiple $SE(3)$ motions, including the camera egomotion, in the presence of occlusion.

The OMD was designed to isolate each of these challenges and provide a scaffold for the development of multimotion estimation techniques. The dataset consists of real-world stereo and RGB-D camera images and IMU data from several different dynamic scenes. Each scene highlights a challenging aspect of the MEP and includes both static and dynamic sensor motions. Ground-truth pose information for every motion in the scene is provided, allowing for metric evaluation of multimotion estimation techniques. This work first appeared in Judd and Gammell (2019a) in RA-L and was presented at ICRA 2019, and it forms the basis for the majority of the quantitative analysis of how well the MVO pipeline addresses the MEP.

This chapter first contextualizes the OMD among other existing estimation, segmentation, and tracking datasets (Section 4.1). Section 4.2 describes the dataset at a high level, and the rest of the chapter illustrates the challenges of the MEP and shortcomings of the methods described in Chapter 3 using sections of the OMD (Sections 4.3 to 4.6).

4.1 Existing Datasets

Several datasets have been developed to address the individual challenges of motion segmentation, pose estimation, and object tracking. While each of these datasets is well-suited to foster the development and evaluation of specialized techniques that address these subproblems, none are designed to address the full MEP.

The TUM RGB-D SLAM dataset (Sturm et al. 2012) is a large collection of RGB-D image sequences and ground-truth egomotion information. The RGB-D images were collected using a Microsoft Kinect, and the motion of the sensor was obtained from a Vicon motion capture system. The dataset includes a variety of $SE(3)$ camera motions and even several dynamic scenes; however, the main third-party motions are non-rigid human motions, and there is no ground-truth information available for any motions other than the camera egomotion.

The KITTI autonomous driving dataset (Geiger et al. 2012) is a suite of different benchmarks addressing various computer vision challenges involved in autonomous driving, such as VO and pedestrian tracking. The data was collected using a car-mounted sensor platform that includes a stereo camera, a 360° lidar, and a GPS. Although the dataset includes several complex urban scenes with multiple dynamic motions, there is no ground-truth information available for those motions. Furthermore, the dataset is tailored toward autonomous driving scenarios and most motions are predominantly $SE(2)$.

The Hopkins 155 dataset (Tron and Vidal 2007) consists of 155 monocular image sequences with two or three independent motions in each. The majority of the sequences involve checkerboard-patterned objects in an indoor environment, but others involve traffic scenes or other piecewise-rigid and non-rigid motions. The key contribution is the inclusion of tracked feature points through each sequence as well as the ground-truth segmentation of those features. The dataset has been used extensively as a motion segmentation benchmark, but it is not appropriate as a multimotion estimation dataset due to the lack of ground-truth trajectory information. Additionally, full $SE(3)$ motions cannot be estimated from monocular images alone.

Part of the Hopkins 155 dataset was incorporated into the Freiburg-Berkeley Motion Segmentation Dataset (Ochs et al. 2014), which contributes dense, pixel-wise annotations of each moving object in 59 different image sequences. The dataset remains unsuitable for multimotion estimation, as it still does not include any ground-truth trajectory information.

Synthetic datasets, e.g., Gaidon et al. (2016) and Dosovitskiy et al. (2017), simulate autonomous driving environments with high levels of photorealism and customization. They can be used to create scenes containing arbitrary 3D objects with arbitrary $SE(3)$ motions. These frameworks are useful for developing data-driven computer-vision techniques, but do not replace the need for real-world data, especially for estimation approaches.

MOT16 (Milan et al. 2016) and MOT19 (Dendorfer et al. 2019) are MOT benchmarks consisting of several highly dynamic scenes with ground-truth annotations for dynamic objects. There are static and dynamic camera segments, and the camera position varies significantly between the segments. The annotations include bounding boxes and labels that describe the position and class of different types of objects in the scene but do not represent the full $SE(3)$ pose. These benchmarks are very useful for developing and evaluating target detection and tracking techniques, but they do not address the underlying motion of the target objects.

As an extension of egomotion estimation which also focuses on estimating the motion of a camera through its environment, the MEP requires both segmentation and estimation in order to determine the full $SE(3)$ trajectory for each motion in a scene. The datasets described above are designed for various other aspects of motion estimation, segmentation, or tracking, and are not well-suited for the full MEP; therefore, a new MEP dataset was required.

4.2 The Oxford Multimotion Dataset (OMD)

The OMD was introduced to lay the foundation for fully addressing the MEP and to support the development and testing of multimotion estimation algorithms by providing experiments with multiple rigid-body motions and ground-truth trajectory information for every moving object within a scene. The dataset consists of over 110 minutes of stereo and RGB-D camera images and IMU data and several different dynamic scenes. The sensors observe the scene with static, mostly translational (i.e., mostly $\mathbb{R}^3$), and unconstrained $SE(3)$ motions. Section 4.2 describes the dataset at a high level, and Sections 4.3 to 4.6 specifically explore challenging aspects of the MEP using segments of the OMD to illustrate the difficulty of the problem.

The dataset was collected from a single, statically calibrated sensor platform in an indoor experimental Vicon room equipped with professional flicker-free lighting. This lighting was chosen because traditional halogen bulbs, and even newer LED bulbs, tend to flicker due to various reasons, e.g., the fluctuations in an alternating-current power supply. This flicker is often imperceptible to or easily ignored by

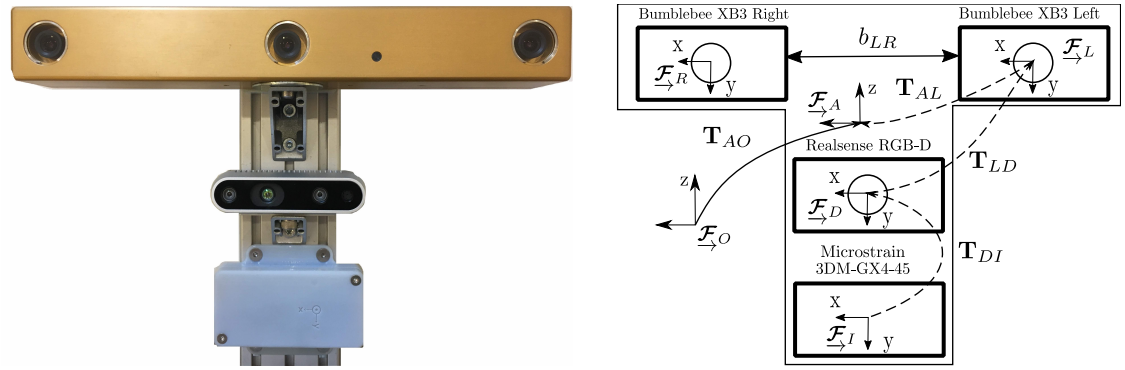


Figure 4.1: The sensor apparatus (a) and calibration diagram (b) used in collecting the OMD. A Bumblebee XB3 provided calibrated stereo images, an Intel Realsense D435 RGB-D camera provided both color and aligned depth images, and a Microstrain 3DM-GX4-45 IMU provided orientation and acceleration information. The baseline between the left and right Bumblebee XB3 camera centers, b_{RL}, is given by the manufacturer, and the transform between the Vicon origin frame and the Vicon frame for the apparatus, $\mathbf{T}_{AO}$, is provided by the Vicon motion capture system as ground truth (solid lines). The calibration between the depth camera and the left stereo camera, $\mathbf{T}_{LD}$, between the IMU and the depth camera, $\mathbf{T}_{DI}$, and between the left stereo camera and the Vicon apparatus frame, $\mathbf{T}_{AL}$, are calculated from the calibration sequences included in the dataset (dashed lines). All sensors are oriented with their z-axis forward and x-axis right, and the Vicon frames are aligned with the z-axis up and x- and y-axes arbitrary.

humans, but visual sensors, such as fixed frame-rate cameras, can be highly sensitive to these effects. While flickering lighting conditions may be representative of some indoor environments, it does not appear in outdoor scenes and represents a distinct, separable challenge from the MEP, so the dataset was designed to remove it.

The sensor apparatus consisted of a Point Grey Bumblebee XB3 stereo camera, an Intel Realsense D435 RGB-D camera, and a Microstrain 3DM-GX4-45 IMU (Fig. 4.1). All sensor data was logged and timestamped by a single computer using the Robotic Operating System (ROS) (Quigley et al. 2009). The Bumblebee XB3 records synchronized stereo image pairs using its wide 24 cm baseline at 16 Hz, and the Realsense records RGB images and 16-bit aligned depth images at 30 Hz. The Microstrain IMU records orientation and acceleration data at 500 Hz (Table 4.1). This sensor combination lends itself readily to a variety of visual motion estimation methods, including stereo and RGB-D VO and VIO.

Ground-truth trajectory information for every motion in each scene, including

Table 4.1: Sensors used in data collection for the OMD.

Sensor Type	Model	Rate	Notes
Stereo RGB	Bumblebee XB3	16 Hz	1280×960
RGB-D	Realsense D435	30 Hz	640×480
IMU	Microstrain 3DM-GX4-45	500 Hz	
$SE(3)$ Pose	Vicon Motion Capture	200 Hz	

the camera egomotion, was collected using a Vicon motion capture system. A Vicon coordinate frame is assigned to each object and its position and orientation relative to the world origin are recorded at 200 Hz. The extrinsic calibrations between each sensor, as well as the Vicon system, are given in the dataset (Section 4.2.1), along with the segments from which they are calculated.

The dataset contains a series of increasingly difficult experiments designed to foster development of multimotion estimation algorithms (Table 4.2). Each section of the dataset highlights a different aspect of the MEP and includes both static and dynamic sensor segments. The *Pinwheel* segments involve a checkerboard pattern rotating about the optical axis of the camera, which requires accurate rotational motion estimation for third-party objects. The *Swinging* segments consist of four independently moving blocks and act as an example of $SE(3)$ multimotion estimation, but do not include any occlusion. The *Occlusion* segments also feature moving blocks but involve significant occlusion and are designed to test the ability of an approach to handle missing observations. There are also *Fixed Occlusion* segments, which are similar but the source of occlusion remains fixed, simplifying the problem to only one third-party motion. The final *Toy Cars* segments include multiple radio-controlled cars that present many different motions and sources of occlusion. This section is intended to represent the type of multimotion estimation problem we believe is necessary to solve in robotics. Each section highlights a specific subset of challenges in the MEP and acts as a scaffold for improving performance on increasingly complex scenes.

Table 4.2: Data segments of the OMD and their characteristics. The type of camera motion, number of third-party object motions, duration, and presence of occlusion are detailed for each segment. More detail can be found in Judd and Gammell (2019a).

Section	Camera Motion	Number of Object Motions	Duration	Object Occlusion
Pinwheel	static	1	1m 05s	N
	unconstrained	1	1m 15s	N
Swinging	static	4	6m 00s	N
	translational	4	3m 45s	N
	unconstrained	4	6m 00s	N
Occlusion	static	2	8m 00s	Y
	translational	2	3m 00s	Y
	unconstrained	2	6m 00s	Y
Fixed Occlusion	static	1	3m 25s	Y
	translational	1	1m 50s	Y
	unconstrained	1	4m 50s	Y
Toy Cars	static	3	3m 00s	Y
	static	6	4m 45s	Y
	translational	3	2m 00s	Y
	translational	6	3m 00s	Y
	unconstrained	1	3m 00s	Y
	unconstrained	3	6m 30s	Y
	unconstrained	6	3m 00s	Y
	robot	6	2m 00s	Y

4.2.1 Calibration

The cameras and IMU were statically affixed to the collection platform, and their coordinate frames were manually aligned with the x-axis right, z-axis forward, and y-axis down, i.e., standard camera coordinate frames (Fig. 4.1b). The Vicon motion capture system maintains a coordinate frame for each tracked object, including the sensor apparatus, but this frame is arbitrarily aligned to the sensor frames. The calibration between individual sensors and between the Vicon system and the sensor apparatus were each calculated from separate calibration sequences in the dataset.

The calibration between the sensors on the apparatus is calculated using Kalibr (Furgale et al. 2013). The sensor apparatus is moved in front of a known, static calibration pattern, e.g., a checkerboard, such that all three cameras can observe the pattern and the IMU is sufficiently exercised in all directions. Because the size and

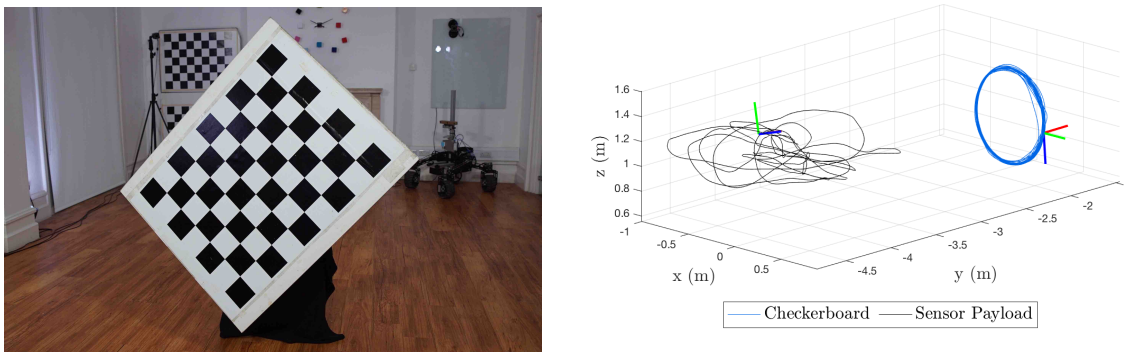
shape of the calibration pattern is known, the calibration problem becomes one of minimizing the geometric reprojection error between each camera. To determine the calibration of the IMU, similar visual geometry techniques are employed to estimate the pose of the camera relative to the calibration pattern, and the camera-IMU calibration is determined by minimizing the difference between the poses of the two sensors. Kalibr was used to determine the extrinsic calibration between the sensors, and the intrinsic calibrations it provides are also included in the dataset, though these are likely to be less accurate than those provided by the cameras themselves.

The calibration between the Vicon system and the sensor apparatus, $\mathbf{T}_{AL}$, was calculated by aligning the VO egomotion estimate of a test sequence with its Vicon ground truth. To do this, the apparatus is moved through a static scene and the trajectory in the camera frame is determined with VO. Assuming the distribution of features and the trajectory estimator are unbiased, the calibration between the camera and Vicon frames is given by the $SE\,(3)$ transform that minimizes the error between the VO trajectory and the Vicon trajectory,

$$\underset{\mathbf{T}_{AL}}{\arg\min} \sum_{k} \ln\left(\mathbf{T}_{L_k L_1}^{-1} \mathbf{T}_{AL}^{-1} \mathbf{T}_{A_k A_1} \mathbf{T}_{AL}\right)^{\vee}, \tag{4.1}$$

where $\mathbf{T}_{L_k L_1}$ is the VO trajectory and $\mathbf{T}_{A_k A_1}$ is the Vicon trajectory. This minimization finds the static transform required to align the two trajectories, though it is dependent on the accuracy of the VO trajectory. This process was used to estimate the extrinsic calibration between the left Bumblebee XB3 camera and the Vicon frame using a stereo VO pipeline, but it could instead be used to calibrate other sensors to the Vicon system using different egomotion estimation techniques.

Unfortunately, it is difficult to perform this kind of calibration for the third-party objects in each scene because the initial pose of each motion assigned by a multimotion estimation pipeline is arbitrary. Each estimated trajectory is therefore calibrated by aligning the first few pose estimates with the Vicon trajectory similar to (4.1). The calibration segments used are included int the dataset for completeness, but are uninteresting in the context of the MEP.



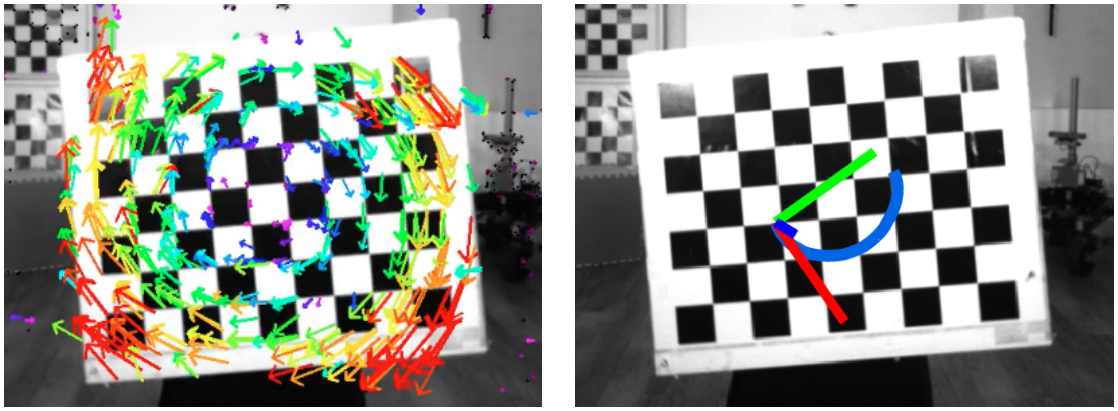
(a) **(b)**

Figure 4.2: The *Pinwheel* scene of the OMD is shown in (a) and the Vicon ground-truth motion trajectories from the unconstrained $SE(3)$ segment are shown in (b). These segments of the dataset consist of a rotating checkerboard pattern (b, blue), observed by a static or dynamic sensor (b, black). The board rotates primarily about the optical axis of the camera, changing rotation direction halfway through each segment. This scene tests the ability of a multimotion estimation technique to estimate rotations about the optical axis of the camera. Example frames from the `pinwheel_1_unconstrained` data segment are shown in Fig. B.1.

Each segment of the dataset consists of data from each of the sensors (Table 4.1). This includes 1280 × 960 RGB image pairs from the Bumblebee XB3; 640 × 480 RGB and 16-bit raw and color-aligned depth images from the Realsense D435; 6-DOF orientation and acceleration information from the Microstrain IMU; and 6-DOF orientation and position information for each object, including the sensor apparatus, from the Vicon motion capture system. Each segment of the dataset is referred to in the format, `<section>_<number of object motions>_<camera motion>`, and more details can be found in Judd and Gammell (2019a). Unless otherwise indicated, this thesis uses the stereo RGB image data for each segment.

4.3 Estimating Rotation

Estimating motion is significantly more complicated when rotations are involved. Largely rotational motion is particularly difficult for flow-based approaches which find the 2D or 3D velocity vector of each pixel or feature point in a scene. These individual velocities are inherently translational and motions involving 3D rotations can only be estimated from segmentations of three or more velocities. In the presence



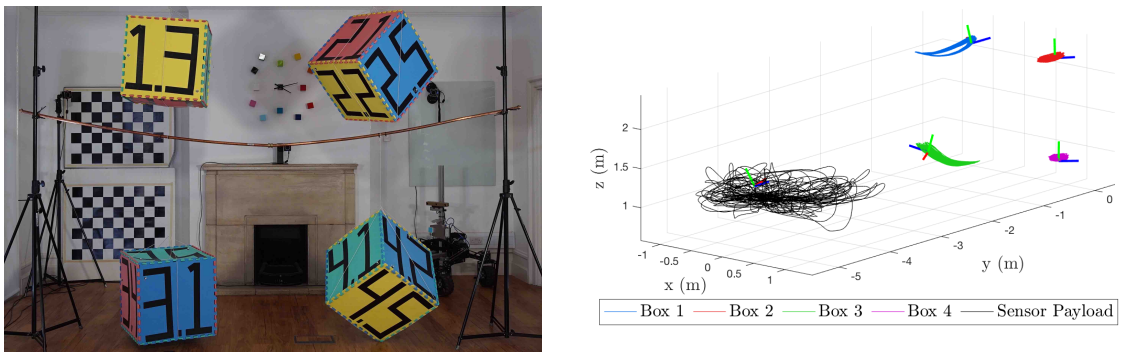
(a) (b)

Figure 4.3: Scene flow, (a), and MVO, (b), motion estimation in the `pinwheel_1_static` segment of the OMD, consisting of a checkerboard rotating about the optical axis of the camera. The magnitude of the flow vector is illustrated by its length and color. Flow techniques inherently calculate individual translational velocities. Since each point on a rotating body has a different tangential velocity, flow techniques fail to segment the velocities into bulk motions. MVO estimates motions as $SE(3)$ transforms even in the presence of rotation. To estimate these motions, flow techniques must solve an equivalent segmentation and estimation problem in the space of velocities.

of small rotations, these segmentations can be achieved using flow discontinuities or the vector distance between velocities (Section 3.4.2).

This limitation is demonstrated clearly with a large rotation around the optical axis of the camera, as in the *Pinwheel* segments of the OMD (Fig. 4.2). The segments consist of a standard calibration checkerboard pattern rotating about the optical axis of the camera. Because traditional flow techniques fundamentally estimate pixel-wise translational velocities, these rotations result in smoothly varying velocities fields that provide no clear segmentations (Fig. 4.3a). Without making specific assumptions to constrain the rotation, an $SO(3)$ rotation requires a minimum of three tracked points to estimate. Because these rotations cannot be estimated in a pixel-wise manner, they require solving a segmentation and estimation problem posed in the intermediate space of pixel-wise velocities.

In contrast to these flow techniques, MVO simultaneously segments and estimates full $SE(3)$ transforms of motions in the scene (Fig. 4.3b). This ability to fully estimate rotational motions is critical to addressing the MEP.



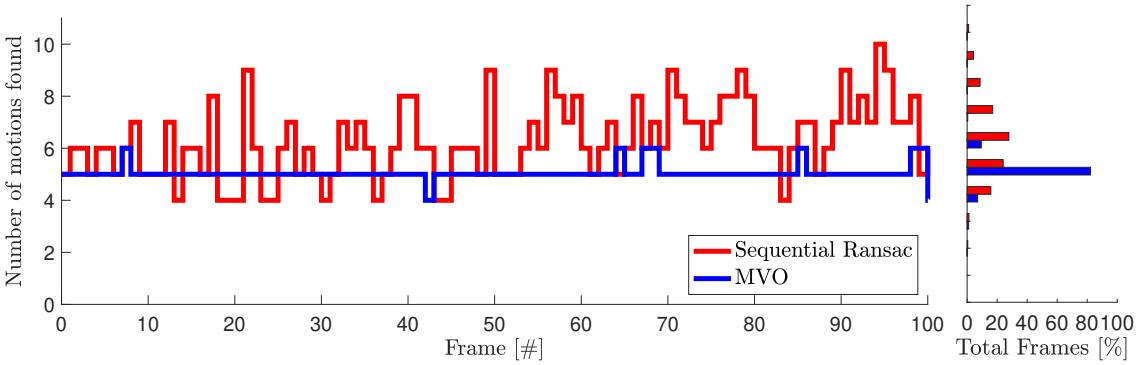
(a) (b)

Figure 4.4: The *Swinging* scene of the OMD is shown in (a) and the Vicon ground-truth motion trajectories from the unconstrained $SE(3)$ segment are shown in (b). These segments of the dataset consist of a static or dynamic sensor (black) observing four independently swinging blocks (other colors). The blocks swing and rotate as pendulums to generate four distinct $SE(3)$ motions. It is a basic scene requiring full $SE(3)$ estimation of multiple motions without complications such as occlusions or collisions. Example frames from the `swinging_4_unconstrained` data segment are shown in Fig. B.2.

4.4 Multimotion Estimation

Central to the MEP is the ability to simultaneously estimate multiple dynamic motions. The *Swinging* data segments of the OMD focus on the challenge of simultaneously estimating multiple $SE(3)$ motions. The scene consists of four blocks that both translate and rotate in pendular motions. The number of objects in the scene reduces the "signal-to-other-signal" ratio described in Section 3.4.3, meaning many sampling-based approaches may fail to correctly segment and estimate the correct trajectories.

Recursive sampling-based methods, such as sequential RANSAC, are efficient at finding the dominant models in a scene, but the ability to sample consistent models decreases as models are removed and the signal-to-noise ratio of the remaining points decreases (Section 3.4.3). Without prior knowledge of the number of motions, sequential RANSAC greedily overfits to noise and finds erroneous or incomplete models (Fig. 4.5). Even when sequential RANSAC finds the correct number of motions, they are unlikely to correspond to the true segmentation of the scene due to greedily over- or undersegmented motions (Fig. 4.6). In contrast, MVO robustly estimates all models simultaneously and requires no *a priori* knowledge of the number of models.



(a) **(b)**

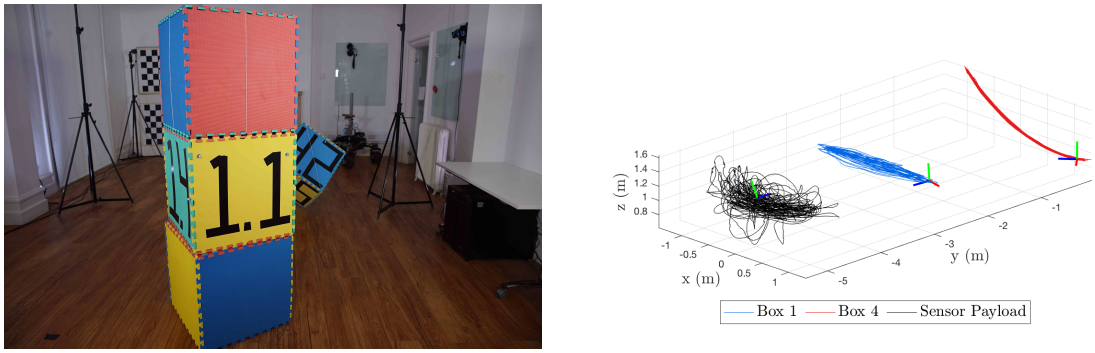
Figure 4.5: A comparison of the number of models found by a sequential RANSAC (a and b, red) and MVO (a and b, blue) over 500 frames of the `swinging_4_unconstrained` segment of the OMD. The first 100-frame section of the sequence is shown in (a) with the cumulative totals for the full sequence in (b). Without prior knowledge of the number of motions, sequential RANSAC greedily overfits to noise and finds an inconsistent number of models. MVO more consistently segments the correct number of motions (82.2% of the time).

(a) **(b)**

Figure 4.6: A comparison of the segmentations found by a sequential RANSAC (a) and MVO (b) in a single frame of the `swinging_4_unconstrained` segment of the OMD. Sequential RANSAC not only struggles to find the correct number of models in complex dynamic scenes, but also finds incorrect and oversegmented models.

4.5 Estimating Through Occlusion

Highly dynamic scenes not only pose difficult motion estimation challenges but also tend to include significant amounts of occlusion. The *Occlusion* segments of the OMD isolate the challenges posed by direct occlusion in the MEP (Fig. 4.7). The segments consist of a sliding block tower that repeatedly occludes a swinging block.

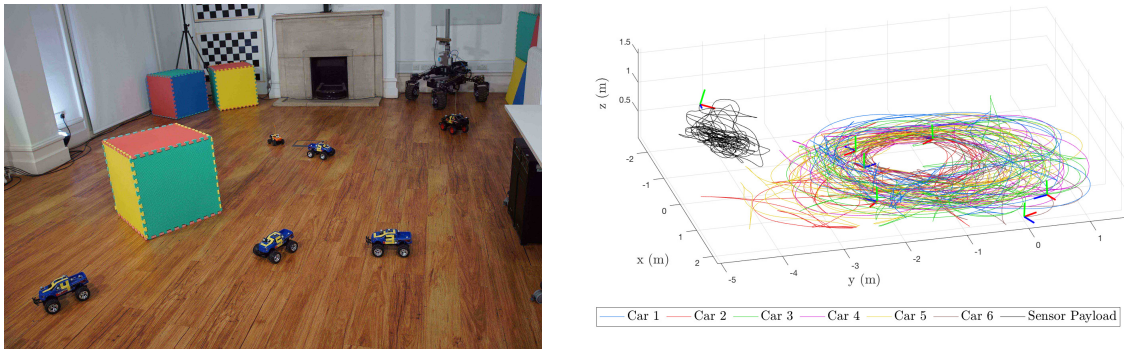


(a) (b)

Figure 4.7: The *Occlusion* scene of the OMD is shown in (a) and the Vicon ground-truth motion trajectories from the unconstrained $SE(3)$ segment are shown in (b). These segments of the dataset consist of a single swinging block (red) observed by the sensor (black) while a sliding block tower (blue) intermittently occludes it. The motion of both blocks creates partial occlusions, where the swinging block can still be partially observed, and total occlusions, where estimates must be extrapolated in the absence of direct observations. This is complicated further by the fact that the occluded block occasionally changes direction while it is occluded. Successfully estimating these motions and interpolating through occlusions is integral to fully addressing the MEP. Example frames from the `occlusion_2_unconstrained` data segment are shown in Fig. B.3.

Additionally, the block tower occasionally stops moving, effectively becoming part of the background. Estimating through occlusion requires the ability to recognize when a previously observed object has been occluded and to predict where and/or when it might reappear. MOT techniques often employ appearance- and motion-based techniques to both predict and recover from these occlusions. In order to address the general MEP, these appearance- and motion-based models must be generalizable to a broad variety of objects and $SE(3)$ motions.

The indirect occlusions caused by sensor or environmental noise, such as motion blur or lighting changes, or by low-level algorithmic failure are difficult to explicitly model or reason about. The high-level object models used by many tracking techniques can be more robust to this type of noise, but object-detection failures can still result in indirect occlusions. Conversely, low-level feature tracking failure commonly results in many short tracklets, which can be problematic for many factorization methods that require points to be tracked for the entirety of the estimation window. Consistently estimating multiple, continuous motions in



(a) **(b)**

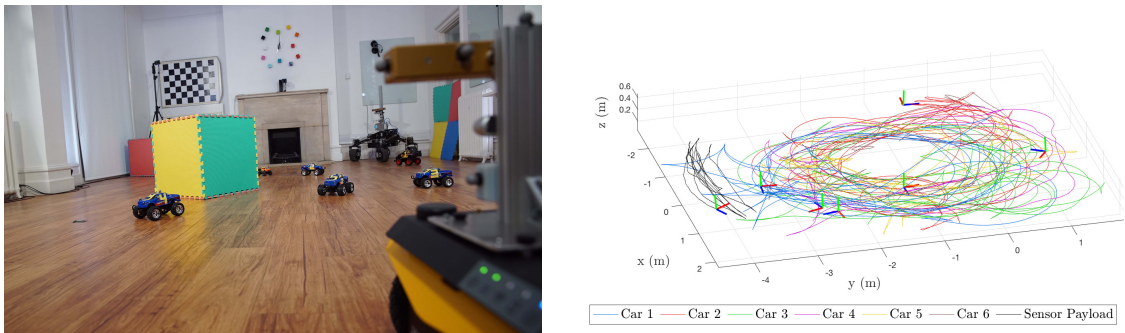
Figure 4.8: The *Toy Cars* scene of the OMD is shown in (a) and the Vicon ground-truth motion trajectories from the unconstrained $SE(3)$ segment are shown in (b). These segments of the dataset consist of a static or dynamic sensor (b, black) observing several independently controlled toy cars (b, other colors) as they maneuver around a block that also adds occlusions. Though motion of each car is restricted to $SE(2)$, these segments of the dataset are the most complex and involve the highest number of independent motions. Example frames from the `cars_3_unconstrained` data segment are shown in Fig. B.4.

the presence of both direct and indirect occlusions is necessary for autonomous navigation in complex dynamic environments.

4.6 Putting it all Together

Addressing the MEP requires addressing each of the aspects detailed in Sections 4.3 to 4.5. The *Toy Cars* data segments of the OMD combine these aspects into a single challenging example of the MEP (Fig. 4.8). The scene consists of several remote-controlled toy cars that move around a static block, frequently changing direction, stopping, and colliding. Occlusions occur when the cars move behind the static block, as well as when they move behind other cars. While the object motions are limited to $SE(2)$, the number and variety of motions present, as well as the frequent occlusions and collisions, still pose a difficult estimation problem.

The cars generally move with relatively constant velocities, but they are controlled by human operators, so they also change direction frequently. This is often in order to avoid collisions, and techniques that explicitly model this type of interaction may be able to predict these changes (Section 3.4.1). These frequent changes in velocity challenge techniques that rely on a constant-velocity assumption,



(a) **(b)**

Figure 4.9: The robot-mounted *Toy Cars* scene of the OMD is shown in (a) and the Vicon ground-truth motion data from the unconstrained $SE(3)$ segment is shown in (b). The scene is similar to the six-car segment shown in Fig. 4.8, but the sensor platform is mounted on a Clearpath Robotics Jackal platform (b, black). This limits the camera motion to $SE(2)$ but significantly increases the perspective effects caused by the cars moving at a broad variety of depths. This also results in more frequent occlusions as the large depth of field causes distant cars to be occluded by nearer ones. Example frames from the `cars_6_robot` data segment are shown in Fig. B.5.

and more accurate and expressive motion models, such as a constant-acceleration prior, may prove more effective (Section 3.5.2).

The cars are also significantly smaller in size than the objects used in the other segments, which poses a greater challenge in segmenting and estimating the motions. Likewise, four of the cars are of the same model, which also challenges appearance-based techniques.

An additional *Toy Cars* segment was recorded with the sensor apparatus mounted on a Clearpath Robotics Jackal (Fig. 4.9). Recording from the perspective of a robotic vehicle reduces the complexity of the sensor egomotion from $SE(3)$ to $SE(2)$, but also greatly increases the perspective effects for the cameras. The broad depth of field in this scene highlights problems with the affine camera model commonly used in factorization and optical flow techniques (Sections 3.4.1 and 3.4.4). Furthermore, occlusions cause by the static block, as well as other cars, are much more frequent in the robot-mounted segment due to the low angle of the camera.

4.7 Summary

This chapter introduces the OMD, a unique dataset designed as a scaffold for the development and evaluation of new multimotion estimation techniques. The dataset contains several different multimotion scenes that each isolates a challenging aspect of the MEP. This data is used throughout the remainder of this thesis to quantitatively evaluate MVO. The experiments follow the design of the OMD, including both investigations of specific challenges and demonstrations on the full MEP.

5

Exploring Multimotion Estimation

Contents

As shown in the previews chapters, the MEP is a multifaceted challenge, and many approaches only address particular aspects of the problem. Fully addressing the MEP requires both *estimation*, i.e., calculating the motion of a set of points, and *segmentation*, i.e., clustering points according to their movement between observations. The interdependence of these tasks creates a *chicken-and-egg* problem that is addressed in single-motion VO systems by using heuristics (e.g., number

of features) to select the egomotion and ignore the other motions in the scene. These heuristics are not readily extensible to *multimotion* estimation problems and analyzing multiple independently moving bodies remains a challenging problem for state-of-the-art vision systems. This chapter introduces the MVO pipeline (Fig. 5.1), which extends the traditional stereo VO pipeline (Fig. 3.2) by applying multimodel-fitting techniques to estimate trajectories for *every* motion in a scene. An earlier version of this work originally appeared in Judd et al. (2018a) at JIRCS 2018 and was expanded in Judd et al. (2018b) at IROS 2018.

As with traditional stereo VO pipelines, incoming RGB stereo images are first rectified and undistorted according to known camera extrinsics and intrinsics. Salient image points are detected and matched across left and right images in each stereo pair and across temporally consecutive pairs of stereo frames. These stereo- and temporally-matched feature points are back-projected into 3D space using the camera intrinsics, forming world- and image-space tracklet histories for each feature point. This set of tracklets, $\mathscr{P}$, becomes the input to the multimotion segmentation and estimation engine (Fig. 5.1b). These tracklets could alternatively be found by using appropriate techniques to associate observations from other 3D sensors (e.g., RGB-D cameras) over time.

The observed motion of each object in the scene, as well as the tracklets detected on that object, is generated by a combination of the camera egomotion and the motion of the object. The multimotion engine segments tracklets by their observed motion, while simultaneously estimating that motion. For tracklets detected on static objects in the scene, this motion corresponds to the true egomotion of the camera.

In the absence of *a priori* information about the scene, each group of tracklets can be used to estimate the camera egomotion by assuming those tracklets belong to a static object. Each of these camera egomotion *hypotheses*, ${}^{\ell}\mathscr{T}_C$, can later be converted into geocentric estimates of the camera egomotion, $\mathscr{T}_C$, and object motions, $\mathscr{T}_\ell$, by identifying the static part of the scene (e.g., as in VO). A key benefit of this approach is that identifying the egomotion label can be deferred until

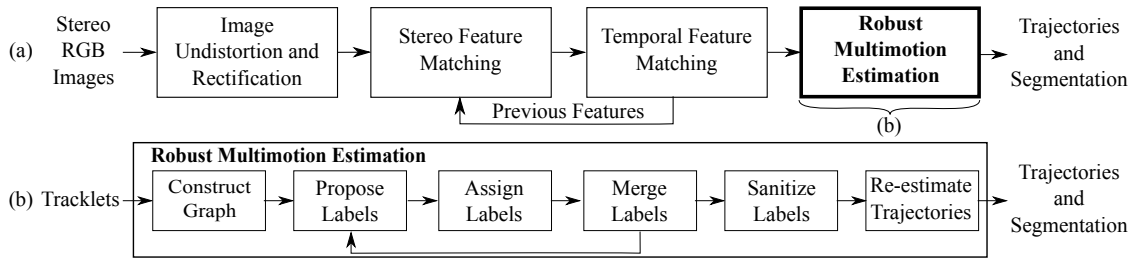


Figure 5.1: An illustration of the stereo MVO pipeline, (a), including details of the multimotion fitting component, (b). The pipeline operates on RGB stereo image pairs and estimates the segmentation and trajectories of all motions in the scene. It extends the standard VO pipeline (Fig. 3.2) by replacing the egomotion estimator with a multimotion estimator that operates on the tracklets produced by the previous stages. The multimotion-fitting pipeline builds a neighborhood graph based on the distances between pairs of points throughout the estimation batch. The pipeline iteratively splits and estimates new labels using the neighborhood graph, assigns labels based on an energy functional, and merges labels that can be considered redundant until convergence. Finally, the label set is sanitized and a full-batch estimation step produces the final segmentation and trajectories.

the segmentation and estimation of all motions is complete. This simultaneous segmentation and estimation maintains the tightly coupled nature of the MEP that is ignored in approaches with separate egomotion and third-party estimation stages. Additionally, the fully segmented and estimated motions can be used in a variety of heuristics to select the egomotion label, such as the dominant motion (as in most VO systems) or the greatest similarity and overlap with previously estimated motions.

Section 5.1 describes how MVO decomposes a set of tracklets into their constituent motions. The segmentation and estimation can be cast as a multilabeling problem where a label, ℓ, represents a motion estimate, ${}^{\ell}\mathcal{T}_C$, calculated from a subset of the tracklets in the scene, $\mathscr{P}_\ell \subseteq \mathscr{P}$. The segmentation is found by minimizing an energy functional using multilabeling approaches, and the set of labels, $\mathscr{L}$, represents the various observed motions present in the scene.

Section 5.2 explains how each motion is individually estimated after the segmentation converges. A full-batch, single-motion estimation approach is used to refine the trajectory estimates after the segmentation has converged because, at that point, the MEP has been decomposed into a set of parallel, single-motion estimation problems.

Section 5.3 explains how the trajectories can be expressed in an egocentric, i.e., camera-attached, or geocentric, i.e., Earth-attached, frame. Throughout the

multilabeling and batch estimation stages of the pipeline, all motion hypotheses are initially treated as potentially belonging to the static portions of the scene (i.e., the camera's egomotion). The geocentric trajectory for each motion, $\mathcal{T}_\ell$, is found in a final step where a label is selected to represent the motion of the camera.

Finally, Section 5.4 demonstrates the performance of the MVO pipeline on scenes from the OMD. The algorithm is capable of simultaneously estimating multiple $SE\,(3)$ motions, including the camera egomotion, in highly dynamic multimotion scenarios. The quantitative estimation error for the camera egomotion and the third-party motions is discussed along with the pipeline limitations in Section 5.5.

The novel contributions of this chapter are:

- Simultaneous motion segmentation and estimation of *every* motion of the scene using low-level feature points, rather than relying on higher-level appearance information or *a priori* motion constraints (Section 5.1);
- Full $SE\,(3)$ trajectory estimation of each motion in the scene using only a rigid-body assumption (Section 5.2);
- Deferral of the designation of a given motion as belonging to the camera (i.e., the egomotion) until all hypothetical egomotion trajectories are estimated, after which their geocentric equivalents can be calculated (Section 5.3);
- Evaluation of the MVO pipeline on complex multimotion scenes from the OMD (Section 5.4).

5.1 Simultaneous $SE\,(3)$ Motion Segmentation and Estimation

Motion segmentation can be cast as a multilabeling problem wherein a label is assigned to groups of tracklets with similar motions. When combined with motion estimation, this label also represents a motion trajectory that explains the motion of those tracklets over time. Herein lies the *chicken-and-egg* problem: the motion trajectories must be estimated from groups of tracklets, but the segmentation of those groups depends on the available motion estimates. To address this interdependency in MVO, motions are iteratively segmented and estimated in

an alternating fashion until the labeling converges. This allows the labeling to deal with varying numbers and types of motions, but requires the set of labels, $\mathscr{L}$, to adapt dynamically to the scene.

These labels are assigned by minimizing a cost function over a graph, $\mathcal{N}$, of all observed tracklets (Section 5.1.1). Each label, $\ell \in \mathscr{L}$, represents a motion trajectory explaining the observed motion of a set of tracklets, ${}^{\ell}\mathscr{T}_C$, and an outlier label, $\mathscr{O}$, is assigned to points whose motions are not well explained by any other label. New labels are proposed for each disconnected component of a label's subgraph through a multiframe RANSAC procedure (Section 5.1.2). Motion labels are assigned to minimize the reprojection residual of the associated trajectory and maximize the label smoothness in the graph (Sections 5.1.3 and 5.1.4). Redundant and oversegmented labels are then merged (Section 5.1.5). Once the segmentation converges, the final labels are then sanitized and any remaining outliers are rejected (Section 5.1.5). Each motion can then be estimated independently using well-known single-motion estimation techniques (Section 5.2).

5.1.1 Defining the Graph

In order to apply existing multilabeling techniques, the tracklets in a scene are embedded in a geometric neighborhood graph, $\mathcal{N}$, where each vertex represents an observed tracklet. The graph is finite and undirected, and its structure forms the basis of both assigning labels and proposing new ones.

The goal of this graph is to embody the rigid-body assumption, i.e., the distance between two points undergoing the same motion is constant. Nodes of the graph represent observed tracklets, and the structure of the graph is dictated by the connectivity of its edges. This connectivity is a function of the cost of an edge between two nodes (a nonexistent edge between nodes can be considered to have infinite cost).

To encode the rigid-body principle, the cost, ω, of an edge should be a function of the distance between two points over time. In rigid-body kinematics, the

instantaneous distance between two points, $\not{p}_C^i$ and $\not{p}_C^j$, at time, k,

$$d\left(\mathbf{p}_{C_k}^{iC_k}, \mathbf{p}_{C_k}^{jC_k}\right) = \left\|\mathbf{p}_{C_k}^{iC_k} - \mathbf{p}_{C_k}^{jC_k}\right\|, \tag{5.1}$$

remains constant over all k. The adherence of two tracklet points to the rigid-body principle can therefore be encoded as a function of the dispersion in the displacement between those points over time. This dispersion is often defined as the deviation from the mean displacement, e.g., the variance, of the set of displacements between two points (Mateus and Horaud 2007).

Computing this deviation for every pair of points quickly becomes costly, so reasonable approximations can be used to improve efficiency. The minimum distance over time poorly approximates the rigid-body principle because points with different motions can be temporarily close, and large objects can have distant motions with similar motions. Conversely, the cost can be defined as the maximum distance between those tracklets.

$$\omega_{ij} \propto \max_k d\left(\mathbf{p}_{C_k}^{i_k C_k}, \mathbf{p}_{C_k}^{j_k C_k}\right). \tag{5.2}$$

This applies a low cost for edges between features that are consistently close while penalizing edges between features that are ever far apart. Though distant points on the same large object will have expensive edges, it is likely that there are sufficient features on the object to form a chain of cheap edges between them, and it is simple and efficient to compute.

Using this definition of the cost between two nodes, the costly edges can be truncated to improve the efficiency of graph computations, as a fully connected graph is unwieldy. Two common forms of this truncation are r-disc and k-nearest-neighbors ($k$nn) graphs. An r-disc graph defines a cost radius, r, around a node and truncates any edges with costs greater than this radius. A $k$nn graph defines a number of neighbors, k, and limits each node's connectivity to its k least-costly edges. A $k$nn graph has advantages over an r-disc graph in that it enforces a specific level of connectivity throughout the graph, and allows for long edges in regions of the graph with low tracklet density, which is common in VO applications.

Unfortunately, a $k$nn graph must be directed, which can be prohibitive for applying certain algorithms. Treating all edges in a $k$nn graph as undirected means all nodes will have at least k edges; conversely, all edges that are not symmetric can be truncated, meaning some nodes will have less than k edges, but none will have more. Using k as an upper-bound on the connectivity of the graph is both efficient and prevents a point's label from being completely dictated by its many neighbors.

The final consideration is the space in which the distance in (5.1) is measured. Given a sensor with minimal noise, such as lidar, it is ideal to compare points in 3-D world space. However, in the case of stereo camera sensors, depth estimates can have significant error, especially as the depth increases. For these sensors, it can be more robust to define this distance in 2-D or 3-D (i.e., with disparity) image space.

Taking this all into account, the MVO pipeline defines the graph as a directed $k$nn graph where the cost between nodes is defined as the maximum distance in image space between those image tracklets over the entire batch,

$$\omega_{ij} \coloneqq \max_{k=1\dots K} d\left(\mathbf{s}\left(\mathbf{p}_{C_k}^{i_k C_k}\right), \mathbf{s}\left(\mathbf{p}_{C_k}^{j_k C_k}\right)\right),$$

where $\mathbf{s}(\cdot)$ applies the nonlinear perspective camera projection in (2.11). This distance is only calculated for tracklets that coexist at some point, meaning there can be no edge between temporally disjoint tracklets. This combines the efficiency of both (5.2) and a $k$nn structure with the robustness of image space to triangulation noise. This connectivity forms the basis for the later steps of label generation and assignment.

5.1.2 Proposing New Trajectories

In addition to being highly varied, dynamic environments are, by their nature, constantly changing. This means multimotion estimation techniques must be able to adapt to a variety of environments, as well dynamically adapt to new and changing motions within those environments. The set of labels describing the currently estimated motions, $\mathscr{L}$, must dynamically grow and adapt to correctly describe a given scene. To accomplish this, new labels are generated by splitting

label support groups whenever their tracklets' motions could more accurately be explained by multiple trajectories. The graph structure in Section 5.1.1 was chosen to approximate the rigid-body principle, so its connectivity can be used as a prior for generating new motion hypotheses.

The points currently assigned to a given label, $\mathscr{P}_\ell$, form a subgraph, $\mathcal{N}_\ell \subseteq \mathcal{N}$. A potential new label, ℓ', is generated for each fully-disjoint component of $\mathcal{N}_\ell$. This ensures a level of spatial smoothness while allowing new labels to be proposed from large label supports comprised of tracklets from spatially or temporally distinct motions in the scene.

The new label proposals are generated by computing both the dominant motion of the given points and the segmentation between inliers and outliers of that motion. In this case, the MEP has been isolated to a single-motion segmentation and estimation problem that can be solved by applying RANSAC in a frame-to-frame fashion, similar to standard VO systems.

Three tracklets are sampled from those visible in the current, k, and previous, $k-1$, frames to estimate the $SE(3)$ transform between the two frames ${}^{\ell'}\mathbf{T}_{C_k,C_{k-1}}$. This transform estimate can be generated using the process detailed in Section 3.3.1. The proposed transform is then evaluated according to how many tracklet reprojection residuals,

$$e_k\left(\mathscr{p}_C^j, {}^{\ell'}\mathscr{T}_C\right) \coloneqq d\left(\mathbf{s}\left(\mathbf{p}_{C_k}^{j_k C_k}\right), \mathbf{s}\left({}^{\ell'}\mathbf{T}_{C_k C_{k-1}} \mathbf{p}_{C_{k-1}}^{j_{k-1} C_{k-1}}\right)\right), \tag{5.3}$$

are within a given threshold error, e_{th}, where d is defined in (5.1). This process of sampling, estimation, and evaluation is repeated many times and the transform with the largest inlier set is appended to the proposed trajectory hypothesis, ${}^{\ell'}T_C$, and the transform corresponding to the next pair of frames in the estimation window is calculated.

This proposal process is repeated for each label, and any tracklets found to be outliers of the newly estimated models are appended to the outlier label, $\mathscr{O}$. New labels are generated from the outlier label through the same process after outliers are culled from each other label.

5.1.3 Defining the Cost Functional

Casting the MEP as a multilabeling problem involves assigning each tracklet, $\mathcal{p}_C \in \mathscr{P}$, to a motion label, $\ell \in \mathscr{L}$, such that some energy functional, $E(\mathscr{L})$, is minimized. This functional is designed to balance the fidelity with which a label describes the motion of a tracklet with a piecewise-smooth model of the environment. An additional complexity term is included to penalize the use of additional labels, i.e., to incentivize a compact solution. The energy functional incorporates the residual error, the label smoothness, and the label complexity term such that,

$$E(\mathscr{L}) \coloneqq \underbrace{\sum_{\mathcal{p}_C \in \mathscr{P}} \rho\left(\mathcal{p}_C, \ell\left(\mathcal{p}_C\right)\right)}_{\text{Residual}} + \lambda \underbrace{\sum_{\left(\mathcal{p}_C^i, \mathcal{p}_C^j\right) \in \mathcal{N}} \omega_{ij} V\left(\mathcal{p}_C^i, \mathcal{p}_C^j\right)}_{\text{Smoothness}} + \underbrace{\sum_{\ell \in \mathscr{L}} \gamma_\ell \psi_\ell}_{\text{Complexity}}, \tag{5.4}$$

where $\ell\left(\mathcal{p}_C\right)$ gives the label currently assigned to $\mathcal{p}_C$ and λ is a user-selected proportionality parameter.

Residual Cost

The residual term,

$$\sum_{\mathcal{p}_C \in \mathscr{P}} \rho\left(\mathcal{p}_C, \ell\left(\mathcal{p}_C\right)\right),$$

penalizes labels that poorly describe the observed data. It is defined as the sum of the residual errors in applying the label trajectories to tracklets. The residual for each point-label pair is defined as

$$\rho\left(\mathcal{p}_C, \ell\right) \coloneqq \max_{k \in 1 \ldots K} e_k\left(\mathcal{p}_C, {}^{\ell}\mathcal{T}_C\right),$$

where e_k as defined in (5.3). The maximum error is used, rather than the mean or another function, because it is important to be conservative in labeling tracklets as belonging to a single motion in order to preserve the consistency of that motion. Tracklets that do not fit that motion, even for a single frame, due to feature mismatches or sensor noise can severely influence the trajectory estimation.

Smoothness Cost

The smoothness term,

$$\sum_{(i,j)\in\mathcal{N}} \omega_{ij} V\left(\mathcal{p}_C^i, \mathcal{p}_C^j\right),$$

penalizes neighboring tracklets that do not share the same label by their edge cost, ω_{ij}. This encourages a spatially piecewise-smooth solution, which is consistent with a geometric model of the scene consisting of contiguous objects separated by depth boundaries. The cost is a weighted sum of all edges penalized according to

$$V\left(\mathcal{p}_C^i, \mathcal{p}_C^j\right) \coloneqq \begin{cases} 1 & \text{if } \ell\left(\mathcal{p}_C^i\right) \neq \ell\left(\mathcal{p}_C^j\right) \\ 0 & \text{otherwise} \end{cases}.$$

This definition of V mirrors the standard Potts model (Boykov et al. 1999).

Complexity Cost

The complexity term,

$$\sum_{\ell\in\mathscr{L}} \gamma_\ell \psi_\ell,$$

encourages a compact solution by penalizing the use of each label. This is consistent with the notion of a *minimum description length*, i.e., Occam's Razor, which favors compactness over complexity, as a solution with many models is likely overfitting to the data. It is the sum of the per-label cost, γ_ℓ, of each label with non-empty support set according to the function,

$$\psi_l \coloneqq \begin{cases} 1 & \text{if } |\mathscr{P}_\ell| > 0 \\ 0 & \text{otherwise} \end{cases}.$$

The cost of using a given label in the solution can be defined in several ways. For example, all motion labels could incur equivalent costs, aside from the outlier label, $\mathscr{O}$, but this means simple motions are just as costly as physically unlikely motions. If prior information about the environment is known, such as the camera or third-party kinematics, label costs could be designed to encourage common motions and penalize kinematically complex motions. A privileged "playbook" of likely motions can also be included in the label set as default options as assigned low costs.

Outlier Label Costs

The outlier label, $\mathscr{O}$, is designed to be attractive to all points whose motions are not well explained by existing labels, so its residual and complexity costs are defined uniquely. The residual energy of the outlier label decays exponentially with that of the best-fitting label,

$$\rho\left(\mathscr{p}_C, \mathscr{O}\right) \coloneqq \alpha \exp\left(-\frac{1}{\beta} \min_{\ell \in \mathscr{L}} \rho\left(\mathscr{p}_C, \ell\right)\right),$$

where α and β are tuning parameters. This cost is designed such that the outlier data cost for tracklets whose motions are well-explained by extant labels will be high, but it will be low for tracklets with high costs for all labels. The label cost, $\gamma_{\mathscr{O}}$, for the outlier label is zero as outliers are assumed to always exist.

5.1.4 Assigning Labels

Given the current label set, $\mathscr{L}$, the multilabeling problem requires assigning a label to each tracklet. This can be done using techniques discussed in Section 3.4.1, such as α-expansion (e.g., PEARL, Isack and Boykov (2012)) or convex optimization (e.g., CORAL, Amayo et al. (2018)). These approaches operate on a graph structure in order to minimize the residual and smoothness energies of (5.4).

The α-expansion approach employed by PEARL proceeds as a series of binary graph cuts. For each label, points not currently assigned to that label can be relabeled if it would reduce the residual and smoothness energies, and this process continues until no new relabeling would reduce those energies.

In contrast, CORAL relaxes this discrete labeling, allowing a point's label to be defined on the continuous range $[0, 1]$. This "soft" labeling leads to a primal-dual formulation of the minimization that involves point-wise calculations, meaning it is well suited for parallelization on a GPU. Once the minimization has converged, the soft labels can be discretized by taking the strongest label for each point. Points with ambiguous soft labels can be labeled as outliers to avoid mislabeling points.

This minimization only addresses the residual and smoothness terms and can result in an oversegmentation due to outliers and poorly estimated intermediate

trajectories. Model merging is therefore used to address the complexity term and improve the motion segmentation.

5.1.5 Merging Redundant Labels

The tools described in Section 5.1.4 only minimize the residual and smoothness terms in (5.4), so the full cost, including the complexity term, must be minimized separately. This is achieved by determining if merging two labels, ℓ and ℓ', i.e., relabeling all $\mathscr{P}_{\ell'}$ as ℓ, would decrease the total energy of (5.4). This occurs when the increase in residual error due to reduced overfitting is less than the cost of using the label, $\gamma_{\ell'}$, and any change in smoothness.

The merging stage only considers label pairs with tracklets adjacent in $\mathcal{N}$, i.e., those where $\mathcal{N}_\ell$ is connected to $\mathcal{N}_{\ell'}$. If they are disconnected, merging the two supports would be undone by the splitting routine (Section 5.1.2). If the two support sets are connected then the new label will persist until the next labeling stage. When more than one merge would reduce the total energy, the one that results in the greatest decrease in cost is chosen. Merging continues until no more merges would reduce (5.4). The outlier label, $\mathscr{O}$, is excluded from merging.

The algorithm iterates the label splitting, assignment, and merging (Sections 5.1.2 to 5.1.5) until the labels converge or a maximum number of iterations have been reached. Label convergence can be defined in a number of ways, such as the total energy, the change in energy, or the change in the labeling. The final label set is then sanitized before being used to estimate the final trajectory hypotheses.

5.1.6 Sanitizing Labels

The final labels are sanitized to refine the segmentation output and remove noisy tracklets before the final model estimation. A merging step first combines any redundant labels regardless of graph connectivity as there is no subsequent splitting stage. This merging stage overcomes the limitation in the approximation of the rigid-motion constraint in Section 5.1.1, which does not allow edges between spatially

distant points. Through merging, large or disconnected objects moving with a single motion can be explicitly merged.

After merging, tracklets whose residual error is greater than a threshold, e_{th}, are relabeled as outliers. Likewise, any label with fewer than a minimum number of support tracklets or that exists for fewer than a minimum number of frames is merged with $\mathcal{O}$. Geometrically, these thresholds are minimally three points and two frames, respectively, but they can be tuned higher to prevent overfitting to spuriously correlated tracklet motions in a small portion of the tracklet set or estimation window. This provides a consistent set of tracklets for the batch estimation of each motion.

5.2 Discrete $SE(3)$ Estimation

Once the label set has converged and been sanitized, the trajectory estimates can be treated independently. Each label represents an egomotion hypothesis, ${}^{\ell}\mathcal{T}_C$, that explains the motion of the tracklets, $\mathscr{P}_\ell$, assuming they are static. Any one of these hypotheses can correspond to the true camera motion, i.e. the tracklets in $\mathscr{P}_\ell$ are actually static. Therefore, the estimates can be refined using the same single-motion bundle adjustment approach described by Barfoot (2017), which is designed to estimate the camera egomotion.

First, the system state, $\mathbf{x}$, of each label is defined to include both the estimated pose transforms, ${}^{\ell}\mathcal{T}_C \coloneqq \left({}^{\ell}\mathbf{T}_{C_kC_1}\right)_{k=2\ldots K}$, and the landmark points, $\left\{\mathbf{p}_{C_1}^{j_1C_1}\right\}_{j=1\ldots|\mathscr{P}_\ell|}$. The state $\mathbf{x}_{jk} \coloneqq \left\{{}^{\ell}\mathbf{T}_{C_kC_1}, \mathbf{p}_{C_1}^{j_1C_1}\right\}$ is defined for each pair of transforms and points belonging to label ℓ. The batch size, K, is the length of the estimation window, which can either be the entire image sequence, or some fixed-length sliding window.

Each observation, $\mathbf{y}_{jk}$, of point $\underrightarrow{p}^{j}$ at pose ${}^{\ell}\mathbf{T}_{C_kC_1}$ is modeled as

$$\begin{aligned}\mathbf{y}_{jk} &\coloneqq \mathbf{g}\left(\mathbf{x}_{jk}\right) + \mathbf{n}_{jk} = \mathbf{s}\left(\mathbf{z}\left(\mathbf{x}_{jk}\right)\right) + \mathbf{n}_{jk} \\ &= \mathbf{s}\left({}^{\ell}\mathbf{T}_{C_kC_1}\mathbf{p}_{C_1}^{j_1C_1}\right) + \mathbf{n}_{jk}.\end{aligned}$$

The measurement model, $\mathbf{g}\left(\cdot\right)$, encompasses both the motion model, $\mathbf{z}\left(\cdot\right)$, which applies $SE(3)$ transforms to observed points, and the perspective camera model, $\mathbf{s}\left(\cdot\right)$. The model assumes additive Gaussian noise, $\mathbf{n}_{jk}$, with zero mean and covariance

$\mathbf{R}_{jk}$. The least-squares cost function is defined as the difference between the measurement model and the observations,

$$J \coloneqq \frac{1}{2} \sum_{jk} \mathbf{e}_{y,jk} \left(\mathbf{x} \right)^T \mathbf{R}_{jk}^{-1} \mathbf{e}_{y,jk} \left(\mathbf{x} \right),$$

where,

$$\mathbf{e}_{y,jk} \left(\mathbf{x} \right) \coloneqq \mathbf{y}_{jk} - \mathbf{s} \left({}^{\ell}\mathbf{T}_{C_k C_1} \mathbf{p}_{C_1}^{j_1 C_1} \right).$$

This cost is linearized about an operating point, $\mathbf{x}_{\text{op}}$, and then minimized using Gauss-Newton. The operating point is perturbed according to the transform perturbations, $\{\boldsymbol{\epsilon}_k \in \mathbb{R}^6\}$, and landmark perturbations, $\{\boldsymbol{\zeta}_j \in \mathbb{R}^3\}$, which together form the full state perturbation, $\delta\mathbf{x}$. An indicator matrix $\mathbf{P}_{jk}$ is defined such that $\delta\mathbf{x}_{jk} = \mathbf{P}_{jk}\delta\mathbf{x}$ is the perturbation of the $\left\{ {}^{\ell}\mathbf{T}_{\text{op},C_k C_1}, \mathbf{p}_{\text{op},C_1}^{j_1 C_1} \right\}$ pair.

The error function is linearized using $\mathbf{G}_{jk}$, the Jacobian of the measurement function, $\mathbf{g}\left(\cdot\right)$,

$$\begin{aligned}
\mathbf{G}_{jk} &\coloneqq \mathbf{S}_{jk} \mathbf{Z}_{jk}, \\
\mathbf{S}_{jk} &= \left. \frac{\partial \mathbf{s}}{\partial \mathbf{z}} \right|_{\mathbf{z}(\mathbf{x}_{\text{op},jk})}, \\
\mathbf{Z}_{jk} &= \begin{bmatrix} \left({}^{\ell}\mathbf{T}_{\text{op},C_k C_1} \mathbf{p}_{\text{op},C_1}^{j_1 C_1} \right)^{\odot} & {}^{\ell}\mathbf{T}_{\text{op},C_k C_1} \mathbf{D} \end{bmatrix}, \\
\mathbf{D} &= \begin{bmatrix} \mathbf{1} \\ \mathbf{0}^T \end{bmatrix},
\end{aligned}$$

where the matrix operator

$$\boldsymbol{p}^{\odot} = \begin{bmatrix} k\mathbf{p} \\ k \end{bmatrix}^{\odot} = \begin{bmatrix} k\mathbf{1} & -\mathbf{p}^{\times} \\ \mathbf{0}^T & \mathbf{0}^T \end{bmatrix} \tag{5.5}$$

and $(\cdot)^{\times}$ is the skew-symmetric matrix operator defined in (2.3). The cost function can then be linearized using

$$\begin{aligned}
J &\approx J(\mathbf{x}_{\text{op}}) - \mathbf{b}^T \delta\mathbf{x}_{jk} + \frac{1}{2} \delta\mathbf{x}_{jk}^T \mathbf{A} \delta\mathbf{x}_{jk}, \\
\mathbf{b} &= \sum_{jk} \mathbf{P}_{jk}^T \mathbf{G}_{jk}^T \mathbf{R}_{jk}^{-1} \mathbf{e}_{y,jk} \left(\mathbf{x}_{\text{op}} \right), \\
\mathbf{A} &= \sum_{jk} \mathbf{P}_{jk}^T \mathbf{G}_{jk}^T \mathbf{R}_{jk}^{-1} \mathbf{G}_{jk} \mathbf{P}_{jk}.
\end{aligned}$$

The optimal perturbation, $\delta\mathbf{x}^*$, for minimizing the cost function, J, is the solution to $\mathbf{A}\delta\mathbf{x}^* = \mathbf{b}$. Each element of the state is then updated according to

$$
\begin{aligned}
{}^{\ell}\mathbf{T}_{\mathrm{op},C_kC_1} &\leftarrow \exp(\boldsymbol{\epsilon}_k^{*\wedge})\,{}^{\ell}\mathbf{T}_{\mathrm{op},C_kC_1} \\
\mathbf{p}_{\mathrm{op},C_1}^{j_1C_1} &\leftarrow \mathbf{p}_{\mathrm{op},C_1}^{j_1C_1} + \mathbf{D}\boldsymbol{\zeta}_j^*.
\end{aligned}
$$

The cost function is then relinearized about the updated operating point and the process iterates until convergence. Here, convergence can be defined as a threshold on the number of iterations, the total cost, the change in cost, or the magnitude of the update.

5.3 Egocentric and Geocentric Trajectories

Egocentric motions are expressed relative to the camera frame, while geocentric motions are expressed in some Earth-attached frame. This requires treating the Earth as an inertial reference frame, which is appropriate for the scale of most autonomous platforms.

The motion hypotheses in Section 5.2 are each estimated as a hypothesis for the geocentric egomotion of the camera. By identifying which motion label corresponds to the static background of the scene, the true egocentric and geocentric motions in the scene can be estimated.

The static label, C, representing the camera egomotion may be selected by heuristics, as in traditional VO. Initially, it can be chosen by finding the label with the largest support, $|\mathscr{P}_\ell|$,

$$
C_0 = \arg\max_{\ell} |\mathscr{P}_\ell|,
$$

after which it can be propagated forward in time by choosing a label that maximizes the overlap in support with the previous label.

$$
C_k = \arg\max_{\ell} |\mathscr{P}_\ell \cup \mathscr{P}_{C_{k-1}}|.
$$

While this support-based threshold is used here, other heuristics could incorporate application-specific information, such as attention masks that prioritize parts of

the camera view that usually contain static background objects. These can be combined with post-estimation heuristics like support size and motion consistency to robustly identify the true egomotion label.

5.3.1 Egocentric Trajectories

Egocentric motions are expressed in the moving camera frame, $\underrightarrow{\boldsymbol{\mathcal{F}}}_{C_k}$. The egocentric motion of the camera is identity by definition and the egocentric motions of the scene are given by

$$\forall \ell \in \mathscr{L},\ {}^{\text{ego}}\mathbf{T}_{\ell_K \ell_1} \coloneqq {}^{\ell}\mathbf{T}^{-1}_{C_K C_1}.$$

One of these motions, ${}^{\text{ego}}\mathcal{T}_C$, is the egocentric motion of the static world *caused* by the camera motion.

5.3.2 Geocentric Trajectories

Geocentric motions are expressed in some Earth-attached frame whose motion is assumed to be identity. The geocentric motion of the camera is given by the motion hypothesis estimated from the static background,

$$\mathcal{T}_C \coloneqq {}^{\ell}\mathcal{T}_C\Big|_{\ell = C_k}.$$

The geocentric motions of the rest of the scene are given by

$$\forall \ell \in \mathscr{L} \setminus C_k,\ \mathbf{T}_{\ell_k \ell_1} = \mathbf{F}_{\ell_k \ell_1} \mathbf{T}_{\ell_1 C_1} {}^{\ell}\mathbf{T}^{-1}_{C_k C_1} \mathbf{T}_{C_k C_1} \mathbf{T}^{-1}_{\ell_1 C_1},$$

where $\mathbf{F}_{\ell_k \ell_1}$ is the object deformation matrix and is assumed to be identity, (i.e., rigid body). The initial camera-to-object transform,

$$\mathbf{T}_{\ell_1 C_1} = \begin{bmatrix} \mathbf{C}_{\ell_1 C_1} & \mathbf{p}^{C_1 \ell_1}_{\ell_1} \\ \mathbf{0}^T & 1 \end{bmatrix},$$

relates the camera frame to the object frame, which is assigned to the center of motion of each object. The object center is calculated from the centroid of all points, $\mathscr{P}_\ell$, projected into the first observed frame,

$$\mathbf{p}^{C_1 \ell_1}_{\ell_1} = -\frac{1}{|\mathscr{P}_\ell|} \mathbf{C}_{\ell_1 C_1} \sum_{j=1}^{|\mathscr{P}_\ell|} {}^{\ell}\mathbf{T}^{-1}_{C_{t_j} C_1} \mathbf{p}^{j_{t_j} C_{t_j}}_{C_{t_j}},$$

where t_j is the first frame where p^j is observed, and $\mathbf{C}_{\ell_1 C_1}$ is arbitrary and assumed to be identity. This averaging allows the centroid estimate to adjust as new points are observed due to rotation or occlusion. In a sliding-window estimation pipeline, this transform can be determined from the previously estimated trajectory estimates.

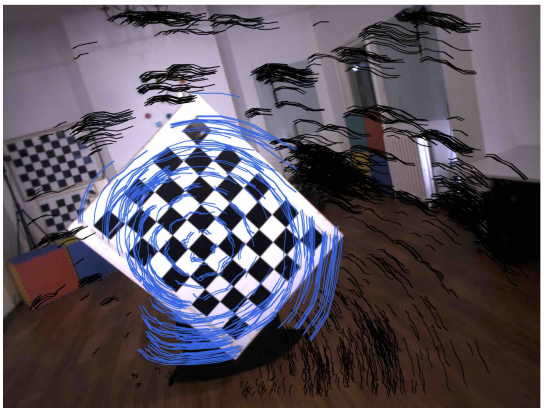
5.4 Evaluation

The accuracy of the MVO algorithm was evaluated on Bumblebee XB3 data from the OMD. Feature detection and matching were performed using LIBVISO2 (Geiger et al. 2011) and the Gauss-Newton minimization was performed with Ceres (Agarwal et al.) using analytical derivatives (Section 5.2). The parameters used in LIBVISO2 are given in Table A.1. The transforms between the Vicon frames and the MVO estimated frames are arbitrary, so the first 10% of each trajectory is used to calibrate this transform (Zhang and Scaramuzza 2018). All errors are reported for geocentric trajectory estimates, so a portion of the geocentric error of each motion is due to the error in the camera motion estimate.

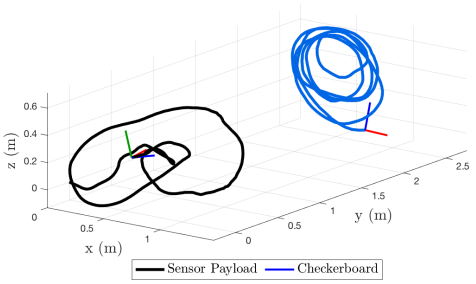
Estimating Rotation

The first evaluation of the performance of the MVO pipeline is through the *Pinwheel* segments of the OMD (Section 4.3). Despite only including two independent motions, these segments require accurate rotational estimation that can be difficult for existing multimotion estimation techniques. Fig. 5.2 illustrates the performance of the MVO pipeline on a 300-frame image sequence from the highly rotational `pinwheel_2_unconstrained` segment, beginning at frame 900. Estimation was performed as an 8-frame sliding window, with 5 neighbors for each point in the graph, 100 RANSAC iterations per new label, $e_{\text{th}} = 4$, $\alpha = 100$, $\beta = 5$, $\psi_\ell = 100$, $\lambda = 1$, and a minimum model size and length of 20 points and 3 frames, respectively.

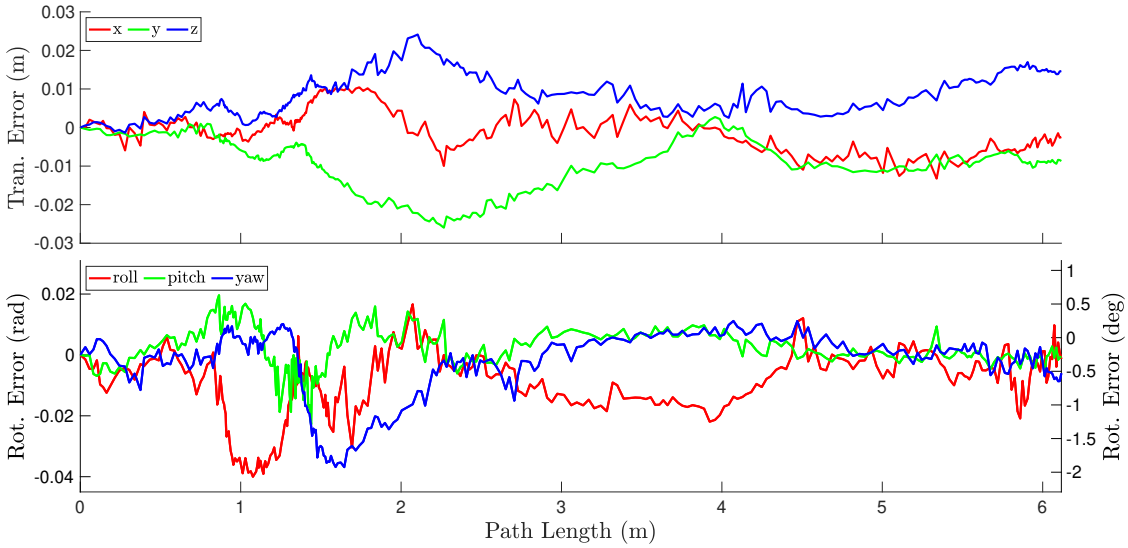
MVO is able to consistently segment and estimate the motions of both the camera and the checkerboard (Figs. 5.2a and 5.2b). The camera egomotion exhibited a maximum total drift of 0.03 m, 1.76% of the total path length (6.12 m), and a maximum rotational error of $-2.29°$, $-1.31°$, and $-2.11°$ in roll-pitch-yaw,



(a) Tracklet Segmentation



(b) Trajectory Plot



(c) Egomotion

Figure 5.2: Performance of the MVO pipeline over a 300-frame portion of the `pinwheel_2_unconstrained` segment of the OMD. The segmentation of a single frame (a) and the total trajectory plot (b) show the consistency of the results qualitatively. The quantitative translational and rotational errors for the estimated egomotion of the camera (c) are shown compared to ground-truth trajectory data. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

respectively (Fig. 5.2c). The maximum translational and rotational errors for the checkerboard were 0.54 m (over a total path of 12.61 m), $-49.50°$, $-14.27°$, and $29.91°$ (Fig. 5.3).

The translational error of the checkerboard estimate is much larger than the corresponding egomotion error, largely because of the volatility of the feature distribution. The distribution of features on the checkerboard changes from frame to frame due to changes in the camera view and lighting, as well as tracking failure.

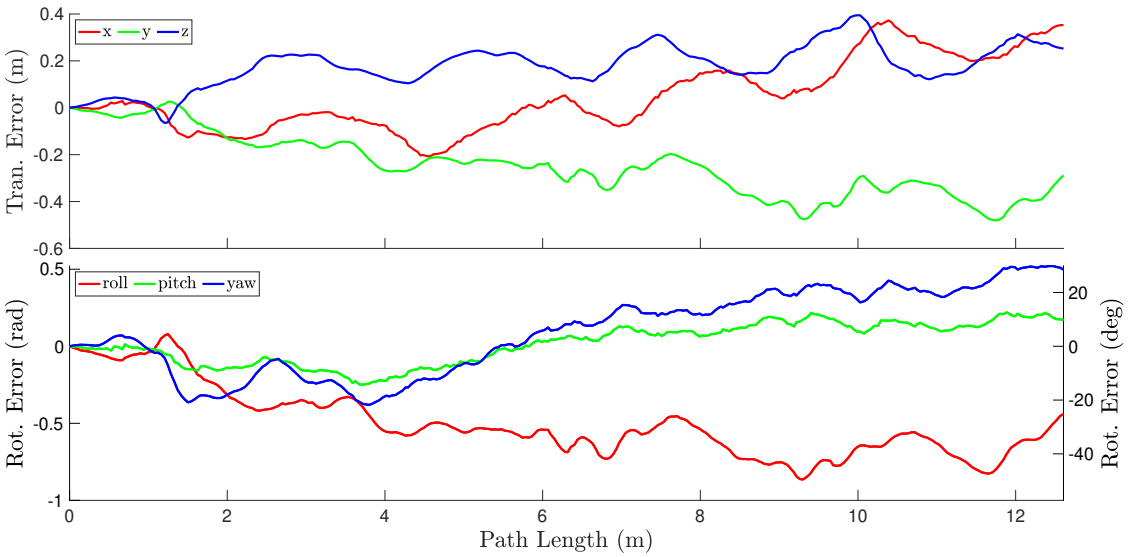


Figure 5.3: The translational and rotational errors for the estimated motion of the checkerboard over a 300-frame portion of the `pinwheel_2_unconstrained` segment of the OMD compared to ground-truth trajectory data. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

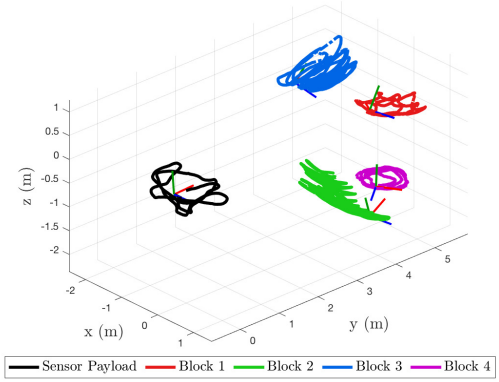
This causes the estimated motion to shift with the feature centroid, resulting in the translational errors shown in Fig. 5.3. The roll and yaw errors of the checkerboard estimates are large and result from the accumulation of small estimation errors orthogonal to the true rotation axis.

Multimotion Estimation

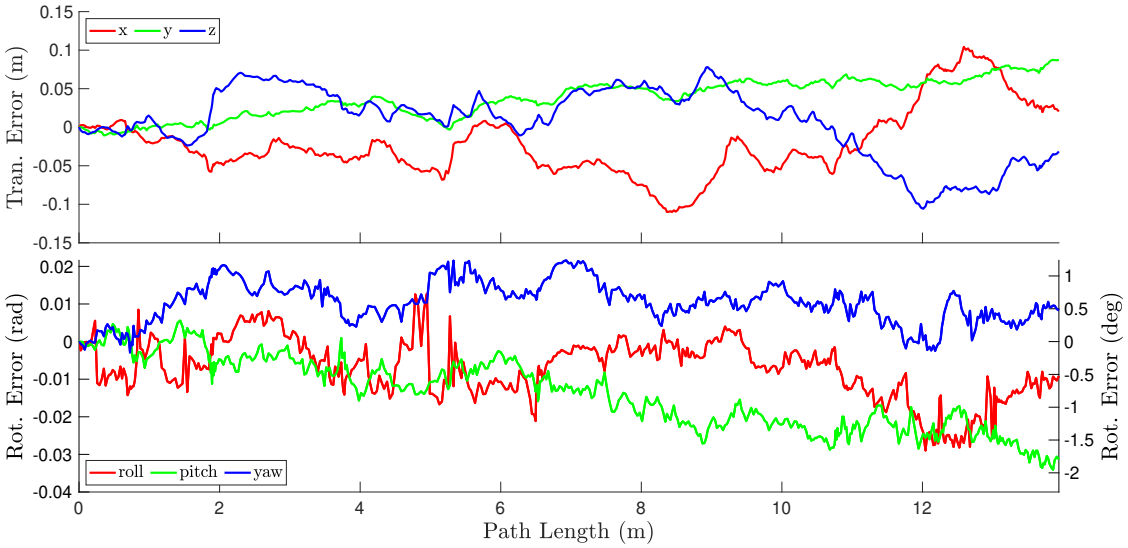
The *Swinging* segments of the OMD directly test the ability of MVO to simultaneously segment and estimate multiple independent motions in a scene (Section 4.4). The results (Figs. 5.4 to 5.6) were produced from a 500-frame image sequence from the `swinging_4_unconstrained` segment, beginning at frame 11.. Estimation was performed as a 12-frame sliding window, with 5 neighbors for each point in the graph, 100 RANSAC iterations per new label, $e_{\text{th}} = 4$, $\alpha = 100$, $\beta = 2$, $\psi_\ell = 100$, $\lambda = 1$, and a minimum model size and length of 10 points and 3 frames, respectively.

As with the *Pinwheel* segments performance, MVO is clearly able to consistently segment the motions of both the camera and the swinging blocks (Fig. 5.4a). The camera egomotion (Fig. 5.4c) exhibited a maximum total drift of 0.15 m, 1.04% of total path length (13.94 m), and a maximum rotational error of $-1.66°$, $-1.95°$,

(a) Tracklet Segmentation



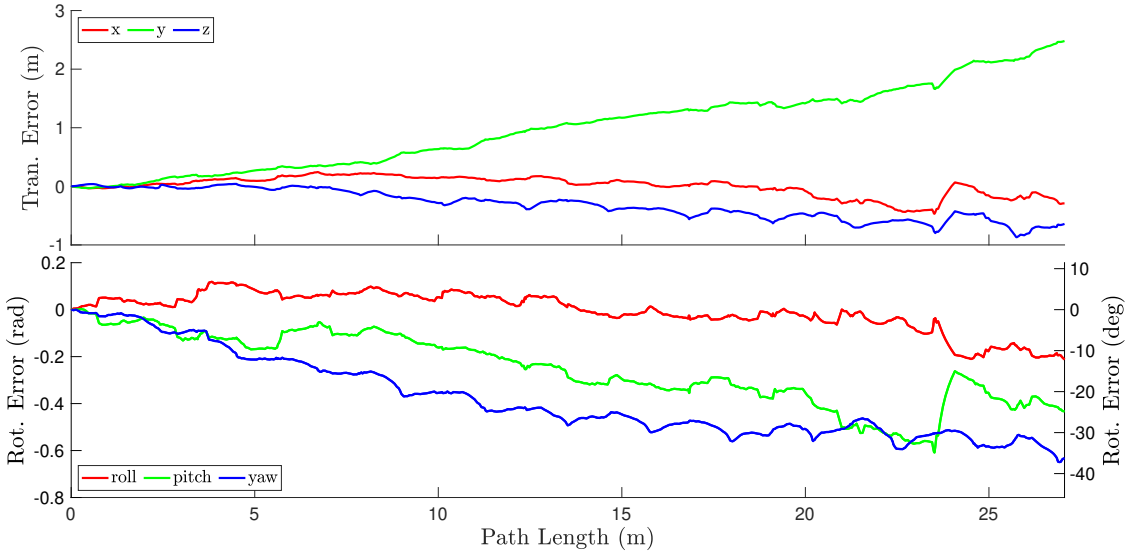
(b) Trajectory Plot



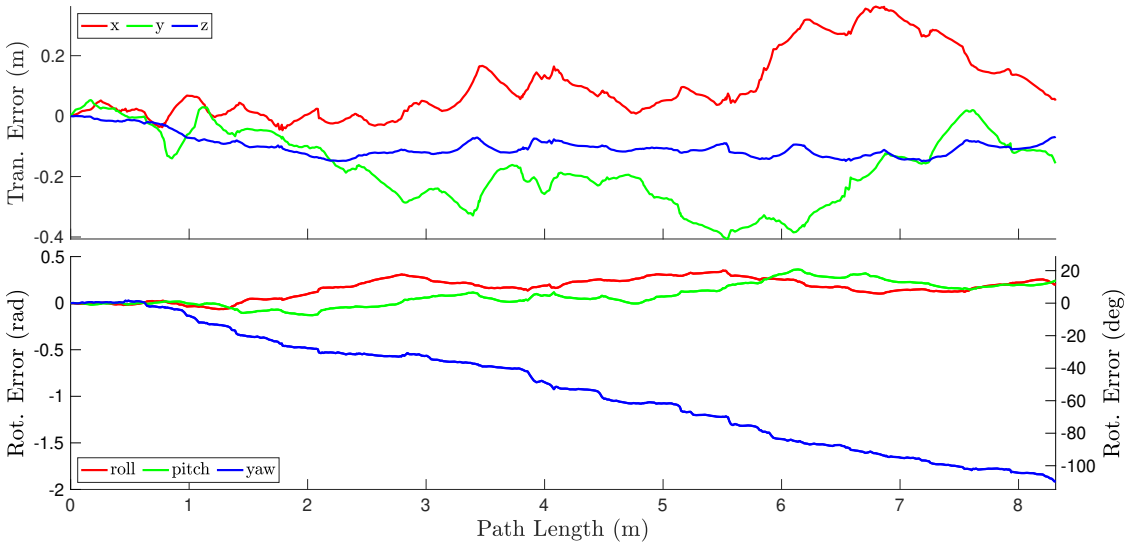
(c) Egomotion

Figure 5.4: Performance of the MVO pipeline over a 500-frame portion of the `swinging_4_unconstrained` segment of the OMD. The segmentation of a single frame (a) and the total trajectory plot (b) show the consistency of the results qualitatively. The quantitative translational and rotational errors for the estimated egomotion of the camera (c) are shown compared to ground-truth trajectory data. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

and 1.24° in roll-pitch-yaw, respectively. The maximum translational and rotational errors for each block are 2.58 m (over a total path of 27.06 m), −12.16°, −34.86°, and −37.19° for Block 1 (top left); 0.49 m (over a total path of 8.31 m), 20.09°, 20.73°, and −109.67° for Block 2 (top right); 1.06 m (over a total path of 19.26 m), 21.94°, −12.33°, and 21.37° for Block 3 (bottom left); and 0.53 m (over a total path of 2.88 m), 8.18°, −4.25°, and 126.73° for Block 4 (bottom right) (Figs. 5.5 and 5.6).



(a) Top-left Box 1

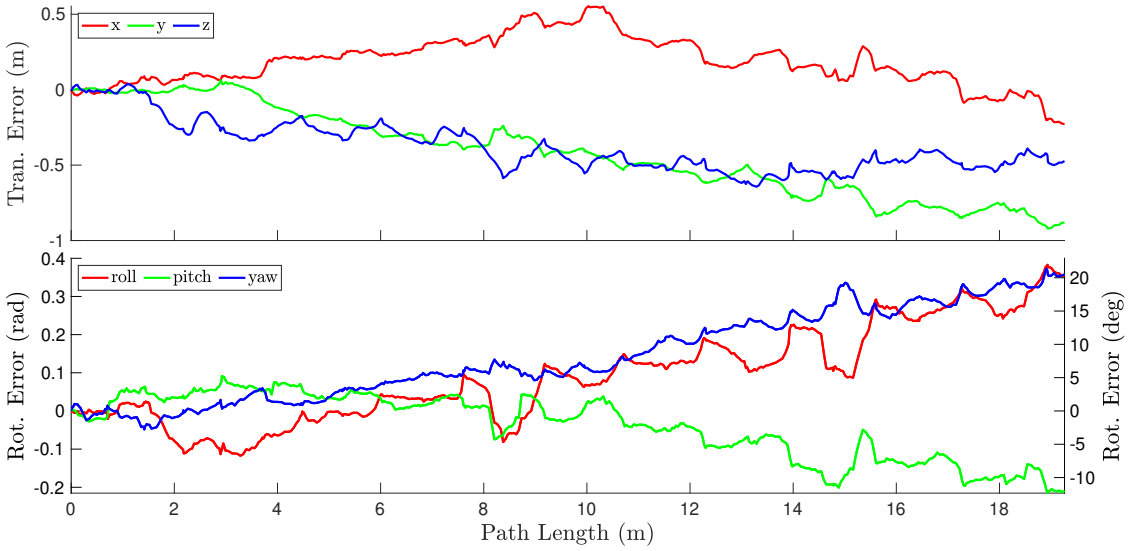


(b) Top-right Box 2

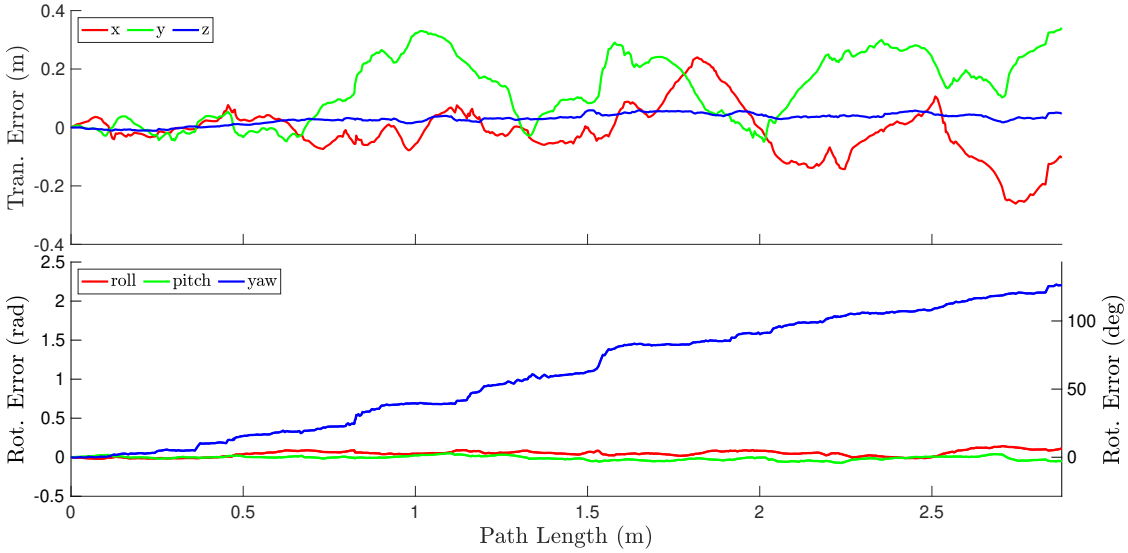
Figure 5.5: Performance of the MVO pipeline over a 500-frame portion of the `swinging_4_unconstrained` segment of the OMD. The quantitative translational and rotational errors for the estimated motion of Box 1 (a) and Box 2 (b) are shown compared to ground-truth trajectory data. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

5.5 Discussion

As shown in Section 5.4, MVO consistently segments and estimates the motions of the camera and third-party objects when they are fully visible. The egomotion errors (Figs. 5.2c and 5.4c) are reasonable compared to the level of drift in other model-free, camera-only VO systems (Geiger et al. 2012), but the accuracy of the third-party



(a) Bottom-left Box 3



(b) Bottom-right Box 4

Figure 5.6: Performance of the MVO pipeline over a 500-frame portion of the `swinging_4_unconstrained` segment of the OMD. The quantitative translational and rotational errors for the estimated motion of Box 3 (a) and Box 4 (b) are shown compared to ground-truth trajectory data. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

estimation deteriorates whenever the checkerboard or one of the blocks partially exits the camera frustum (Figs. 5.3, 5.5 and 5.6). When part of an object leaves the field of view of the camera, it is more difficult for MVO to accurately segment and estimate that object's motion in that frame. Recovering from this failure is similar to recovering from an occlusion and is an important limitation of the MVO pipeline.

When estimating the rotation of the checkerboard, the distribution of features heavily influences the accuracy of the trajectory estimation. Despite accurately segmenting the motion of the checkerboard from that of the background, the imbalance of features biases the estimator toward a largely translational motion, rather than rotational. This is an inherent limitation of using a sparse, feature-based approach that is highlighted in this heavily rotational scene.

The distribution of features also influences the estimate of the center of the motion. It is difficult to infer the structure of an object beyond the surfaces that are observed in the batch without *a priori* knowledge of the object's shape. This means the centroid of the observed feature points for a label can often be a bad estimate of the the label's true center of motion. This is particularly evident in the estimated trajectory for Block 2 and Block 4 (Fig. 5.6b). The blocks are rotating about an axis perpendicular to the optical axis, in contrast to the checkerboard rotating about the optical axis, meaning (at most) half of each block is visible to the camera at any given time. For the same reasons it is difficult to estimate the rotation of the checkerboard with an incomplete feature distribution, it is difficult to estimate the true rotation of the blocks from these partial observations without developing a model for their shape. This results in estimated trajectories that *orbit* an axis of rotation, rather than pivot around it, which contributes to the high errors for these blocks.

Reliable feature distribution is an important factor in the performance of any sparse approach (Farboud-Sheshdeh et al. 2014). This challenge is exacerbated for dynamic objects that take up a small portion of the scene. Additionally, the appearance of dynamic objects is often more volatile making feature association even more difficult. The MVO algorithm is dependent on the accuracy of the input tracklet set and cannot estimate motions for which there are insufficient features. The fact that the MVO pipeline is not robust to full, or even some partial, occlusions severely limits its ability to fully address the MEP.

5.6 Summary

This chapter introduces the MVO pipeline and attempts to address the MEP by simultaneously segmenting and estimating *all* rigid motions in a scene. It does so by applying multimodel-fitting techniques to the traditional VO pipeline and its performance is evaluated on highly dynamic sections of the OMD. MVO is shown to be able to segment and estimate the full $SE(3)$ trajectory of *every* motion in a scene, including the egomotion, without *a priori* appearance information or motion constraints.

The estimation accuracy of MVO is comparable to similarly defined egomotion-only VO systems while also exhibiting similar limitations. Most of these limitations stem from the observability of objects in the scene, either due to poor feature distribution or occlusions. As described in Section 3.1, highly dynamic scenes not only pose difficult motion estimation challenges but also tend to include significant amounts of occlusion. Consistently estimating multiple, continuous motions despite observation dropouts and occlusions is necessary for autonomous navigation in complex dynamic environments. The novel contributions of this chapter are:

- Simultaneous motion segmentation and estimation of *every* motion of the scene using low-level feature points, rather than relying on higher-level appearance information or *a priori* motion constraints (Section 5.1);
- Full $SE(3)$ trajectory estimation of each motion in the scene using only a rigid-body assumption (Section 5.2);
- Deferral of the designation of a given motion as belonging to the camera (i.e., the egomotion) until all hypothetical egomotion trajectories are estimated, after which their geocentric equivalents can be calculated (Section 5.3);
- Evaluation of the MVO pipeline on complex multimotion scenes from the OMD (Section 5.4).

To address the challenges posed by occlusions, it is necessary to extend the estimation tools described in this chapter to allow for under-observed motions. This requires employing prior assumptions to describe how objects move through the environment in the absence of direct observations. Chapter 6 explores the considerations involved in defining such a prior and how it can be used to address these limitations.

6

Extending Multimotion Estimation Through Occlusion

Contents

The ability to safely navigate through a dynamic environment is a crucial task in autonomous robotics that requires simultaneously analyzing the egomotion of the sensor and the third-party motions in the environment. As shown in Chapter 5, multimodel-fitting techniques can be applied to the traditional VO pipeline to extend it to *multimotion* estimation. This MVO pipeline simultaneously segments and estimates the full $SE(3)$ trajectory of every motion in a complex, dynamic scene.

Highly dynamic scenes not only pose difficult motion estimation challenges, but

also tend to include significant amounts of occlusion. These occlusions make it hard to maintain consistent trajectory estimates due to the lack of direct observations. Multimotion estimation algorithms must be able to detect occlusions in order to avoid misassociating observations, as well as predict when an object becomes unoccluded in order to resume tracking it and maintain a consistent trajectory.

The MVO pipeline relies on direct observations, and while it can estimate through some partial occlusions, it is unable to handle significant observation dropouts. While simultaneously *segmenting* and *estimating* multiple motions in a scene is crucial to addressing the MEP, *tracking* those motions through potential occlusions is also important for maintaining accurate trajectory estimates.

For the MVO pipeline to be robust to occlusions, it must be able to estimate and track motions in the absence of direct observations. As explained in Section 3.5.3, objects can be tracked through occlusions by extrapolating their position until they can be observed again. This requires accurately estimating both the pose and velocity of an object. By defining a prior that models the expected motion of an object, these velocities can be used to extrapolate motions and predict how and where they will reappear.

Motion priors are also commonly employed in continuous-time estimation to estimate the egomotion of high-rate or asynchronous sensors, or of multiple unsynchronized sensors (Barfoot et al. 2014). For these sensors, traditional discrete-estimation methods are impractical or inadequate, but the incorporation of a motion prior sufficiently constrains the motion estimate in addition to the various sensor measurements available. Similarly, such a prior can be used to constrain the trajectory estimate of an object that is currently occluded.

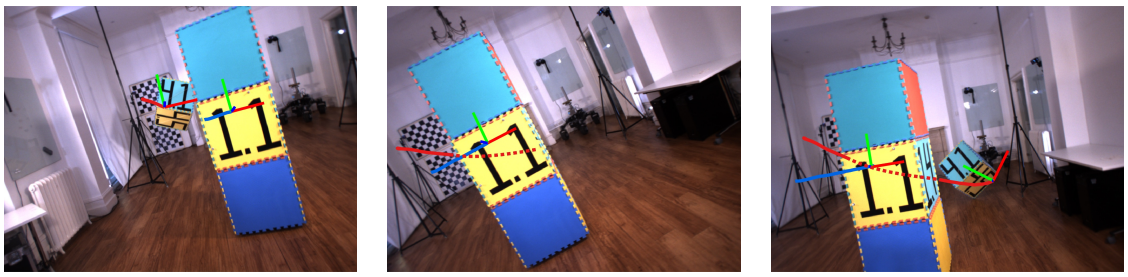
This chapter extends the MVO pipeline to handle occlusions using a continuous $SE(3)$ white-noise-on-acceleration prior. The prior enforces a locally-constant-velocity assumption, and is used both to estimate directly observed trajectories and to extrapolate occluded motions. The occlusion-robust MVO pipeline estimates both the camera egomotion and third-party motions even in the presence of temporary occlusions. Earlier versions of this work originally appeared in Judd and Gammell

(2019b) at the Long-term Human Motion Planning Workshop at ICRA 2019 and was expanded in Judd and Gammell (2020) submitted to ICRA 2020.

Section 6.1 introduces the white-noise-on-acceleration prior. The prior injects a white-noise Gaussian process into the acceleration of a continuous-time $SE(3)$ motion model. This models the velocity as locally constant, which is representative of many real-world motions, and the prior can be used to extrapolate motions in the absence of direct observations, i.e., during occlusions. If tracking is successfully resumed after an occlusion, the occluded trajectory estimates can be interpolated, which further improves the accuracy of the estimation. This section also expands on the concepts introduced in Section 3.3.3 concerning relative motions in noninertial frames. While the constant-velocity assumption may be valid for many motions expressed in an inertial frame, it breaks down if the reference frame, i.e., the camera frame, is rotating.

Section 6.2 explains how the white-noise-on-acceleration prior is incorporated into the occlusion-robust MVO pipeline. The prior can be used to both estimate continuous motion trajectories from direct observations, as well as extrapolate those trajectories in the presence of occlusion. The pipeline proceeds similarly to the original MVO pipeline, decomposing a set of tracklets in a scene into their constituent $SE(3)$ motions. The segmentation initially uses egocentric motion estimates (Section 6.2.1); however, it is necessary to express the motions in a geocentric frame to accurately apply the white-noise-on-acceleration prior. This requires identifying the camera motion and extending the egocentric estimation techniques to geocentric estimation (Section 6.2.2).

Section 6.3 introduces *motion closure*, which maintains trajectory consistency through occlusions using motion-based tracking. If a newly discovered motion is sufficiently similar to the extrapolated estimate of an occluded motion, then it is taken to be the reappearance of that previous motion. The newly discovered motion is used to interpolate the occluded trajectory estimates, which refines the trajectory estimates and improves their accuracy without relying on appearance-based metrics (Fig. 6.1).



(a) (b) (c)

Figure 6.1: A demonstration of the occlusion-robust MVO motion closure procedure showing trajectory estimates produced before (a), during (b), and after (c) an occlusion in the `occlusion_2_unconstrained` segment of the OMD. The trajectory of the swinging block (4, red) is directly estimated when it is visible in (a) and (c) and is extrapolated using the constant-velocity motion prior (dashed line) when the block is occluded by the moving tower (1, blue) in (b). When the block becomes unoccluded in (c), it is rediscovered through *motion closure* and the estimates are interpolated to match the directly estimated trajectory.

Finally, Section 6.4 demonstrates the performance of the occlusion-robust MVO pipeline on scenes from the OMD featuring significant occlusion. The algorithm is capable of simultaneously estimating multiple $SE(3)$ motions and tracking them through temporary occlusions in highly dynamic multimotion scenarios. The quantitative estimation error for the camera egomotion and the third-party motions is discussed along with the pipeline limitations in Section 6.5.

The novel contributions of this chapter are:

- Application of a continuous white-noise-on-acceleration prior to the MVO pipeline to egocentrically segment and estimate a smooth trajectory for every motion in the scene (Sections 6.1 and 6.2);
- Extension of this prior to the continuous-time, geocentric estimation of third-party motions (Section 6.2.2);
- Reacquisition of previously occluded motions in *motion closure* using a motion-based tracking metric (Section 6.3);
- Evaluation of the occlusion-robust MVO pipeline on complex multimotion scenes with significant occlusion from the OMD (Section 6.4).

6.1 White-Noise-on-Acceleration Motion Prior

The occlusion-robust MVO pipeline employs the $SE(3)$ white-noise-on-acceleration (i.e., locally constant-velocity) motion prior described by Barfoot et al. (2014). This prior effectively penalizes the trajectory's deviation from a constant body-centric velocity. It is physically founded because objects tend to move smoothly throughout their environment.

At a high level, the white-noise-on-acceleration prior injects a zero-mean, white-noise Gaussian process into the acceleration of a trajectory model. This models the velocity as being locally constant, but permits deviation from this prior given enough support from the observations.

The continuous-time trajectory of the motion ℓ, $\mathcal{S}_\ell(t) \coloneqq \{\mathbf{T}_\ell(t), \boldsymbol{\varpi}_\ell(t)\}$, is comprised of both the $SE(3)$ poses, $\mathbf{T}_\ell(t)$, and the local, body-centric velocities, $\boldsymbol{\varpi}_\ell(t)$. The trajectory state is assumed to vary smoothly over time via the Lie algebra, $\mathfrak{se}(3)$. This prior takes the form

$$\begin{aligned} \dot{\mathbf{T}}_\ell(t) &= \boldsymbol{\varpi}_\ell(t)^\wedge \mathbf{T}_\ell(t) \\ \dot{\boldsymbol{\varpi}}_\ell(t) &= \mathbf{w}'(t), \\ \mathbf{w}'(t) &\sim \mathcal{GP}\left(\mathbf{0}, \mathbf{Q}_c' \delta(t - t')\right), \end{aligned} \tag{6.1}$$

where $\mathbf{w}'$ is a zero-mean, white-noise Gaussian process with power spectral density matrix, $\mathbf{Q}_c' \in \mathbb{R}^{6\times 6}$, and $\boldsymbol{\varpi}^\wedge$ is the $\mathfrak{se}(3)$ representation of $\boldsymbol{\varpi} \in \mathbb{R}^6$ as defined in (2.4).

This continuous-time trajectory can be estimated at a collection of discrete time steps, $t_1, \ldots, t_K$, such that,

$$\mathcal{S}_{\ell_k} \coloneqq \{\mathbf{T}_{\ell_k \ell_1}, \boldsymbol{\varpi}_{\ell_k}\} \equiv \mathcal{S}_\ell(t_k), \qquad t_1 \leq t_k \leq t_K,$$

where $\mathbf{T}_{\ell_k \ell_1} \coloneqq \mathbf{T}_\ell(t_k)$ and $\boldsymbol{\varpi}_{\ell_k} \coloneqq \boldsymbol{\varpi}_\ell(t_k)$. These time steps correspond to observation times when measurements of the scene are collected.

The system in (6.1) is nonlinear and finding a numerical solution is costly. By assuming the motion between measurement times is small, the system can be recast

as a set of local, linear time-invariant stochastic differential equations of the form,

$$\begin{aligned}\dot{\boldsymbol{\gamma}}_{\ell_k}(t) &= \underbrace{\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}}_{\mathbf{A}}\boldsymbol{\gamma}_{\ell_k}(t) + \mathbf{B}\mathbf{u}(t) + \underbrace{\begin{bmatrix}0\\ 1\end{bmatrix}}_{\mathbf{L}}\mathbf{w}(t),\\ \mathbf{w}(t) &\sim \mathcal{GP}\left(\mathbf{0}, \mathbf{Q}_c\delta(t-t')\right),\end{aligned} \tag{6.2}$$

where $\boldsymbol{\gamma}_{\ell_k}$ is the local GP state, $\mathbf{u}$ is the exogenous input, and $\mathbf{w}$ is the *local* Gaussian white-noise defined similarly to $\mathbf{w}'$ in (6.1). The local state is defined as

$$\boldsymbol{\gamma}_{\ell_k}(t) \coloneqq \begin{bmatrix}\ln\left(\mathbf{T}_\ell(t)\,\mathbf{T}_{\ell_k\ell_1}^{-1}\right)^\vee\\ \boldsymbol{\mathcal{J}}\left(\ln\left(\mathbf{T}_\ell(t)\,\mathbf{T}_{\ell_k\ell_1}{}^{-1}\right)^\vee\right)^{-1}\boldsymbol{\varpi}_\ell(t)\end{bmatrix},$$

where $t_k \leq t \leq t_{k+1}$ and $\boldsymbol{\mathcal{J}}(\cdot)$ is the left Jacobian of $SE(3)$ as defined in (2.5). The *global* noise of the nonlinear system, $\mathbf{w}'$, and the *local* noise of the linear time-invariant systems, $\mathbf{w}$, model the trajectory's adherence to the constant-velocity prior. In the zero-noise case, the velocity is exactly constant and the piecewise-defined system matches the nonlinear system in (6.1).

Assuming there are no net forces externally acting on the object, i.e., $\mathbf{u}(t) = \mathbf{0}$, the solution to the linear, time-invariant stochastic differential equation in (6.2) is

$$\check{\boldsymbol{\gamma}}_{\ell_k}(\tau) = \boldsymbol{\Phi}(\tau, t_k)\,\check{\boldsymbol{\gamma}}_{\ell_k}(t_k), \tag{6.3}$$

where $\check{\boldsymbol{\gamma}}_{\ell_k}$ is the local GP prior mean, and $\boldsymbol{\Phi}(\tau, t_k)$ is the state transition function from t_k to τ,

$$\boldsymbol{\Phi}(\tau, t) \coloneqq \exp\left(\mathbf{A}(\tau - t)\right) = \begin{bmatrix}\mathbf{1} & (\tau - t)\,\mathbf{1}\\ \mathbf{0} & \mathbf{1}\end{bmatrix}.$$

Applying this prior *locally* at each time step represents the *global* nonlinear system as a piecewise sequence of linear, time-invariant systems (Barfoot et al. 2014).

6.1.1 Trajectory Extrapolation and Interpolation

The continuous-time prior can be used to augment the discrete state estimates that coincide with sensor observations. In egomotion estimation, the model is commonly used to interpolate the sensor pose between discrete *keyframes*. In addition to interpolation, poses can be extrapolated when only one side of the interpolation is

constrained by a discrete state. Similarly, extrapolation and interpolation can be used to infer the motion of an occluded object in the absence of direct observations.

Extrapolation involves propagating an estimated state forward or backward in time, whereas interpolation involves estimating intermediate states of that motion between directly observed estimates. Both extrapolation and interpolation require modeling the expected motion of the object given that there are no direct observations available to constrain its motion. Interpolation tends to be more accurate than extrapolation, as it bounds the inferred state estimates with directly observed estimates on both sides, but the accuracy of both extrapolated and interpolated estimates is dependent on the fidelity of the motion priors to the true motions of the objects in the scene.

The white-noise-on-acceleration prior is a physically founded assumption that expects objects to move with locally constant velocity. The prior can therefore be used to extrapolate and interpolate motion estimates in the absence of direct observations due to occlusions.

The local state, $\boldsymbol{\gamma}_k$, at time t_k can be used in (6.3) to estimate the extrapolated state,

$$\check{\boldsymbol{\gamma}}_{\ell_k}(\tau) = \boldsymbol{\Phi}(\tau, t_k)\,\boldsymbol{\gamma}_{\ell_k}(t_k),$$

at time τ (Anderson and Barfoot 2015). The extrapolated state is then transformed to the global state, consisting of the extrapolated transform, $\mathbf{T}_{\check{\ell}}$, and velocity, $\boldsymbol{\varpi}_{\check{\ell}}$, via

$$\begin{aligned}\mathbf{T}_{\check{\ell}}(\tau) &= \exp\left(\begin{bmatrix}\mathbf{1} & \mathbf{0}\end{bmatrix}\check{\boldsymbol{\gamma}}_{\ell_k}(\tau)\right)\mathbf{T}_{\ell}(t_k),\\ \boldsymbol{\varpi}_{\check{\ell}}(\tau) &= \begin{bmatrix}\mathbf{0} & \mathbf{1}\end{bmatrix}\check{\boldsymbol{\gamma}}_{\ell_k}(\tau).\end{aligned}$$

Estimates can be extrapolated forward or backward in time depending on whether $\tau > t_k$ or $\tau < t_k$, respectively. As the length of the extrapolation grows, the estimates will drift from the true motion of the object, especially if it exhibits significant changes in velocity.

Once tracking is resumed after an occlusion, these extrapolated estimates can be improved via interpolation. The occluded trajectory state can be interpolated

between t_j and t_k according to

$$\check{\boldsymbol{\gamma}}_j(\tau) = \boldsymbol{\Lambda}(\tau)\,\boldsymbol{\gamma}_j(t_j) + \boldsymbol{\Omega}(\tau)\,\boldsymbol{\gamma}_j(t_k),$$

where

$$\boldsymbol{\Lambda}(\tau) = \boldsymbol{\Phi}(\tau, t_j) - \boldsymbol{\Omega}(\tau)\,\boldsymbol{\Phi}(t_k, t_j),$$
$$\boldsymbol{\Omega}(\tau) = \mathbf{Q}_j(\tau)\,\boldsymbol{\Phi}(t_k, \tau)^T\,\mathbf{Q}_j(t_k)^{-1},$$

$t_j < \tau < t_k$, and the block covariance matrix, $\mathbf{Q}_j$, is defined using the white-noise power spectral density, $\mathbf{Q}_c$,

$$\mathbf{Q}_j(\tau) \coloneqq \begin{bmatrix} \frac{1}{3}(\tau - t_j)^3\,\mathbf{Q}_c & \frac{1}{2}(\tau - t_j)^2\,\mathbf{Q}_c \\ \frac{1}{2}(\tau - t_j)^2\,\mathbf{Q}_c & (\tau - t_j)\,\mathbf{Q}_c \end{bmatrix}.$$

This interpolated estimation can explain the occluded motion of the object better than extrapolation because it includes direct estimates on both sides of the occlusion.

6.1.2 Relative Velocities and Accelerations

As mentioned in Section 3.3.3, motions can be defined in arbitrary frames, and a simple motion in one frame may become complex when expressed in another. The locally-constant-velocity assumption is valid for many real-world motions moving relative to an inertial reference frame, but this breaks down when the reference frame is rotating. A point, $\underrightarrow{p}^{\,a}$, moving with constant velocity relative to the inertial frame, $\underrightarrow{\mathcal{F}}_I$, moves relative to the noninertial frame, $\underrightarrow{\mathcal{F}}_C$, with velocity,

$$\frac{d}{dt}\left(\mathbf{p}_C^{aC}\right) = \frac{d}{dt}\left(\mathbf{p}_I^{aI}\right) - \boldsymbol{\omega}_{CI} \times \mathbf{p}_I^{aI},$$

where $\boldsymbol{\omega}_{CI}$ is the angular velocity of $\underrightarrow{\mathcal{F}}_C$, relative to $\underrightarrow{\mathcal{F}}_I$. This representation shows that a constant velocity in $\underrightarrow{\mathcal{F}}_I$ may not be constant in $\underrightarrow{\mathcal{F}}_C$. This is further illustrated in the acceleration,

$$\frac{d^2}{dt^2}\left(\mathbf{p}_C^{aC}\right) = \underbrace{\frac{d^2}{dt^2}\left(\mathbf{p}_I^{aI}\right)}_{\text{Inertial}} - \underbrace{2\boldsymbol{\omega}_{CI} \times \frac{d}{dt}\left(\mathbf{p}_C^{aC}\right)}_{\text{Coriolis}} - \underbrace{\boldsymbol{\omega}_{CI} \times \boldsymbol{\omega}_{CI} \times \mathbf{p}_I^{aI}}_{\text{Centrifugal}} - \underbrace{\frac{d}{dt}\left(\boldsymbol{\omega}_{CI}\right) \times \mathbf{p}_C^{aC}}_{\text{Euler}},$$

where the Coriolis, centrifugal, and Euler accelerations are present in the noninertial frame. Even if the inertial and Euler accelerations are zero, i.e., the point has

no inertial acceleration and the reference frame is rotating with constant angular velocity, the Coriolis and centrifugal accelerations will not be zero if the reference frame has nonzero angular velocity. This means the constant-velocity assumption breaks down in the nonintertial frame and it is therefore often necessary to express models in an inertial or quasi-inertial (e.g., geocentric) frame.

6.2 Continuous-Time Motion Segmentation and Estimation

As with the original MVO pipeline (Chapter 5), the occlusion-robust MVO pipeline operates on sparse 3D tracklets and estimates all motion hypotheses egocentrically until the segmentation converges. However, the motion of the camera means the egocentric frame is noninertial, making the constant-velocity prior less valid. In order to estimate the third-party motion trajectories using the white-noise-on-acceleration prior, the egomotion label must first be identified. This label represents the egomotion of the camera, which can be egocentrically estimated because the static background of the scene constitutes an inertial frame (Section 6.2.1). The egomotion is then used to perform a full-batch estimation of every other trajectory in a geocentric frame (Section 6.2.2).

As in Section 5.3, the egomotion label, C, is chosen using prior information or heuristics. It can be initialized as the largest label, as in VO,

$$C_0 = \arg\max_{\ell} |\mathscr{P}_\ell|,$$

after which it can be propagated forward in time by choosing the label that maximizes the overlap in support with the previous egomotion label

$$C_k = \arg\max_{\ell} |\mathscr{P}_\ell \cup \mathscr{P}_{C_{k-1}}|.$$

Motion-based similarity metrics can also be used to identify or validate the choice of C_k and maintain a consistent egomotion trajectory.

6.2.1 Egocentric Motion Estimation

The egomotion of the camera is estimated using the approach described in Anderson and Barfoot (2015). The system state, $\mathbf{x}$, comprises the estimated pose transforms and body-centric velocities, $\{\mathcal{S}_{C_k}\}_{k=1\ldots K}$, and the labeled landmark points, $\left\{\mathbf{p}_{C_1}^{j_1 C_1}\right\}_{j=1\ldots|\mathscr{P}_C|}$. The estimated state, $\mathbf{x}$, is found by minimizing an objective function, $J(\mathbf{x}) = J_y(\mathbf{x}) + J_p(\mathbf{x})$, consisting of the measurement and prior terms. As in Section 5.2, the batch size, K, can either be the entire image sequence, or some fixed-length sliding window.

The measurement term, $J_y(\mathbf{x})$, constrains the estimated state using the observations,

$$J_y(\mathbf{x}) \coloneqq \frac{1}{2}\sum_{jk} \mathbf{e}_{y,jk}(\mathbf{x})^T \mathbf{R}_{jk}^{-1} \mathbf{e}_{y,jk}(\mathbf{x}),$$

where the error is the residual of the measurement model, $\mathbf{g}(\cdot)$, compared to the observations, $\mathbf{y}_{jk}$.

$$\begin{aligned} \mathbf{e}_{y,jk}(\mathbf{x}) &\coloneqq \mathbf{y}_{jk} - \mathbf{g}(\mathbf{x}_{jk}) \\ &= \mathbf{y}_{jk} - \mathbf{s}(\mathbf{z}(\mathbf{x}_{jk})) \\ &= \mathbf{y}_{jk} - \mathbf{s}\left({}^{\ell}\mathbf{T}_{C_k C_1}\mathbf{p}_{C_1}^{j_1 C_1}\right), \end{aligned} \tag{6.4}$$

and $\mathbf{x}_{jk} \coloneqq \left\{\mathcal{S}_{C_k}, \mathbf{p}_{C_1}^{j_1 C_1}\right\}$ is the estimation state. As in Section 5.2, the measurement model applies the perspective camera model, $\mathbf{s}(\cdot)$, to landmark points transformed by the transform model, $\mathbf{z}(\cdot)$, and $\mathbf{R}_{jk}$ is the covariance of the zero-mean additive Gaussian noise in the measurements.

The prior term, $J_p(\mathbf{x})$, constrains the trajectory and velocity estimates by the constant-velocity assumption,

$$J_p(\mathbf{x}) \coloneqq \frac{1}{2}\sum_{k} \mathbf{e}_{p,k}(\mathbf{x})^T \mathbf{Q}_k(t_{k+1})^{-1} \mathbf{e}_{p,k}(\mathbf{x}),$$

where the error penalizes deviation from the constant-velocity prior defined in (6.3),

$$\mathbf{e}_{p,k}(\mathbf{x}) = \boldsymbol{\gamma}_k(t_{k+1}) - \boldsymbol{\Phi}(t_{k+1}, t_k)\,\boldsymbol{\gamma}_k(t_k), \tag{6.5}$$

and the covariance block matrix, $\mathbf{Q}_k(t)$ is defined in (6.1.1).

The total cost, $J(\mathbf{x})$, is minimized by linearizing the error about an operating point, $\mathbf{x}_{\text{op}}$. The operating point is perturbed according to the transform perturbations, $\{\delta\boldsymbol{\xi}_k \in \mathbb{R}^6\}$, velocity perturbations $\{\delta\boldsymbol{\varpi}_k \in \mathbb{R}^6\}$, and landmark perturbations, $\{\delta\boldsymbol{\zeta}_j \in \mathbb{R}^3\}$, which are stacked to form the full state perturbation, $\delta\mathbf{x}$.

Linearizing the cost function requires linearizing (6.4) and (6.5). Using the Jacobians of the measurement error function, $\mathbf{G}_{jk}$, and the prior error function, $\mathbf{E}_k$, the linearized cost is given by

$$J(\mathbf{x}) \approx J(\mathbf{x}_{\text{op}}) - \mathbf{b}^T\delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T\mathbf{A}\delta\mathbf{x}, \tag{6.6}$$

where,

$$\mathbf{b} = \sum_{jk}\mathbf{P}_{jk}^T\mathbf{G}_{jk}^T\mathbf{R}_{jk}^{-1}\mathbf{e}_{y,jk}(\mathbf{x}_{\text{op}}) + \sum_k \mathbf{P}_k^T\mathbf{E}_k^T\mathbf{Q}_k^{-1}\mathbf{e}_{p,k}(\mathbf{x}_{\text{op}}),$$
$$\mathbf{A} = \sum_{jk}\mathbf{P}_{jk}^T\mathbf{G}_{jk}^T\mathbf{R}_{jk}^{-1}\mathbf{G}_{jk}\mathbf{P}_{jk} + \sum_k \mathbf{P}_k^T\mathbf{E}_k^T\mathbf{Q}_k^{-1}\mathbf{E}_k\mathbf{P}_k.$$

The indicator matrices $\mathbf{P}_{jk}$ and $\mathbf{P}_k$ are defined such that $\delta\mathbf{x}_{jk} = \mathbf{P}_{jk}\delta\mathbf{x}$ and $\delta\mathbf{x}_k = \mathbf{P}_k\delta\mathbf{x}$.

The Jacobian of the measurement function is given by

$$\mathbf{G}_{jk} \coloneqq \left.\frac{\partial\mathbf{g}}{\partial\mathbf{x}}\right|_{\mathbf{x}_{\text{op},jk}} = \left.\frac{\partial\mathbf{s}}{\partial\mathbf{z}}\right|_{\mathbf{z}(\mathbf{x}_{\text{op},jk})}\left.\frac{\partial\mathbf{z}}{\partial\mathbf{x}}\right|_{\mathbf{x}_{\text{op},jk}}, \tag{6.7}$$

where

$$\left.\frac{\partial\mathbf{z}}{\partial\mathbf{x}}\right|_{\mathbf{x}_{\text{op},jk}} = \begin{bmatrix}\left(\mathbf{T}_{\text{op},C_kC_1}\mathbf{p}_{\text{op},C_1}^{j_1C_1}\right)^{\odot} & \mathbf{0} & \mathbf{T}_{\text{op},C_kC_1}\mathbf{D}\end{bmatrix},$$

$\mathbf{D} = \begin{bmatrix}\mathbf{1} & \mathbf{0}\end{bmatrix}^T$, and $(\cdot)^{\odot}$ is defined in (5.5).

The Jacobian of the prior error function is

$$\mathbf{E}_k = \begin{bmatrix}\boldsymbol{\mathcal{J}}_{C_{k+1}C_k}^{-1}\boldsymbol{\mathcal{T}}_{C_{k+1}C_k} & \Delta t_k\mathbf{1} & -\boldsymbol{\mathcal{J}}_{C_{k+1}C_k}^{-1} & \mathbf{0} \\ \frac{1}{2}\boldsymbol{\varpi}_{C_{k+1}}^{\curlywedge}\boldsymbol{\mathcal{J}}_{C_{k+1}C_k}^{-1}\boldsymbol{\mathcal{T}}_{C_{k+1}C_k} & \mathbf{1} & -\frac{1}{2}\boldsymbol{\varpi}_{C_{k+1}}^{\curlywedge}\boldsymbol{\mathcal{J}}_{C_{k+1}C_k}^{-1} & -\boldsymbol{\mathcal{J}}_{C_{k+1}C_k}^{-1}\end{bmatrix},$$

where $(\cdot)^{\curlywedge}$ is defined in (2.6), $\boldsymbol{\mathcal{J}}_{C_{k+1}C_k}^{-1} \coloneqq \boldsymbol{\mathcal{J}}\left(\boldsymbol{\xi}_{C_{k+1}C_k}\right)^{-1}$, and $\boldsymbol{\mathcal{T}}_{C_{k+1}C_k} \in \mathbb{R}^{6\times6}$, the adjoint of $\mathbf{T}_{C_{k+1}C_k}$, is defined in (2.7)

The optimal perturbation, $\delta\mathbf{x}^*$, to minimize the linearized cost, $J(\mathbf{x})$, is the solution to $\mathbf{A}\delta\mathbf{x}^* = \mathbf{b}$. Each element of the operating point is then updated using

$$\mathbf{T}_{\text{op},C_kC_1} \leftarrow \exp(\delta\boldsymbol{\xi}_k^{\wedge})\mathbf{T}_{\text{op},C_kC_1},$$
$$\boldsymbol{\varpi}_{C_k} \leftarrow \boldsymbol{\varpi}_{C_k} + \delta\boldsymbol{\varpi}_k^*,$$
$$\mathbf{p}_{\text{op},C_1}^{j_1C_1} \leftarrow \mathbf{p}_{\text{op},C_1}^{j_1C_1} + \mathbf{D}\delta\boldsymbol{\zeta}_j^*,$$

and the cost is relinearized about the updated operating point. The process iterates until the state convergences and $\mathbf{x} \leftarrow \mathbf{x}_{\text{op}}$.

6.2.2 Geocentric Motion Estimation

The third-party geocentric motions in the scene, $\{\mathcal{S}_\ell\}_{\ell \in \mathscr{L} \setminus C}$, are calculated using the estimated egomotion, $\mathcal{S}_C$. As in Section 6.2.1, each label's state, $\mathbf{x}$, comprises its estimated poses and velocities, $\{\mathcal{S}_{\ell_k}\}_{k=1\ldots K}$, and its associated landmark points, $\left\{\mathbf{p}_{C_1}^{j_1 C_1}\right\}_{j=1\ldots|\mathscr{P}_\ell|}$.

The transform model, $\mathbf{z}$, used by Anderson and Barfoot (2015) must be adjusted for third-party motions to transform egocentrically observed points through a geocentrically estimated state,

$$\mathbf{z}'\left(\mathbf{x}_{jk}\right) \coloneqq \mathbf{T}_{C_k C_1} \mathbf{T}_{\ell_1 C_1}^{-1} \mathbf{T}_{\ell_k \ell_1}^{-1} \mathbf{F}_{\ell_k \ell_1} \mathbf{T}_{\ell_1 C_1} \mathbf{p}_{C_1}^{j_1 C_1},$$

where $\mathbf{F}_{\ell_k \ell_1}$ is the object deformation matrix (identity for rigid bodies), and $\mathbf{T}_{C_k C_1}$ is the camera egomotion as estimated in Section 6.2.1. The transform from the camera to the object centroid when it is first observed is given by

$$\mathbf{T}_{\ell_1 C_1} = \begin{bmatrix} \mathbf{C}_{\ell_1 C_1} & \mathbf{p}_{\ell_1}^{C_1 \ell_1} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{C}_{\ell_1 C_1} & -\mathbf{C}_{\ell_1 C_1} \mathbf{p}_{C_1}^{\ell_1 C_1} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{6.8}$$

The rotation, $\mathbf{C}_{\ell_1 C_1}$, is arbitrary and initially assumed to be identity for newly discovered motions. The translation, $\mathbf{p}_{C_1}^{\ell_1 C_1}$, is calculated as the centroid of the labeled points, $p^j \in \mathscr{P}_\ell$, observed in the first frame. In a sliding-window pipeline, this transform can be determined from the previously estimated trajectory estimates.

The motion model part of the measurement Jacobian is now given by the block-row vector,

$$\left.\frac{\partial \mathbf{z}'}{\partial \mathbf{x}}\right|_{\mathbf{x}_{\text{op},jk}} = \Big[-\mathbf{T}_{C_k C_1} \mathbf{T}_{\ell_1 C_1}^{-1} \mathbf{T}_{\text{op},\ell_k \ell_1}^{-1} \left(\mathbf{F}_{\ell_k \ell_1} \mathbf{T}_{\ell_1 C_1} \mathbf{p}_{\text{op},C_1}^{j_1 C_1}\right)^{\odot} \quad \mathbf{0} \quad \mathbf{T}_{C_k C_1} \mathbf{T}_{\ell_1 C_1}^{-1} \mathbf{T}_{\text{op},\ell_k \ell_1}^{-1} \mathbf{F}_{\ell_k \ell_1} \mathbf{T}_{\ell_1 C_1} \mathbf{D} \Big]$$

This Jacobian is used to estimate $\mathbf{G}_{jk}$ in (6.7), and (6.6) is used to estimate the continuous-time geocentric trajectory, $\mathcal{S}_\ell$, of every third-party motion in the scene.

6.3 Motion-Based Tracking Through Occlusion

Most tracking techniques overcome occlusions by extrapolating the estimated location of an occluded object using simple motion models (Section 3.5.3). These models often poorly approximate the true motion of the object, but they can be used alongside appearance-based object detectors to identify when the object becomes unoccluded. This reliance on appearance-based metrics limits their ability to track generic objects and deal with volatile lighting conditions. In contrast, *motion closure* uses motion-based tracking metrics to maintain trajectory consistency for arbitrary objects in the presence of occlusion.

Once the trajectories are estimated in a geocentric frame, the motion prior can be used to extrapolate previously estimated trajectories that are not found in the current frame due to occlusion or estimation failure. These extrapolated trajectories are then used to determine if any new trajectory can be explained by the reappearance of a previously observed motion. Trajectories found to belong to the same motion are used to correct occluded estimates through interpolation.

A newly discovered trajectory, $\mathcal{S}_{\ell'_k} \coloneqq \left\{\mathbf{T}_{\ell'_k}, \boldsymbol{\varpi}_{\ell'_k}\right\}$, is compared to an occluded motion's extrapolated trajectory, $\mathcal{S}_{\check{\ell}_k} \coloneqq \left\{\mathbf{T}_{\check{\ell}_k}, \boldsymbol{\varpi}_{\check{\ell}_k}\right\}$, at time t_k using a motion-based threshold incorporating both position and velocity. The position of the extrapolated motion, $\mathbf{p}_{C_k}^{\check{\ell}_k C_k}$, can be found from the extrapolated transform, $\mathbf{T}_{\check{\ell}_k}$, via (2.1) and (2.2). The position of the newly discovered motion, $\mathbf{p}_{C_k}^{\ell'_k C_k}$, is calculated from the centroid of labeled points in the first frame the motion was observed as in (5.3.2). The positions are compared along with the velocities,

$$\left\|\mathbf{p}_{C_k}^{\check{\ell}_k C_k} - \mathbf{p}_{C_k}^{\ell'_k C_k}\right\| < \epsilon_{\text{pos}}$$

and

$$\left\|\boldsymbol{\varpi}_{\check{\ell}_k} - \boldsymbol{\mathcal{T}}_{\check{\ell}_k \ell'_k} \boldsymbol{\varpi}_{\ell'_k}\right\| < \epsilon_{\text{vel}},$$

where the rotational adjoint, $\boldsymbol{\mathcal{T}}_{\check{\ell}_k \ell'_k}$, is used to compare the velocities in the same frame and is defined as

$$\boldsymbol{\mathcal{T}}_{\check{\ell}_k \ell'_k} = \exp\left(\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{\phi}_{\check{\ell}_k \ell'_k} \end{bmatrix}^\curlywedge\right) = \begin{bmatrix} \mathbf{C}_{\check{\ell}_k \ell'_k} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{\check{\ell}_k \ell'_k} \end{bmatrix}.$$

If the positions and velocities are sufficiently similar according to the thresholds, ϵ_{pos} and ϵ_{vel}, the motions are closed. While these thresholds are presented as independent, they could alternatively be linearly combined as a single threshold with a proportionality parameter,

$$\left\| \mathbf{p}_{C_k}^{\check{\ell}_k C_k} - \mathbf{p}_{C_k}^{\ell'_k C_k} \right\| + \lambda_\epsilon \left\| \boldsymbol{\varpi}_{\check{\ell}_k} - \boldsymbol{\mathcal{T}}_{\check{\ell}_k \ell'_k} \boldsymbol{\varpi}_{\ell'_k} \right\| < \epsilon_{\text{combined}}.$$

Upon successful motion closure, the trajectory is reestimated using the extrapolated estimate of the transform from the camera to the object centroid, $\mathbf{T}_{\check{\ell}_1 C_1}$ (Section 6.2.2). The corrected trajectory, $\mathcal{S}_{\ell_k} \coloneqq \{\mathbf{T}_{\ell_k}, \boldsymbol{\varpi}_{\ell_k}\}$, is then estimated from the extrapolated trajectory, $\mathcal{S}_{\check{\ell}_k}$, and a correction transform, $\mathbf{T}_{\ell_k \check{\ell}_k}$,

$$\begin{aligned} \mathbf{T}_{\ell_k} &\coloneqq \mathbf{T}_{\ell_k \ell_1} = \mathbf{T}_{\ell_k \check{\ell}_k} \mathbf{T}_{\check{\ell}_k \check{\ell}_1} \mathbf{T}_{\check{\ell}_1 \ell_1}, \\ \boldsymbol{\varpi}_{\ell_k} &= \boldsymbol{\varpi}_{\ell'_k}, \end{aligned}$$

where $\mathbf{T}_{\check{\ell}_1 \ell_1}$ is identity because the corrected and extrapolated trajectories are equivalent before the occlusion.

The correction transform uses the observed centroid of the rediscovered trajectory, $\mathbf{p}_{C_k}^{\ell'_k C_k}$, to adjust the extrapolated trajectory position, $\mathbf{p}_{C_k}^{\check{\ell}_k C_k}$,

$$\mathbf{T}_{\ell_k \check{\ell}_k} = \begin{bmatrix} \mathbf{C}_{\ell_k \check{\ell}_k} & \mathbf{p}_{\ell_k}^{\check{\ell}_k \ell_k} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

where

$$\begin{aligned} \mathbf{p}_{\ell_k}^{\check{\ell}_k \ell_k} &= \mathbf{C}_{\ell_k C_k} \mathbf{p}_{C_k}^{\check{\ell}_k \ell_k} = \mathbf{C}_{\ell_k C_k} \mathbf{p}_{C_k}^{\check{\ell}_k \ell'_k} \\ &= \mathbf{C}_{\ell_k \check{\ell}_k} \mathbf{C}_{\check{\ell}_k C_k} \left(\mathbf{p}_{C_k}^{\check{\ell}_k C_k} - \mathbf{p}_{C_k}^{\ell'_k C_k} \right), \end{aligned}$$

and the extrapolated camera-object rotation, $\mathbf{C}_{\check{\ell}_k C_k}$, comes from

$$\mathbf{T}_{C_k \check{\ell}_k} = \mathbf{T}_{C_k C_1} \mathbf{T}_{\check{\ell}_1 C_1}^{-1} \mathbf{T}_{\check{\ell}_k \check{\ell}_1}^{-1} = \begin{bmatrix} \mathbf{C}_{\check{\ell}_k C_k}^T & \mathbf{p}_{C_k}^{\check{\ell}_k C_k} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{6.9}$$

It is difficult to determine the true rotation of the object after the occlusion without using appearance-based metrics, so the extrapolated trajectory rotation is taken directly, i.e., $\mathbf{C}_{\ell_k \check{\ell}_k}$ is identity.

6.4 Evaluation

The accuracy of the MVO algorithm was evaluated on real-world data from the OMD, collected using a Bumblebee XB3 stereo camera and a Vicon motion capture system. As with the original pipeline in Section 5.4, feature detection and matching were performed using LIBVISO2 (Geiger et al. 2011) and the Gauss-Newton minimization was performed with Ceres (Agarwal et al.) using analytical derivatives (Section 6.2). The transforms between the Vicon frames and the MVO estimated frames are arbitrary, so the first 10% of each trajectory is used to calibrate this transform (Zhang and Scaramuzza 2018). All errors are reported for geocentric trajectory estimates, so a portion of the geocentric error of each motion is due to the error in the camera motion estimate.

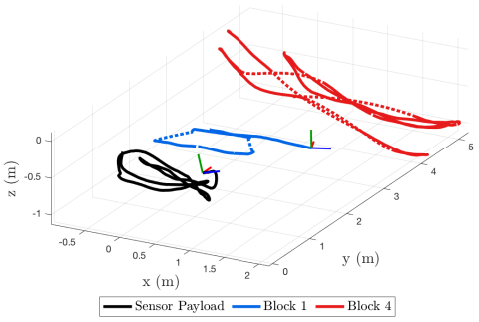
This section extends the evaluation in Section 5.4 using the *Occlusion* and *Toy Cars* segments of the OMD, which each represent challenging examples of the MEP that exhibit significant occlusion. Estimation for both segments was performed as a 12-frame sliding window, with 5 neighbors for each point in the graph, 100 RANSAC iterations per new label, $e_{\text{th}} = 4$, $\alpha = 100$, $\beta = 5$, $\psi_\ell = 100$, $\lambda = 1$, $\epsilon_{\text{pos}} = 1$, $\epsilon_{\text{vel}} = 1$, and a minimum model size and length of 20 points and 3 frames, respectively. The parameters used in LIBVISO2 are given in Table A.1.

6.4.1 Tracking Through Occlusion

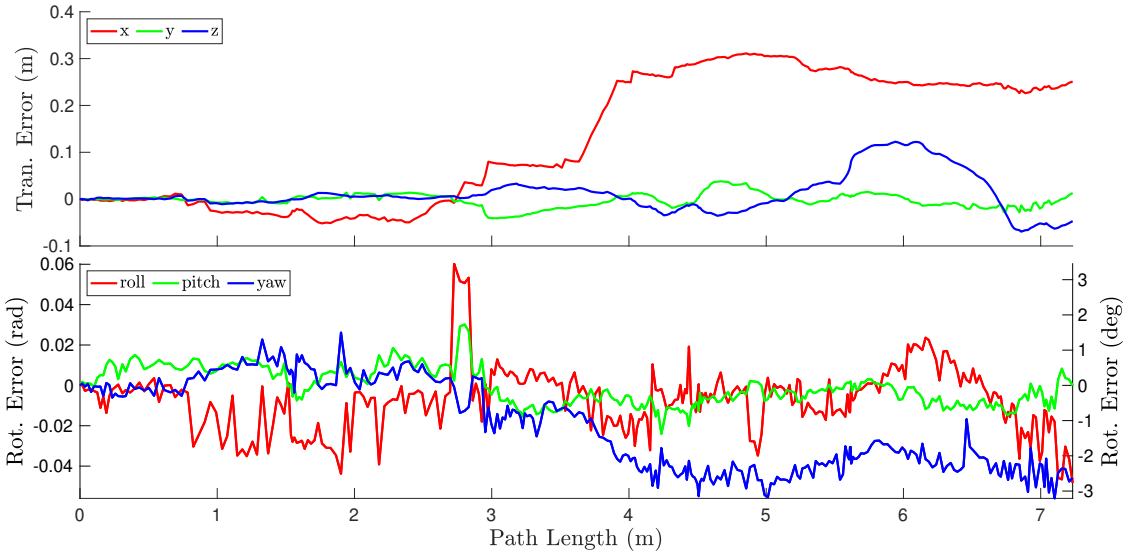
The *Occlusion* segments include three independent motions, a sliding block tower that occasionally becomes static, a swinging block that is repeatedly occluded by the tower, and the camera egomotion (Section 4.5). The sliding block tower is frequently partially occluded when it partially leaves the field of view of the camera, further complicating the estimation. It also occasionally stops moving and becomes part of the static background, which in motion tracking is effectively the same as becoming occluded. The frequent occlusion and the complex $SE\,(3)$ motions of the blocks and camera pose a challenging estimation and tracking problem.

Figs. 6.2 and 6.3 illustrate the performance of the occlusion-robust MVO pipeline on a 300-frame image sequence from the `occlusion_2_unconstrained` segment.

(a) Tracklet Segmentation



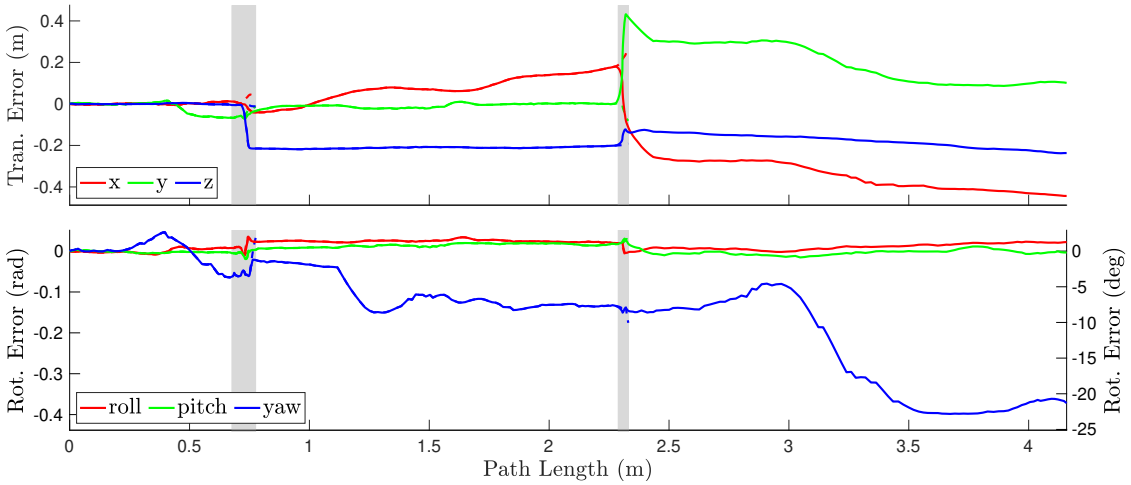
(b) Trajectory Plot



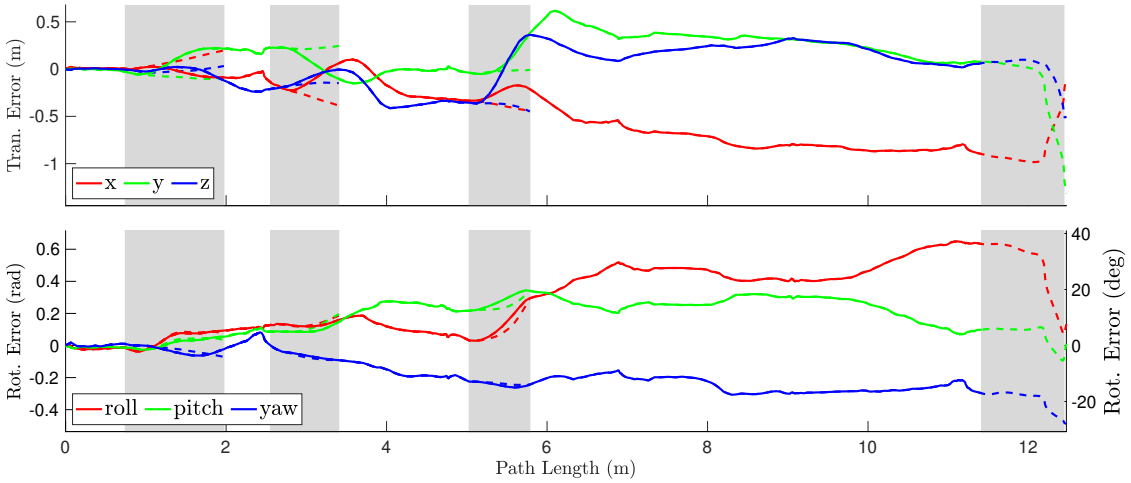
(c) Egomotion

Figure 6.2: The segmentation (a) and trajectory estimates (b) of the occlusion-robust MVO pipeline for the `occlusion_2_unconstrained` segment of the OMD, as well as the translational and rotational errors for the estimated motion of the camera (c). The scene consists of the static background (black), the sliding block tower (red, 1), and the swinging block (blue, 4). The estimation results are compared to ground-truth trajectory data over a 300-frame section of the segment, and errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

The grey regions represent times when the swinging block was occluded by the moving tower (Fig. 6.3b), or when the tower was stationary and effectively part of the static background (Fig. 6.3a). In these regions, the dashed lines represent the error in extrapolation and the solid lines represent that of the interpolated estimates. Elsewhere, the solid lines represent the errors in the directly estimated trajectories. Near the end of the segment, the swinging block changed direction while occluded by



(a) Block 1



(b) Block 4

Figure 6.3: The translational and rotational errors for the estimated motion of the moving block tower (a) and the swinging block (b) for the `occlusion_2_unconstrained` segment of the OMD. Grey regions represent times when the swinging block was occluded by the tower, or when the tower was stationary and effectively part of the static background. Dashed lines represent the error in extrapolation and the solid lines represent the error in the direct or interpolated estimates. Each object is compared to ground-truth trajectory data over a 300-frame section of the segment. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

the block tower, which caused the motion closure stage to fail because the constant-velocity prior no longer described its motion. This resulted in the trajectory error for Block 4 (Fig. 6.3b) changing drastically at the end of the segment.

Occlusion-robust MVO is clearly able to consistently estimate the motions of both the camera and the moving blocks through motion closure (Figs. 6.2a and 6.2b); however, the newly calculated centroid of a motion after an occlusion does not

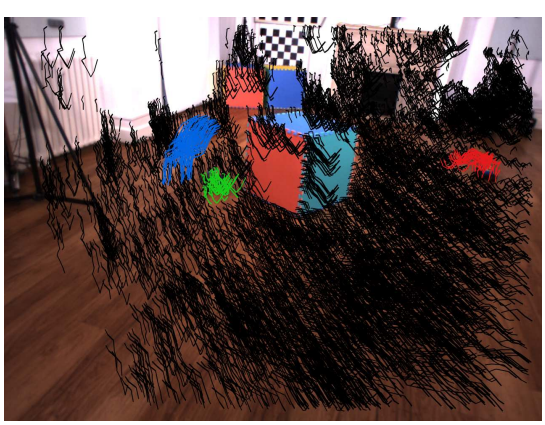
always match that of the original motion. This discrepancy causes jumps in the trajectory, as the original centroid is projected forward to a different location than the location calculated in the current frame. This greatly affects the calculation of the correction transform in (6.9) and is a major source of error in the estimates of the block tower (Fig. 6.3a), as it is often partially outside the view of the camera, which changes its observable centroid.

The camera egomotion (Fig. 6.2c) exhibited a maximum total drift of 0.31 m (over a 7.23 m path) and a maximum rotational error of 3.45°, 1.73°, and −3.21° in roll-pitch-yaw, respectively. The interpolated error of the block tower and the swinging block was occasionally worse than the extrapolated error due to the shifting centroid estimates used in *motion closure*, but both motions were consistently tracked and estimated. The block tower (Fig. 6.3a) exhibited a maximum total drift of 0.51 m (over a 4.16 m path). It exhibited a maximum rotational error of 1.94°, 1.75°, and −22.83° in roll-pitch-yaw, respectively. The swinging block (Fig. 6.3b) exhibited a maximum total drift of 1.41 m, (over a 12.47 m path). It exhibited a maximum rotational error of 37.22°, 19.72°, and −27.83° in roll-pitch-yaw, respectively.

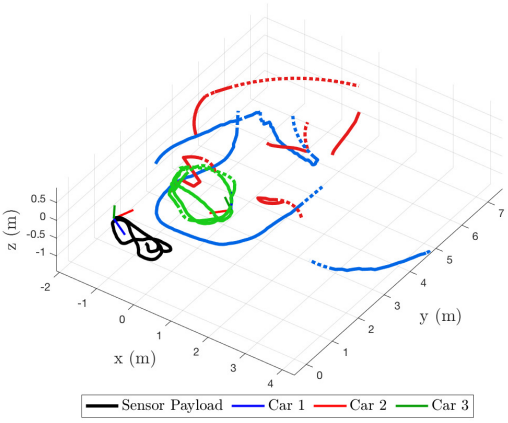
6.4.2 The Full Motion Estimation Problem

The occlusion-robust MVO pipeline overcomes the main limitation of the original MVO and is able to address the full MEP in dynamic scenes with significant occlusion. The *Toy Cars* segments of the OMD represent this kind of challenging dynamic scene (Section 4.6). The segments include several independent toy cars driving around a static source of occlusion. While the object motions are limited to $SE(2)$, the number and variety of motions present, as well as the frequent occlusions, still pose a difficult estimation problem.

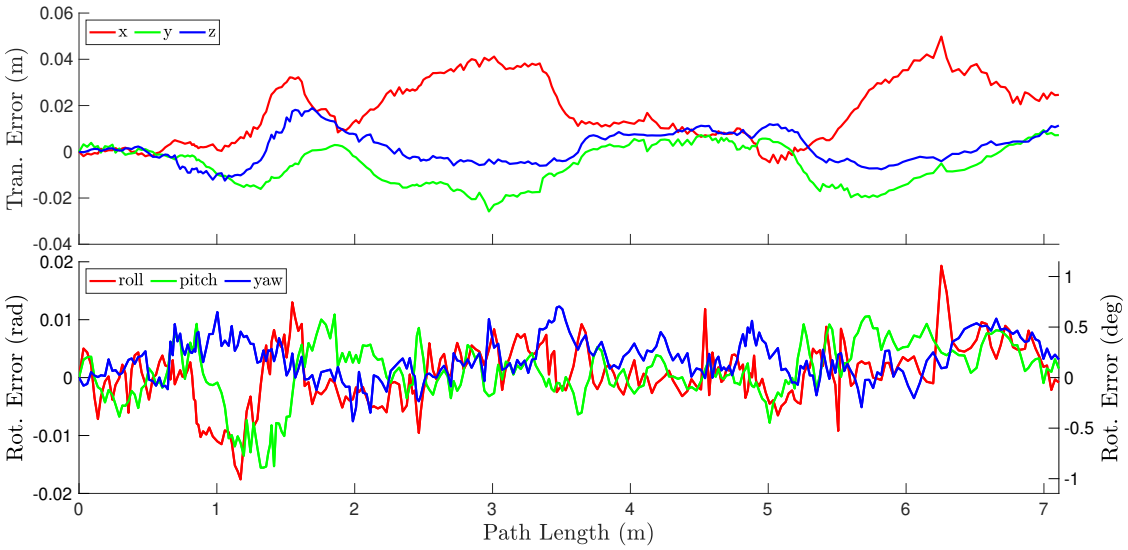
The cars generally move with relatively constant velocities, but they are controlled by human operators, so they may also change direction arbitrarily. This poses a significant challenge for the occlusion-robust MVO pipeline because the white-noise-on-acceleration prior is not always accurate. Figs. 6.4 to 6.6 illustrate the performance of the occlusion-robust MVO pipeline on a 275-frame image sequence



(a) Tracklet Segmentation



(b) Trajectory Plot



(c) Egomotion

Figure 6.4: The segmentation (a) and trajectory estimates (b) of the occlusion-robust MVO pipeline for the `cars_3_unconstrained` segment of the OMD, as well as the translational and rotational errors for the estimated egomotion of the camera (c). The scene consists of the static background (black) and three independently controlled toy cars, Car 1 (blue), Car 2 (red), and Car 3 (green). The trajectories for Cars 1 and 2 are disjointed because they were not successfully tracked throughout the segment due to their small size, repeated occlusion, and varying motions. The estimates are compared to ground-truth trajectory data over a 275-frame section of the segment. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

from the `cars_3_unconstrained` segment. The portion of the segment was chosen in order demonstrate the abilities of the occlusion-robust MVO and intentionally does not include any collisions, which the pipeline was not designed to address.

As in Section 6.4.1, the grey regions represent times when a car was occluded or

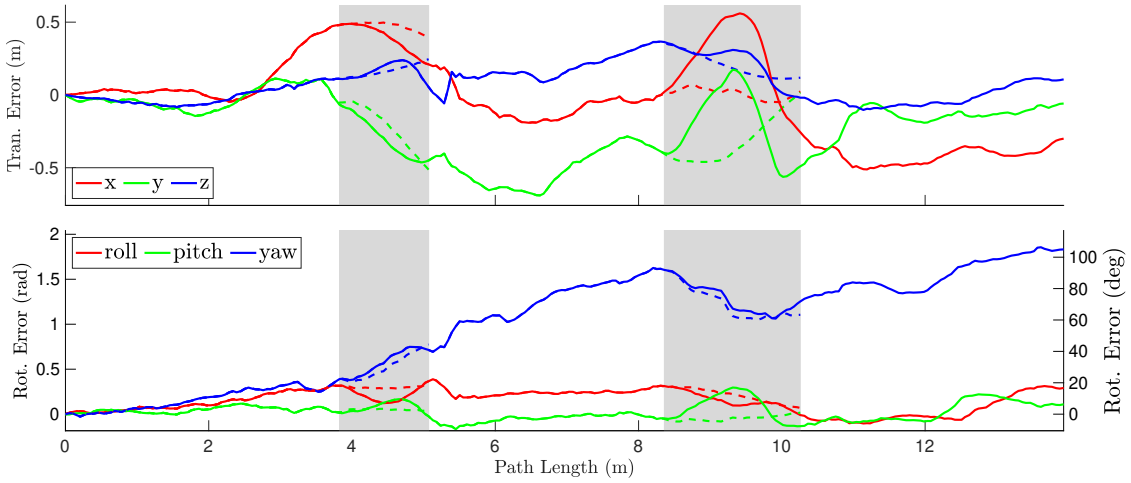
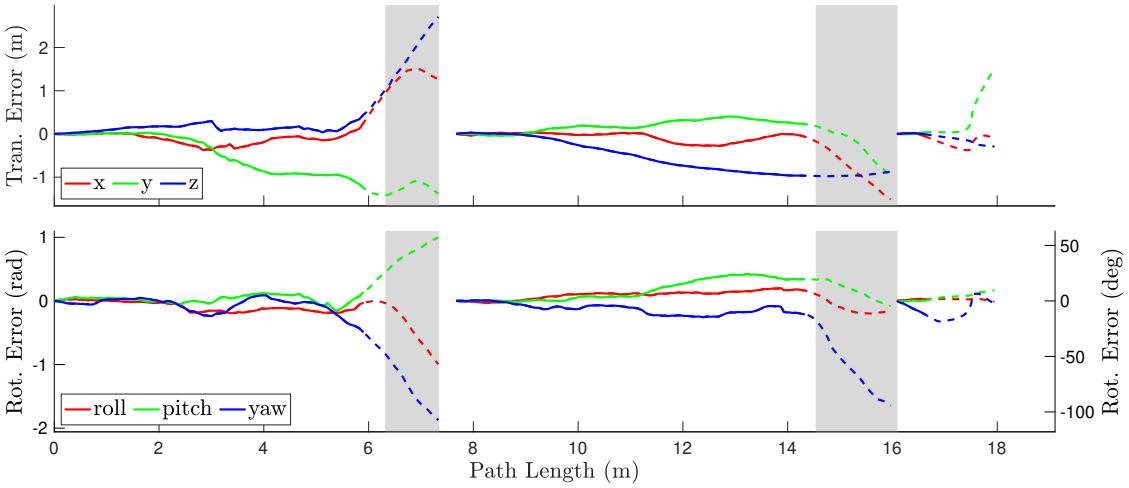


Figure 6.5: The translational and rotational errors for the estimated motion of Car 3 for the `cars_3_unconstrained` segment of the OMD. Grey regions represent times when a car was stationary, outside the camera frustum, or occluded by the static block. Dashed lines represent the error in extrapolation and the solid lines represent the error in the direct or interpolated estimates. The estimates are compared to ground-truth trajectory data over a 275-frame section of the segment. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

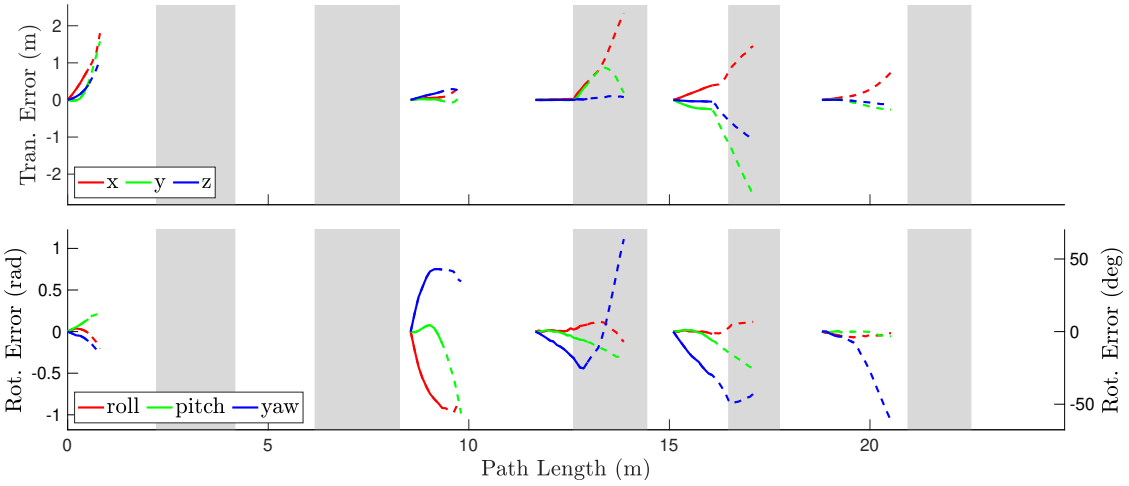
outside the camera frustum. In these regions, the dashed lines represent the error in extrapolation and the solid lines represent that of the interpolated estimates. Elsewhere, the solid lines represent the errors in the directly estimated trajectories.

The camera egomotion (Fig. 6.4c) exhibited a maximum total drift of 0.05 m 0.01% of total path length (7.11 m), and a maximum rotational error of 1.11°, −0.89°, and 0.70° in roll-pitch-yaw, respectively. The cars (Fig. 6.5) exhibited a maximum total drift of 0.73 m (over a 13.93 m path), 0.66 m (over a 3.89 m path), 0.70 m (over a 10.45 m path), for Car 1, 2, and 3, respectively. It exhibited a maximum rotational error of 22.25°, 16.96°, and 106.34° roll-pitch-yaw, respectively. The yaw error amounts to 11.22% of the total rotation.

Car 3 was consistently estimated and tracked through multiple occlusions, but Car 1 and Car 2 were not consistently tracked (Fig. 6.6). Motion closure for Car 1 failed when it was occluded because the constant-velocity prior did not match its motion well. Likewise, Car 2 was repeatedly occluded and changed direction often. Furthermore, both Car 1 and Car 2 moved around the far side of the room, meaning it was particularly difficult to segment them from the static background,



(a) Car 1



(b) Car 2

Figure 6.6: The translational and rotational errors for the estimated motion of Car 1 (a) and Car 2 (b) for the `cars_3_unconstrained` segment of the OMD. Grey regions represent times when a car was stationary, outside the camera frustum, or occluded by the static block. Dashed lines represent the error in extrapolation and the solid lines represent the error in the direct or interpolated estimates. The estimates are compared to ground-truth trajectory data over a 275-frame section of the segment. Errors are reported in a geocentric frame with the x-axis right, y-axis forward, and z-axis up.

and depth triangulation errors lead to poor estimation accuracy.

6.5 Discussion

The occlusion-robust MVO pipeline consistently segmented the motions of the camera and the blocks or cars while also estimating the trajectories through occlusions; however, it is still a sparse, feature-based technique, meaning estimating

the motion of small or distant objects can be challenging. Occlusion-robust MVO also maintains its reliance on motion, meaning objects that temporarily have the same motion will be given the same label, such as when the sliding block tower becomes stationary (Fig. 6.3a). A form of *motion permanence* is achieved by extrapolating the motion forward in time, despite the fact that it is largely redundant with the background motion, because tracking can be resumed when the tower begins moving again. Unfortunately, the observable shape (and centroid) of an object changes as it moves, affecting the geocentric estimate of its trajectory, which can be seen in the interpolated motions of the block tower.

Similar to Section 5.5, the distribution of features is critical to the accuracy of the estimation. The calculation of the object centroid in (6.9) is dependent on this distribution, and the centroid calculated before an occlusion often does not match the centroid calculated after it. This is caused by partial occlusions, either due to the object partially leaving the camera frustum, as often occurred with the sliding block tower, or due to other objects, as often occurred when the swinging block was becoming occluded or unoccluded. As the object becomes more occluded, the quality of the trajectory estimation will degrade. It is unlikely that an object will become occluded or unoccluded instantaneously, but this could be mitigated by predicting occlusions (Mitzel et al. 2010). The original MVO pipeline partially mitigated this through a rolling-average calculation of the centroid, but this is difficult in geocentric estimation as the centroid is required for the calculation of third-party trajectories. This could be further mitigated using part-to-whole extrapolation techniques (Section 3.5.3), but this would also require the generation of accurate object models.

The results show that it is particularly difficult to estimate the yaw rotation for the cars and the block tower. This is largely due to the fact that yaw represents an out-of-plane rotation that is hard to observe for small objects like the cars or tall, proportionally thin objects like the block tower. This is made even more difficult by the challenge of estimating depth from stereo cameras. The translational errors

were much more accurate, even through extrapolation, which proved crucial for maintaining consistent trajectories through motion closure.

The accuracy of the extrapolated estimates is dependent on the fidelity of the motion priors to the true motions of the objects in the scene. As the length of the extrapolation grows, the estimates will diverge from the true motion of the occluded object, especially if it exhibits significant changes in velocity. The applicability of the white-noise-on-acceleration motion prior is limited in scenes where objects change direction or speed. The *Toy Cars* data segments test this prior because the cars are each independently controlled and often change direction and speed according to the agency of the operator. So long as the car maintains a near-constant velocity while it is occluded, tracking and estimation can be resumed through motion closure when it becomes unoccluded; however, motion closure understandably fails if the car significantly changes its velocity. Future work could focus on introducing a white-noise-on-jerk prior (Tang et al. 2019), which is more applicable to motions with smoothly varying velocities, but it is ultimately very difficult to accurately infer complex motion without direct observations.

6.6 Summary

This chapter extends the MVO pipeline to address the challenges posed by occlusion in highly dynamic environments and demonstrates its performance on challenging segments from the OMD with significant occlusion and highly dynamic $SE(3)$ motions. The occlusion-robust MVO pipeline extend a continuous $SE(3)$ white-noise-on-acceleration prior to both to geocentrically estimate directly observed trajectories and to extrapolate occluded motions. Temporarily occluded motions can be tracked and reacquired through *motion closure* using the continuous prior. This maintains trajectory consistency and allows the occlusion-robust MVO pipeline to estimate and track multiple $SE(3)$ motions, even in the presence of occlusion. The novel contributions of this chapter are:

- Application of a continuous white-noise-on-acceleration prior to the MVO pipeline to egocentrically segment and estimate a smooth trajectory for every motion in the scene (Sections 6.1 and 6.2);
- Extension of this prior to the continuous-time, geocentric estimation of third-party motions (Section 6.2.2);
- Reacquisition of previously occluded motions in *motion closure* using a motion-based tracking metric (Section 6.3);
- Evaluation of the occlusion-robust MVO pipeline on complex multimotion scenes with significant occlusion from the OMD (Section 6.4).

Thus far, this thesis has primarily focused on stereo RGB data from the OMD, but the MVO pipeline operates on 3D tracklets and is largely agnostic to the sensor that generates them. Each sensor archetype, such as cameras or lidar, poses a unique set of challenges in the context of the MEP. Chapter 7 extends the MVO pipeline to stereo event cameras, which are an alternative type of sensor that asynchronously measure pixel-wise changes in brightness. These cameras present several key advantages over traditional RGB cameras, but their unique data structure introduces several important challenges as well.

7

Adapting Multimotion Estimation to Event Data

Contents

The previous chapters have focused on addressing the MEP using stereo RGB cameras. The cameras are well calibrated and synchronized, making them convenient for visual navigation; however, they are designed for recording static images rather than sensing dynamic motion. These global shutter cameras can record dense image frames at a fixed frame rate, but much of the image information is likely redundant and uninformative due to poor or repetitive texture in the environment. Processing these images can be time-intensive, especially with a high-resolution camera, and textureless regions represent wasted computation.

Highly dynamic scenes can also result in motion blur, and changing lighting conditions can lead to over- or underexposed images that are unsuitable for estimation. The Bumblebee XB3 used in the OMD is particularly susceptible to

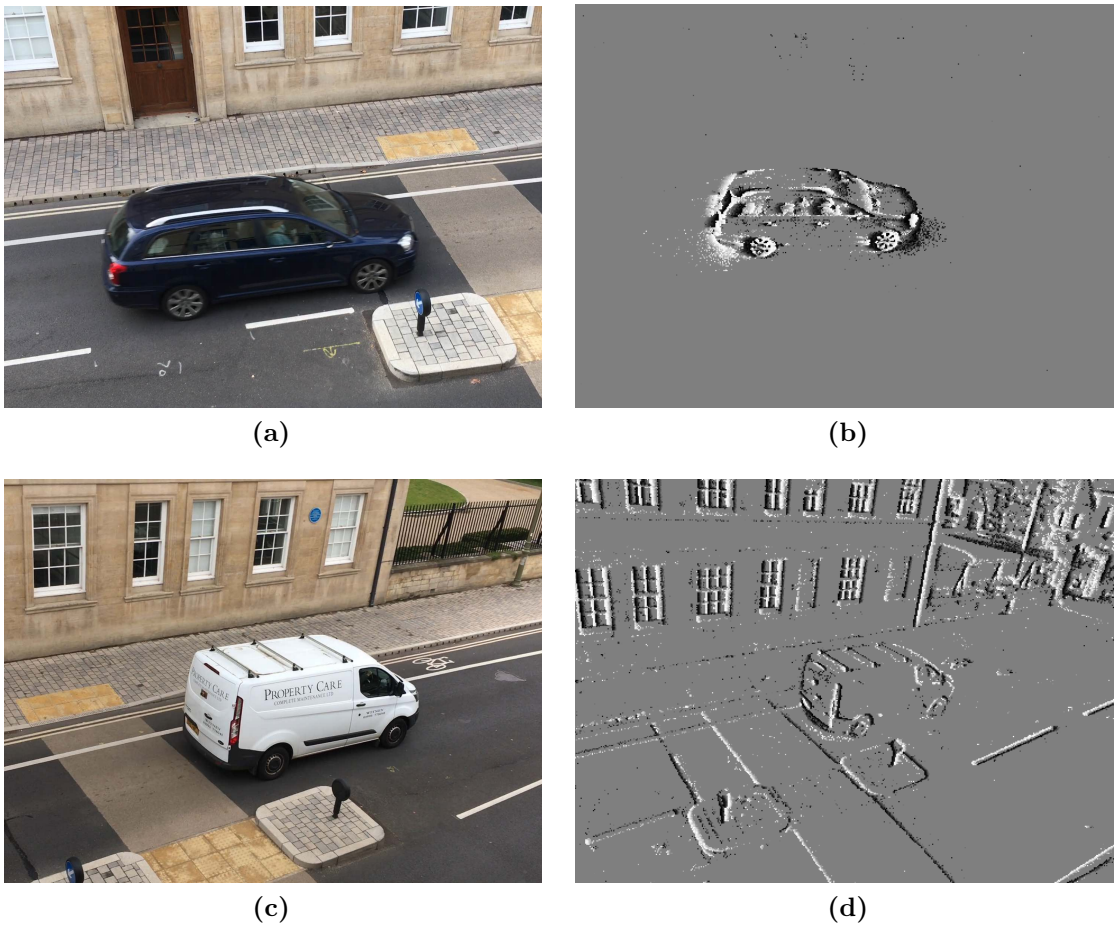


Figure 7.1: A comparison of traditional RGB camera frames (a and c) and event camera data (b and d). The top row of images were taken while the cameras were static, and the bottom row were taken while the cameras were moving. The RGB camera records the entire scene, even textureless areas, even when the camera is static. In contrast, the event camera only records changes in the observed scene. When the camera is static, this only corresponds to dynamic objects in the scene, and the static background can only be observed when the camera is moving.

motion blur and it has a low frame rate (16 Hz). Monocular cameras used for visual estimation often have higher frame rates, but still suffer from these limitations. While automatic exposure features can handle some lighting changes, visual estimation in low-light and outdoor environments remains difficult for traditional camera sensors.

Event cameras are an alternative type of sensor that asynchronously measure pixel-wise changes in brightness. These sensors provide several key advantages over traditional cameras in the context of the MEP.

Event cameras only record changes in brightness, or *events*. The camera asyn-

chronously records these events as a serial stream, which is a sparse representation of the scene that only includes observed changes over time. This directly aligns with the focus of motion estimation techniques because the cameras inherently measure the *dynamism* of the observed scene (Fig. 7.1).

Event cameras consist of sensor arrays that record pixel-wise changes in brightness, compared to the absolute intensity images recorded by traditional cameras. This means an event camera can detect when a dark part of the scene becomes darker, and vice versa for light parts of the scene. Event cameras have much larger dynamic ranges than traditional cameras and can be used to navigate through over- or underlit environments, so long as the lighting is not flickering. In contrast, traditional cameras often have to over- or underexpose parts of a scene in order to produce a useful image. This also means that event cameras are able to perceive shadows and reflections that are hard for traditional cameras, or even human eyes, to detect.

Each event consists of the timestamp, the image coordinates, and the polarity of the brightness change (bright-to-dark or dark-to-bright). This data structure is very compact, meaning events can be recorded at extremely high rates (>10 kHz) and the cameras tend to have much lower power requirements. Though each individual event provides little information, multiple events can provide image context at a much higher rate than traditional cameras.

Event cameras also have a unique set of limitations compared to traditional cameras, such as high cost and low resolution. Additionally, each individual pixel sensor is sensitive to noise and can often records spurious events. A single event also provides little information, so groups of events must be analyzed together to understand a scene, much like a single pixel of an RGB frame is not informative without the context of its neighbors. Unfortunately, the asynchronous event stream does not have the same structure as traditional image frames, so a new class of algorithms must be constructed for this type of sensor.

This chapter explores how the techniques detailed in Chapter 5 and Chapter 6 can be applied to a stereo event camera sensor. Section 7.1 details the challenges posed by event data for motion estimation, specifically in the context of the MEP.

Section 7.2 describes how the MVO pipeline can be applied to event data and Section 7.3 qualitatively illustrates its ability to segment and estimate multiple motions from event data using a real-world experiment.

The novel contributions of this chapter are:

- Discussion of the challenges inherent to addressing the MEP in event data (Section 7.1);
- Extension of the MVO pipeline to simultaneously segment and estimate multiple $SE(3)$ motions from sparse event streams (Section 7.2);
- Qualitative evaluation of the MVO pipeline on real-world event camera data segments with multiple motions (Section 7.3).

7.1 Event-Based Motion Analysis

Event cameras have only recently been leveraged for motion analysis, and most approaches are either relatively simple or rely on external models. The novel structure and limitations of the event stream present a sharp contrast to the traditional frames that most motion analysis techniques depend on. Some approaches mitigate these limitations by incorporating additional sensor types, such as traditional cameras (Brandli et al. 2014) or IMUs (Vidal et al. 2018). Addressing the tasks of feature detection and tracking, as well as motion estimation, segmentation, and tracking, using only event cameras poses a novel set of challenges in addition to those inherent in the MEP.

A single event has little context so events are often compared to models of the environment that are generated separately or accumulated into "frames" in order to process them in groups. Accumulated frames can consist of a fixed number of recent events or of every event recorded over a fixed time period. Using a fixed *number* of events for each frame adheres better to the asynchronous nature of event cameras but results in a varying frame rate. The varying frame rate can be addressed using the continuous-time estimation techniques described in Chapter 6, but it also means that highly textured scenes may be separated into multiple frames and sparsely textured scenes may have significant motion blur. Using a

fixed *duration* for each frame breaks the asynchronous nature of event cameras in that it results in a fixed frame rate with latency comparable to traditional cameras. This method aligns better with traditional image processing techniques and highly textured scenes can still be processed in their entirety, but it means sparsely textured scenes may result in largely empty frames.

One of the most fundamental stages of many motion analysis techniques is the tracking of low-level feature points across multiple images. Traditional feature-tracking techniques detect salient points in the image, extract local image patches at those points, and compare those patches across each image (Section 2.7). Tracking features in event cameras is significantly more difficult because of the unstructured nature of the event streams and the high noise levels of the sensors.

While traditional feature matching techniques can be applied to image frames generated from event cameras with limited success (Kogler et al. 2011), synchronized stereo event cameras can also exploit the temporal precision of the events to determine corresponding features. Events corresponding to the same world point may not arrive at the same time, so stereo matching must be more nuanced than simply matching timestamps. For example, Schraml et al. (2010) compare sequences of events according to the number, order, and temporal spacing of events to determine correspondences.

Tracking over time is much more difficult because there is no temporal synchronization to constrain the correspondence search. By treating the event stream as a sparse volumetric flow, corners can be detected as the intersection of two edges, which form intersecting planes in the event flow (Clady et al. 2015). Traditional corner detectors can also be extended to the event stream (Vasco et al. 2016; Mueggler et al. 2017), and Alzugaray and Chli (2018) propose corner detection methods based on finding corners in the event flow volume. These techniques show promising results, but event-based feature tracking is much more difficult than tracking in traditional RGB camera frames..

Event-based motion estimation approaches were originally limited to highly constrained motions, but recent techniques are able to estimate the full $SE(3)$

motion of the camera. Mueggler et al. (2014) track line segments in event data to estimate the full $SE(3)$ pose of an event camera with very low latency. Mueggler et al. (2015) improve the accuracy and robustness of this technique using continuous-time estimation techniques, which are well-suited to the high-speed event data. The key challenge in estimating motion from event data is balancing the trade-off between minimizing latency and maximizing accuracy, which are each a function of the number of events accumulated for estimation.

Event-based motion segmentation techniques often focus on tracking objects from static cameras, where the events are only generated by dynamic objects (Fig. 7.1b). Segmentation is much more challenging when the image contains many more events due to the dynamic camera (Fig. 7.1d). Vasco et al. (2017) track low-level features and segment those generated by the camera motion from dynamic objects using a learning method based on the kinematics of the actuator moving the camera. Stoffregen and Kleeman (2018) calculate optical flow from the event stream and segment it along flow discontinuities, similar to the methods discussed in Section 3.4.2. Segmentation can also be achieved using motion compensation (Mitrokhin et al. 2018). By warping all events detected over a given window to compensate for the camera motion, the events generated by individual static edges in the scene can be aligned. Any independent dynamic objects will generate motion blur in the motion-compensated event image and can be segmented.

Event-based tracking is a nascent research area. Glover and Bartolozzi (2016) and Glover and Bartolozzi (2017) are able to track simple shapes from event-based optical flow using Hough transforms and particle filters, respectively. Mitrokhin et al. (2018) use a traditional Kalman filter to track dynamic objects through occlusions and missed detections following the techniques described in Section 3.5.1. These approaches only track objects in image space and do not estimate their $SE(3)$ pose.

Event cameras present novel challenges for each individual aspect of the MEP and several techniques have been designed to address individual aspects of the problem, but thus far none has attempted to address it in its entirety. This chapter

adapts the occlusion-robust MVO pipeline to simultaneously segment and estimate multiple $SE(3)$ motions from a stereo event camera.

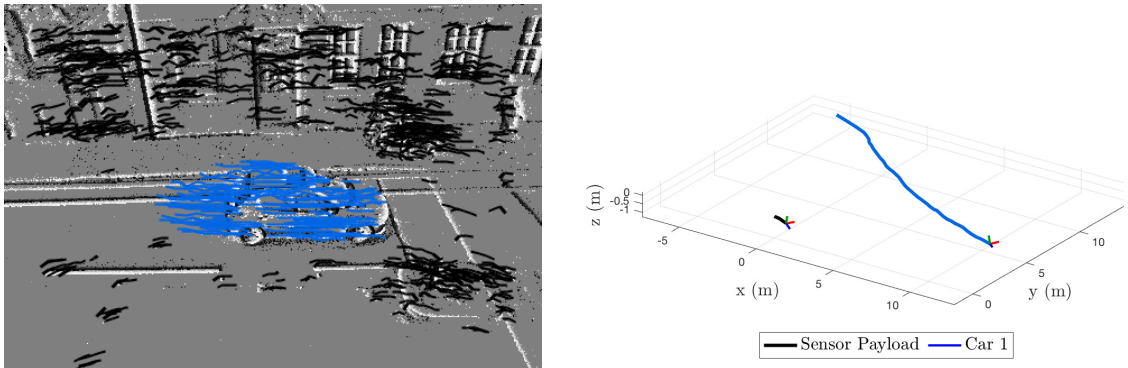
7.2 MVO in Event Data

The MVO pipeline described in the previous chapters operates on 3D tracklets and is largely agnostic to the sensor that generates them. This means it can be applied to the stereo event camera as long as the two asynchronous event streams can be processed into 3D tracked feature points. The cameras are statically calibrated, so the triangulation tools described in Section 2.6 are valid, and the primary challenge is associating points between stereo images and temporally across frames.

In order to analyze the scene and its constituent motions, multiple events must first be accumulated and analyzed in bulk. Each incoming event stream is grouped into "frames" at a fixed frame rate. Each event corresponds to a change in a pixel's brightness, so each pixel can be modeled with a ternary value based on whether it got brighter, darker, or did not change, i.e. there were no events recorded for that pixel.

Traditional feature tracking techniques used to generate tracklets from RGB images (Section 2.7) do not perform well on these highly discrete images. Noisy events cause spurious corner detections and can greatly alter the feature descriptors extracted for these points. Treating pixel values as ternary also ignores the temporal information available in the event timestamps. The event-based MVO pipeline instead treats each pixel as a greyscale value that shifts high or low upon the arrival of a new event depending on its polarity and decays toward a neutral value over the "duration" of the event. This incorporates the temporal information directly into the event frame and results in a smoother image that is more conducive to feature matching.

The tracked features are still very volatile, resulting in many short tracklets. This, combined with the sparsity of the data and the low image resolution, means the tuning parameters used in the previous chapters must be adjusted (Table A.2).



(a) (b)

Figure 7.2: The segmentation (a) and trajectory estimates (b) produced by the occlusion-robust MVO pipeline on a real-world event-data segment. MVO is able to process the stereo event streams and simultaneously segment and estimate the motion of the car and the camera egomotion. The car moves down the road and across the camera's field of view with relatively constant velocity while the camera translates and pans to keep it in view. The trajectory plot in (b) is shown relative to the initial camera pose, which is arbitrary as no ground truth data is available for reference.

7.3 Evaluation

To evaluate the event-based MVO performance, a set of real-world data segments were collected using a calibrated stereo pair of Prophesee event cameras. The cameras have a 480×360 resolution and their clocks were hardware synchronized. The baseline of the cameras was chosen to match that of the Bumblebee XB3 stereo camera used in the OMD.

The cameras dynamically observe a busy city street with vehicles moving in both directions, as well as cyclists and pedestrians (Fig. 7.1). The sequences are short (roughly three seconds each) due to the difficulty of keeping multiple motions of interest within the cameras' fields of view in real-world scenarios. Figs. 7.2 and 7.3 illustrate the qualitative results of the occlusion-robust MVO pipeline on two different real-world data segments. The full trajectory plot for each segment is shown along with the segmentation for a single frame of the sequence. The event data was accumulated at an effective frame rate of 25 Hz and the event duration was 20 ms. Estimation for both segments was performed as an 8-frame sliding window, with 5 neighbors for each point in the graph, 1000 RANSAC iterations

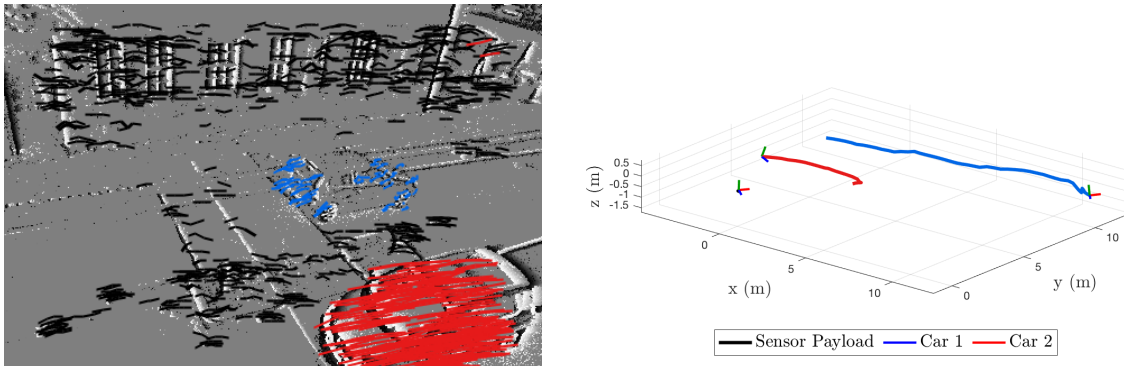
per new label, $e_{\text{th}} = 5$, $\alpha = 100$, $\beta = 5$, $\psi_\ell = 100$, $\lambda = 2$, $\epsilon_{\text{pos}} = 1$, $\epsilon_{\text{vel}} = 1$, and a minimum model size and length of 10 points and 3 frames, respectively.

The first sequence includes a single moving car and several stationary and moving pedestrians. The camera moves horizontally and pans to keep the car in view (Fig. 7.2). The car moves with relatively constant velocity, is reliably segmented, and the estimated trajectory is qualitatively consistent with its motion. The camera egomotion is estimated to span 0.98 m with 9.27°, 21.91°, and 45.40° of rotation in roll-pitch-yaw, respectively. The car trajectory is estimated to span 19.56 m with −12.87°, 4.38°, and 40.49° of rotation in roll-pitch-yaw, respectively. In contrast, the pedestrians move more irregularly and less distinctly; which, combined with their small size and the low image resolution, results in their motions not being accurately segmented or estimated.

The second sequence includes a two cars moving in opposite directions and the camera moves and rotates to keep both cars mostly in view (Fig. 7.3). Both cars move with relatively constant velocity and are consistently segmented and estimated. Car 1 moves in the background from left to right across the field of view of the camera, and Car 2 moves from right to left in the foreground. The camera egomotion is estimated to span 0.49 m with 2.59°, 7.21°, and 38.30° of rotation in roll-pitch-yaw, respectively. The trajectory of Car 1 is estimated to span 17.34 m with 4.89°, 1.94°, and 50.09° of rotation in roll-pitch-yaw, respectively; and the trajectory of Car 2 is estimated to span 7.12 m with 8.61°, 2.75°, and 0.88° of rotation in roll-pitch-yaw, respectively. The trajectory of Car 2 is shorter because the speed, size, and position of the cars make it difficult to keep them both in the camera's field of view for long, and the camera was moved to track Car 1.

7.4 Discussion

The results described above are qualitative and serve to demonstrate that the MVO pipeline is applicable to stereo event cameras, but there are still significant areas of research to be pursued. Using stereo event cameras is particularly difficult due to the nature of the sparse, asynchronous event stream, as well as their low resolution.



(a) **(b)**

Figure 7.3: The segmentation (a) and trajectory estimates (b) produced by the occlusion-robust MVO pipeline on a real-world event-data segment. Car 1 (background, blue) moves from left to right across the field of view of the camera, and Car 2 (foreground, red) moves from right to left. As in Fig. 7.2, MVO is able to simultaneously segment and estimate the motion of both cars and the camera egomotion from the event data. The trajectory plot in (b) is shown relative to the initial camera pose, which is arbitrary as no ground truth data is available for reference.

MVO can also be applied to other 3D sensor types, such as RGB-D cameras and scanning lidar, and they each pose their own set of challenges.

The reliance on changes in brightness means event cameras are well-suited for motion analysis as they only record dynamic parts of the image. This yields a sparse information stream for efficient segmentation and estimation, but it also means that objects and even entire scenes can be occluded if there is no observed motion, i.e., the camera is static or near-perfectly tracking an object. A notable example of this is when objects move away from the camera along the optical axis, which is a common motion in applications such as autonomous driving. In the real-world segments in Section 7.3, the camera does not move with constant velocity, so the density of features varies significantly over the course of the segment. The events corresponding to the motion of the cars are even more volatile as the relative motion of an object observed in the image plane changes as it and the camera move. These variations lead to difficulties in accurately segmenting and estimating the motion of the cars, and these situations make the occlusion-robust techniques discussed in Chapter 6 significantly more important for event cameras.

Tracking low-level features is also difficult in event cameras. The precise temporal information in each event means that stereo matching can be more accurate than in RGB cameras, but noise and variations in the sensor arrays of different cameras can easily corrupt the triangulation. Temporal matching is also difficult, as events arrive asynchronously and it can be difficult to understand the local context of a scene from a small number of events. These difficulties, along with the low resolution of the cameras relative to comparably priced RGB cameras, make it difficult to segment small motions that could otherwise be detected in traditional RGB images. For example, it is difficult to segment and estimate the motions of the pedestrians in Fig. 7.2 because they are both small and slow, so they do not constitute a distinct-enough motion to be segmented from the static background. This is especially true when the camera pans because the rotation causes large observed motions for distant points, making the pedestrians' motions even less distinct compared to the noise in the image and tracked features.

The area of feature tracking within event data is an ongoing area of research, and MVO is shown to be able to simultaneously segment and estimate multiple distinct motions, given that a suitable number features can be reliably tracked on them. Event cameras are a relatively nascent technology but are rapidly advancing in both hardware and software capabilities, and improvements in sensor design and resolution will continue to drive their value in robotics applications. A continued research focus on addressing the challenges posed by the MEP using event cameras will address their current limitations and capitalize on their unique advantages.

7.5 Summary

This chapter qualitatively illustrates that the MVO pipeline can be applied to stereo event cameras to accurately segment and estimate multiple $SE(3)$ motions, including the camera egomotion, in real-world environments. The results in this chapter indicate an exciting new direction in multimotion estimation research using event cameras. The novel contributions of this chapter are:

- Discussion of the challenges inherent to addressing the MEP in event data (Section 7.1);
- Extension of the MVO pipeline to simultaneously segment and estimate multiple $SE(3)$ motions from a sparse event stream (Section 7.2);
- Qualitative evaluation of the MVO pipeline on real-world event camera data segments with multiple motions (Section 7.3).

8
Conclusion

Accurate multimotion estimation and tracking is critically important for autonomous navigation in complex, dynamic environments. Addressing this MEP involves segmenting and estimating the dynamic motions in a scene, including the egomotion of the sensor, and tracking those motions throughout the scene, even in the presence of occlusion. Broad fields of robotics and computer vision research have focused on estimating the motion of a dynamic sensor, segmenting a dynamic scene into its constituent objects, and tracking multiple objects through occlusions; however, far less effort has been directed at *unifying* these tasks into a simultaneous approach.

The process of estimating the egomotion of a dynamic sensor relative to its static environment is a fundamental requirement in mobile robotics, and VO is a common approach for this using cameras. This estimation inherently requires segmenting the static portions of the scene from dynamic portions, and most approaches simply ignore this "dynamic noise." In complex, dynamic scenes, this segmentation can be difficult to determine, as independent dynamic motions contribute to a low signal-to-noise-ratio. In these situations, it becomes necessary to analyze and estimate *every* motion in the scene simultaneously in order to accurately determine the egomotion of the sensor. This multimotion-estimation approach additionally calculates the trajectories of all other third-party motions in the scene, which yields important information for safe navigation throughout an environment.

Estimating third-party motions is much more difficult than estimating the sensor egomotion because they cannot be assumed to be static. The observed motion of a static background is generated by the motion of the sensor, but the observed motion of a dynamic object is generated by both the sensor motion and the object's own motion. The challenges posed by this MEP are identified and explored in Chapter 3. Most previous work has focused on a subset of this problem, and very little work has been focused on the MEP in its entirety. Some approaches apply simplifying constraints to address the MEP, but these techniques do not generalize well to other applications where those assumptions do not hold.

Chapter 4 further dissects the MEP and the challenges it poses through segments of the OMD. Designed as a scaffold for the development and evaluation of multimotion estimation techniques, the dataset explores the individual aspects of $SE(3)$ estimation, multimotion segmentation and estimation, and tracking through occlusion. This dataset was first published as

> Kevin Judd and Jonathan Gammell (Apr. 2019a). "The Oxford Multimotion Dataset: Multiple SE(3) Motions With Ground Truth". In: *IEEE Robotics and Automation Letters (RA-L)* 4.2. Presented at ICRA 2019, pp. 800–807

This thesis introduces MVO, a feature-based approach for addressing the MEP in a stereo pipeline. Chapter 5 explores this MVO pipeline, which extends the traditional VO pipeline by casting the MEP as a multilabeling problem and employing multimodel-fitting techniques. Each independent $SE(3)$ trajectory is represented by a label, and sparse tracklets are assigned to motion labels in order to minimize an energy functional incorporating reprojection residual, local smoothness, and solution complexity costs. The pipeline is shown to be able to segment and estimate the full $SE(3)$ trajectory of every motion in the scene, including the egomotion, using only a rigid-motion assumption. This work first appeared in the following publications:

> Kevin Judd et al. (June 2018a). "Visual multimotion estimation". In: *Joint Industry and Robotics CDTs Symposium (JIRCS)*

Kevin Judd et al. (Oct. 2018b). "Multimotion Visual Odometry (MVO): Simultaneous Estimation of Camera and Third-Party Motions". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. arXiv (corrected version):1808.00274 [cs.RO], pp. 3949–3956

The pipeline relies on direct observations and therefore cannot estimate motions through occlusion. Likewise, the pipeline uses sparse, feature-based techniques, meaning it can estimate motions through some partial occlusions, but an imbalance of tracked features on a rotating body can bias the estimator toward translational, rather than rotational, motion. While the pipeline does not rely on *a priori* appearance information for detecting object motions, it does require a sufficient number of features to be tracked on any dynamic object, so it is difficult to track small or textureless objects.

Chapter 6 extends the MVO pipeline to be robust to occlusion by exploiting an $SE(3)$ white-noise-on-acceleration motion prior. The prior is physically founded and penalizes deviation from a locally constant, body-centric velocity, which is a reasonable approximation for many real-world motions. This prior is used to extrapolate previously observed motion estimates until the object becomes visible again. Extrapolated estimates are used in *motion closure* to recover tracking when objects reappear in the predicted location, and the extrapolated estimates can be corrected using interpolation. The full $SE(3)$ trajectory of every motion in the scene is estimated through both direct and indirect occlusions. This work first appeared in the following publications:

Kevin Judd and Jonathan Gammell (May 2019b). "SE(3) multimotion estimation through occlusion." In: *Long-Term Human Motion Prediction (LHMP) Workshop, IEEE International Conference on Robotics and Automation (ICRA)*

Kevin Judd and Jonathan Gammell (2020). "Occlusion-robust MVO: Multimotion estimation through occlusion via motion closure". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. To appear, arXiv:1905.05121 [cs.RO]

The occlusion-robust MVO pipeline is still a sparse, feature-based technique, so partial occlusions can still degrade estimation accuracy by changing the distribution

of features and biasing the estimator. Likewise, objects rarely become instantaneously occluded or unoccluded, and these periods of partial occlusion change the observed shape of the object and affect the estimated trajectory. These limitations are partially mitigated by the constant-velocity prior because it penalizes strong changes in the local, body-centric velocity.

The reliance of the occlusion-robust MVO pipeline on motion means that objects that temporarily have the same motion will be given the same label, such as when a dynamic object becomes stationary. This is often desirable, as it implicitly handles trajectory merging, but appearance-based object descriptors can also be used to enforce a form of *motion permanence*. The accuracy of the extrapolated estimates is also dependent on how well the prior models the true object motions, and the applicability of the white-noise-on-acceleration motion prior is limited in scenes where objects change direction or speed. These applications may require more expressive motion priors.

Chapter 7 extends the occlusion-robust MVO pipeline to event camera data. The asynchronous event data stream records pixel-wise changes in brightness and presents a challenging format that is largely incompatible with many common image-processing techniques. At its core, the MVO pipeline operates on 3D tracklets and is largely agnostic to the sensor modality that generates these tracklets. Event cameras inherently measure dynamic changes within a scene, so they are particularly well suited for motion estimation, but they pose difficult calibration and processing challenges. Though it requires addressing these challenges, the MVO pipeline was demonstrated to be capable of segmenting and estimating multiple motions in real-world scenarios using this new event-based sensor archetype.

This thesis demonstrates an approach for unifying the challenges of motion estimation, segmentation, and tracking. The Multimotion Visual Odometry pipeline simultaneously segments and estimates the full $SE\,(3)$ motion of every motion in highly dynamic scenes while also tracking those motions through occlusions. Several avenues of future research are presented to further extend the techniques described

in this thesis, and this research area will no doubt become increasingly important as autonomous navigation applications become more demanding.

In summary, the major contributions of this thesis are:

- Definition and exploration of the MEP (Chapter 3 and Chapter 4).
- Simultaneous motion segmentation and estimation of *every* motion of the scene using low-level feature points, rather than relying on higher-level appearance information or *a priori* motion constraints (Chapter 5);
- Full $SE(3)$ trajectory estimation of each motion in the scene using only a rigid-body assumption (Chapter 5);
- Deferral of the designation of a given motion as belonging to the camera, i.e., the egomotion, until all hypothetical egomotion trajectories are estimated, after which their geocentric equivalents can be calculated (Chapter 5).
- Evaluation of the MVO pipeline on complex multimotion scenes from the OMD (Chapter 5);
- Application of a continuous white-noise-on-acceleration prior to the MVO pipeline to egocentrically segment and estimate a smooth trajectory for every motion in the scene (Chapter 6);
- Extension of this prior to the continuous-time, geocentric estimation of third-party motions (Chapter 6);
- Extrapolation and interpolation of occluded trajectories using the white-noise-on-acceleration prior (Chapter 6);
- Reacquisition of previously occluded motions in *motion closure* using a motion-based tracking metric (Chapter 6);
- Evaluation of the occlusion-robust MVO pipeline on complex multimotion scenes with significant occlusion from the OMD (Chapter 6);
- Extension of the occlusion-robust MVO pipeline to event data (Chapter 7);
- Evaluation of the event-based MVO pipeline on real-world multimotion scenes (Chapter 7).

Appendices

A

Feature Detection and Tracking Parameters

The MVO pipeline operates on 3D tracklets and is largely agnostic to the method by which those tracklets are generated. This thesis focuses on stereo image pipelines to generate these tracklets, and relies on LIBVISO2 (Geiger et al. 2011) to detect and match image features across stereo pairs and between consecutive stereo frames. The tables below list the LIBVISO2 parameters used in the results presented in Chapters 4 to 6 (Table A.1) and in Chapter 7 (Table A.2). The stereo_radius parameter was introduced for the purpose of differentiating between the search window for stereo and temporal feature correspondences.

Table A.1: LIBVISO2 parameter values used for the OMD results presented in this thesis.

Parameter	Value	Notes
nms_n	2	non-maximal-suppression minimum distance between maxima (pixels)
nms_tau	50	non-maximal-suppression peakiness threshold of maxima
match_binsize	50	matching bin width/height (pixels)
match_radius	30	temporal matching radius (pixels)
stereo_radius	150	stereo matching radius (pixels)
match_disp_tolerance	3	vertical tolerance for stereo matches (pixels)
outlier_disp_tolerance	10	disparity tolerance for outlier removal (pixels)
outlier_flow_tolerance	10	flow tolerance for outlier removal (pixels)
multi_stage	1	multistage matching (denser and faster)
half_resolution	1	match at half resolution, refine at full resolution
refinement	1	pixel-level refinement

Table A.2: LIBVISO2 parameter values used for real-world event camera results presented in this thesis.

Parameter	Value	Notes
nms_n	1	non-maximal-suppression minimum distance between maxima (pixels)
nms_tau	5	non-maximal-suppression peakiness threshold of maxima
match_binsize	50	matching bin width/height (pixels)
match_radius	30	temporal matching radius (pixels)
stereo_radius	20	stereo matching radius (pixels)
match_disp_tolerance	10	vertical tolerance for stereo matches (pixels)
outlier_disp_tolerance	10	disparity tolerance for outlier removal (pixels)
outlier_flow_tolerance	10	flow tolerance for outlier removal (pixels)
multi_stage	1	multistage matching (denser and faster)
half_resolution	1	0 match at half resolution, refine at full resolution
refinement	1	pixel-level refinement

B

OMD Example Frames

The OMD consists of real-world stereo and RGB-D camera images and IMU data from several different dynamic scenes. Each scene highlights a challenging aspect of the MEP and includes both static and dynamic sensor motions. The scenes are detailed in Chapter 4 and frame sequences of several of the data segments are included here to illustrate the observed motions involved in each.

Figure B.1: Several example frames from the `pinwheel_1_unconstrained` segment of the OMD. The frames are taken from the left camera of the Bumblebee XB3 stereo camera and are each 4 frames (0.25 s) apart.

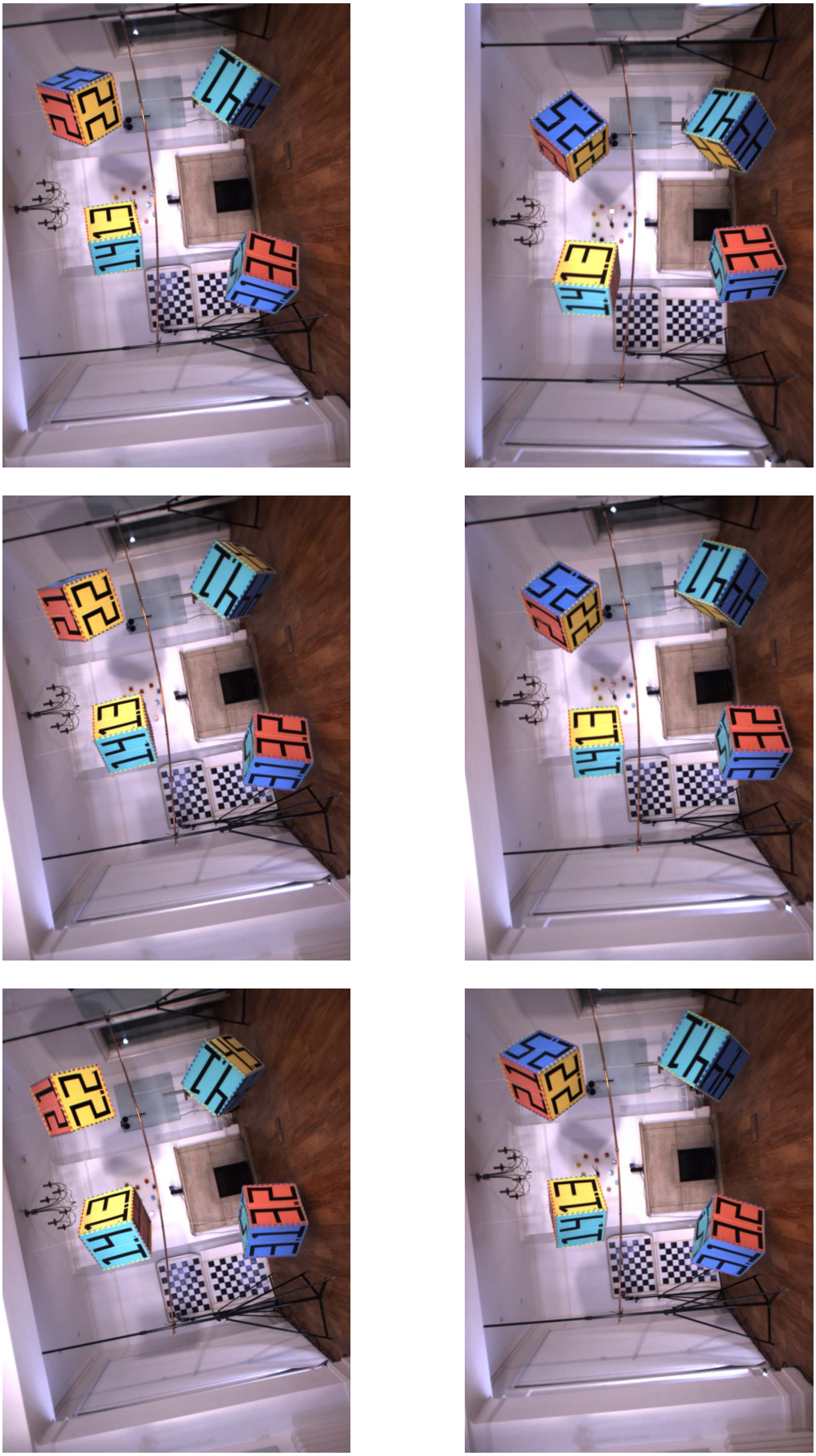


Figure B.2: Several example frames from the `swinging_4_unconstrained` segment of the OMD. The frames are taken from the left camera of the Bumblebee XB3 stereo camera and are each 4 frames (0.25 s) apart.

Figure B.3: Several example frames from the `occlusion_2_unconstrained` segment of the OMD. The frames are taken from the left camera of the Bumblebee XB3 stereo camera and are each 8 frames (0.5 s) apart.

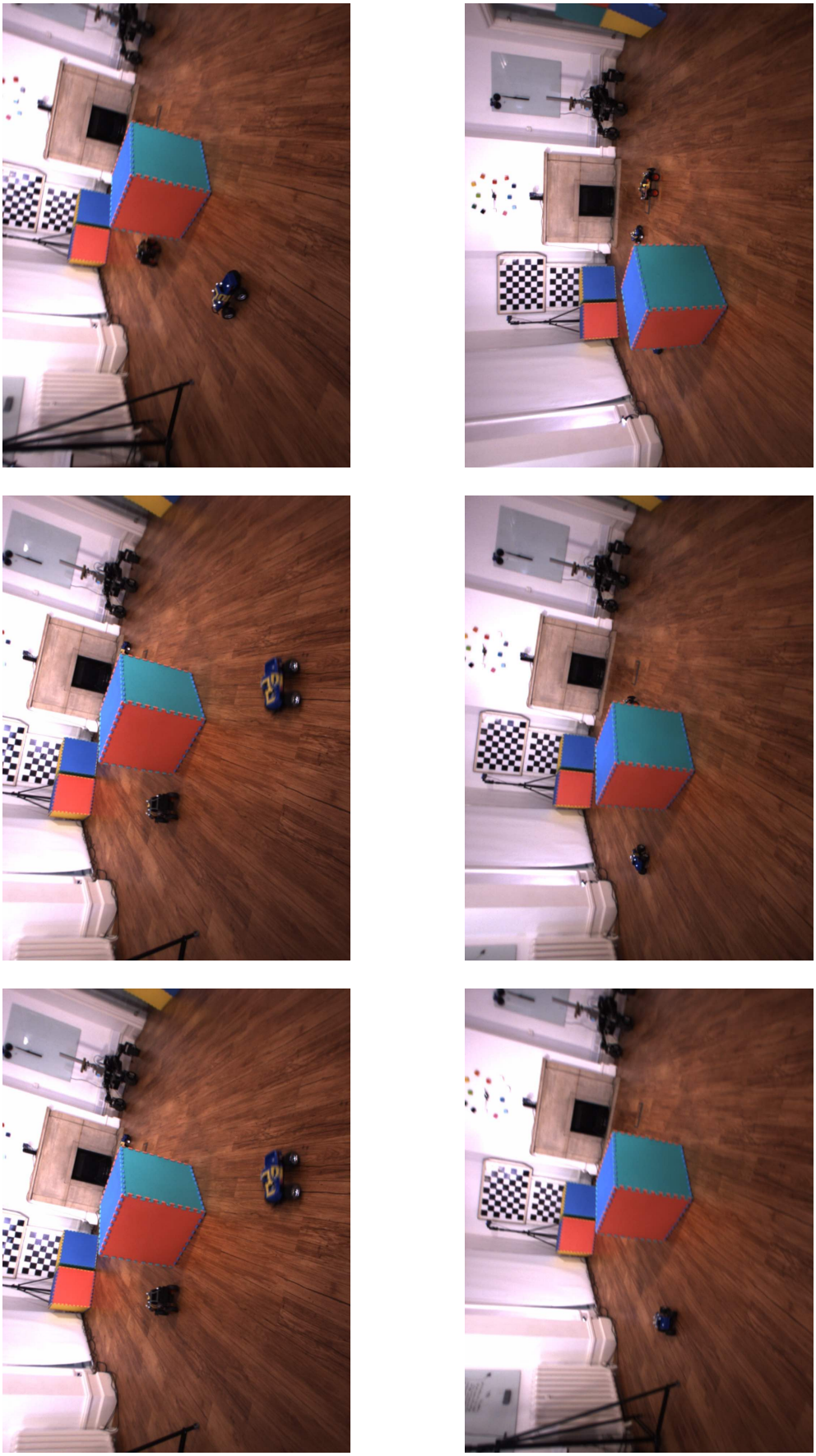


Figure B.4: Several example frames from the `cars_3_unconstrained` segment of the OMD. The frames are taken from the left camera of the Bumblebee XB3 stereo camera and are each 8 frames (0.5 s) apart.

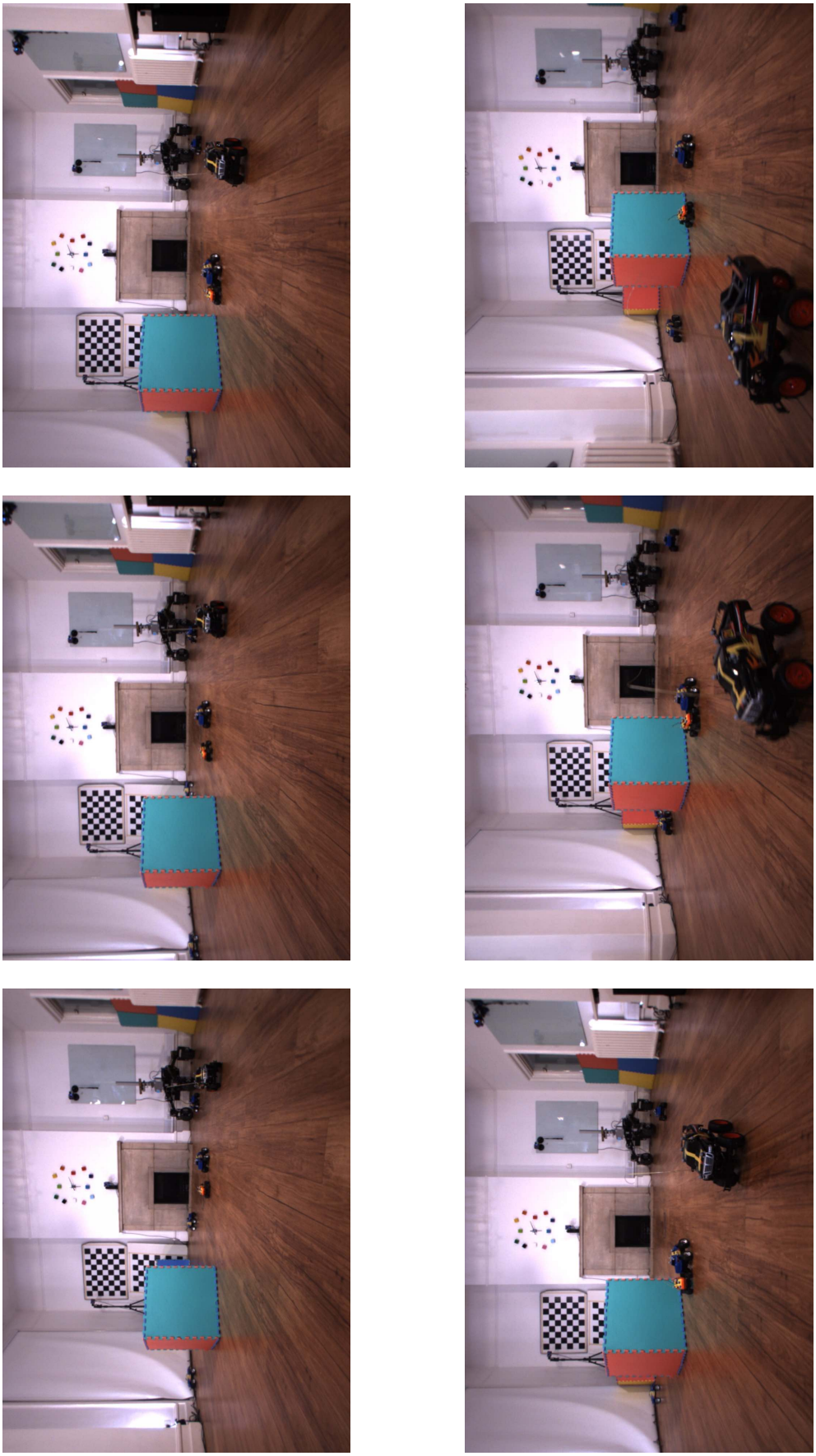


Figure B.5: Several example frames from the `cars_6_robot` segment of the OMD. The frames are taken from the left camera of the Bumblebee XB3 stereo camera and are each 8 frames (0.5 s) apart.

References

Agarwal, Sameer, Keir Mierle, and Others. *Ceres Solver*.

Alahi, Alexandre, Raphael Ortiz, and Pierre Vandergheynst (2012). "FREAK: Fast retina keypoint". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 510–517.

Alzugaray, Ignacio and Margarita Chli (2018). "Asynchronous corner detection and tracking for event cameras in real time". In: *IEEE Robotics and Automation Letters (RA-L)* 3.4, pp. 3177–3184.

Amayo, Paul, Pedro Piniés, Lina Maria Paz, and Paul Newman (2018). "Geometric Multi-Model Fitting with a Convex Relaxation Algorithm". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Anderson, S. and T. D. Barfoot (Sept. 2015). "Full STEAM ahead: Exactly sparse gaussian process regression for batch continuous-time trajectory estimation on SE(3)". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 157–164.

Barfoot, Timothy D (2017). *State Estimation for Robotics*. Cambridge University Press.

Barfoot, Timothy D., Chi Hay Tong, and Simo Särkkä (July 2014). "Batch Continuous-Time Trajectory Estimation as Exactly Sparse Gaussian Process Regression". In: *Robotics: Science and Systems (RSS)*. Berkeley, USA.

Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool (2006). "Surf: Speeded up robust features". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 404–417.

Besl, P. J. and N. D. McKay (Feb. 1992). "A method for registration of 3-D shapes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 14.2, pp. 239–256.

Blake, Andrew and Michael Isard (1997). "The CONDENSATION algorithm-conditional density propagation and applications to visual tracking". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 361–367.

Boult, Terrance E and L Gottesfeld Brown (1991). "Factorization-based segmentation of motions". In: *IEEE Workshop on Visual Motion*, pp. 179–186.

Boykov, Yuri, Olga Veksler, and Ramin Zabih (1999). "Fast approximate energy minimization via graph cuts". In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 1, pp. 377–384.

Boykov, Yuri Y and Marie-Pierre Jolly (2001). "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images". In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 1, pp. 105–112.

Brandli, Christian, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck (Oct. 2014). "A 240 × 180 130 dB 3 μs latency global shutter spatiotemporal vision sensor". In: *IEEE Journal of Solid-State Circuits* 49.10, pp. 2333–2341.

Breitenstein, M. D., F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool (Sept. 2009). "Robust tracking-by-detection using a detector confidence particle filter". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1515–1522.

Broida, Ted J and Rama Chellappa (1986). "Estimation of object motion parameters from noisy images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 1, pp. 90–99.

Byeon, Moonsub, Haanju Yoo, Kikyung Kim, Songhwai Oh, and Jin Young Choi (2018). "Unified optimization framework for localization and tracking of multiple targets with multiple cameras". In: *Computer Vision and Image Understanding* 166, pp. 51 –65.

Calonder, Michael, Vincent Lepetit, Christoph Strecha, and Pascal Fua (2010). "BRIEF: Binary robust independent elementary features". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 778–792.

Chung, Fan RK and Fan Chung Graham (1997). *Spectral graph theory*. 92. American Mathematical Soc.

Ciaparrone, Gioele, Francisco Luque Sánchez, et al. (2019). "Deep learning in video multi-object tracking: A survey". In: *Neurocomputing*.

Clady, Xavier, Sio-Hoi Ieng, and Ryad Benosman (2015). "Asynchronous event-based corner detection and matching". In: *Neural Networks* 66, pp. 91–106.

Costeira, João Paulo and Takeo Kanade (1998). "A multibody factorization method for independently moving objects". In: *International Journal of Computer Vision (IJCV)* 29.3, pp. 159–179.

Cox, Ingemar J. and Sunita L. Hingorani (1996). "An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 18.2, pp. 138–150.

Del Moral, Pierre (1997). "Nonlinear filtering: Interacting particle resolution". In: *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics* 325.6, pp. 653–658.

Dendorfer, Patrick, Hamid Rezatofighi, et al. (2019). "CVPR19 Tracking and Detection Challenge: How crowded can it get?" In: *arXiv preprint arXiv:1906.04567*.

Dosovitskiy, Alexey, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun (2017). *CARLA: An open urban driving simulator*. arXiv: 1711.03938 [cs.CV].

Eckenhoff, K., Y. Yang, P. Geneva, and G. Huang (Apr. 2019). "Tightly-Coupled Visual-Inertial Localization and 3-D Rigid-Body Target Tracking". In: *IEEE Robotics and Automation Letters (RA-L)* 4.2, pp. 1541–1548.

Engel, Jakob, Jurgen Sturm, and Daniel Cremers (2013). "Semi-dense visual odometry for a monocular camera". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1449–1456.

Engel, Jakob, Vladlen Koltun, and Daniel Cremers (2017). "Direct sparse odometry". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 40.3, pp. 611–625.

Farboud-Sheshdeh, Sara, Timothy D Barfoot, and Raymond H Kwong (2014). "Towards estimating bias in stereo visual odometry". In: *Conference on Computer and Robot Vision*, pp. 8–15.

Faugeras, Olivier D (1992). "What can be seen in three dimensions with an uncalibrated stereo rig?" In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 563–578.

Fischler, Martin A and Robert C Bolles (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6, pp. 381–395.

Fortun, Denis, Patrick Bouthemy, and Charles Kervrann (2015). "Optical flow modeling and computation: a survey". In: *Computer Vision and Image Understanding* 134, pp. 1–21.

Furgale, P., J. Rehder, and R. Siegwart (Nov. 2013). "Unified temporal and spatial calibration for multi-sensor systems". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1280–1286.

Gaidon, Adrien, Qiao Wang, Yohann Cabon, and Eleonora Vig (June 2016). "Virtual worlds as proxy for multi-object tracking analysis". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4340–4349.

Gear, C William (1994). "Feature grouping in moving objects". In: *IEEE Workshop Motion of Non-Rigid and Articulated Objects*, pp. 214–219.

Geiger, Andreas, Julius Ziegler, and Christoph Stiller (2011). "Stereoscan: Dense 3D reconstruction in real-time". In: *IEEE Intelligent Vehicles Symposium (IV)*, pp. 963–968.

Geiger, Andreas, Philip Lenz, and Raquel Urtasun (2012). "Are we ready for autonomous driving? the KITTI vision benchmark suite". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3354–3361.

Geiger, Andreas, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun (2014). "3D traffic scene understanding from movable platforms". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36.5, pp. 1012–1025.

Glover, Arren and Chiara Bartolozzi (2016). "Event-driven ball detection and gaze fixation in clutter". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2203–2208.

Glover, Arren and Chiara Bartolozzi (2017). "Robust visual tracking with a freely-moving event camera". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3769–3776.

Greig, Dorothy M, Bruce T Porteous, and Allan H Seheult (1989). "Exact maximum a posteriori estimation for binary images". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 51.2, pp. 271–279.

Gruber, Amit and Yair Weiss (2004). "Multibody factorization with uncertainty and missing data using the EM algorithm". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hahnel, Dirk, Rudolph Triebel, Wolfram Burgard, and Sebastian Thrun (2003). "Map building with mobile robots in dynamic environments". In: *IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2, pp. 1557–1563.

Han, Mei and Takeo Kanade (1999). *Perspective factorization methods for Euclidean reconstruction*.

Han, Xufeng, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg (2015). "Matchnet: Unifying feature and metric learning for patch-based matching". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3279–3286.

Harris, Christopher G and Mike Stephens (1988). "A combined corner and edge detector". In: *Alvey Vision Conference*. Citeseer.

Hartley, Richard (1995). "In defence of the 8-point algorithm". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1064–1070.

Hartley, Richard and Andrew Zisserman (2003). *Multiple view geometry in computer vision*. Cambridge university press.

Hartley, Richard, Rajiv Gupta, and Tom Chang (1992). "Stereo from uncalibrated cameras". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 761–764.

Hedborg, Johan, Per-Erik Forssén, Michael Felsberg, and Erik Ringaby (2012). "Rolling shutter bundle adjustment". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1434–1441.

Helbing, Dirk and Peter Molnar (1995). "Social force model for pedestrian dynamics". In: *Physical review E* 51.5, p. 4282.

Horn, Berthold KP (1987). "Closed-form solution of absolute orientation using unit quaternions". In: *Journal of the Optical Society of America* 4.4, pp. 629–642.

Horn, Berthold KP and Brian G Schunck (1981). "Determining optical flow". In: *Artificial intelligence* 17.1-3, pp. 185–203.

Hu, Min, Saad Ali, and Mubarak Shah (2008). "Detecting global motion patterns in complex videos". In: *International Conference on Pattern Recognition (ICPR)*, pp. 1–5.

Hu, W., X. Li, et al. (Dec. 2012). "Single and Multiple Object Tracking Using Log-Euclidean Riemannian Subspace and Block-Division Appearance Model". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 34.12, pp. 2420–2440.

Huber, Peter J. (1964). "Robust Estimation of a Location Parameter". In: *The Annals of Mathematical Statistics* 35.1, pp. 73–101.

Ichimura, N. (Sept. 1999). "Motion segmentation based on factorization method and discriminant criterion". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 600–605.

Ilg, Eddy, Nikolaus Mayer, et al. (2017). "Flownet 2.0: Evolution of optical flow estimation with deep networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2462–2470.

Inoue, Kohei and Kiichi Urahama (2001). "Separation of multiple objects in motion images by clustering". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 219–224.

Isack, Hossam and Yuri Boykov (2012). "Energy-based geometric multi-model fitting". In: *International Journal of Computer Vision (IJCV)* 97.2, pp. 123–147.

Jaimez, Mariano, Christian Kerl, Javier Gonzalez-Jimenez, and Daniel Cremers (2017). "Fast odometry and scene flow from RGB-D cameras based on geometric clustering". In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3992–3999.

Judd, Kevin and Jonathan Gammell (Apr. 2019a). "The Oxford Multimotion Dataset: Multiple SE(3) Motions With Ground Truth". In: *IEEE Robotics and Automation Letters (RA-L)* 4.2. Presented at ICRA 2019, pp. 800–807.

Judd, Kevin and Jonathan Gammell (May 2019b). "SE(3) multimotion estimation through occlusion." In: *Long-Term Human Motion Prediction (LHMP) Workshop, IEEE International Conference on Robotics and Automation (ICRA)*.

Judd, Kevin and Jonathan Gammell (2020). "Occlusion-robust MVO: Multimotion estimation through occlusion via motion closure". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. To appear, arXiv:1905.05121 [cs.RO].

Judd, Kevin, Jonathan Gammell, and Paul Newman (June 2018a). "Visual multimotion estimation". In: *Joint Industry and Robotics CDTs Symposium (JIRCS)*.

Judd, Kevin, Jonathan Gammell, and Paul Newman (Oct. 2018b). "Multimotion Visual Odometry (MVO): Simultaneous Estimation of Camera and Third-Party Motions". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. arXiv (corrected version):1808.00274 [cs.RO], pp. 3949–3956.

Kalman, Rudolph Emil (1960). "A New Approach to Linear Filtering and Prediction Problems". In: *Transactions of the ASME–Journal of Basic Engineering* 82.D, pp. 35–45.

Kanatani, Kenichi (2001). "Motion segmentation by subspace separation and model selection". In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 2, pp. 586–591.

Kaucic, R., A. G. Amitha Perera, G. Brooksby, J. Kaufhold, and A. Hoogs (June 2005). "A unified framework for tracking through occlusions and across sensor gaps". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 990–997.

Keat, James E (Feb. 1977). "Analysis of least-square attitude determination routine". In: *Computer Sciences Corporation Report CSC/TM-77/6034*.

Kendall, Alex, Matthew Grimes, and Roberto Cipolla (2015). "Posenet: A convolutional network for real-time 6-dof camera relocalization". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2938–2946.

Khan, Zia, Tucker Balch, and Frank Dellaert (2004). "An MCMC-based particle filter for tracking multiple interacting targets". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 279–290.

Kim, In Su, Hong Seok Choi, Kwang Moo Yi, Jin Young Choi, and Seong G Kong (2010). "Intelligent visual surveillance—A survey". In: *International Journal of Control, Automation and Systems* 8.5, pp. 926–939.

Kim, Jong Bae and Hang Joon Kim (2003). "Efficient region-based motion segmentation for a video monitoring system". In: *Pattern Recognition Letters* 24.1-3, pp. 113–128.

Kogler, Jürgen, Christoph Sulzbachner, Martin Humenberger, and Florian Eibensteiner (2011). "Address-event based stereo vision with bio-inspired silicon retina imagers". In: *Advances in Theory and Applications of Stereo Vision*, pp. 165–188.

Kottke, Dane P and Ying Sun (1994). "Motion estimation via cluster matching". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 16.11, pp. 1128–1132.

Kruppa, E (1913). *Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung*. Hölder.

Kuhn, Harold W (1955). "The Hungarian method for the assignment problem". In: *Naval Research Logistics* 2.1-2, pp. 83–97.

Kuo, Cheng-Hao and Ram Nevatia (2011). "How does person identity recognition help multi-person tracking?" In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1217–1224.

Lenz, Philip, Julius Ziegler, Andreas Geiger, and Martin Roser (2011). "Sparse scene flow segmentation for moving object detection in urban environments". In: *IEEE Intelligent Vehicles Symposium (IV)*, pp. 926–932.

Leutenegger, Stefan, Margarita Chli, and Roland Siegwart (2011). "BRISK: Binary robust invariant scalable keypoints". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 2548–2555.

Lin, Kuen-Han and Chieh-Chih Wang (2010). "Stereo-based simultaneous localization, mapping and moving object tracking". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3975–3980.

Liu, Wei, Dragomir Anguelov, et al. (2016). "Ssd: Single shot multibox detector". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 21–37.

Longuet-Higgins, H Christopher (1981). "A computer algorithm for reconstructing a scene from two projections". In: *Nature* 293.5828, p. 133.

Lowe, David G (1999). "Object recognition from local scale-invariant features." In: *IEEE International Conference on Computer Vision (ICCV)*.

Markley, F Landis (1988). "Attitude determination using vector observations and the singular value decomposition". In: *Journal of the Astronautical Sciences* 36.3, pp. 245–258.

Mateus, Diana and Radu Horaud (2007). "Spectral Methods for 3-D Motion Segmentation of Sparse Scene-Flow". In: *IEEE Workshop on Motion and Video Computing*, pp. 14–14.

Memin, E. and P. Perez (May 1998). "Dense estimation and object-based segmentation of the optical flow with robust techniques". In: *IEEE Transactions on Image Processing* 7.5, pp. 703–719.

Menze, Moritz and Andreas Geiger (2015). "Object scene flow for autonomous vehicles". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3061–3070.

Milan, Anton, Stefan Roth, and Konrad Schindler (2013). "Continuous energy minimization for multitarget tracking". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36.1, pp. 58–72.

Milan, Anton, Laura Leal-Taixé, Ian D. Reid, Stefan Roth, and Konrad Schindler (2016). *MOT16: A Benchmark for Multi-Object Tracking.* arXiv: 1603.00831 [cs.CV].

Milan, Anton, S Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler (2017). "Online multi-target tracking using recurrent neural networks". In: *AAAI Conference on Artificial Intelligence.*

Milella, Annalisa and Roland Siegwart (2006). "Stereo-based ego-motion estimation using pixel tracking and iterative closest point". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 21–21.

Mitrokhin, Anton, Cornelia Fermüller, Chethan Parameshwara, and Yiannis Aloimonos (2018). "Event-based moving object detection and tracking". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9.

Mitzel, Dennis, Esther Horbert, Andreas Ess, and Bastian Leibe (2010). "Multi-person tracking with sparse detection and continuous segmentation". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 397–410.

Moravec, Hans Peter (1980). "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover". PhD thesis. Stanford, CA, USA.

Mourikis, Anastasios I and Stergios I Roumeliotis (2007). "A multi-state constraint Kalman filter for vision-aided inertial navigation". In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3565–3572.

Mueggler, Elias, Basil Huber, and Davide Scaramuzza (2014). "Event-based, 6-DOF pose tracking for high-speed maneuvers". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2761–2768.

Mueggler, Elias, Guillermo Gallego, and Davide Scaramuzza (2015). *Continuous-time trajectory estimation for event-based vision sensors.* Tech. rep.

Mueggler, Elias, Chiara Bartolozzi, and Davide Scaramuzza (2017). "Fast Event-based Corner Detection." In: *British Machine Vision Conference.*

Newcombe, Richard A, Steven J Lovegrove, and Andrew J Davison (2011). "DTAM: Dense tracking and mapping in real-time". In: *international conference on computer vision (ICCV)*, pp. 2320–2327.

Newman, Paul and Kin Ho (2005). "SLAM-loop closing with visually salient features". In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 635–642.

Ng, Andrew Y, Michael I Jordan, and Yair Weiss (2002). "On spectral clustering: Analysis and an algorithm". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 849–856.

Nieuwenhuis, Claudia, Eno Töppe, and Daniel Cremers (2013). "A survey and comparison of discrete and continuous multi-label optimization approaches for the Potts model". In: *International Journal of Computer Vision (IJCV)* 104.3, pp. 223–240.

Nistér, David (2004). "An efficient solution to the five-point relative pose problem". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 26.6, pp. 756–777.

Nistér, David, Oleg Naroditsky, and James Bergen (2004). "Visual odometry". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ochs, P., J. Malik, and T. Brox (June 2014). "Segmentation of Moving Objects by Long Term Video Analysis". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36.6, pp. 1187–1200.

Ortin, Diego and José María Martínez Montiel (2001). "Indoor robot motion based on monocular images". In: *Robotica* 19.3, pp. 331–342.

Oth, Luc, Paul Furgale, Laurent Kneip, and Roland Siegwart (2013). "Rolling shutter camera calibration". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1360–1367.

Piccardi, Massimo (2004). "Background subtraction techniques: a review". In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 4, pp. 3099–3104.

Qiu, K., T. Qin, W. Gao, and S. Shen (Aug. 2019). "Tracking 3-D Motion of Dynamic Objects Using Monocular Visual-Inertial Sensing". In: *IEEE Transactions on Robotics (T-RO)* 35.4, pp. 799–816.

Quigley, Morgan, Ken Conley, et al. (May 2009). "ROS: an open-source Robot Operating System". In: *ICRA Workshop on Open Source Software*.

Quiroga, Julian, Thomas Brox, Frédéric Devernay, and James Crowley (2014). "Dense semi-rigid scene flow estimation from rgbd images". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 567–582.

Redmon, Joseph and Ali Farhadi (2017). "YOLO9000: better, faster, stronger". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7263–7271.

Reid, Donald (1979). "An algorithm for tracking multiple targets". In: *TAC* 24.6, pp. 843–854.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 91–99.

Rosten, Edward and Tom Drummond (2006). "Machine learning for high-speed corner detection". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 430–443.

Rother, Carsten, Vladimir Kolmogorov, and Andrew Blake (2004). "Grabcut: Interactive foreground extraction using iterated graph cuts". In: *ACM Transactions on Graphics*. Vol. 23. 3, pp. 309–314.

Roussos, Anastasios, Chris Russell, Ravi Garg, and Lourdes Agapito (2012). "Dense multibody motion estimation and reconstruction from a handheld camera". In: *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 31–40.

Rublee, Ethan, Vincent Rabaud, Kurt Konolige, and Gary Bradski (2011). "ORB: An efficient alternative to SIFT or SURF". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 2564–2571.

Runz, M., M. Buffier, and L. Agapito (Nov. 2018). "MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects". In: *IEEE International Symposium on Mixed and Augmented Reality*, pp. 10–20.

Rünz, Martin and Lourdes Agapito (2017). "Co-Fusion: Real-time Segmentation, Tracking and Fusion of Multiple Objects". In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4471–4478.

Ryoo, Michael S and Jake K Aggarwal (2008). "Observe-and-explain: A new approach for multiple hypotheses tracking of humans and objects". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Sabzevari, Reza and Davide Scaramuzza (2016). "Multi-body motion estimation from monocular vehicle-mounted cameras". In: *IEEE Transactions on Robotics (T-RO)* 32.3, pp. 638–651.

Scaramuzza, Davide (2011). "1-point-ransac structure from motion for vehicle-mounted cameras by exploiting non-holonomic constraints". In: *International Journal of Computer Vision (IJCV)* 95.1, pp. 74–85.

Schindler, Konrad, U James, and Hanzi Wang (2006). "Perspective n-view multibody structure-and-motion through model selection". In: *European Conference on Computer Vision*. Springer, pp. 606–619.

Schraml, S., A. N. Belbachir, N. Milosevic, and P. Schön (May 2010). "Dynamic stereo vision system for real-time tracking". In: *IEEE International Symposium on Circuits and Systems*, pp. 1409–1412.

Shi, Jianbo and Jitendra Malik (1997). *Motion segmentation and tracking using normalized cuts*.

Shi, Jianbo and Carlo Tomasi (1993). *Good features to track*. Tech. rep.

Shu, Guang, Afshin Dehghan, Omar Oreifej, Emily Hand, and Mubarak Shah (2012). "Part-based multiple-person tracking with partial occlusion handling". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1815–1821.

Sim, Kristy and Richard Hartley (2006). "Removing outliers using the L\infty norm". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1, pp. 485–494.

Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman (2014). "Learning local feature descriptors using convex optimisation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36.8, pp. 1573–1585.

Stauffer, Chris (2003). "Estimating tracking sources and sinks". In: *IEEE Computer Vision and Pattern Recognition Workshop (CVPR) Workshop*. Vol. 4, pp. 35–35.

Stoffregen, Timo and Lindsay Kleeman (2018). "Simultaneous optical flow and segmentation (SOFAS) using dynamic vision sensor". In: *arXiv:1805.12326 [cs.CV]*.

Stühmer, Jan, Stefan Gumhold, and Daniel Cremers (2010). "Real-time dense geometry from a handheld camera". In: *Joint Pattern Recognition Symposium*. Springer, pp. 11–20.

Sturm, J., N. Engelhard, F. Endres, W. Burgard, and D. Cremers (Nov. 2012). "A Benchmark for the Evaluation of RGB-D SLAM Systems". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Sugimura, Daisuke, Kris M Kitani, Takahiro Okabe, Yoichi Sato, and Akihiro Sugimoto (2009). "Using individuality to track individuals: Clustering individual trajectories in crowds using local appearance and frequency trait". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1467–1474.

Tang, Tim Yuqing, David Juny Yoon, and Timothy D Barfoot (Apr. 2019). "A white-noise-on-jerk motion prior for continuous-time trajectory estimation on SE (3)". In: *IEEE Robotics and Automation Letters (RA-L)* 4.2, pp. 594–601.

Tomasi, Carlo and Takeo Kanade (1990). "Shape and motion without depth". In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 91–95.

Tomasi, Carlo and Takeo Kanade (1991). *Detection and tracking of point features*. Tech. rep.

Tomasi, Carlo and Takeo Kanade (1992). "Shape and motion from image streams under orthography: a factorization method". In: *International Journal of Computer Vision (IJCV)* 9.2, pp. 137–154.

Tordoff, Ben J and David W Murray (2005). "Guided-MLESAC: Faster image transform estimation by using matching priors". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 27.10, pp. 1523–1535.

Torr, Philip HS (1998). "Geometric motion segmentation and model selection". In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 356.1740, pp. 1321–1340.

Torr, Philip HS and Andrew Zisserman (2000). "MLESAC: A new robust estimator with application to estimating image geometry". In: *Computer Vision and Image Understanding* 78.1, pp. 138–156.

Triggs, Bill, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon (1999). "Bundle adjustment—a modern synthesis". In: *International workshop on vision algorithms*. Springer, pp. 298–372.

Tron, Roberto and René Vidal (2007). "A benchmark for the comparison of 3-D motion segmentation algorithms". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Valgaerts, Levi, Andrés Bruhn, Markus Mainberger, and Joachim Weickert (2012). "Dense versus sparse approaches for estimating the fundamental matrix". In: *International Journal of Computer Vision (IJCV)* 96.2, pp. 212–234.

Valin, J., F. Michaud, J. Rouat, and D. Letourneau (Nov. 2003). "Robust sound source localization using a microphone array on a mobile robot". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2, 1228–1233 vol.2.

Vasco, Valentina, Arren Glover, and Chiara Bartolozzi (2016). "Fast event-based Harris corner detection exploiting the advantages of event-driven cameras". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4144–4149.

Vasco, Valentina, Arren Glover, et al. (2017). "Independent motion detection with event-driven cameras". In: *IEEE International Conference on Advanced Robotics*, pp. 530–536.

Vedula, Sundar, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade (1999). "Three-dimensional scene flow". In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 2, pp. 722–729.

Vidal, Antoni Rosinol, Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza (2018). "Ultimate SLAM? Combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios". In: *IEEE Robotics and Automation Letters (RA-L)* 3.2, pp. 994–1001.

Von Luxburg, Ulrike (2007). "A tutorial on spectral clustering". In: *Statistics and computing* 17.4, pp. 395–416.

Wahba, Grace (1965). "A least squares estimate of satellite attitude". In: *SIAM review* 7.3, pp. 409–409.

Wang, Chieh-Chih, Charles Thorpe, Sebastian Thrun, Martial Hebert, and Hugh Durrant-Whyte (2007). "Simultaneous localization, mapping and moving object

tracking". In: *The International Journal of Robotics Research (IJRR)* 26.9, pp. 889–916.

Wang, John YA and Edward H Adelson (1994). "Representing moving images with layers". In: *IEEE Transactions on Image Processing* 3.5, pp. 625–638.

Wang, Qing Hua, Teodor Ivanov, and Parham Aarabi (2004). "Acoustic robot navigation using distributed microphone arrays". In: *Information Fusion* 5.2, pp. 131–140.

Wedel, Andreas, Annemarie Meißner, Clemens Rabe, Uwe Franke, and Daniel Cremers (2009). "Detection and segmentation of independently moving objects from dense scene flow". In: *Computer Vision and Pattern Recognition Workshop (CVPR) Workshop on Energy Minimization Methods*. Springer, pp. 14–27.

Wikimedia Commons (2003). *NASA Mars Rover*. URL: `https://commons.wikimedia.org/wiki/File:NASA_Mars_Rover.jpg`.

Wikimedia Commons (2011). *iRobot Roomba*. URL: `https://commons.wikimedia.org/wiki/File:IRobot_Roomba_780.jpg`.

Wikimedia Commons (2017). *A Waymo self-driving car on the road in Mountain View*. URL: `https://commons.wikimedia.org/wiki/File:Waymo_self-driving_car_front_view.gk.jpg`.

Wikimedia Commons (2018). *Quadcopter camera drone in flight*. URL: `https://commons.wikimedia.org/wiki/File:Quadcopter_camera_drone_in_flight.jpg`.

Wu, Bo and Ram Nevatia (Nov. 2007). "Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet based Part Detectors". In: *International Journal of Computer Vision (IJCV)* 75.2, pp. 247–266.

Wu, Ying, Zhengyou Zhang, Thomas S. Huang, and John Y. Lin (2001). "Multibody Grouping via Orthogonal Subspace Decomposition". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2, pp. 252–257.

Xu, Binbin, Wenbin Li, et al. (2019). "Mid-fusion: Octree-based object-level multi-instance dynamic SLAM". In: *IIEEE nternational Conference on Robotics and Automation (ICRA)*, pp. 5231–5237.

Yan, Jingyu and Marc Pollefeys (2006). "A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 94–106.

Yang, B., C. Huang, and R. Nevatia (June 2011). "Learning affinities and dependencies for multi-target tracking using a CRF model". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1233–1240.

Yang, Bo and Ram Nevatia (2012). "Online Learned Discriminative Part-Based Appearance Models for Multi-human Tracking". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 484–498.

Yi, Kwang Moo, Eduard Trulls, Vincent Lepetit, and Pascal Fua (2016). "Lift: Learned invariant feature transform". In: *European Conference on Computer Vision (ECCV)*. Springer, pp. 467–483.

Yilmaz, Alper, Omar Javed, and Mubarak Shah (2006). "Object tracking: A survey". In: *ACM Computing Surveys* 38.4, p. 13.

Zhang, Li, Yuan Li, and Ramakant Nevatia (2008). "Global data association for multi-object tracking using network flows". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Zhang, Zhengyou (2000). "A flexible new technique for camera calibration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 22.

Zhang, Zichao and Davide Scaramuzza (2018). "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7244–7251.

Zhou, Tinghui, Matthew Brown, Noah Snavely, and David G. Lowe (June 2017). "Unsupervised Learning of Depth and Ego-Motion From Video". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.