MOTION SEGMENTATION

SEGMENTATION OF INDEPENDENTLY MOVING OBJECTS IN VIDEO

A Dissertation Presented

by

PIA KATALIN BIDEAU

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2019

College of Information and Computer Sciences

MOTION SEGMENTATION

SEGMENTATION OF INDEPENDENTLY MOVING OBJECTS IN VIDEO

A Dissertation Presented

by

PIA KATALIN BIDEAU

Approved as to style and content by:

Erik Learned-Miller, Chair

Subhransu Maji, Member

Evangelos Kalogerakis, Member

David Huber, Member

James Allan, Chair of the Faculty
College of Information and Computer Sciences

To my brother Til and my sister Leonie.

ACKNOWLEDGMENTS

I would like to express my gratitude to everybody who joined me on my way towards this dissertation, an important time for me.

First I would like to thank my advisor Erik Learned-Miller, who was a wonderful support on this way - personally and also taught me with passion how to conduct high quality research. We have had intense discussions continuously fighting for the same goal. Thank you. Then I would like to thank my committee Subhransu Maji, Evangelos Kalogerakis and David Huber for supporting my work and giving intermediate feedback which significantly contributed to the final outcome of this work. Cordelia Schmid and Karteek Alahari have been a great guidance during my internship at INRIA in Grenoble during fall 2018. Thank you for good collaboration. Thanks to Mark Corner, Zhipeng Tang and Fabien Delattre for interesting discussions and exploring research areas combining computer vision and mobile systems.

Not forgetting all my lab colleagues and friends Aruni, SouYoung, Huaizu, Tsung-Yu, Hang, Matheus, Archan, Jong-Chyi and Ashish, who made this time a lot of fun. Thanks for company on many hikes, fun evenings, interesting discussions and joining me for contra dances. Nina and Wayne have been wonderful friends, housemates and landlords during these years of my PhD. Thank you so much!

ABSTRACT

MOTION SEGMENTATION

SEGMENTATION OF INDEPENDENTLY MOVING OBJECTS IN VIDEO

DECEMBER 2019

PIA KATALIN BIDEAU

B.Sc., UNIVERSITY OF APPLIED SCIENCE DÜSSELDORF

M.Sc., RUHR UNIVERSITY BOCHUM

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Erik Learned-Miller

The ability to recognize motion is one of the most important functions of our visual system. Motion allows us both to recognize objects and to get a better understanding of the 3D world in which we are moving. Because of its importance, motion is used to answer a wide variety of fundamental questions in computer vision such as: (1) Which objects are moving independently in the world? (2) Which objects are close and which objects are far away? (3) How is the camera moving?

My work addresses the problem of moving object segmentation in unconstrained videos. I developed a probabilistic approach to segment independently moving objects [4] in a video sequence, connecting aspects of camera motion estimation, relative depth and flow statistics. My work consists of three major parts:

- Modeling motion using a simple (rigid) motion model strictly following the principles of perspective projection and segmenting the video into its different motion components by assigning each pixel to its most likely motion model in a Bayesian fashion. [5]

- Combining piecewise rigid motions to more complex, deformable and articulated objects, guided by learned semantic object segmentations. [7]

- Learning highly variable motion patterns using a neural network trained on synthetic (unlimited) training data. Training data is automatically generated strictly following the principles of perspective projection. In this way well-known geometric constraints are precisely characterized during training to learn the principles of motion segmentation rather than identifying well-known structures that are likely to move. [6]

This work shows that a careful analysis of the motion field not only leads to a consistent segmentation of moving objects in a video sequence, but also helps us understand the scene geometry of the world we are moving in.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

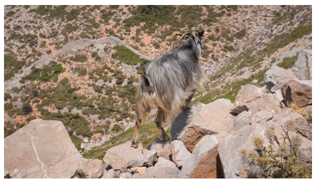
INTRODUCTION

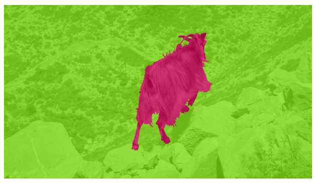
The human visual system has an incredible ability to detect, analyze and act on motion perceived as motion stimuli (light changes) on the human eye's retina. These motion stimuli might arise when the observer himself moves or an object moves in the world. Of course both of these situations might happen at once, but often it is helpful to consider these cases separately first.

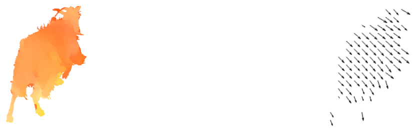
To get a sense for the complexity of the task of motion processing happening in the human visual cortex let's consider three situations that result in three very different stimuli on the retina.

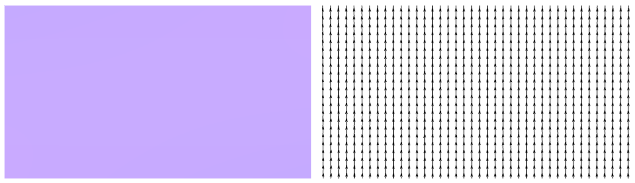
Consider an observer looking at a stationary scene in which one object is moving (see Figure 4.1(c)). This might be a person walking in the world, which is pictured as a person moving across the observer's retina. This case is simple to interpret. The perceived motion on the retina exactly corresponds to the motion in the world.

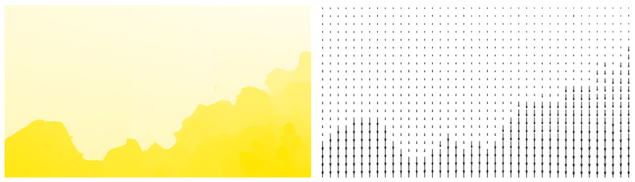
Now let's consider the second case, a moving observer looking at a stationary scene (see Figure 4.1(d-f)). The observer might be just turning his head (rotating), walking through world (translating) or both. If the observer only turns his head the entire image projected onto the retina moves across the retina according to the observer's motion. If the observer is walking through the world, the pictured motion on the retina is far more complex. The change on the retina highly depends on the scene geometry that is pictured. Objects that are close, for example a nearby tree, lead to a "faster" motion than farther objects. Objects at the horizon create no change on the retina. Observing this type of motion on the retina might be interpreted in several ways: (1) the entire world is moving while the observer stands still, (2) if

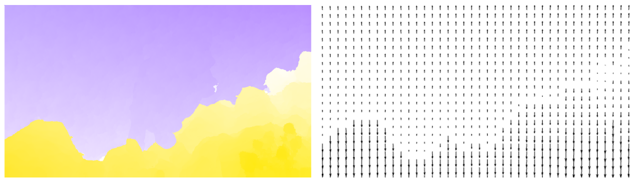

(a) frame


(b) motion segmentation


(c) no observer motion - moving goat


(d) observer rotation - stationary scene


(e) observer translation - stationary scene


(f) observer rotation and translation - stationary scene

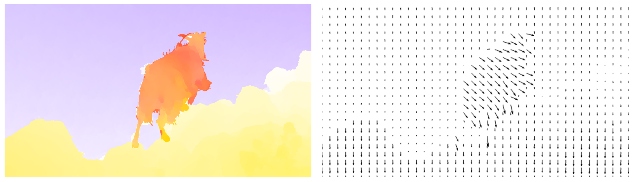

(g) observer rotation and translation - moving goat

Figure 1: What is moving? It is extremely challenging to distinguish between moving and stationary objects from optical flow directly. Three different scenarios are pictured: (1) stationary observer watches a moving goat (c), (2) observer is moving while watching a stationary scene without moving objects (d-f) and (3) observer is moving while watching a stationary scene with a moving goat (g).

we observe different “speeds” of motion, some objects might be moving faster than their environment or alternatively, objects might be located at different depth, (3) the observer moves while the world is standing still. In most cases the last option might appear to be the most reasonable interpretation of the observed motion on the retina.

The third situation to consider is a moving observer and a moving object in the world. The motion due to the observer as well as the motion due to the object results in a motion of the pictured scene on the retina as described in the previous two cases. However if we track the moving object with our eyes, the object will not create any motion on the retina. On the retina, the object will appear to be stationary whereas the world appears to be moving. However, in reality, the world is stationary and the object is moving.

This means that just because something is moving across our retina, doesn’t mean that it is actually moving in the world. Conversely, just because something appears to be stationary on our retina doesn’t mean that it is actually stationary in the world.

How do humans know what is moving in the world and what is not?

This question is the subject of current research in many different areas such as neuroscience, psychology, and computer science [10, 9, 2, 122, 26, 40, 8, 35].

In the human visual system motion perception begins on the human eye’s retina. Photoreceptor cell’s in the retina respond to light changes, which often correspond to motion but also many other possible causes. One can think of a photoreceptor as a pixel on a camera sensor. At this stage visual information is simply received. Further information processing of the visual signal is done in different areas of the visual cortex [10, 9, 2], where pure visual information is combined with additional information from the vestibular system and eye movement signals to process and interpret the perceived visual information correctly [122, 26].

The strong ability of humans to detect motion can be especially highlighted by tackling the task of spotting camouflaged animals in nature. In those cases, where environment and object share similar appearance, it is quite challenging to spot the object solely based on its appearance, instead we need additional cues such as motion to reliable detect the object. Figure 2 shows two examples of camouflaged animals in the wild. The goatweed leafwing (Figure 2: left) is a butterfly which can be often found sitting on the ground, with wings folded over its back. In this position it mimics almost perfectly a dry leaf on the ground. This well developed adaptation ability allows the goatweed leafwing to rest on the ground quite safely. Natural enemies are not able to detect the goat leafwing as long as it is not moving. The other scenario shows an owl sitting in a tree. The owl's feathering mimics the tree's bark, thus the owl can sit quietly and look for prey without being seen. These are two scenarios showing animals that are almost invisible as long as they stand still, however due to the great sensitivity to motion of our visual system we are able to spot them as soon as they move. For this reason we selected videos of camouflaged animals and formed a new data set - *camouflaged animals*, the purpose of which is to evaluate the ability of motion segmentation in challenging tasks where the object's appearance is a rather weak clue to spot and segment the moving object correctly in video [5].

This dissertation presents an approach that aims to accurately interpret the perceived motion on the retina. Of course we are not able to receive an image directly from our eye, that we could process to segment moving objects. Instead we help our self by using video sequences taken by a camera and methods to estimate the motion field between two consecutive frames. The task is then to develop a method that correctly interprets the observed motion field regardless of the complexity of the scene's geometry and segments objects that are moving in the world.

The document is organized as follows Chapter 1 defines the problem of motion segmentation and makes a distinction to other closely related topics such as fore-

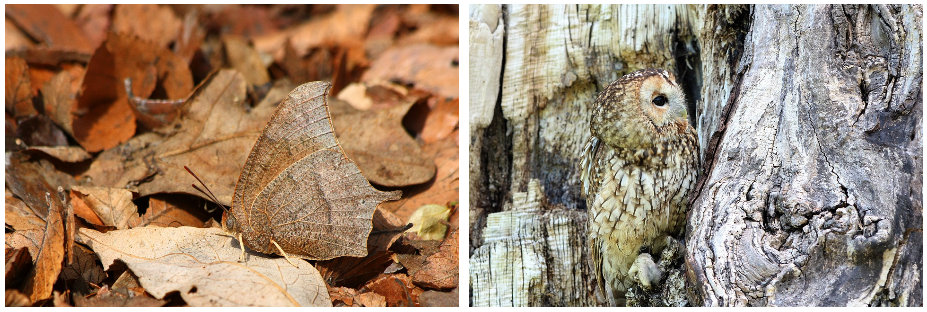


Figure 2: The challenge of object detection in scenarios of well-camouflage. Camouflage is one of multiple adaptation mechanisms found in nature. If an animal is well camouflaged, it is almost impossible to spot them just based on their appearance. This makes it possible for animals to live in environments together with their natural enemies. Left: goatweed leafwing *(Accessed 22 May, 2018.* `https://c2.staticflickr.com/6/5180/5538111287_44b43d58c2_o.jpg` *)*, right: owl *(Accessed 22 May, 2018.* `https://www.maxpixel.net/static/photo/1x/Tree-Nature-Owl-Wildlife-Camouflaged-Bird-Prey-1738971.jpg` *)*

ground/background segmentation or video object segmentation. Chapter 2 reviews closely related literature dealing with the problem of motion segmentation as well as motion estimation. Basic background information about the motion field are provided in Chapter 3. A probabilistic approach for the motion segmentation task, the segmentation of a motion field into its rigid motion components, is presented in Chapter 4.1. Building on this work Chapter 4.2 introduces a more advanced approach that combines piecewise rigid motions to more complex, deformable and articulated objects, guided by learned semantic segmentations. Chapter 5.1 presents an approach to learn motion segmentation in a self-supervised manner. A procedure to automatically generate an unlimited amount of synthetic training data while strictly following the principles of perspective projection is proposed. The goal of this training data is to precisely characterize well-known geometric constraints during training to learn the principles of motion segmentation rather than identifying well-known structures that are likely to move. Motion estimation as a fundamental sub-task of motion segmentation also finds its application in other related areas such as mobile computing and research related to UAVs, details are described in Chapter 6. I conclude my work with a discussion and an outlook into possible future research directions in Chapter 7.

CHAPTER 1
PROBLEM STATEMENT

The goal of my thesis is to develop a fully automatic motion segmentation system, that works in diverse scenarios showing different scene geometries (objects located at different depths) and irrespectively of the object's camouflage.

Given a video of a natural scene, this can be for example a video showing road traffic, animals in nature or people, the goal is to segment all *independently* moving objects. Besides the motion of the object itself there might be image motion [1] due to the observer's motion (camera motion). As shown in Figure 1 the optical flow field couples motion information and scene depth in case of present camera motion, which often makes the distinction between moving object and static environment a challenging task. Related topics such as depth estimation and camera motion estimation will be examined in the context of motion segmentation.

In Section 1.1 we define the problem of motion segmentation. *What exactly should be segmented?* This appears to be not always an obvious question. Should leaves wiggling in the wind be segmented as a moving object? The following Section provides a guideline for motion segmentation and distinguishes between the binary (Section 1.1.1) and the more general version of multi-label motion segmentation (Section 1.1.2). Several situations still appear to debatable whether an object (or part of

[1]We refer to image motions, if the environment or the objects move over the image plane. Image motion might occur due to object motion or due to the observer's motion. Image motion does not necessarily require true 3D motion in the world. We might observe image motion in areas of static environment solely due to the observer's motion and not due to a 3D motion of the environment in the world.

an object) should be segmented as moving or not, those cases are discussed in the remainder of Section 1.1.

1.1 Definition of motion segmentation

General image segmentation is the subject of current research in computer vision and machine learning. The goal is to produce k connected regions by pooling pixels, which share one or multiple common criteria. k refers to the number of predefined labels the image should be segmented in. Those criteria (labels) might be for example color, texture or motion. Motion segmentation groups pixels which share the same motion.

Object motion as well as the observer's motion, can be observed as a pixel displacement on the image plane. If observed motions are tiny it is often hard to decide whether those motions actually belong to a moving object or not. The border between stationary objects and moving objects can be quite fluent. If a person is walking there are short time periods where one foot is moving, but the other foot stands still. Do we want to segment just the part of the person, that is moving or the entire moving person? Those difficulties *(1) are those pixels moving or not?* and *(2) Is just part of the object moving or the entire object?* turns creating a ground truth to evaluate motion segmentation algorithms into a challenging problem.

Since the criterion "motion" (referring to all kinds of motion including motions of just object parts) alone is hard to evaluate we will instead refer to object motion. This is a useful and practical simplification. Using this simplification the entire object needs to be segmented even if just part of it is moving. If just one foot of a walking person is moving, we'll segment the entire object or if just the shovel of a digger is moving, we'll segment the entire digger - not the shovel only. This decision is for sure debatable and whether it makes sense depends upon the higher-level task motion information is used for.

In the following we'll address motion segmentation as a binary segmentation problem and extend this task to the more general task of motion segmentation where it is distinguished between individual objects that are independently moving.

1.1.1 Definition: Binary Motion Segmentation

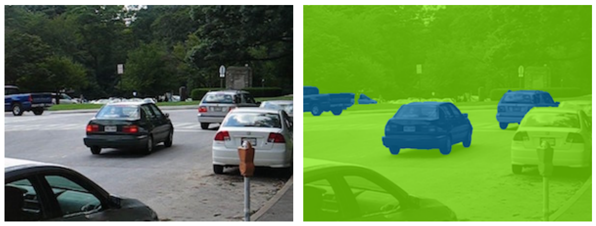


Figure 1.1: Binary motion segmentation. Every pixel is given one of two labels: static environment or moving objects. Left to Right: original image, correct binary segmentation

Binary motion segmentation segments each video frame into two components. It is distinguished between (1) static environment and (2) independently moving objects moving differently than the camera motion. The environment itself is not moving, however the pixels describing static environment can show a displacement from frame t to $t + 1$ due to the camera's motion. We define motion segmentation as follows:

(I) Every pixel is given one of **two labels**: static environment or moving objects.

(II) If only part of an object is moving (like a moving person with a stationary foot), the **entire object** should be segmented.

(III) **All freely moving objects** (not just one) should be segmented, but nothing else. We do not considered tethered objects such as trees to be freely moving.

(IV) Stationary objects are not segmented, even when they moved before or will move in the future. We consider segmentation of previously moving objects to be *tracking*. Our focus is on segmentation by motion analysis.

1.1.2 Definition: Motion Segmentation

Motion segmentation is a task that groups pixels sharing the same motion. Just following the criteria of motion it's often not possible to distinguish between different objects, since those might move together. Due to their shared motion they form

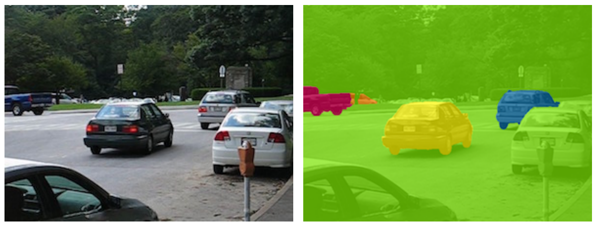


Figure 1.2: Motion segmentation. Every pixel is given one of k labels: k is the number of independently moving objects and static environment. Left to right: original image, motion segmentation

a common pixel group which is segmented together. How to segment a video into k independently moving objects is clear if the moving objects are not touching in 3D and clearly move independently like the four cars shown in Figure 1.2. However there are a few cases where segmenting a frame into k independently moving objects is challenging: *(1) two persons walking hand in hand, (2) a person jumping onto a carriage, which is pulled by a horse or (3) laundry fluttering in the wind.* After defining motion segmentation as a general segmentation problem of a video into k moving objects, we'll take a closer look at those broader cases.

We build upon our previous definition of binary motion segmentation:

(I) Every pixel is given one of k **labels**. k is the number of observed independently moving objects and static environment.

(II) If only part of an object is moving (like a moving person with a stationary foot), the **entire object** should be segmented.

(III) **All freely moving objects** (not just one) should be segmented, but nothing else. We do not considered tethered objects such as trees to be freely moving.

(IV) Stationary objects are not segmented, even when they moved before or will move in the future. We consider segmentation of previously moving objects to be *tracking*. Our focus is on segmentation by motion analysis.

(V) If objects are moving together and are connected in 3D, they should be segmented together since they share the same motion. A person carrying a basket - here the person and the basket are forming one common motion component.

(VI) If objects are connected in 3D but move independently from each other, they should be segmented separately since they do not share the same motion. A person walking with a leashed dog. The person and the dog are connected in 3D, but move independently from each other. Thus the person and the dog get their own motion segment.

(VII) An object which is not moving (but could be connected in 3D with an other moving object) should be not segmented unless it is considered an integral part of an other object. Is a person sitting on a stationary chair, then the chair should not be segmented. However if the chair is moving with that person (for example a wheelchair), then the person should be segmented together with the chair following rule (V).

Based on the provided definition (I-VII) of motion segmentation, we discuss the three previously mentioned challenging cases.

- *Two persons walking hand in hand* The two persons are connected in 3D and move pretty much independently even if there might be some influence from one person to the other. Basically there are two independent ”motion sources” thus we observe two independent motions and label both persons separately according to (VI). This is a challenging case since as long as the persons are connected in 3D it is hard to judge based on the video whether this is one complex motion component or two simpler and probably similar motion components.

- *A person jumping onto a carriage, which is pulled by a horse* We start with carriage and horse. This situation corresponds to (V). The horse is pulling the carriage such that the carriage is moving with the horse. Carriage and horse form together one independently moving object. Now we consider the man jumping onto the carriage. This situation gets significantly more tricky. The man is for sure moving independently at the beginning and thus rule (VI) can be applied however in the later run if the man doesn't move significantly independent anymore - sitting on the carriage, the man can be considered to be segmented with the carriage and horse (V). This is a very difficult case and can be interpreted differently, however we consider together moving objects as one moving object, if they move as a whole independently.

- *laundry fluttering in the wind.* Even though laundry does not belong to a tethered object such as the leaves of a tree, which a wiggling in the wind, we do not consider laundry as an independently moving object. This situation is quite similar as described in (III). Thus laundry fluttering in the wind is not considered to be a freely moving object.

CHAPTER 2
LITERATURE REVIEW

Many works tackling the problem of motion segmentation focus on *binary motion segmentation*, where pixels are classified as either moving or part of the background, but no distinction is made between separate moving objects [5, 72, 81, 22]. Others [109, 48, 25] address *multi-label motion segmentation*, where a separate label is given to each independently moving object. In the following sections, we do not distinguish between binary and multi-label motion segmentation.

In most cases information about motion is derived from matched pixels across consecutive frames. This could be in the form of either *sparse point trajectories* or *optical flow*. We review four fundamentally different approaches to tackle the problem of motion segmentation. All of them rely on point-to-point correspondences (matched pixels) from one frame to an other. We start with methods based on motion trajectories (Section 2.1.1), followed by methods based on projective geometry (Section 2.1.2) and perspective projection (Section 2.1.3) concluding with the most recent approaches based on convolutional neural networks (Section 2.2) to learn general motion patterns throughout a video.

2.1 Classical approaches for motion segmentation

2.1.1 Methods based on motion trajectories

Methods based on *point trajectories* [48, 12, 25, 75, 47, 132, 101, 58] have shown good results for tracking and time consistent video segmentation. Point trajectories are either formed by tracked image features or dense optical flow. Feature tracks are produced by tracking sparse image features (key-points) across multiple frames, whereas optical flow is a dense estimate of the motion field between two consecutive frames. Trajectories typically end if the tracked feature moves outside of the image plane or gets occluded by a different object (this can happen due to object or camera motion), they can arise any time with a new detected image feature. Trajectories based on either feature tracks or optical flow are able to track motion patterns coherently over multiple frames and are thus very effective for motion segmentation. Trajectories sharing similar motion characteristics are grouped into coherent motion clusters describing the motion of a particular object.

Different approaches vary in defining typical motion characteristics based on which the trajectories are clustered. [132] propose to cluster trajectories based on geometric constraints (trajectories of the same motion lie in a manifold) and locality. In [47, 48] the segmentation problem is represented as a minimum cost multicut graph problem following underlying principles that trajectories belonging to the same motion share fundamental criteria like motion similarity, color similarity and spatial distance between trajectories. Objects are detected by clustering the trajectories and are tracked naturally which leads to a time consistent segmentation of a short video sequence.

Trajectories based on image features are mostly quite computational efficient since those are sparse motion representations other than than a dense optical flow field. But due to their sparsity obtained segmentations are sparse as well and need further post processing to turn these sparse segmentations into a dense video segmentation.

Trajectory based approaches reach their limit if scene understanding is necessary to segment a moving object correctly. Trajectories perfectly represent individual pixel displacements, however drawing a geometrically plausible conclusion based on more or less individual tracks rather than a coherent motion field remains challenging. Pixel displacements from one frame to the next are a function of depth and motion. Thus motion-trajectory based clustering methods often form clusters not only for independently moving objects, but also for objects at different depths. Methods based on occlusions [78, 109] are subject to similar depth-related problems. Trajectory-based methods are *non-causal*, since they require information of the entire video. To segment earlier frames, one must wait for trajectories which are computed over future frames.

2.1.2 Methods based on projective geometry

Projective geometry is an extension of the Euclidean and affine space and contains properties of perspective projection. Often projective geometry is used as a geometrical model to explain the properties of perspective projection. In Euclidean space valid transformations are restricted to the transformations of rotation and translation, thus Euclidean space is not sufficient anymore to model the more complex imaging process (perspective projection). Affine geometry adds transformations covering shearing and scaling. Projective geometry seems to be a reasonable extension of the Euclidean and affine geometry. Its is widely used as a mathematical formalism to describe the geometry of cameras and its associated transformations [112, 134, 123, 46, 45, 130, 118, 131]. Leading to elegant mathematics on one hand, the universality of projective geometry comes with several problems being inconsistent with the true physical world [36, 5].

Different from trajectory based motion segmentation approaches projective geometry methods analyze the optical flow between a pair of frames, grouping pixels into regions whose flow is consistent with various motion models consistent with projective geometry [112, 134, 123, 46, 45, 130, 131]. Torr [112] develops a sophisticated probabilistic model of optical flow, building a mixture model that explains an arbitrary number of rigid components within the scene. Interestingly, he assigns different types of motion models to each object based on model fitting criteria. His approach is fundamentally based on projective geometry rather than based directly on perspective projection equations. Zamalieva et al. [134] and Xun Xu et al. [131] present a combination of methods that rely on homographies and fundamental matrix estimation. The two methods have complimentary strengths, and the authors attempt to select among the best dynamically.

Horn has identified drawbacks of using projective geometry in such estimation problems and has argued that methods based directly on perspective projection are less prone to overfitting in the presence of noise [36]. In contrast to the quite universal approaches based on projective geometry there are methods based on perspective projection only. These methods focus more on realistic representations of the physical world rather than nice mathematical models.

All these methods relying on projective geometry perform well in cases of planar motion (motion obtained by a translating or rotating camera picturing a planar scene or a very distant scene, where effects of 3D parallax are negligible), however if the camera undergoes arbitrary translational or rotational motions and the scene shows a more complex geometry with objects located at different depth the motion field gets quiet complex and methods based ob projective geometry reach their limits.

2.1.3 Methods based on perspective projection

A human eye, a painter and a pin hole camera - all deal with the same task, which is the projection of the three-dimensional world onto a two-dimensional image plane. Artists and scientists like Albrecht Dürer, Leon Batista Alberti, Filippo Brunelleschi or Leonardo da Vinici - to name just a few, have made a significant contribution [87, 20, 21] to the current successes in computer vision. Understanding the process of image formation (perspective projection) is essential for most computer vision problems like optical flow estimation, ego-motion estimation or segmentation. Perspective geometry allows us to mathematically explain and model the process of how the three dimensional world is projected on to a just two-dimensional image plane. One of the key aspects of perspective projection is, that the larger the distance to the viewer the smaller appears the imaged object leading to an observation that parallel lines (in the euclidean space) that undergo the transformation of perspective projection lead to two lines that intersect in the vanishing point at the horizon on the image plane.

Approaches based on perspective projection [40, 5, 7, 72, 119, 135] are in general more accurate (in terms of model agreement to the physical world) than those based on projective geometry, since the latter omits certain constraints in modeling image transformations [36, 5]. Having a model that is confirm with the physical world might be especially critical for certain tasks where interaction with the physical world is required in a second step such as in robotics or autonomous driving. A commonly first step towards motion segmentation is to estimate ego-motion, which is equivalent to camera motion in video.

Widely used are approaches fro ego-motion estimation relying on epipolar geometry [116, 125, 105]. *Epipolar geometry [32]* describes the correspondence between image points originating from two different images taken from two different points of view at the same time - typically taken with stereo cameras. Camera calibration and the their relative position to each other is usually known.

Bundle adjustment is an alternative approach that jointly estimates 3D structure and camera poses and calibration via optimization procedures [113]. The optimization procedure is often formulated as nonlinear least square problem [30, 127, 11, 102, 53]. Different from epipolar geometry stereo vision is not required, instead structure is estimated from camera motion. Also often referred to as structure from motion (SfM). For stereo vision the relative camera position between the two stereo frames is known, however in case of bundle adjustment the relative camera motion between two consecutive frames is jointly estimated estimated together with 3D feature coordinates.

Unlike previous works on motion segmentation relying on either bundle adjustment or epipolar geometry, our work presented in this thesis incorporates an optimization procedure to estimate the motion field due to camera rotation, without directly reconstructing the three dimensional scene structure. The idea is to simplify the observed motion field by compensating for motion due camera rotation. Given a motion field formed by only camera translation and scene structure, all effects of camera rotation and the need for camera calibration can be completely eliminated. Further more the image motion *direction* is only determined by the translational camera motion direction and only shows minor influences by the scene depth (due to the optical flows noise, which is showing larger influence if image motion is small corresponding to either no camera motion or a distant scene). These fundamental characteristics make the translation only motion field attractive for the motion segmentation task.

In Chapter 4.1 we present a fully automatic motion segmentation method [5] based on optical flow. Following the geometry of perspective projection, a frame is segmented based on the optical flow's direction. Assuming that the underlying motion field magnitude is *equal* to the optical flow magnitude, we use the motion field magnitude to model the informativeness of the direction of each flow vector. In Chapter 4.2 the previous presented approach is extended by dealing with the unknown motion field magnitude in a Bayesian fashion, rather than assuming its value is equal to the flow magnitude[7]. This naturally leads to a confirmation of the previous statement made in Chapter 4.1, that small flow vectors are less informative and allows us to segment a video sequence into static environment and independently moving objects regardless the complexity of the scene structure.

2.2 Learning motion segmentation using convolutional neural networks

2.2.1 Supervised approaches for motion segmentation

Recent approaches as [110, 111, 42, 18, 19, 91, 117] use deep neural networks to learn characteristic motion patterns and produce binary motion masks distinguishing whether a pixel is moving or not. Most approaches propose a two-stream architecture [111, 42, 19] to separately process motion and appearance. [19] is the first fully learning based approach for spatio-temporal grouping. Other than previous approaches they segment individual object instances.

Theses approaches learn motion patterns given the optical flow, the raw video frames or optical flow together video frames. Rather than following the true physics of image formation, which are described by perspective projection, fully convolutional neural networks are able to learn high level motion patterns such as background motion or object motion. Without analyzing specifically in which direction the camera or an object moves these approaches are able to distinguish whether a pixel in an image belongs to a background motion pattern or not. This ability has the clear advantage of not being dependent upon technical camera parameters such as the focal length or image distortions due to various lens characteristics or constraints induced by technical parts of the camera (mechanical or electronic). Different lenses may lead to significant image distortions an extreme example is the fisheye lens. Exact parameters of a lens are provided in rare cases only, however we as humans are still able to detect independently moving objects quite reliable where as classical motion segmentation methods might fail completely. Technical constraints leading to unpredictable image distortions can be induced by camera sensors. A CMOS sensor records an image gradually over time (line by line), which results in wired looking image distortion not explainable by perspective projection.

General concerns of deep-learning based approaches are overfitting to a particular type of object category [19] and the lack of large amounts of training data [42]. Instead of learning typical objects categories that are likely to move one has to ensure that also never-before-seen objects are segmented based on their motion and regardless of what particular semantic name it happens to be associated with.

The other well known problem of learning based approaches is the lack of large amount of training data. This might enforce the problem of overfitting and makes the task of learning universal motion models a challenging task. Two possible ideas often used in latest approaches are either using synthetic training data from animated videos [110, 111] or relying on noisy optical flow [42] estimated by other algorithms [63, 107, 39, 99, 106]. The lack of sufficient training data rises the need to consider self-supervised approaches for the task of motion segmentation, some of which we will review in the following Section.

2.2.2 Self-supervised approaches for motion segmentation

High capacity convolutional neural networks [56] and large-scale labeled data [52, 62, 51] have lead to significant changes in computer vision. Vision tasks like object detection [28, 27, 95, 93, 94] or semantic segmentation [86, 85, 33] have experienced a great boost in performance, but requiring large amounts of labeled training data. Labeling training data is time consuming and expensive. However a significant portion of human learning happens actually unsupervised - without the need of labeled data. A child learns from observations as well as explorations and interactions with the world [29, 104]. These natural learning techniques are not restricted by the need of labels. How can one minimize the need of labeled training data? Self-supervised learning methods like [136, 74, 121, 44, 126] attempt to address this question and develop approaches that are able to learn from unlabeled data. Mostly a pretext task like colorization [136, 121], solving jigsaw puzzles [74] or predicting if a video plays forward or backwards [126] is explored to learn some fundamental underlying structure of the data before fine-tuning the network one a smaller data set for the actual task that has to be solved. There have been multiple approaches to tackle the motion segmentation task in an self-supervised or unsupervised manner [121, 6, 24, 138, 82, 69, 133]. Other than traditional self-supervised learning approaches, that use some hidden information present in the data as a learning signal, we propose a self-supervised learning approach using knowledge about the underlying physics behind the process of motion field formation. Given these very basic principles we generate training data to train a neural network for the motion segmentation tasks relying on motion information only.

[120, 138] attempt to incorporate knowledge about the physical image formation process into the learning pipeline of neural networks. They developed an fully unsupervised approach (nor requiring any labeled data) to learn depth, ego-motion and motion segmentation simultaneously by warping nearby images to the target image using single-view depth and multi-view pose estimation results.

Incorporating knowledge about the real world physics into the training procedure of a neural network is subject of current and is far from being solved yet [115].

CHAPTER 3
THE MOTION FIELD

Object motion or camera motion produce changes in images captured at different time. These changes lead to a motion field - a purely geometrical concept. The motion field assigns a velocity vector to every image point, which describes the displacement of each pixel. There are a few exceptions where true physical motions do not necessarily lead to image changes as for example a perfect sphere rotating under constant illumination. In those cases the true motion field doesn't correspond with it apparent (perceived) motion field. Table 3.1 shows a specific camera motion and the corresponding motion field. These motion fields represent a pure camera motion assuming continuous depth. We normalized the focal length, such that ($f = 1$). A normalized focal length leads to an exaggerated field of view. The motion field is represented with the Middlebury color coding [3], here the magnitude is visualized using color intensity and different angles are shown in different color. Besides the Middlebury color coding the vector plot of motion field is shown. A flow field of realistic scenes look much more complicated than the synthetically generated motion fields pictured in Table 3.1. To estimate the motion field of a realistic scene, containing objects at different depth, an typically an essential assumption is made. Brightness pattern in the image move accordingly to the object motion [37]. Motion corresponds to an apparent motion of the brightness pattern. This apparent motion pattern is called the optical flow [37]. We provide here background information on motion fields that occur due to camera rotation, translation or a combination of both.

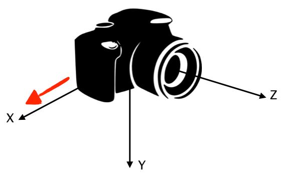
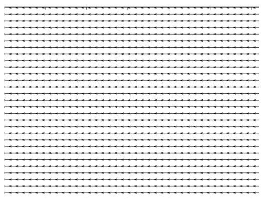
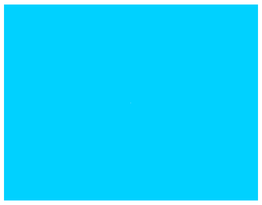
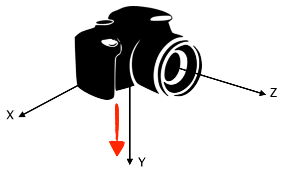
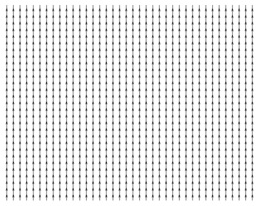
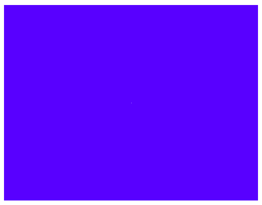
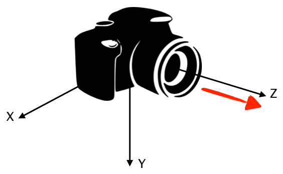
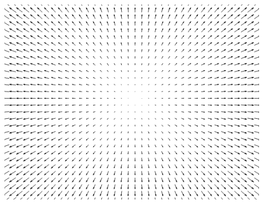
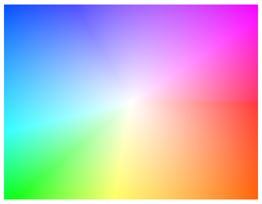
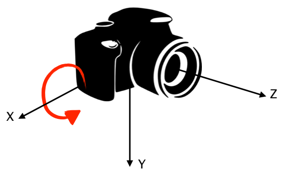
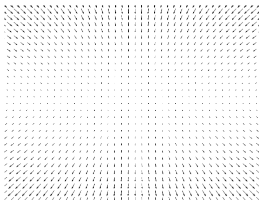
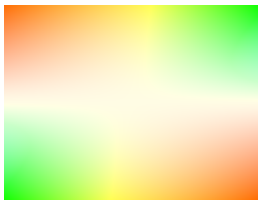
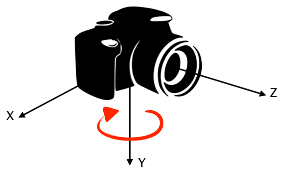
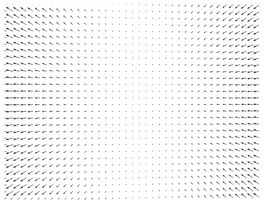
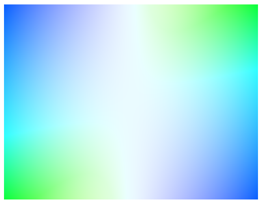
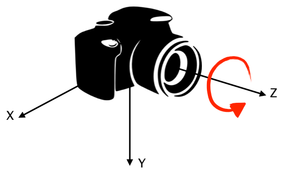
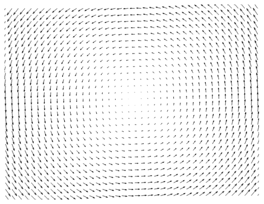
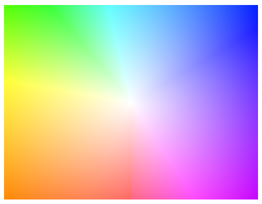
		optical flow	
	camera motion	vector plot	Middlebury color coding
			
translation			
			
			
rotation			
			

Table 3.1: Camera motions and the associated motion field with normalized focal length ($f = 1$)

3.1 The motion field of a moving camera

Suppose we took a short video with a camera. We assume a moving camera in a static environment. When the camera moves, the pixels belonging to the static background no longer maintain their positions in consecutive frames. Pixels move according to a the motion of the camera (Figure 3.1). If the camera moves to the left, then all image pixels will move to the right. If the camera moves along the Z-axis, pixels will spread out from the image center to the border. The camera is moving with 6 degrees of freedom. (U, V, W) describe the linear velocity (translation) along the three axes of the coordinate system and (A, B, C) describe the angular velocity (rotation).

Why are we talking about velocity? We are starting with the idea of pixel displacements [66]. P_0 is a point in 3D at time t. P_1 is a point in 3D at time $t+1$. So the pixel displacement can be described as follows.

$$\text{3D position at time } t: \quad P_0 \tag{3.1}$$

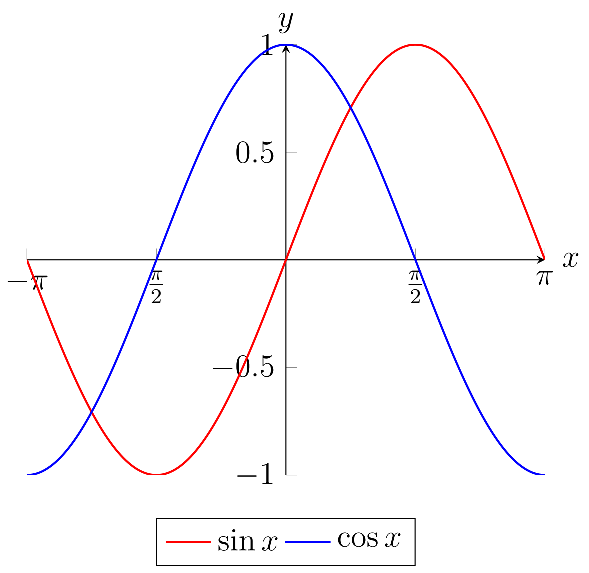
$$\text{3D position at time } t+1: \quad P_1 = P_0 R + T \tag{3.2}$$

$$\text{3D displacement:} \quad \Delta P = R P_0 + T - P_0 \tag{3.3}$$

A typical frame rate for videos is 30 frames per second (fps). Due to the small time period from frame to frame we assume a small rotation angle between frame a at time t and b at time $t+1$.

In case of small angle approximation:

$$\cos(x) = 1$$
$$\sin(x) = x$$
$$\sin(x)\sin(y) = 0$$



Small angle approximation allows us to write a multiplication of three rotation matrices R_x, R_y and R_z as one matrix, which can be separated in two parts - one skew symmetric matrix and one identity matrix. Let α, β and γ be the three rotation angles in 3D, then one can write the corresponding rotation matrices as follows:

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

$$R_y(\gamma) = \begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.4}$$

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

$$R = R_y(\gamma)R_z(\beta)R_x(\alpha)$$

$$= \begin{pmatrix} \cos(\beta)\cos(\gamma) & \cos(\gamma)\sin(\alpha)\sin(\beta) - \cos(\alpha)\sin(\gamma) & \cos(\alpha)\cos(\gamma)\sin(\beta) + \sin(\alpha)\sin(\gamma) \\ \cos(\beta)\sin(\gamma) & \cos(\alpha)\cos(\gamma) + \sin(\alpha)\sin(\beta)\sin(\gamma) & \cos(\alpha)\sin(\beta)\sin(\gamma) - \cos(\gamma)\sin(\alpha) \\ -\sin(\beta) & \cos(\beta)\sin(\alpha) & \cos(\alpha)\cos(\beta) \end{pmatrix}$$

$$= \begin{pmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{pmatrix} = I + \begin{pmatrix} 0 & -\gamma & \beta \\ \gamma & 0 & -\alpha \\ -\beta & \alpha & 0 \end{pmatrix} = I + S \tag{3.5}$$

Assuming that the rotation angles are small, we could replace the rotation matrix R and rewrite equation 3.1.

$$\text{3D displacement: } \Delta P = (I + S)P_0 + T - P_0 \tag{3.6}$$

$$\Delta P = T + SP_0 \tag{3.7}$$

In limit displacement become a velocity. SP_0 can be written as a cross product $\omega \times P_0$, where $\omega = (A, B, C)^T$ is the angular velocity. $t = (U, V, W)^T$ is the linear velocity. *Under small angle approximation* we can interpret a pixel displacement as a velocity, such that the flow field assigns a velocity vector to each point in the image. The velocity of $P_0 = (X, Y, Z)^T$ in 3D with respect to the object coordinate system is

$$V = -t - \omega \times P_0 \tag{3.8}$$

$$\begin{aligned} \frac{dX}{dt} &= -U - BZ + CY \\ \frac{dY}{dt} &= -V - CX + AZ \\ \frac{dZ}{dt} &= -W - AY + BX \end{aligned} \tag{3.9}$$

The optical flow assigns a 2D velocity vector to each image point (x, y) in the image plane. Therefor we project the point P in the object coordinate system onto the image plane using the equations for perspective projection. The coordinates of the corresponding point p are

$$x = \frac{Xf}{Z} \qquad\qquad y = \frac{Yf}{Z} \tag{3.10}$$

A velocity is the derivation of location with respect to time t. So that the optical flow at point (x, y), denoted by (u, v) is

$$u = \frac{dx}{dt} = \frac{f \cdot (\frac{dX}{dt}Z - \frac{dZ}{dt}X)}{Z^2} \qquad\qquad v = \frac{dy}{dt} = \frac{f \cdot (\frac{dY}{dt}Z - \frac{dZ}{dt}Y)}{Z^2} \tag{3.11}$$

After using the derivatives 3.9 it becomes visible that we can separate the optical flow into a translational and rotational component.

$$\begin{aligned}
u = u_t + u_r &= \frac{dx}{dt} && (3.12)\\
&= \frac{f \cdot ((-U - BZ + CY)Z - (-W - AY + BX)X)}{Z^2} && (3.13)\\
&= \frac{f \cdot (-UZ - BZ^2 + CYZ + WX + AYX - BX^2)}{Z^2} && (3.14)\\
&= -\frac{fU}{Z} - fB + \frac{fCY}{Z} + \frac{fWX}{Z^2} + \frac{fAYX}{Z^2} - \frac{fBX^2}{Z^2} && (3.15)\\
&= -\frac{fU}{Z} - fB + Cy + \frac{Wx}{Z} + \frac{Ayx}{f} - \frac{Bx^2}{f} && (3.16)\\
&= \frac{-fU + xW}{Z} + \frac{Ayx}{f} - fB - \frac{Bx^2}{f} + Cy && (3.17)
\end{aligned}$$

$$v = v_t + v_r = \frac{dy}{dt} = \frac{-Vf + yW}{Z} + fA + \frac{Ay^2}{f} - \frac{Bxy}{f} - Cx \tag{3.18}$$

In the following we explicitly address the geometry of the motion field due to camera rotation, which contains no information about the scene structure (depth).

We continue with motion field due to camera translation, which is informative in many regards, and thus is very valuable for the motion segmentation task and conclude with the motion field produced by general camera motion - the combination of camera translation and rotation.

3.1.1 Camera rotation

Let f be the camera's focal length. A camera rotation is defined by its three rotational parameters (A, B, C). Given the three rotational parameters (A, B, C), we can compute the rotational optical flow vector at each pixel position (x, y) as follows:[1]

$$\vec{v_r} = \begin{pmatrix} u_r \\ v_r \end{pmatrix} = \begin{pmatrix} \frac{A}{f}xy - Bf - \frac{B}{f}x^2 + Cy \\ Af + \frac{A}{f}y^2 - \frac{B}{f}xy - Cx \end{pmatrix} \tag{3.19}$$

The rotational flow vector $\vec{v_r}$ is independent of the scene depth (see Figure 3.1(b)), thus it can be simply subtracted from the optical flow $\vec{v}$ to "stabilize" the image.

3.1.2 Camera translation

Let (U, V, W) be the translational motion of the camera relative to an object. Let (X, Y, Z) be the real world coordinates in 3D of a point that projects to (x, y) in the image. The motion field vector (u, v) at the image location (x, y) due to a translational motion is given by

$$\vec{v_t} = \begin{pmatrix} u_t \\ v_t \end{pmatrix} = \frac{1}{Z} \begin{pmatrix} -fU + xW \\ -fV + yW \end{pmatrix}. \tag{3.20}$$

The translational flow vector $\vec{v_t}$ is inversely proportional to the scene depth Z, thus a large flow magnitude might be due to high motion speed, or the pictured object is

[1]This equation only holds if rotation angles are small. However camera rotation is always independent of the scene depth regardless their amount.

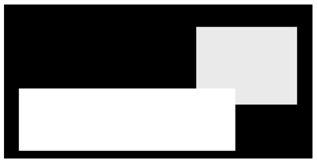
just very close to the camera (see Figure 3.1(c)). Just based on the flow magnitude, we are not able to distinguish between the two possible sources - speed and depth. The 2D translational motion direction at each point in the image is then given by the angle of the motion field vector (u, v) at image location (x, y):

$$\theta = \begin{cases} \arccos\left(W \cdot x - U \cdot f\right), & \text{if } (W \cdot y - V \cdot f) > 0 \\ 2\pi - \arccos\left(W \cdot x - U \cdot f\right), & \text{otherwise} \end{cases} \tag{3.21}$$

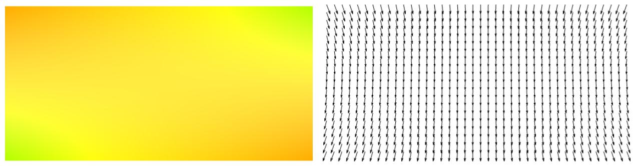
The translational flow direction at a particular pixel (x, y) however is purely determined by the parameters (U, V, W) and the focal length f.

3.1.3 Camera rotation and translation

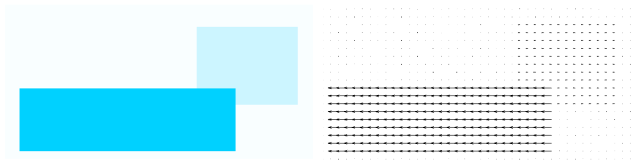
The motion field emerging due to combined camera translation and rotation is difficult to interpret. Since the motion angles (direction) are in this case influenced by the camera motion as well as the scene depth. The motion magnitudes are dependent upon the scene depth and the camera's motion speed. All theses interacting dependencies make the interpretation of a raw motion field given an arbitrary camera motion a challenging task (see Figure 3.1(d)).



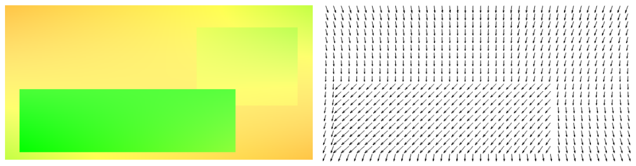
(a) depth



(b) rotation



(c) translation



(d) rotation and translation

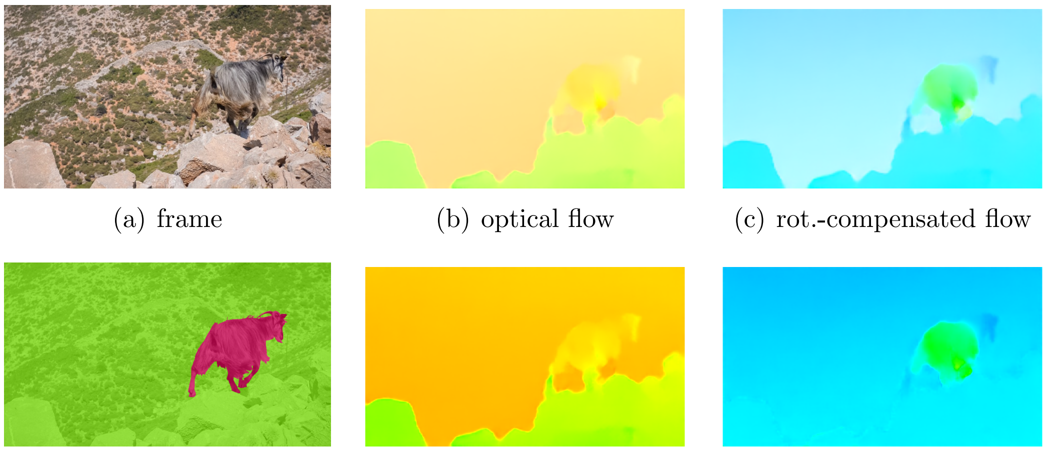
Figure 3.1: Motion fields of different camera motions of a *static scene* containing objects at different depth. (a) Shows a synthetically generated depth map. The color black describes a pictured region that is far away, white pixels correspond to nearby regions. (b), (c) and (d) show three different motion fields corresponding to the depth pictured in (a). On the left the motion field using the middleburry color coding [3] is used, where color encodes the direction of a motion field vector and color intensity the speed of a motion field vector. On the right the motion field is shown using a vector plot representation. The motion field shown in (b) corresponds to camera rotation around the horizontal axis. As one can see the motion field is purely determined by the camera's motion and independent of the scene depth. The motion field shown in (c) corresponds to camera translation to the left. The motion field is determined by camera motion and scene depth. Objects close to the camera lead to a larger image motion than objects far away. (d) pictures the motion field due to camera rotation and translation together. Motion field and depth are coupled. The motion field's direction at each image location as well as the its magnitude are depth and motion dependent.

CHAPTER 4
A PROBABILISTIC MODEL FOR MOTION SEGMENTATION

We perceive motion as flow fields (see Figure 4.1) that emerge due to the observer's motion, object motions or both. Depending upon the scene geometry these flow fields can be quite difficult to interpret correctly. Given a flow field as pictured in Figure 3.1(d), can one tell which objects (symbolized as rectangles) are moving? None of these objects are moving. Interaction of depth and camera motion lead to highly complex flow fields. Without further knowledge about the scene or the camera's motion, understanding the scene and answering questions addressing objects and their motion are hard to answer.

In nature the principle of gaze stabilization (compensation for rotational ego-motion) is a very wide spread mechanism that can be found in the vision system of humans and animals, which has the goal to reduce the complexity of the perceived flow field. Similar to gaze stabilization of the human visual system performed by little human eye movements, we propose an approach that first rectifies the flow field for camera rotation to obtain a new flow field, that is much easier to interpret in terms of observer motion and independent object motion.

In our work on probabilistic models for causal motion segmentation [7, 5] we developed a motion model to model independent object motion in short video sequences given the rotation compensated flow field. To each pixel the most likely motion model is assigned in a Bayesian fashion leading to a dense motion segmentation mask as shown in Figure 4.1(d). Our work can be devided into two large parts:



(a) frame (b) optical flow (c) rot.-compensated flow

(d) segmentation (e) angle of (c) (f) angle of (e)

Figure 4.1: Decide what is moving given the different optical flow fields. It is extremely challenging to distinguish between moving and stationary objects solely from optical flow. Notice that neither the raw optical flow (c) nor the angle of the raw optical flow (d) makes it clear that the goat is the only thing moving in this video. However, the angle of the rotation-compensated flow (f) strongly highlights the only moving object.

- **Modeling motion using rigid motion models while re-examining classical methods based upon perspective projection** [5]. Rigid motion models are defined by a 3D unit vector representing a particular translational motion direction. Following rules of perspective projection, this 3D motion direction leads to a unique *angle field*, which is independent of technical camera parameters such as focal length and the scene depth. We use these motion models to approximate the true object motion observed in a video sequence. Focusing on the true physics of perspective projection and combining these physical constraints with a Bayesian approach for segmentation, has led to high quality object segmentation results even in challenging camera videos with complex scene geometry and the lack of strong appearance cues.

- **Combining piecewise rigid motions to complex deformable and articulated objects, guided by learned semantic object segmentations [7].**

The motion of objects can be quite varied. Human body motion for example combines the motion of legs and arms which might move in contrary directions for a short period of time and thus create complex motion patterns in the flow field. To deal with these kind of complex motion patterns we use a set of rigid motion models together with objectness knowledge obtained from a network for semantic segmentation. Each part of an object can easily be described by a single rigid motion, but how does one know that these parts like legs and arm of a person actually move together? Knowing about objects and their appearance is one possible approach that I pursued in my work. Another possibility would be a temporal (long term) analysis of motion patterns to retrieve the characteristic's of a motion pattern for a particular object. Using additional knowledge about "objectness" we combine individual motion components to segment the object as a whole although its parts might move independently for a short period of time. This way a scene can be segmented into its independently moving objects regardless of the complexity of their motion pattern.

In Chapter 4.1 we will present an approach to segment a video into static environment and moving objects (binary segmentation mask) using rigid motion models. In Chapter 4.2 we extend our developed model by combining piecewise rigid motion models guided by learned semantic object segmentations to a more complex object motion model to be able to model deformable and articulated motions accurately. This way we are able not only to segment the video into moving object and static background (two labels), furthermore we are able to distinguish between differently moving objects and to track them over time.

Figure 4.2: Where is the camouflaged insect? Before looking at Figure 4.3, which shows the ground truth localization of this insect, try identifying the insect. While it is virtually impossible to see without motion, it immediately "pops out" to human observers as it moves in the video.

4.1 Binary Segmentation: Segmentation of a video sequence based upon different motion directions

How can we match the ease and speed with which humans and other animals detect motion? This remarkable capability works in the presence of complex background geometry, camouflage, and motion of the observer. Figure 4.2 shows a frame from a video of a "walking stick" insect. Despite the motion of the camera, the rarity of the object, and the high complexity of the background geometry, the insect is immediately visible as soon as it starts moving.

To develop such a motion segmentation system, we re-examined classical methods based upon perspective projection, and developed a new probabilistic model which accurately captures the information about 3D motion in each observed optical flow vector $\vec{v}$. First, we estimate the portion of the optical flow due to rotation, and subtract it from $\vec{v}$ to produce $\vec{v_t}$, the translational portion of the optical flow. Next, we derive a new *conditional flow angle likelihood* $\mathcal{L} = p(\theta_{\vec{v_t}} \mid M, \|\vec{v_t}\|)$, the probability of observing a particular flow angle $\theta_{\vec{v_t}}$ given a model M of the angle part of a particular object's (or the background's) motion field and the flow magnitude $\|\vec{v_t}\|$.

M, which we call an *angle field*, describes the motion *directions* of an object in the image plane. It is a function of the object's relative motion (U, V, W) and the camera's focal length f, but can be computed more directly from a set of *motion field parameters* $(U', V', W) = (fU, fV, W)_2$, where the "2" subscript indicates L_2 normalization.

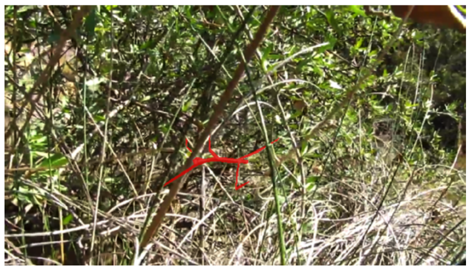


Figure 4.3: Answer: the insect from Figure 1 in shown in red. The insect is trivial to see in the original video, though extremely difficult to identify in a still image. In addition to superior results on standard databases, our method is also one of the few that can detect objects is such complex scenes.

Our angle likelihood helps us to address a fundamental difficulty of motion segmentation: the ambiguity of 3D motion given a set of noisy flow vectors. While we cannot eliminate this problem, the angle likelihood allows us to weigh the evidence for each image motion properly based on the optical flow. In particular, when the underlying image motion is very small, moderate errors in the optical flow can completely change the apparent motion direction (i.e., the angle of the optical flow vector). When the underlying image motion is large, typical errors in the optical flow will not have a large effect on apparent motion direction. This leads to the critical observation that small optical flow vectors are less informative about motion than large ones. Our derivation of the angle likelihood (Section 4.1.1) quantifies this notion and makes it precise in the context of a Bayesian model of motion segmentation.

We evaluate our method on three diverse data sets, achieving state-of-the-art performance on all three. The first is the widely used Berkeley Motion Segmentation (BMS-26) database [12, 114], featuring videos of cars, pedestrians, and other common scenes. The second is the Complex Background Data Set [72], designed to test algorithms' abilities to handle scenes with highly variable depth. Third, we introduce a new and even more challenging benchmark for motion segmentation algorithms: the *Camouflaged Animal Data Set*. The nine (moving camera) videos in this benchmark exhibit camouflaged animals that are difficult to see in a single frame, but can be detected based upon their motion across frames.

4.1.1 Methods

The *motion field* of a scene is a 2D representation of 3D motion. Motion vectors, describing the displacement in 3D, are projected onto the image plane forming a 2D motion field. This field is created by the movement of the camera relative to a stationary environment and the additional motion of independently moving objects. We use the optical flow, or estimated motion field, to segment each video image into static environment and independently moving objects.

The observed flow field consists of the flow vectors $\vec{v}$ at each pixel in the image. Let $\vec{m}$ be the flow vectors describing the motion field caused only by a rotating and translating camera in its stationary 3D environment. $\vec{m}$ does not include motion of other independently moving objects. The flow vectors $\vec{m}$ can be decomposed in a *translational component* $\vec{m_t}$ and a *rotational component* $\vec{m_r}$. Let the direction or angle of a flow vector of a translational camera motion at a particular pixel (x, y) be $\theta_{\vec{m_t}}$.

When the camera is only translating, there are strong constraints on the optical flow field – the *direction* $\theta_{\vec{m_t}}$ of the motion at each pixel is determined by the camera translation (U, V, W), the image location of the pixel (x, y), and the camera's focal length f, and has no dependence on scene depth [35].

$$\theta_{\vec{m_t}} = \begin{cases} \arccos\left(W \cdot x - U \cdot f\right), & \text{if } (W \cdot y - V \cdot f) > 0 \\ 2\pi - \arccos\left(W \cdot x - U \cdot f\right), & \text{otherwise} \end{cases} \tag{4.1}$$

The collection of $\theta_{\vec{m_t}}$ forms a translational angle field M representing the camera's translation direction on the 2D image plane.

Simultaneous camera rotation and translation, however, couple the scene depth and the optical flow, making it much harder to assign pixels to the right angle field M described by the estimated translation parameters (U', V', W).

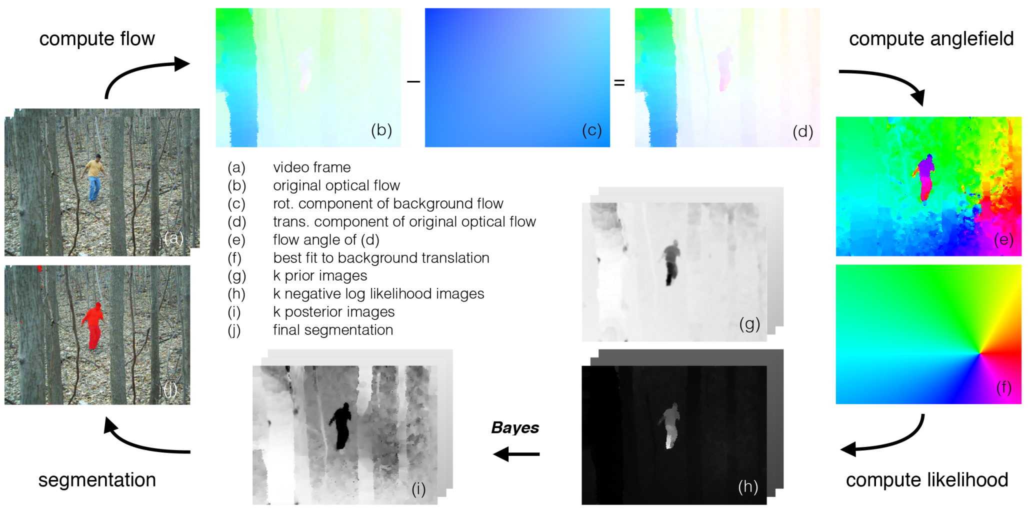


Figure 4.4: An overview: A probabilistic model for binary motion segmentation. Given the optical flow (b) the camera rotation is estimated. Then, the flow $\vec{m_r}$ due to camera rotation defined by the motion parameters (A, B, C) (c) is subtracted from the optical flow $\vec{v}$ to produce a translational flow $\vec{v_t}$. The flow angles $\theta_{\vec{v_t}}$ of $\vec{v_t}$ are shown in (e). The best fitting translation parameters (U', V', W) to the static environment of $\vec{v_t}$ yield an estimated angle field M (f), which clearly shows the forward motion of the camera (rainbow focus of expansion pattern) not visible in the original angle field. The motion component priors (g) and negative log likelihoods (h) yield the posteriors (i) and the final segmentation (j).

To address this, we wish to subtract off the flow vectors $\vec{m_r}$ describing the rotational camera motion field from the observed flow vectors $\vec{v}$ to produce a flow $\vec{v_t}$ comprising camera translation only. The subsequent assignment of flow vectors to particular angle fields is thus greatly simplified. However estimating camera rotation in the presence of multiple motions is challenging.

In Chapter 4.1.1.1, we describe how all frames after the first frame are segmented, using the segmentation from the previous frame and our angle likelihood. After reviewing Bruss and Horn's motion estimation technique [14] in Chapter 4.1.1.2, Chapter 4.1.1.3 describes how our method is initialized in the first frame, including a process for estimating camera motion in the presence of multiple motions.

4.1.1.1 A probabilistic model for motion segmentation

Given a prior motion segmentation of frame $t-1$ into k different motion components and an optical flow from frames t and $t+1$, segmenting frame t requires several ingredients: **a)** the *prior* probabilities $p(M_j)$ for each pixel that it is assigned to a particular angle field M_j, **b)** the estimate of the translational angle field M_j, $1 \leq j \leq k$ to be able to model the motion for each of the k motion components from the previous frame, **c)** for each pixel position, a *likelihood* $\mathcal{L}_j = p(\vec{v_t} \mid M_j)$, the probability of observing a flow vector $\vec{v_t}$ under an estimated angle field M_j, and **d)** the prior probability $p(M_{k+1})$ and angle likelihoods $\mathcal{L}_{k+1}$ given an angle field M_{k+1} to model a *new motion*. Given these priors and likelihoods, we use Bayes' rule to obtain a *posterior* probability for each translational angle field at each pixel location. We have

$$p(M_j \mid \vec{v_t}) \propto p(\vec{v_t} \mid M_j) \cdot p(M_j) \tag{4.2}$$

We directly use this posterior for segmentation. We now describe how the above quantities are computed.

Propagating the posterior for a new prior. We start from the optical flow of Sun et al. [106] (Figure 4.4(b)). We then create a prior at each pixel for each angle field M_j in the new frame (Figure 4.4(g)) by propagating the posterior from the previous frame (Figure 4.4(i)) in three steps.

1. Use the previous frame's flow to map posteriors from frame $t-1$ (Figure 4.4(i)) to new positions in frame t.

2. Smooth the mapped posterior in the new frame by convolving with a spatial Gaussian, as done in [72, 73]. This implements the idea that object locations in future frames are likely to be close to their locations in previous frames.

3. Renormalize the smoothed posterior from the previous frame to form a proper probability distribution at each pixel location, which acts as the prior on the k motion components for the new frame (Figure 4.4(g)). Finally, we set aside a probability of $1/(k+1)$ for the prior of a new motion component, while rescaling the priors for the pre-existing motions to sum to $k/(k+1)$.

Estimating and removing rotational flow. We use the prior for the motion component of the static environment to weight pixels for estimating the current frame's flow due to the camera motion. We estimate the camera translation parameters (U', V', W) and rotation parameters (A, B, C) using a modified version of the Bruss and Horn algorithm [14] (Section 4.1.1.2). As described above, we then render the flow angle independent of the unknown scene depth by subtracting the estimated rotational flow (Figure 4.4(c)) from the original flow (Figure 4.4(b)) to produce an estimate of the flow without influences of camera rotation (Fig. 4.4(d)). For each flow vector we compute:

$$\vec{\hat{v}}_t = \vec{v} - \vec{\hat{m}}_r(\hat{A}, \hat{B}, \hat{C}) \tag{4.3}$$

$$\theta_{\vec{v_t}} = \measuredangle(\vec{\hat{v}}_t, \vec{n}) \tag{4.4}$$

, where $\vec{n}$ is a unit vector $[1, 0]^T$.

For each additional motion component j besides the static environment, we estimate 3D translation parameters (U', V', W) using the segment priors to select pixels, weighted according to the prior, such that the motion perceived from video frame t to $t+1$ is described by j independent angle fields M_j.

The flow angle likelihood. Once we have obtained a translational flow field by removing the rotational flow, we use each flow vector $\vec{v_t}$ to decide which motion component it belongs to. Most of the information about the 3D motion direction is contained in the flow angle, not the flow magnitude. This is because for a given translational 3D motion direction (relative to the camera), the flow angle is completely determined by that motion and the location in the image, whereas the flow magnitude is a function of the object's depth, which is unknown. However, as discussed above, the *amount of information* in the flow angle depends upon the flow magnitude–flow vectors with greater magnitude are much more reliable indicators of true motion direction. This is why it is critical to formulate the angle likelihood conditioned on the flow magnitude.

Other authors have used flow angles in motion segmentation. For example, Papazoglou and Ferrari [81] use both a gradient of the optical flow and a separate function of the flow angle to define motion boundaries. Narayana et al. [72] use *only* the optical flow angle to evaluate motions. But our derivation gives a principled and effective method of using the flow angle and magnitude together to mine accurate information from the optical flow. In particular, we show that while the translational magnitudes alone have no information about which motion is most likely, the magnitudes play an important role in specifying the *informativeness* of the flow angles. In our experiments section, we demonstrate that failing to condition on flow magnitudes in this way results in greatly reduced performance over our derived model.

We now derive the key element of our method, the *conditional flow angle likelihood* $p(\theta_{\vec{v_t}} \mid M_j, \|\vec{v_t}\|)$, the probability of observing a flow direction $\theta_{\vec{v_t}}$ given that a pixel was part of a motion component undergoing the 2D motion direction M_j, and that the flow magnitude was $\|\vec{v_t}\|$. We make the following modeling assumptions:

1. We assume the observed translational flow $\vec{v_t} = (\|\vec{v_t}\|, \theta_{\vec{v_t}})$ at a pixel is a noisy observation of the translational motion field $\vec{m_t} = (\|\vec{m_t}\|, \theta_{\vec{m_t}})$:

$$\vec{v_t} = \vec{m_t} + \eta, \tag{4.5}$$

   where η is independent 2D Gaussian noise with zero mean and circular but unknown covariance.

2. We assume the translational motion field magnitude $\|\vec{m_t}\|$ is statistically independent of the translation angle field M created by the estimated 3D translation parameters (U', V', W). It follows that $\|\vec{v_t}\| = \|\vec{m_t}\| + \eta$ is also independent of M, and hence $p(\|\vec{v_t}\| \mid M) = p(\|\vec{v_t}\|)$.

With these assumptions, we have

$$p(\vec{v_t} \mid M_j) \overset{(1)}{=} p(\|\vec{v_t}\|, \theta_{\vec{v_t}} \mid M_j) \tag{4.6}$$

$$= p(\theta_{\vec{v_t}} \mid \|\vec{v_t}\|, M_j) \cdot p(\|\vec{v_t}\| \mid M_j) \tag{4.7}$$

$$\overset{(2)}{=} p(\theta_{\vec{v_t}} \mid \|\vec{v_t}\|, M_j) \cdot p(\|\vec{v_t}\|) \tag{4.8}$$

$$\propto p(\theta_{\vec{v_t}} \mid \|\vec{v_t}\|, M_j), \tag{4.9}$$

where the numbers over each equality give the assumption that is invoked. Equation (4.9) follows since $p(\|\vec{v_t}\|)$ is constant across all estimated angle fields.

We model $p(\theta_{\vec{v_t}} \mid \|\vec{v_t}\|, M)$ using a *von Mises* distribution $\mathcal{V}(\mu, \kappa)$ with parameters μ, the preferred direction, and concentration parameter κ. We set $\mu = \theta_{\vec{m_t}}$, since $\theta_{\vec{m_t}}$ is the most likely direction assuming a noisy observation of a translational motion θ_{v_t}. To set κ, we observe that when the ground truth flow magnitude $\|\vec{m_t}\|$ is small, the distribution of observed angles θ_{v_t} will be near uniform (see Figure 4.5, $\vec{m_t} = (0, 0)$), whereas when $\|\vec{m_t}\|$ is large, the observed angle $\theta_{\vec{v_t}}$ is likely to be close to the flow angle $\theta_{\vec{m_t}}$ (Figure 4.5, $\vec{m_t} = (2, 0)$). We can achieve this basic relationship by setting $\kappa = a(\|\vec{m_t}\|)^b$, where a and b are parameters that give added flexibility to the model. Since we don't have direct access to $\|\vec{m_t}\|$, we use $\|\vec{v_t}\|$ as a surrogate, yielding

$$p(\theta_{\vec{v_t}} \mid \|\vec{v_t}\|, M_j) \propto \mathcal{V}(\theta_{\vec{v_t}}; \mu = \theta_{\vec{m_t}}, \kappa = a\|\vec{v_t}\|^b). \tag{4.10}$$

Note that this likelihood treats zero-length translation vectors as uninformative–it assigns them the same likelihood under all motions. This makes sense, since the

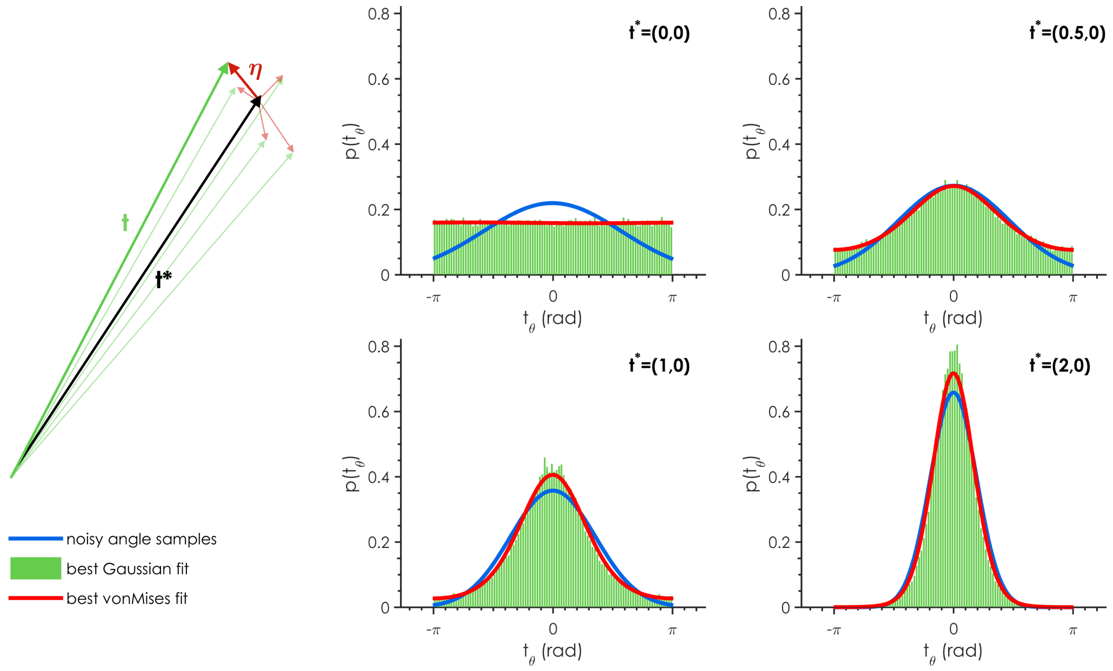


Figure 4.5: The von Mises distribution. When a motion field vector $\vec{m}_t$ is perturbed by added Gaussian noise η (figure top left), the resulting distribution over optical flow angles θ_{v_t} is well-modeled by a *von Mises* distribution. The figure shows how small motion field vectors result in a broad distribution of angles after noise is added, while larger magnitude motion field vectors result in a narrower distribution of angles. The red curve shows the best von Mises fit to these sample distributions and the blue curve shows the lower quality of the best Gaussian fit.

direction of a zero-length optical flow vector is essentially random. Similarly, the longer the optical flow vector, the more reliable and informative it becomes.

Likelihood of a new motion. Lastly, with no prior information about new motions, we set $p(\theta_{\vec{v_t}} \mid \|\vec{v_t}\|, M_j) = \frac{1}{2\pi}$, a uniform distribution.

Once we have priors and likelihoods, we compute the posteriors (Equation 4.2) and label each pixel as

$$L = \arg\max_j p(M_j \mid \vec{v_t}). \tag{4.11}$$

4.1.1.2 Bruss and Horn's motion estimation.

To estimate the translation parameters (U', V', W) of the camera relative to the static environment, we use the method of Bruss and Horn [14] and apply it to pixels selected by the prior of M_j. The observed optical flow vector $\vec{v_i}$ at pixel i can be decomposed as $\vec{v_i} = \vec{p_i} + \vec{e_i}$, where $\vec{p_i}$ is the component of $\vec{v_i}$ in the predicted direction θ_{m_t} and $\vec{e_i}$ is the component orthogonal to $\vec{p_i}$. The authors find the motion parameters that minimizes the sum of these "error" components $\vec{e_i}$. The optimization for translation-only is

$$\operatorname*{arg\,min}_{U',V',W} \sum_i \|\vec{e_i}(\vec{v_i}, U', V', W)\|, \tag{4.12}$$

where $(U', V', W) = (Uf, Vf, W)$ are the three translation parameters. Since we do not know the focal length it's not possible to compute the correct 3D translation, but we are able to estimate the parameters (U', V', W), which show the same angular characteristics in 2D as the true 3D translation (U, V, W). Bruss and Horn give a closed form solution to this problem for the translation-only case.

Recovering camera rotation. Bruss and Horn also outline how to solve for rotation, but give limited details. We implement our own estimation of rotations (A, B, C) and translation as a nested optimization:

$$\hat{M} = \operatorname*{arg\,min}_{A,B,C} \left[\min_{U',V',W} \sum_i \|\vec{e_i}\left(\vec{v_i}, A, B, C, U', V', W\right)\| \right]. \tag{4.13}$$

Given (A, B, C) one can compute the flow vectors $\vec{m_r}$ describing the rotational motion field of the observed flow, one can subtract off the rotation since it does not depend on scene geometry: $\vec{v_t} = \vec{v} - \vec{m_r}(A, B, C)$. Subtracting the rotation (A, B, C) from the observed flow reduces the optimization to the translation only case. We solve the optimization over the rotation parameters A, B, C by using Matlab's standard gradient descent optimization, while calling the Bruss and Horn closed form solution

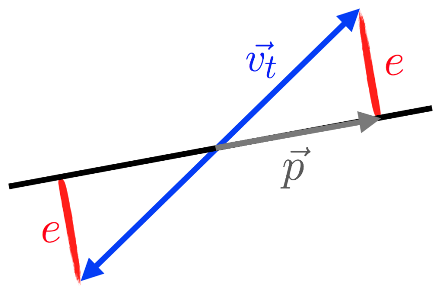


Figure 4.6: The Bruss and Horn error. Let $\vec{p}$ be a vector in the direction of preferred motion with respect to a motion hypothesis (U, V, W). The Bruss and Horn error assigned to a translational flow vector $\vec{v}_t$ is then the distance of its projection onto $\vec{p}$. However, this *same error* would be assigned to a vector $-\vec{v}_t$ pointing in the opposite direction, which should have much lower compatibility with the motion hypothesis. direction, its error is computed as its full magnitude, rather than the distance of projection (left side of Figure). This new error function keeps objects moving in opposite directions from being confused with each other.

for the translation variables given the rotational variables as part of the internal function evaluation. Local minima are a concern, but since we are estimating camera motion between two video frames, the rotation is almost always small and close to the optimization's starting point.

4.1.1.2.1 Modified Bruss and Horn Error We introduce a modification to the error function of the Bruss and Horn algorithm that we call the modified Bruss and Horn (MBH) error. We first describe the Bruss and Horn error function and a particular issue that makes it problematic in the context of motion segmentation, and then describe our modification to the algorithm.

The Bruss and Horn Error Function. The goal of the Bruss and Horn algorithm (translation-only case) is to find the motion direction parameters (U, V, W) that are as compatible as possible with the observed optical flow vectors. Let $\vec{p}$ be a vector in the direction of the flow expected from a motion (U, V, W) (see Figure 4.6). Then the Bruss and Horn error for the observed flow vector $\vec{v}_t$ is the distance of the projection of $\vec{v}_t$ onto $\vec{p}$, shown by the red segment e on the right side of the figure.

The problem with this error function is that this distance is small not only for vectors which are close to the preferred direction, but also for vectors that are in a direction *opposite* the preferred direction. That is, *observed optical flow vectors that point in exactly the wrong direction with respect to a motion* (U, V, W) *get a small error* in the Bruss and Horn algorithm. In particular, the error assigned to a vector $\vec{v}_t$ is the same as the error assigned to a vector $-\vec{v}_t$ in the opposite direction (See Figure 4.6).

Because the Bruss and Horn algorithm is intended for motion estimation in scenarios where there is only a single motion (the camera motion), such motions in the opposite direction to the preferred motion are not common, and thus, this "problem" we've identified has little impact. However, in the motion segmentation setting, where flows of objects may point in opposite directions, this can make the flow of a separately moving object, look as though it is compatible with the background. We address this problem by introducing a modified version of the error.

The modified Bruss and Horn error. For the first frame we do not have any estimate about static environment and moving objects. As stated above, the Bruss and Horn error is the distance of the projection of an optical flow vector onto the vector $\vec{p}$ representing the preferred direction of flow according to a translational motion (U, V, W). This can be written simply as

$$e_{BH}(\vec{v}_t, \vec{p}) = \|\vec{v}_t\| \cdot |\sin(\measuredangle(\vec{v}_t, \vec{p})|. \tag{4.14}$$

This error function has the appropriate behavior when the observed optical flow is within 90 degrees of the expected flow direction, i.e., when $\vec{v}_t \cdot \vec{p} \geq 0$. However, when the observed flow points *away* from the preferred direction, we assign an error equal to the magnitude of the entire vector, rather than its projection, since no component of this vector represents a "valid direction" with respect to (U, V, W). This results in the modified Bruss and Horn error (see Figure 4.7):

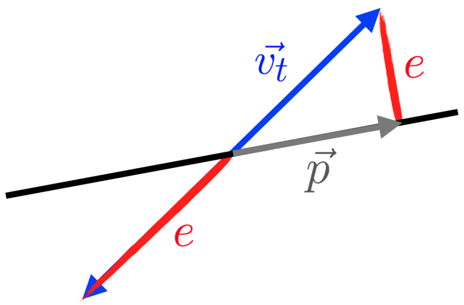


Figure 4.7: The modified Bruss and Horn error. When an observed translation vector $\vec{v}_t$ is within 90 degrees of the preferred direction, its error is computed in the same manner as the traditional Bruss and Horn error (right side of Figure). However, when the observed vector is more than 90 degrees from the preferred direction, its error is computed as its full magnitude, rather than the distance of projection (left side of Figure). This new error function keeps objects moving in opposite directions from being confused with each other.

$$e_{MBH} = \begin{cases} \|\vec{v}_t\|, & \text{if } \vec{v}_t \cdot \vec{p} < 0 \\ \|\vec{v}_t\| \cdot |\sin(\measuredangle(\vec{v}_t, \vec{p})|, & \text{otherwise.} \end{cases}$$

This error has the desired behavior of penalizing flows in the opposite direction to the expected flow.

4.1.1.3 Initialization: Segmenting the first frame

The goals of the initialization are a) estimating translation parameters (U', V', W) and the rotation (A, B, C) of the motion of static environment due to camera motion, b) the estimated set of parameters (U', V', W) form an angle field M corresponging to the observed flow c) finding pixels whose flow is consistent with M, and d) assigning inconsistent groups of contiguous pixels to additional angle fields. Bruss and Horn's method was not developed to handle scenes with multiple different motions, and so large or fast-moving objects can result in poor motion estimates (Figure 4.10).

Constrained RANSAC. To address this problem we use a modified version of RANSAC [23] to robustly estimate motion of static environment (Figure 4.8). We use 10 random SLIC superpixels [1][1] to estimate camera motion (Section 4.1.1.2). We modify the standard RANSAC procedure to force the algorithm to choose three of

[1]We use the http://www.vlfeat.org/api/slic.html code with regionSize=20 and regularizer=0.5.

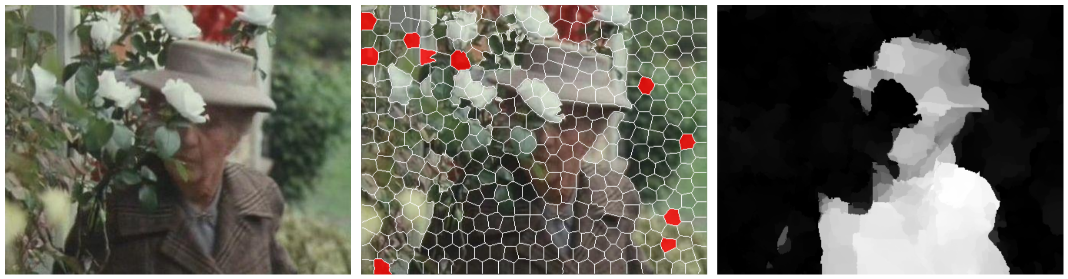


Figure 4.8: Initialization: Segmenting the first frame using random sample consensus (RANSAC). The result of our RANSAC procedure is to find image patches of the static environment. Notice that none of the patches are on the person moving in the foreground. Also notice that we force the algorithm to pick patches in three of the four image corners (a "corner" is 4% of the image). The right figure shows the negative log likelihood of the static environment.

the 10 patches from the image corners, because image corners are prone to errors due to a misestimated camera rotation. 5000 RANSAC trials are run, and the camera motion resulting in the fewest outlier pixels according to the *modified Bruss-Horn* (MBH) error is retained, using a threshold of 0.1.

Otsu's Method. While using the RANSAC threshold on the MBH image produces a good set of pixels to estimate the motion of the static environment due to camera motion, the method often excludes some pixels that should be included in the motion component of static environment. We use Otsu's method [79] to separate the MBH image into a region of low error (static environment) and high error: (1) Use Otsu's threshold to divide the errors, minimizing the intraclass variance. Use this threshold to do a binary segmentation of the image. (2) Find the connected component C with highest average error. Remove these pixels ($I \leftarrow I \setminus C$), and assign them to an additional angle field M. These steps are repeated until Otsu's *effectiveness* parameter is below 0.6.

Algorithm 1: A causal motion segmentation algorithm.

```
Input: video with $n$ frames
Output: binary motion segmentation
1  for $t \leftarrow 1$ to $n-1$ do
2      compute optical flow from frame $t$ to frame $t+1$
3      if first frame then
4          foreach RANSAC iteration do
5              find best set of translation parameters $(U', V', W)$ for 10 random
                 patched (3 in corners)
6              retain best angle field for the static environment $M_k$
7          end
8          $p(M) \leftarrow$ segment MBH error image into $k$ comp. using Otsu's method
9      else
10         $p(M) \leftarrow$ propagate posterior $p(M \mid \vec{v_t})$
11         find $(U', V', W)$ and rotation $(A, B, C)$ of static environment using
             gradient descent
12         foreach flow vector $\vec{v}$ do
13             $\vec{v_t} = \vec{v} - \vec{m_r}(A, B, C)$
14         end
15     end
16     for $j \leftarrow 1$ to $k$ do
17         compute angle field $M_j$ of motion component $j$
18         foreach flow vector $\vec{v_t}$ do
19             $p(\theta_{\vec{v_t}} \mid M_j, \|\vec{v_t}\|) \leftarrow \mathcal{V}(\theta_{\vec{v_t}}; \mu = \theta^j_{\vec{m_t}}, \kappa = a\|\vec{v_t}\|^b)$
20         end
21     end
22     foreach flow vector $\vec{v_t}$ do
23         $p(M_{k+1}) \leftarrow \frac{1}{k+1}$
24         $p(\theta_{\vec{v_t}} \mid M_{k+1}, \|\vec{v_t}\|) \leftarrow \frac{1}{2\pi}$
25         normalize $p(M_j)$ that they sum up to $1 - p(M_{k+1})$
26         $p(M \mid \vec{v_t}) \leftarrow p(\theta_{\vec{v_t}} \mid M, \|\vec{v_t}\|) \cdot p(M)$
27     end
28     given the posteriors $p(M \mid \vec{v_t})$ assign every pixel one of two labels: static
         environment or moving objects
29 end
```

4.1.2 Experiments

Several motion segmentation benchmarks exist, but often a clear definition of what people intend to segment in ground truth is missing. The resulting inconsistent segmentations complicate the comparison of methods. We define motion segmentation as follows.

(I) Every pixel is given one of two labels: static background or moving objects.

(II) If only part of an object is moving (like a moving person with a stationary foot), the entire object should be segmented.

(III) *All* freely moving objects (not just one) should be segmented, but nothing else. We do not considered tethered objects such as trees to be freely moving.

(IV) Stationary objects are not segmented, even when they moved before or will move in the future. We consider segmentation of previously moving objects to be *tracking*. Our focus is on segmentation by motion analysis.

Experiments were run on two previous data sets and our new camouflaged animals videos. The first was the Berkeley Motion Segmentation (BMS-26) database [12, 114] (Figure 4.11, rows 5,6). Some BMS videos have an inconsistent definition of ground truth from both our definition and from the other videos in the benchmark. An example is *Marple10* whose ground truth segments a wall in the foreground as a moving object (see Figure 4.9). While it is interesting to use camera motion to segment static objects (as in [123]), we are addressing the segmentation of objects that are moving differently than the background, and so we excluded ten such videos from our experiments (see [4] for more details). The second database used is the Complex Background Data Set [72], which includes significant depth variation in the background and also significant amounts of camera rotation (Figure 4.11, rows 3,4). We also introduced the Camouflaged Animals Data Set (Figure 4.11, rows 1,2). These videos were ground-truthed every 5th frame.

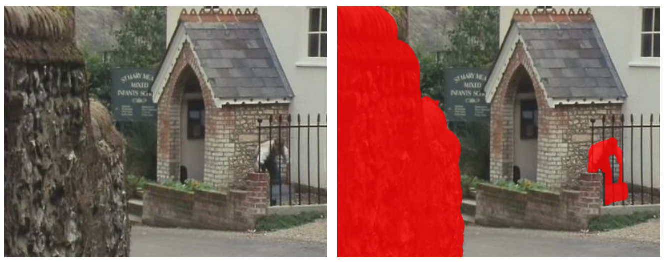


Figure 4.9: Example of common segmentation inconsistencies in widely used motion segmentation data sets. Some BMS-26 videos contain significant ground truth errors, such as this segmentation of the foreground wall, which is clearly not a moving object.

Setting von Mises parameters. There are two parameters a and b that affect the von Mises concentration $\kappa = ar^b$. To set these parameters for each video, we train on the remaining videos in a leave-one-out paradigm, maximizing over the values $0.5, 1.0, 2.0, 4.0$ for multiplier parameter a and the values $0, 0.5, 1, 2$ for the exponent parameter b. Cross validation resulted in the selection of the parameter pair ($a = 4.0, b = 1.0$) for most videos, and we adopted these as our final values.

4.1.2.1 Results: Binary motion segmentation

In Tab. 4.1, we compare our model to five different state-of-the-art methods [81, 134, 72, 25, 48]. We compared against methods for which either code was available or that had results on either of the two public databases that we used. However, we excluded some methods (such as [109]), as their published results were less accurate than [48], to whom we compared.

Some authors have scored algorithms using the number of correctly labeled pixels. However, when the moving object in the foreground is small, a method can achieve a very high score simply by marking the entire video as background. The F-measure is also not symmetric with respect to foreground and background, and is not well-defined when a frame contains no foreground pixels. Matthew's Correlation Coefficient (MCC) handles both of these issues, and is recommended for scoring such binary classification problems when there is a large imbalance between the number of pixels in each category [89]. However, in order to enable comparison with [72], and

		Keuper [48]	**Papaz.** [81]	**Frag.** [25]	**Zama.** [134]	**Naray.** [72]	**ours**
Camouflage	MCC	**0.4305**	0.3517	0.1633	0.3354	-	**0.5344**
	F	**0.4379**	0.3297	0.1602	0.3007	-	**0.5276**
BMS-26	MCC	0.6851	0.6112	**0.7187**	0.6399	-	**0.7576**
	F	**0.7306**	0.6412	0.7276	0.6595	0.6246	**0.7823**
Complex	MCC	0.4752	**0.6359**	0.3257	0.3661	-	**0.7491**
	F	0.4559	**0.6220**	0.3300	0.3297	0.3751	**0.7408**
Total avg.	MCC	**0.5737**	0.5375	0.4866	0.5029	-	**0.6918**
	F	**0.5970**	0.5446	0.4911	0.4969	-	**0.6990**

Table 4.1: Binary motion segmentation - Comparison to state-of-the-art. We compare our final method to other motion segmentation approaches using the Matthew's correlation coefficient and F-measure. Numbers for each data set and the total average across all valid videos are provided. Best viewed in color (**1st-best**, **2nd-best**).

to allow easier comparison to other methods, we also included F-measures. Table 4.1 shows the highest average accuracy per data set highlighted in **yellow** and the second best in **blue**, for both the F-measure and MCC. We were not able to obtain code for Narayana et al. [72], but reproduced F-measures directly from their paper. The method of [25] failed on several videos (only in the BMS data set), possibly due to the length of these videos. In these cases, we assigned scores for those videos by assigning all pixels to background.

4.1.2.2 Ablation study

Conditioning our angle likelihood on the flow magnitude is an important factor in our method. Table 4.2 shows the detrimental effect of using a constant von Mises concentration κ instead of one that depends upon flow magnitude. In this experiment, we set the parameter b which governs the dependence of κ on $\vec{t}_r$ to 0, and set the value of κ to maximize performance. Even with the optimum constant κ, the drop in performance was 7%, 5%, and a whopping 22% across the three data sets.

	final	constant κ	no RANSAC
BMS-26	**0.7576**	0.6843	0.6450
complex background	**0.7491**	0.7000	0.5757
camouflaged animals	**0.5344**	0.3128	0.5176

Table 4.2: Ablation study. The effect of RANSAC and the concentration parameter κ.

We also show the consistent gains stemming from our constrained RANSAC initialization procedure. In this experiment, we segmented the first frame of video without rejecting any pixels as outliers. In some videos, this had little effect, but sometimes the effect was large, as shown in Figure 4.10.

The method by Keuper et al. [48] performs fairly well, but often makes errors in segmenting rigid parts of the foreground near the observer. This can be seen in the third and fourth rows of Figure 4.11, which shows sample results from the Complex Background Data Set. In particular, note that Keuper et al.'s method segments the tree in the near foreground in the third row and the wall in the near foreground in the fourth row. The method of Fragkiadaki et al., also based on trajectories, has similar behavior. These methods in general seem to have difficulty with high variability in depth.

4.1.3 Summary

We developed a new motion segmentation algorithm based on Bayesian statistics. A new angle likelihood function is presented that accurately captures the amount of independent object motion information contained in the flow angle using the flow magnitude as an informativeness measure. The larger the flow magnitude the more reliable the information contained in the optical flow angle. We are not making any compromises in modeling motion. We are directly using the perspective projection equations leading to the translational angle field to analyze motion, as has been

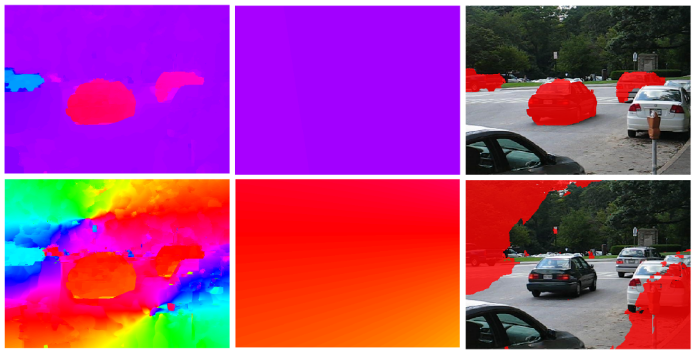


Figure 4.10: Ablation study: Initialization of the first frame with and without RANSAC. Top row: robust initialisation with RANSAC. Bottom row: using Bruss and Horn's method directly on the entire image. Left to right: flow angles of translational flow, flow angles of estimated background translation and segmentation. Note that without RANSAC the estimated background translation is the best fit for the car instead of background.

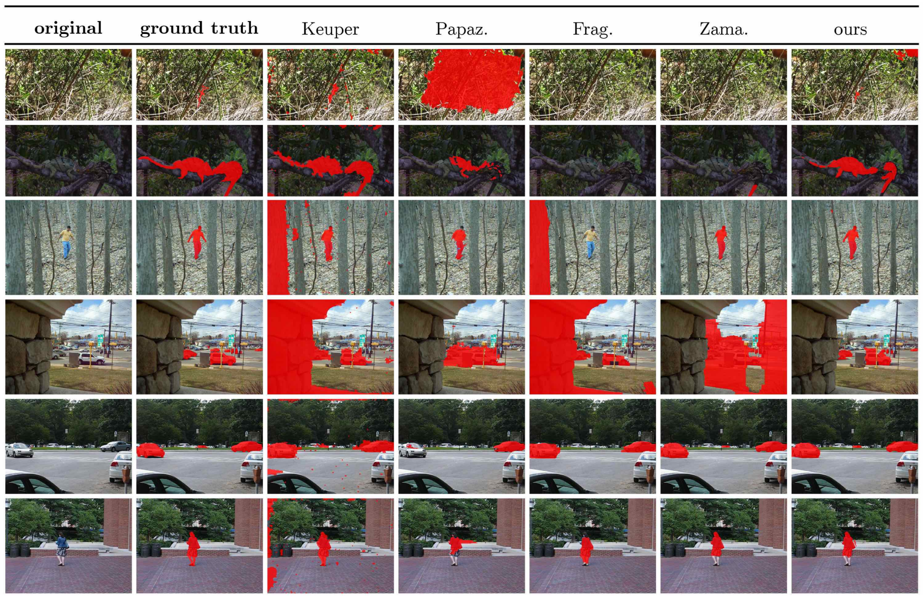


Figure 4.11: Qualitative segmentation results. Left to right: original image, ground truth, [48], [81], [25], [134] and our binary segmentations. Rows 1-2: sample results on the Animal Camouflage Data Set (chameleon and stickinsect). Rows 3-4: sample results on Complex Background (traffic and forest). Rows 5-6: sample results on BMS-26 (cars5 and people1).

advocated by Horn [36], rather than approximations based on projective geometry.We showed great performance on three data sets, one of which is the new introduced Camouflaged Animal data set. This data set focuses on scenes where motion is the strongest cue (exceeding appearance significantly) to detect otherwise invisible animals.

4.2 Multi-label Segmentation: Adding semantic information to distinguish among differently moving objects

Motion segmentation is an intriguing problem in that it combines subareas of vision in which geometry is a powerful constraint–the understanding of how images will change under camera motion–with “messy” problems like segmentation and the deformation of flexible moving objects, in which there are virtually no hard geometric constraints. This has given rise to a range of methods–some that use mostly geometric techniques while largely ignoring appearance [40, 5, 134], and others that try to learn the entire pipeline using CNN architectures [110, 111, 42] attempting to learn both the image patterns and the flow patterns in CNNs.

Methods that use motion cues alone, without appearance models of moving objects, are likely to fail in cases where flow is noisy, ambiguous, or hard to determine. Such purely “geometric” approaches are often not sufficient to understand motion well. The appearance of what is moving must also be considered. This suggests using deep learning methods to incorporate high-level semantic object information besides motion cues alone.

Of course, CNNs are excellent at modeling the appearance of objects [52, 103, 34]. They excel at finding objects in static images and videos [27, 64, 92]. They are also very good at segmenting objects [17, 137, 61, 31, 65, 97], exceeding performance of pre-CNN methods. However, there are cases where *appearance alone* is simply not enough to segment well. Such cases are highlighted by the Camouflaged Animal motion segmentation data set [5], in which moving objects are virtually invisible in many of the static frames.

In this Chapter we extend our work presented in Chapter 4.1 and combine careful motion modeling using classical ideas with a modern CNN for appearance modeling, yielding excellent results. Towards this end, we design a hierarchical motion segmentation system in which the first phase identifies simple rigid motion components,

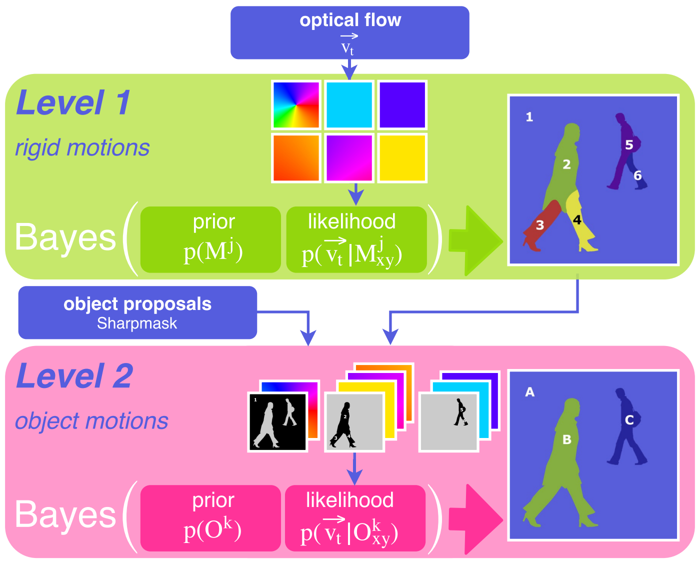


Figure 4.12: A hierarchical model for motion segmentation. The first level of our method estimates rigid motion components from optical flow. The second level groups these components based upon object proposals from SharpMask [86] to form object motion models.

and the second phase assembles these rigid motion components into full objects, guided by a semantic segmentation of each frame [86] (see Figure 4.12). This new hierarchical system allows the first low-level phase to focus on the geometry of perspective projection, segmenting the frame into its rigid motions. Then, in the second phase, deformable and articulated objects, like pedestrians and animals, are modeled as a combination of a number of rigid motion components, as suggested by the semantic segmentation results. While neither the motion analysis nor the semantic segmentations are error free, their combination results in a significant improvement in performance on the multi-label motion segmentation problem. Our contributions include:

- A new hierarchical model for motion segmentation with two steps:
    1. segmenting a frame into rigid motions;

2. using *objectness* knowledge from SharpMask [86] to combine these rigid parts into object models that describe the motions of articulated and deformable objects such as people or animals.

- A new statistical model for optical flow as a noisy measurement of the underlying motion field. We set noise distribution parameters using statistics of the Sintel database [15].
- A Bayesian approach to computing the likelihood of a *3D motion direction* associated with an optical flow vector, in which we integrate over the unobservable motion field magnitude. This allows us to assign pixels to rigid motion models in a fashion consistent with perspective projection and our statistical model.

We report results on three motion segmentation benchmarks that are consistent with the classical definition of motion segmentation: Freiburg-Berkeley Motion Segmetnation [12], Complex Background [72], and Camouflaged Animals [5]. The Davis data set [83, 88] is a popular *video segmentation* benchmark which focuses on segmenting prominent objects rather than all moving objects. While our method is not designed for such benchmarks, we still discuss results on that benchmark and the relationship between object segmentation and motion segmentation.

4.2.1 Methods

Our approach is not limited to a certain type of scenes. Our intention is to propose a new method to segment any video into static environment and moving objects without any prior knowledge about the video with its objects, motions and scene structure. To get an initial estimate about the scene, our algorithm is (automatically) initialized with an estimate of the background region and a set of rigid objects. We adopt the initialization procedure presented in Chapter 4.1.1.3 for this purpose.

Throughout our system, we consider two separate notions of movement:

- **Rigid motions**: motions that can be described by translating rigid 3D regions.[2]
- **Object motions**: motions of real objects (e.g., pedestrians or cars) that are modeled as compositions of one or more rigid motions.

Throughout the video, we maintain a set of rigid motions. This set may be expanded, to contain newly discovered motions, or contracted, if we find there is no more evidence of a previously seen rigid motion. Multiple rigid motions together can describe highly complex object motions. These means complex object motions such as a walking person is often split into simple rigid motions describing just the motion of the arm or the leg. We maintain a set of such object motions, which typically correspond to real world objects such as cars, pedestrians, or animals. The "background", which is typically the static environment, can be modeled with a single rigid motion. Depending upon the specific task the entire object motion, but also the piece wise rigid motion might be of interest.

Algorithm 2 gives the overview of our main loop. Given the optical flow (Sun et al. [106]), object proposals from SharpMask [86] and information from the previous time steps we first segment the video into its different rigid motions and then use object proposals provided by Sharpmask to segment the video into different object motions.

The main steps of our method are (1) removing rotational flow (Sec. 4.2.1.1), (2) estimating rigid motion components and assigning pixels in each frame to rigid motion components (Sec. 4.2.1.2), (3) grouping rigid motion components into sets to form object models (Sec. 4.2.1.3) and (4) assigning the pixels in each frame to objects for a final segmentation (Sec. 4.2.1.3.4).

[2]Object rotations are not modeled.

Algorithm 2: Estimate motion models and segment frame into objects

```
Input:
    Optical flow.
    Rigid components of previous frame.
    Moving objects of previous frame.
    Assignment history of rigid motions to objs.
    SharpMask object proposals for current frame.
Output:
    Current rigid components.
    Current moving objects.
1  // Estimate rotational flow and remove it 4.2.1.1.
2  // Estimate rigid motion components 4.2.1.2.
3  for each rigid component region from prev. frame do
4      Est. current rigid motion model for that region.
5  end
6  for each pixel in current image do
7      Assign it to a rigid motion model.
8  end
9  // Grouping rigid motion components 4.2.1.3.
10 for moving object mask in object proposals do
11     Assign rigid motion models to object mask.
12     Check consistency with assignment history.
13 end
14 Create object motion models
15 //Assign pixels to moving objects 4.2.1.3.4.
16 for each pixel in current image do
17     Assign it to an object motion model.
18 end
```

4.2.1.1 Removing rotational flow

We seek a camera rotation such that, after subtracting off this rotation from the optical flow, the remaining flow corresponds to purely translational motion (details of this basic idea are described in the previous Chapter 4.1). Unless specified otherwise, all remaining optical flows discussed in this Chapter 4.2 refer to the rotation compensated flow, i.e. the optical flow after camera rotation has been removed.

4.2.1.2 Rigid motions

We want to discover the "set of motions" of rigid structures in the image, and then to determine which pixels belong to each motion, as shown in Figure 4.12 leading to a video segmentation into piece wise rigid motions. The next step of our system is to estimate a set of J rigid motion models M^j, $j = 1 \dots J$, and to assign each pixel in the current image to one of the motion models.

4.2.1.2.1 The rigid motion model. We use the translational angle field as introduced in Chapter 4.1.1 to model the nature of rigid motions. Equation 3.21 essentially leads us to our rigid motion model M, which is a $h \times w$ matrix (h and w are the image height and width), defined by a 3D translational motion (U, V, W). The elements of this matrix are the motion directions at each pixel location (x, y) in the image.

Independence of the set of rigid motion models from the focal length f. The rigid motion model M is dependent upon the focal length (Equation 3.21). Thus, a motion field alone, without the focal length, is not enough to infer the 3D motion direction of an object. While our method segments objects based upon different 3D motions projected on a 2D image plane, it is not important for the method to infer the exact 3D direction; rather, it is only important that for each focal length f there is a unique mapping from 3D directions to rigid motion models (at each pixel location). We show that for any fixed but unknown focal length, each rigid motion model maps to a unique motion direction in 3D. Thus, the rigid motion models are enough to distinguish among *different motions* even though they are not enough to distinguish the exact 3D motion. In other words, if our goal is merely to separate different types of motions, the rigid motion models are sufficient. We present a proof that for any camera focal length f, there is a one-to-one mapping from rigid motion models to 3D motions.

Notation and Preliminaries: Let $S(f)$ be the set of all possible rigid motion models in a static environment for a camera with focal length f. $M(f,T)$ is a motion model defined by the focal length f and the motion direction $T=(U,V,W)$. Let $\mathcal{T}$ be the set of all translational directions, i.e., the set of points on the unit sphere. That is

$$S(f)=\{s : s=M(f,T), T\in\mathcal{T}\}, \tag{4.15}$$

Consider the set S^* of rigid motion models generated by the set of all possible motion directions T when the focal length f is equal to 1. We are interested in the question of how the set $S(f)$ of motion models differs from S^*, due to the difference of focal length.

Theorem 1. *Let f and f' be two different focal lengths. Let $M(f,T)$ be a canonical rigid motion model that results from the focal length f and motion direction T. The same rigid motion model can be obtained for another focal length $f'=cf$ and a different motion direction $T'=(U,V,cW)$, as $M(f',T')$. We show that*

$$M(f,T) \;=\; M(f',T'). \tag{4.16}$$

Thus the direction $\theta(x,y,f,U,V,W)$ at each pixel location (x,y) can be obtained with different focal length $f'=cf$ and a different motion direction $T'=(U,V,cW)$, or

$$\theta(x,y,f',U,V,cW) \;=\; \theta(x,y,f,U,V,W) \tag{4.17}$$

Proof.

$$
\begin{aligned}
& \theta(x, y, f', U, V, cW) && (4.18)\\
= \;& \arctan(cW \cdot y - V \cdot f', cW \cdot x - U \cdot f') && (4.19)\\
= \;& \arctan(c(W \cdot y - V \cdot f), c(W \cdot x - U \cdot f)) && (4.20)\\
= \;& \arctan(W \cdot y - V \cdot f, W \cdot x - U \cdot f) && (4.21)\\
= \;& \theta(x, y, f, U, V, W). && (4.22)
\end{aligned}
$$

□

Since this establishes a one-to-one mapping among rigid motion models governed by the two focal lengths, it establishes that the total set S of rigid motion models is independent of focal length. In particular, while the rigid motion model $M(f, T)$ for a particular motion direction is affected by the focal length, the *set of all possible rigid motion models* S is the same for all focal lengths.

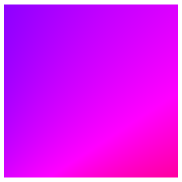
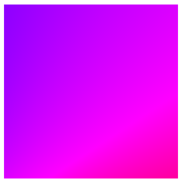
3D trans. direction $[U, V, W]$	**focal length in pixel**	**rigid motion model** M
$[-1, 1, 1]$	1000	
$[-1, 1, 0.001]$	1	

Table 4.3: Independence of the set of rigid motion models from the focal length. Same rigid motion model can be obtained using a different focal length and a different motion direction $[U, V, W]$.

4.2.1.2.2 Estimating a rigid motion model for each rigid motion segment.

We examine the regions from the rigid motion segmentation of the previous frame to estimate a rigid motion model that describes the current optical flow in each rigid motion region. First, we "flow forward" the previous frame's rigid motion regions to obtain the approximate positions of the same rigid structures in the current frame. We then use the optical flow vectors in each region to estimate the motion model by using Horn's method [14], which gives a closed form solution for the best fit to the current translational flow of each region using a least-squares estimation procedure. Given the estimates $[U, V, W]$ for each region, we can substitute them into Eq. 3.21 to obtain a set of rigid motion models for the current frame.

4.2.1.2.3 Assigning pixels to rigid motion models. Given the set of rigid motion models M^j, $j = 1 \dots J$, we assign each pixel to one of the estimated rigid motion models in a Bayesian fashion. Let $\vec{v_t} = (u_t, v_t)$ be an observed translational flow vector at a particular pixel position (x, y), containing only motion due to camera translation and object motion. The current goal is to choose from among J motion models at each pixel location the one with highest probability given the observed flow vector:

$$L_{rigid} = \arg\max_j p(M^j_{xy} \mid \vec{v_t}). \tag{4.23}$$

Each pixel in the image will be assigned to its maximum a posteriori motion model, resulting in the segmentation of a frame into its J rigid motion components. We compute these posteriors using Bayes' rule as

$$p(M^j_{xy} \mid \vec{v_t}) \propto p(\vec{v_t} \mid M^j_{xy}) \cdot p(M^j_{xy}). \tag{4.24}$$

To compute this posterior, we introduce a new model for the flow likelihood $p(\vec{v_t}|M^j_{xy})$ and the prior $p(M^j_{xy})$, details of which are described in Section 4.2.1.4.

4.2.1.3 Object motions

The segmentation L_{rigid} (Eq. 4.23) segments a frame into its rigid motion components. Common objects such as pedestrians, cars or animals show more deformable, articulated and unstructured motion patterns. These type of motion patterns exceeds the complexity one can model using a single rigid motion model. A complex object motion is broken into pieces (segments) if one attempts to model object motion using rigid motions. This over segmentation might not be unreasonable and of interest for certain tasks, however an understanding of an object itself that moves as a whole is not accessible at this point. To be able to model a characteristic motion for an object accurately we incorporate the strength of convolutional neural networks for the task of object detection and segmentation leveraging the semantics of high level image understanding. According to object proposals generated by [86] - a network trained for object segmentation capturing both object-level information as well as low-level pixel data - we join rigid motion models into sets that belong to a specific object. Thus a set of rigid motion models is used to model an object's motion.

Given the rigid motion models M^j, the segmentation L_{rigid} (Eq.4.23) of a frame into J rigid motions and a set of object proposal masks for objects in this frame, we form mutually exclusive subsets $\mathcal{M}^k$ of the rigid motion models M^j. Each $\mathcal{M}^k$, $k = 1 \dots K$ comprises a set of rigid motion models describing a specific object's motion. The steps are as follows:

1. Generate object proposals using the SharpMask segmentation method [86] to create candidate masks of objects and select masks corresponding to moving objects only.

2. Join rigid motion models into sets that belong to a specific object motion guided by semantic segmentations of [86].

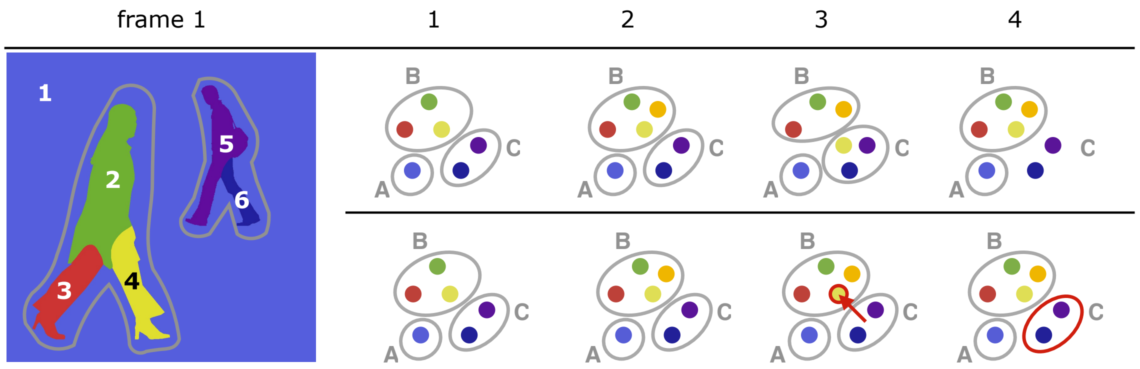


Figure 4.13: Grouping rigid motion models with temporal consistency. Colored dots represent rigid motions, that are grouped to object motions A, B and C. *Top row:* tracking objects over time: (i) Two rigid motion components, dark blue and violet assigned to Object C previously, become isolated in frame 4; (ii) The yellow component suddenly shifts from Object B to C in frame 3. *Bottom row:* time consistent assignment of rigid motions to object motions addresses both of these issues.

4.2.1.3.1 Generating moving object proposals. We first generate a large set of *object proposals* and *objectness scores* using the SharpMask segmentation method [86], and keep the top 100 proposals (based on objectness score). We analyze these object proposal masks and select a subset that best covers the non-background portions of the image, the latter being estimated from the rigid motion models.

4.2.1.3.2 Joining rigid motion models into sets that describe a specific object motion. Given moving object proposal masks and the segmentation L_{rigid}, we could simply assign each motion model M^j to the object proposal mask that has the highest intersection with the rigid motion region corresponding to M^j. However object proposal masks (based on single frames) are not necessarily time consistent – they might arise, disappear or cover part of other objects in single frames. Thus a more sophisticated approach than simply assigning each motion model M^j to its object proposal mask is required. To achieve a temporally consistent segmentation, we address the following three consistency constraints: (1) *tracking objects consistently over time*, (2) *each object owns a consistent set of independently moving parts*, where

each part is modeled by a rigid motion M^j and (3) *motion components* M^j *are assigned to objects they belong to in a time consistent manner*.
We will address consistency constraint (1) and (2). Following up with describing an approach to incorporate consistency constraint (3).

Rigid motions of the current frame are estimated based on the propagated posterior of the previous frame (see Chapter 4.2.1.2.2), thus it is easy to track these motions over time which form the basis of our segmentation system. High level object motions are tracked in a time consistent manner by evaluating key criteria like each object owns a consistent set of independently moving parts. A rigid motion describing the motion of a persons leg is very unlikely to change its owner (the object), unless the leg is a prosthesis the owner changes occasionally. However we call these the rare cases. We track objects by evaluating shared rigid motion components among objects in the current frame and objects detected in the past.
Let Q be the number of object proposal masks (i.e., the output of SharpMask at the current frame) and $q \in 1, .., Q$ its index. Let K be the number of all different objects detected till the current video frame T, indexed by $k \in 1, ..., K$. Given the Q object proposal masks, the segmentation of all frames into rigid motion components $\{L^t_{rigid}\}_{t=1,..,T}$, and the object segmentations from all previous frames $\{L^t_{object}\}_{t=1,...,T-1}$,[3] the problem is to find the lowest-cost way to assign each object proposal mask at the current frame to its corresponding object segmentation. This problem can be represented in a matrix of the *component similarity* - the number of common rigid motion components between the object k and a motion mask q. This leads to a $Q \times K$ matrix. Then the Hungarian algorithm is used to find the best matching such that the component similarity is maximized.

[3]We do not have L^T_{object}, the object segmentation of the current frame, at this point.

The third consistency requirement we address is time consistent assignment of rigid motion models M^j of the current frame to the K objects detected in the video sequence so far. To guarantee a time consistency it is important to assign detected rigid motions to the K objects detected in the video sequence so far, instead of assigning to the Q object proposals of the current frame only, which are independent object proposals based on a single frame. We assign a rigid motion component M^j to an object according to its conditional probability,

$$p(M^j \mid \mathcal{M}_T^k) = \frac{\sum_{t=1}^{T} \mathbb{1}[M_t^j \in \mathcal{M}_t^k]}{T}. \tag{4.25}$$

In words, the probability that a rigid motion M^j is part of the set $\mathcal{M}_T^k$ (set of rigid motions that define a specific object's motion of the current frame T) is the number of frames t, with $tv \in 1, ..., T$, where M^j was assigned to $\mathcal{M}_t^k$, out of the total number of frames seen so far, T.

In summary we first assign rigid motions to Q motion masks of the current frame based on its *component similarity* (top row of Figure 4.13). We then re-assign rigid motions to the K moving objects that have been seen so far (bottom row of Figure 4.13).

4.2.1.3.3 The object motion model. $\mathcal{M}_T^k$ is a set of rigid motion models belonging to a specific object's motion. Each rigid motion model describes part of that object's motion at the current frame T. Let r be the index over elements (rigid motions) in the set $\mathcal{M}_T^k$. We now explain how a new high level object motion model O^k is generated from a set of rigid motion models $M^r \in \mathcal{M}_T^k$.

Similar to a rigid motion model M^j, an object motion model O^k determines a motion direction at each pixel location. M^j often models just a *part* of an object's motion due to its rigidity constraint, whereas the *high level object motion model* overcomes this limitation by modeling the entire object's direction of motion as a whole.

The object motion model O^k is a MAP-estimate at each pixel over the set of rigid motion models in $\mathcal{M}_T^k$. We compute the probability of each rigid motion $M^r \in \mathcal{M}_T^k$ given the observed flow $\vec{v_t}$ at a particular pixel position (x, y) (Eq. 4.26) and assign the most likely motion model to that pixel (Eq. 4.27). An example of this is shown in Figure 4.14.

$$p(M_{xy}^r|\vec{v_t}) = \frac{p(\vec{v_t}|M_{xy}^r) \cdot p(M_{xy}^r)}{p(\vec{v_t})} \tag{4.26}$$

$$O_{xy}^k = \arg\max_{M_{xy}^r}(p(M_{xy}^r|\vec{v_t})) \tag{4.27}$$

4.2.1.3.4 Assigning pixels to moving objects Given the object motion models O^k we segment a frame into its independently moving objects. Similar to how we assign pixels to rigid motion models (Eq. 4.23), the goal is now to choose among K high level object motion models at each pixel location (x, y), the one with highest probability given the optical flow vector $\vec{v_t}$:

$$p(O_{xy}^k|\vec{v_t}) = \frac{p(\vec{v_t}|O_{xy}^k) \cdot p(O_{xy}^k)}{p(v_t)}. \tag{4.28}$$

This leads to a moving object segmentation,

$$L_{object} = \arg\max_k(p(O_{xy}^k|\vec{v_t})). \tag{4.29}$$

Likelihoods and priors are computed similarly as for the segmentation procedure of a frame into rigid motion components (Equation 4.23) and are derived in the following section 4.2.1.4.

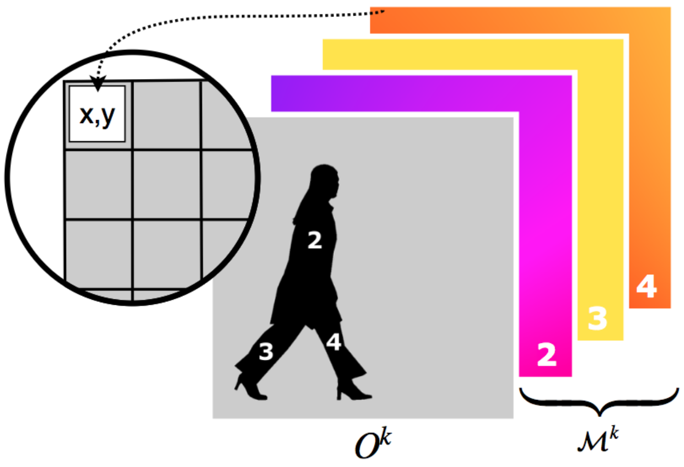


Figure 4.14: The object motion model. In this figure, the k-th object's motion *(walking person)* is described by three rigid motion models forming the set $\mathcal{M}^k = \{M^2, M^3, M^4\}$. The object motion model O^k is a MAP-estimate at each pixel (x, y) over rigid motion models in $\mathcal{M}^k$.

4.2.1.4 Flow likelihood and prior

In this section we describe our new flow likelihood and a prior that are used to assign to each pixel its most likely motion model. The motion model can be either a rigid motion model M^j or an object motion model O^k, the principle how to compute the flow likelihood in both cases remains the same.

4.2.1.4.1 Flow likelihood. Let $\vec{q} = (r, \theta)$ be the *true translational motion field vector* (with magnitude r and angle θ), representing the motion field at a particular pixel location less the component due to camera rotation. Let $\vec{v_t}$ be the translational component of the *observed* [4] *optical flow vector* $\vec{v}$. We model $\vec{v_t}$ as a noisy observation of $\vec{q}$:

$$\vec{v_t} = \vec{q} + \vec{n}. \tag{4.30}$$

[4]We refer to the flow vector as "observed", but it is the output of an optical flow algorithm which has access to a pair of frames.

Inspired by [41], we model flow noise $\vec{n} = (n_u, n_v)$ as a product of Laplacian distributions (for the u and v components), where the parameters depend upon the motion field magnitude r:

$$\vec{n} \sim \mathsf{Laplace}(b_{n_u}(r)) \cdot \mathsf{Laplace}(b_{n_v}(r)). \tag{4.31}$$

With these assumptions we derive our new flow likelihood, the probability of $\vec{v_t}$ given a rigid motion model M^j (or given an object motion model O^k, respectively):[5]

$$\begin{aligned}
p(\vec{v_t} \mid M^j_{xy}) &= \int_0^\infty p(\vec{v_t}, r \mid M^j_{xy})\, dr && (4.32)\\
&= \int_0^\infty p(\vec{v_t} \mid r, M^j_{xy})\, p(r \mid M^j_{xy})\, dr && (4.33)\\
&\overset{(a)}{=} \int_0^\infty p(\vec{v_t} \mid \vec{q})\, p(r \mid M^j_{xy})\, dr && (4.34)\\
&\overset{(b)}{=} \int_0^\infty p(\vec{n}; r)\, p(r \mid M^j_{xy})\, dr. && (4.35)
\end{aligned}$$

The equality (a) follows since the motion field vector $\vec{q}$ is just a combination of the motion field magnitude r and the motion direction M^j_{xy}. The final equality (b) expresses the fact that the only uncertainty in $\vec{v_t}$ is due to the flow noise $\vec{n}$. The noise variance depends upon r. Parameters of the flow noise distribution are estimated from the Sintel database [15], details of which can be found in Section *Modeling the flow noise.*

$p(r \mid M^j_{xy})$ is the probability of flow magnitude r given a particular motion direction M^j_{xy}. We assume that $p(r)$ is independent of the flow direction θ and approximate it as an exponential distribution with parameter b_r:

[5]We define the likelihood of a "new motion" that was not observed before to be $p(\vec{v_t} \mid M^{new}) = \frac{1}{2\pi}\int_0^{2\pi} p(\vec{v_t} \mid M)\, dM$. The likelihood of a new motion direction is the average likelihood over all possible motion directions.

$$p(r \mid M_{xy}^j) \approx \mathsf{Exp}(r; b_r). \tag{4.36}$$

The scale parameter b_r is learned using the FBMS-59 training data set [12, 5]. We discuss the relationship between the variance of the flow noise and the magnitude r of the motion field in the subsequent paragraph.

Modeling the flow noise. We use the ground truth optical flow provided by the Sintel [15] data set for modeling the characteristics of optical flow computed by the algorithm of Sun et al. [106].

We measure the variance of the observed flow noise for different magnitudes r of the ground truth flow. Figure 4.15 shows four histograms of the flow noise (u-component) for different ground truth flow magnitudes. The last plot shows the observed variances as blue dots and in red the exponential function that best models the relationship between flow noise variance and the motion field magnitude r. A significant relation between the variance of the flow noise and magnitude can be observed – the larger the flow magnitudes, the larger the covariance of the flow noise. For large pixel displacements the computation of optical flow becomes very noisy. To incorporate this relationship into our model, we model the variance as a function of r with an exponential function of the form $s(r) = a \cdot e^{br}$.
The least squares fit for a and b are:[6]

$$\mathrm{Var}(n_u(r)) : a = 1.145 \times 10^{-4}, b = 35.85 \tag{4.37}$$

$$\mathrm{Var}(n_v(r)) : a = 1.635 \times 10^{-4}, b = 45.8 \tag{4.38}$$

Additionally we introduce a multiplier m, to add flexibility to our noise model. This is supportive for real world videos, since the measurements rely on the synthetic action

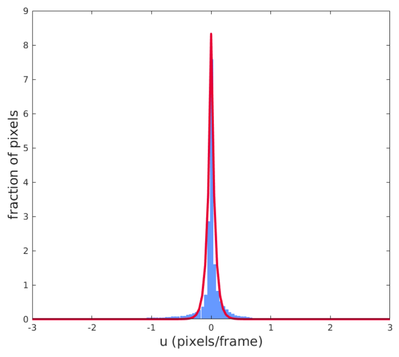
[6]Parameters a and b are measured based on the *normalized flow* – the flow relative to the frame size.

movie Sintel which comes with additional challenges, like textureless regions, artificial motion blur effects and large pixel displacements. We learn the parameter m using the FBMS-59-3D motion training data set [4, 77].

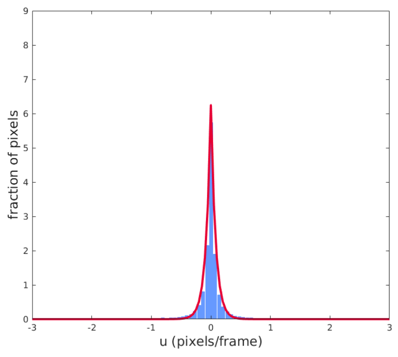
4.2.1.4.2 Prior. The prior $p(M^j_{xy})$ on a particular rigid motion model at each pixel includes information about the posterior probability of each motion from the previous frame (the *motion prior*) and another factor that restricts the position of that component in the next frame to a position close to its expected position (the *location prior*). The location prior is important because each rigid motion model should model exactly one rigid motion component of an object. The number of rigid motion models correspond to the number of existing rigid motion components in the image. This way a new motion gets only introduced by a prior for a new motion and not by priors for existing motion components.

Motion prior. To get a rough estimate about the motion modeled by M^j we proceed as follows: (1) We propagate the posterior of $p(M^j_{xy}|\vec{v_t})$ from the previous frame along the previous frame's optical flow. (2) We interpolate regions of disocclusion by iteratively smoothing from adjacent unoccluded regions. (3) Then we spatially distribute the probability that each motion component is presents by smoothing the prior with a 7x7 Gaussian.

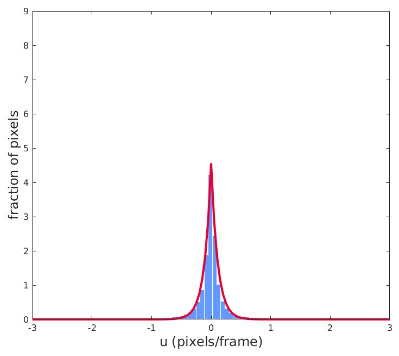
Location prior. The location prior restricts the location of a motion component to being near its former location. If there are multiple rigid motion components with similar motion, it is important that each object motion is described by its own set of rigid motion components. A rigid motion model cannot be shared among multiple objects. Therefore we propagate the hard segmentation from the previous frame and distribute it spatially in a manner similar to the motion prior.



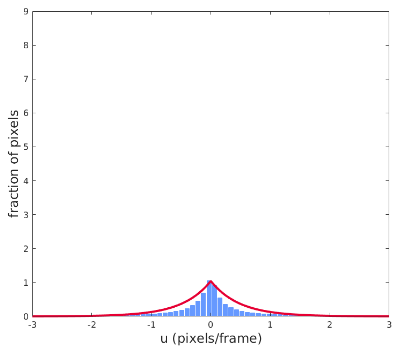
(a) first quartile



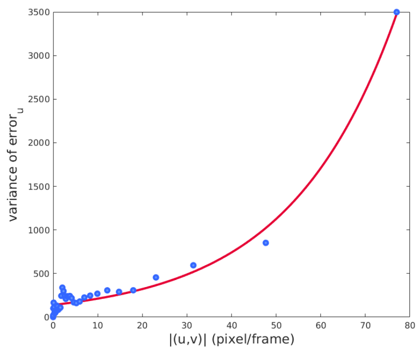
(b) second quartile



(c) third quartile



(d) fourth quartile



(e) variance of flow noise

Figure 4.15: Variance of flow noise. (a)-(d): Histograms of the optical flow noise of the first, second, third and fourth quartile of motion field magnitudes (Q_1, Q_2, Q_3, Q_4). (e): Visualization of the dependence of the flow noise variance and the corresponding motion field magnitude r. The blue dots show the flow noise variance for a particular motion field magnitude.

	Motion Segmentation							
	Testset: FBMS-motion (30 sequences)				**complex background** (5 sequences)			
	P	R	F	ΔObj	P	R	F	ΔObj
[48]	**74.64**	**62.03**	**63.59**	**7.7**	**67.62**	**58.28**	**60.27**	**3.4**
[109]	72.69	54.36	56.32	11.7	60.79	44.74	45.83	**3.4**
ours+CRF	**74.23**	**63.07**	**64.97**	**4**	**64.85**	**67.28**	**65.60**	**3.4**

	camouflaged animals (9 sequences)				**all** (44 sequences)			
	P	R	F	ΔObj	P	R	F	**ΔObj**
[48]	77.78	**68.10**	**69.97**	**5.7**	**74.48**	**62.84**	**64.52**	**6.8**
[109]	**84.71**	59.40	61.52	22.2	73.80	54.29	56.19	12.9
ours+CRF	**83.84**	**69.99**	**72.15**	**5**	**75.13**	**64.96**	**66.51**	**4.1**

Table 4.4: Motion segmentation: Comparison to state-of-the-art. We compare motion segmentation approaches (multi-label) [48, 109]. Best viewed in color (**1st-best**, **2nd-best**).

4.2.2 Experiments

We evaluate our work on three motion segmentation data sets: FBMS-59 [12], the Complex Background data set [72], and the Camouflaged Animals data set [5]. As discussed in [4], FBMS-59 shows a significant number of annotation errors. We use a corrected version of the data set that is linked on the original data set's web site. Our main results are for multi-label segmentation, but we also convert our results to a binary segmentation form for comparison with previous work on binary motion segmentation. In addition, we show segmentation results of each stage of our moving object segmentation algorithm – segmentation into rigid motion models (rMM), segmentation of the video using object proposals mask of SharpMask directly ($objP$), segmentation of the video using a constant variance of the optical flow error for all flow magnitudes ($cVar$) and results of our final moving object segmentation algorithm *(ours)*.

4.2.2.1 Evaluation scheme

We adopt the multi-label evaluation scheme from [77] and add an additional measure ΔObj that represents the accuracy of the segmented object count. ΔObj is the average absolute difference of the ground truth object count in each frame and the number of objects identified by the algorithm. A drawback of the evaluation scheme proposed by [77] is that it does not penalize algorithms much for large numbers of unnecessary (additional) segmented objects. If there is a large background in the image and the algorithm identifies 10 false positive moving objects, these will only affect the score for the background region according to the proportion of the area taken up by the false positive objects. Thus, the F-score of [77] alone does not entirely capture whether the algorithm has an accurate count of the number of objects and the additional ΔObj measure is necessary for a representative evaluation.

	Multi-label Video Segmentation				**Binary Video Segmentation**		
	all (44 sequences)				**all** (44 sequences)		
	P	R	F	ΔObj	P	R	F
cVar	**76.43**	62.19	64.86	**3.4**	**85.78**	81.09	81.15
rMM	76.01	50.11	52.69	85.88	81.05	81.81	78.91
objP	-	-	-	-	77.15	**85.03**	78.78
ours	74.75	64.70	66.45	4.3	83.66	82.68	81.27
ours+CRF	75.13	**64.96**	**66.51**	4.1	84.72	82.65	**81.49**

Table 4.5: Ablation study: Intermediate results of our motion segmentation algorithm. We compare five versions of our algorithm to show how each part of the algorithm affects the performance of the overall motion segmentation method.

4.2.2.2 Results: Multi-label motion segmentation

We outperform [48, 109] by significant margins on FBMS-59, Complex Background and Camouflaged Animals data set (see Tab. 4.4). The Complex Background data set shows videos with high variance in depth, which is particularly challenging for trajectory based motion segmentation approaches such as [48], as well as for

	Binary Motion Segmentation					
	Testset: FBMS-motion (30 sequences)			**complex background** (5 sequences)		
	P	R	F	P	R	F
[5]	79.94	80.76	77.33	84.31	**91.74**	**86.56**
[81]	83.86	79.96	79.56	**87.57**	84.95	80.64
[22]	86.24	76.25	77.33	79.91	69.31	73.65
[110]	**87.29**	72.19	74.79	86.78	77.49	78.19
[111]	**92.40**	**85.07**	**86.96**	74.58	77.02	70.52
ours+CRF	85.53	**83.14**	**81.85**	**87.69**	**93.13**	**90.11**

	camouflaged animals (9 sequences)			**all** (44 sequences)		
	P	R	F	P	R	F
[5]	**81.86**	**74.55**	**76.31**	80.83	**80.74**	**78.17**
[81]	73.31	56.65	60.38	82.12	75.76	75.76
[22]	**82.34**	68.45	72.48	84.72	73.92	75.91
[110]	77.82	62.03	64.84	**85.30**	70.71	73.14
[111]	77.62	51.08	50.82	**87.35**	77.20	77.67
ours+CRF	80.37	**75.21**	**75.95**	84.72	**82.65**	**81.49**

Table 4.6: Binary motion segmentation: Comparison to state-of-the-art. We compare to binary [22, 5, 81, 110] motion segmentation approaches. Best viewed in color (**1st-best**, **2nd-best**).

occlusion-based object segmentation approaches [109]. Over all the videos in these data sets combined, we gain an average improvement of 2% in F-score compared to the second best performing segmentation method [48]. Our ΔObj results are on par or better for Complex Background and Camouflaged Animals; on FBMS, we are more accurate than either of the other methods in segmenting the correct number of objects (Fig.4.16 for qualitative results).

4.2.2.3 Results: Binary motion segmentation

In these experiments, we segment each frame into either static background or moving objects, but do not distinguish among the moving objects, enabling us to compare to other methods that address the binary segmentation problem. We outperform other methods based on overall F-score and recall, and on all three performance metrics on the Complex Background data set. On FBMS we are in second

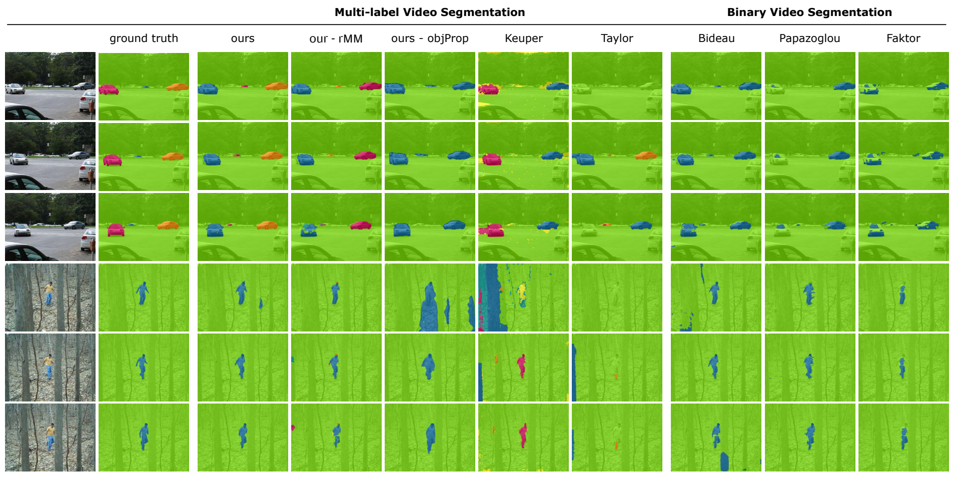


Figure 4.16: Qualitative segmentation results. Row 1-3: *cars5* from the FBMS-59 test set. Rows 4-6: *forest* from the complex background data set. For both videos we show frames 1, 10 and 20. We show results on our final version of our algorithm ("ours") as well as for intermediate results of our final algorithm ("ours - rMM" and "ours - objProp"). "ours - rMM" shows the segmentation of a frame into its rigid motions. "ours - objProp" shows the object segmentations produced by SharpMask [86], which are used to join rigid motions to object motions. Comparisons to state of the art methods on multi-label segmentation and binary segmentation are shown in columns 6-10 [48, 109, 5, 81, 22].

place behind Tokmakov et al. [111] and on Camouflaged Animals the method from Bideau et al. [5] is slightly better (0.36%) than ours. On average over all videos we have a lead of 3.32% over the next best method [5].

4.2.2.4 Ablation study

To show the contribution of each part of our algorithm separately, we evaluate intermediate results of our method and specific variants, shown in Tab. 4.5:

1. *Constant variance (cVar):* Modeling the variance of optical flow error as a function of the optical flow magnitude leads to an improvement of about 2% over all data sets. Regarding precision and ΔObj, we outperform our final motion segmentation approach – cVar segments fewer objects and, due to less false positives, the precision increases. However, the overall performance is worse due to low recall.

2. *Segmentation into rigid motion models (rMM):* Simple rigid motion models are not sufficient to model complex object motion. After the first stage – segmentation of a frame into its rigid motion models – complex motion patterns are broken into multiple simple rigid motion models. Thus, it is not surprising that ΔObj increases dramatically to 85.55.
3. *Segmentation into moving object proposals (objP):* Moving object proposals are generated from a subset of the object proposals out of SharpMask[86]. In Figure 4.16 ("ours - objProp"), it can be seen that the obtained proposals are covering the object completely (high recall); however the object boundaries are very rough. Those inaccurate boundaries – where a large part of the static background is segmented along with a moving object – lead to low performance. Therefore a composed motion model for modeling the motion of an object accurately is necessary and leads to an improved performance.[7]
4. *Conditional Random Field (ours+CRF):* We add a fully-connected CRF [50] on top of our method to refine the segmentations [110, 16]. The CRF hyperparameters were set by cross-validation on the FBMS Training set.

4.2.3 Summary

Many previous methods have shown impressive results in motion segmentation using just low-level or low and mid-level cues [5, 48, 25, 81, 109, 78, 76, 134, 22]. Like recent work in optical flow [100] that uses the power of CNNs to condition optical flow on semantic regions, it seems logical to incorporate this type of high-level information into motion segmentation. We presented an hierarchical statistical method that leverages perspective geometry to model low level parts and semantic segmentation results from a CNN, and combines these parts in a logical way to form higher level objects.

[7]Since the object proposal masks of SharpMask might be overlapping or describe the same object, an evaluation of multi-label segmentation is not directly possible for *objP*.

We demonstrated best average results across three major motion segmentation data sets and showed strong performance on a wide variety of challenging videos.

CHAPTER 5
LEARNING MOTION PATTERNS

Modern learning based methods of motion analysis excel at identifying well-known structures, but may not precisely characterize well-known geometric constraints. In this Chapter we present an approach to learn highly variable motion patterns using a convolutional neural network. We start with answering the following question: *is there a formal and principled description of a moving object, that one could use as a recipe to learn what a moving object is?* If one had this kind of recipe maybe one could generate training data accordingly in a synthetic manner to train a neural network for the motion segmentation task. This leads us to our following work presented in Chapter 5.1 and 5.2 which attempts to learn motion patterns of object's using a neural network.

- **Learning highly variable motion patterns in a self-supervised manner for moving object segmentation [6].** A possible high level description of a moving object in a static scene could sound as follows: *A moving object is a connected image region that undergoes some independent motion. The connected image region can be of any size and shape.* Having a good motion model for the camera motion (a rigid motion model as introduced in Chapter 4.2.1.2) and a motion model for moving objects (following ideas presented in Chapter 4.2.1.3), one can synthesize large amount of training data consistent with the definition of a moving object and consistent with true physics of perspective projection.

  This approach has been shown very promising results on synthetic (perfect) flow fields. In particular, in scenarios where the scene is rather complex and objects are located at different depths our approach outperforms state of the art motion segmentation methods trained on original flow fields directly.

- **Learning object motion from noisy estimated motion fields** Making the transfer from synthetic data to realistic noisy flow fields remains still challenging. How can one train a network being conform with the true physics of the world (scene structure) and interpreting independent objects motions robustly. This is topic of Chapter 5.2.

5.1 Self-Supervised Motion Segmentation

The human visual system has an incredible ability to detect motion, regardless of its complexity. While we are moving through the world our eye captures an enormous number of images over time. Images are projected onto our retina and the perceived motion (image change over time) is processed by the brain. In computer vision, optical flow is used to describe the motion between two consecutive images. Low level optical flow methods are based on two images alone [68, 37, 106, 13, 96, 107]; other methods attempt to incorporate object knowledge and the knowledge about object motions [100, 129, 38]. In this work we propose an approach to learn motion segmentation given a motion field as input. A motion field describes the (perfect) motion between two consecutive frames, where as the optical flow is its noisy estimate.

For the human visual system little eye movements play a key role in simplifying the motion field on the eye's retina and making it easier to interpret for our brain [122]. Motion fields produced by eye motions (rotations) contain no information about the scene's geometry (see Equation 3.19) and thus can be used for motion compensation without adding or reducing critical information. The two major reasons for eye movements are (1) to stabilize vision and (2) to change direction of gaze.

In this work we aim to develop an approach that accurately interprets the perceived motion field on the retina. Inspired by visual ecology, we start with vision

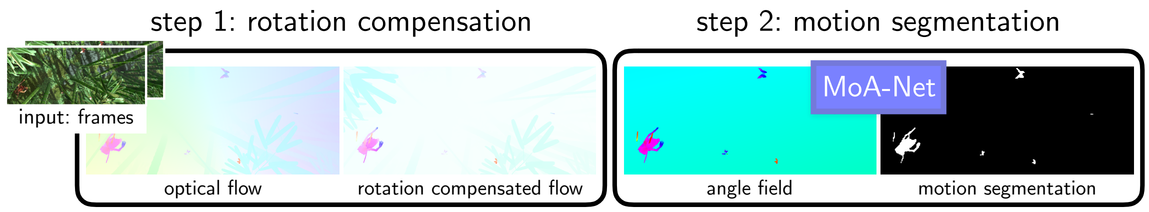

step 1: rotation compensation
step 2: motion segmentation
input: frames
optical flow
rotation compensated flow
MoA-Net
angle field
motion segmentation

Figure 5.1: An overview: Self-supervised motion segmentation. Given a motion field our goal is to segment a frame into independently moving objects and static environment. Due to the complexity of motion fields, previous neural network models have had difficulty segmenting motion directly, in addition semantic cues to handle challenging motion cases. To deal also with highly complex motion field we use a two step approach (like in previous work [5]), which first involves adjusting the optical flow for camera rotation (left) and then segments the angle of the compensated flow into static environment and moving objects (right). We show strong potential of training a network to segment the angle field rather than the raw motion field to be able to mutually interpreting true physics of the scene and motion correctly.

stabilization before processing the motion field to segment independently moving objects. Of course one is not able to receive an image directly from the human's eye. Instead one typically uses video sequences taken by a camera and methods to estimate the optical flow between two consecutive frames.

Unlike most learning-based approaches, we are not relying on labeled training data, which is limited. Instead we carefully analyse the underlying geometry of the motion field and break down the problem of motion segmentation into two subproblems: compensating the motion field for rotation (similar to vision stabilization of our eye movements) and segmenting the remaining motion field into static background and moving objects. The step of compensating the image motion for camera rotation is a challenging step especially because only a noisy estimate of the motion field is accessible for real world videos [7, 5]. Estimating the camera rotation given the optical flow as input is not further explored in this work; possible approaches to estimate the camera rotation are presented in [7, 5].

As stated already in several previous literature [40, 72, 7, 5] the remaining motion field (after compensating for rotation) has a well interpretable geometrical pattern. We follow geometrical principles behind the rotation compensated flow to synthesize training data in large amounts while consistently following rules of perspective projection. In this way we do not rely on any training data set for motion segmentation, which are limited in size, the variety of shown scene structures, or quality.

Our contributions are as follows:

- Inspired by visual ecology, we present a two step approach for motion segmentation, which first involves compensation of the optical flow for camera rotation and then segments the compensated flow into static environment and independently moving objects. While this two step approach is a well established approach for motion segmentation [7, 5, 40], we present in this Chapter its great potential for *learning based* video segmentation methods. We aim to leverage the strength of classical geometrical approaches (based on perspective projection) and learning based approaches for motion segmentation.

- For evaluation purposes, motion segmentation ground truth for the optical flow data set Sintel [15] is generated.

- A new self-supervised training approach is presented that does not rely on limited training data. Instead the problem of motion segmentation is broken down into two smaller subproblems. Guided by perspective projection, we provide a principled (abstract) definition of a moving object, which allows us to generate an unlimited amount of training data in a synthetic way that covers the fundamental principles of motion.

- We show state-of-the-art performance on ground truth optical flow (the motion field) of Sintel [15] and FlyingThings3D [71].

This Chapter is organized as follows. The principles behind the formation process of a motion field (see Chapter 3) lead us to our approach of training a neural network for motion segmentation as described in Chapter 5.1.1. Rather than relying on labeled training data, synthetic training data is automatically generated considering the geometry of optical flow. We explain the automatic generation procedure for training data in chapter 5.1.1.2.1. In Chapter 5.1.2 we evaluate our motion segmentation network and compare its performance to two other networks for motion segmentation.

5.1.1 Methods

Motion patterns of the motion field are often quite difficult to interpret directly. Camera rotation and translation couple the scene depth, which makes it impossible to judge whether an object is moving or not. Motion magnitude as well as direction are dependent on camera motion, object motion and depth, when the camera is rotating and translating simultaneously. Inspired by visual ecology and the purpose of human eye movements, we use a two step approach for motion segmentation (see Figure 5.1). The two steps are as follows:

1. Compensate the motion field for rotation
   - Compensate the motion field for the rotational component of the observer's motion, similar to the way that image stabilization is done on the human retina, which is done via small eye rotations. The rotation compensated motion vector is $\vec{v}$.
2. Segment the motion field into static environment and moving objects
   - Given a motion vector $\vec{v}$ compute its direction θ at each pixel location (Equation 3.21).

- A neural network MoA-Net (*Mo*tion *A*ngle - *Net*work) takes an angle image as input and generates per-pixel motion labels.

Rather than having the network learn complex geometrical dependencies, the fundamental idea is to break down the motion field into a pattern that is easier to interpret. The input to the network - the *angle image* - is simpler and contains all of the motion information that can be obtained from motion field. Rather than adding additional information we reduce the amount of information required to train a neural network successfully to its minimum.

In this work we assume the rotation to be known and present an approach that automatically segments the motion field into static environment and moving objects.

5.1.1.1 Network architecture

Our basic network architecture is adopted from [110, 111]. It its a U-Net architecture [97], which is a well established architecture for image segmentation. Originally [110, 111] the networks input was the motion field's angle and magnitude - leading to a three dimensional input of size $[height \times width \times 2]$. Instead our network takes the angle image of the rotation compensated motion field, which just has two dimensions $[height \times width]$, as input. The angles are in the range of $[-\pi, \dots, \pi]$. The network is trained using the binary cross-entropy loss. Its output are "soft" motion segmentation masks with values in the range of $[0, 1]$, which are rounded in a second step to obtain binary motion segmentation masks for final evaluation.

5.1.1.2 Training: Incorporating the Basics of Perspective Projection

Training a neural network for the task of motion segmentation usually requires large amounts of optical flow and its corresponding motion segmentations. The problem of using those data sets for training is that those data sets are often limited in size and the variety in scene geometry and motion is often restricted.

Rather than relying on these limited data set we present an approach to generate training data in an automatic way incorporating the physics of perspective projection and independent motion.

5.1.1.2.1 Generating Training Data We start with a definition of a moving object and guided by this definition we introduce a procedure to generate training data to learn to segment independently moving objects.

> ***Definition: Moving Object.*** *A moving object is a connected image region that undergoes some independent motion. The connected image region can be of any size and shape.*

True object motion can be quite complex, since objects can be deformable and articulated. If an object is articulated, each part might move independently of the other parts, e.g. a walking person. In case of a walking person, one arm might move forward while the other is standing still - here, although the body parts are physically connected, each part can move relatively independently of each other. The static environment undergoes a single pure translational motion due to the observers motion. Training data should contain these key criteria reflecting object motion along with observer motion.

We generate training data for motion segmentation in 5 steps:

1. Generating connected object regions: To cover a large variety of different possible shapes and sizes, we use the binary segmentations masks of FlyingThings3D [110, 71] (Figure 5.2(a)).

2. Modeling articulated object motion: To model object motion, each object region is split into n subregions using superpixels. n is a random number between one and ten. Splitting objects into subregions as shown in Figure 5.2(b) leads to

multiple different motion regions. In Figure 5.2(b) we have eight motion regions including the region of static environment.

3. We assign to each motion region a translational 3D direction (Figure 5.2(c)). A 3D translational direction is represented as a 3D unit vector. We generate a set of equally distributed translational motion direction on a sphere using the vertices of an icosahedron as approximation. Each vertex of an icosahedron represents a translational motion direction. To generate a large set of possible translational motion direction, we generate an icosahedron of frequency 50 which has 25002 vertices representing the set of translational motion directions.

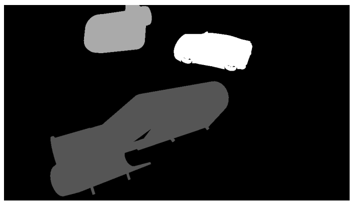
4. Smoothing motion boundaries: To smooth motion boundaries within an object, we use a Gaussian filter with standard deviation $\sigma = 50$ (Figure 5.2(d)). Object boundaries remain sharp.

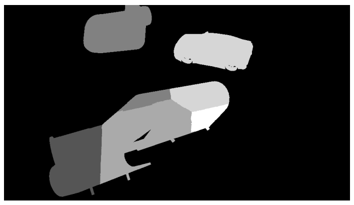
5. We add random Gaussian noise with zero mean and standard deviation $\sigma = 0.1$ (Figure 5.2(e)).

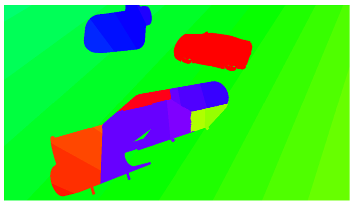
This procedure to generate training data is entirely independent of any color images or other labeled training data. It incorporates all geometrical information required to segment independently moving objects. This abstraction - reducing objects to *connected image regions* that undergo *independent motion* - allows us to train a network with unlimited training data in a self-supervised manner.

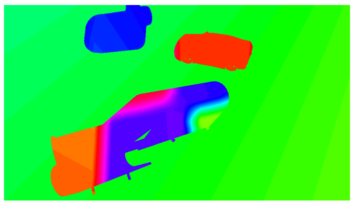
5.1.2 Experiments

We evaluate our work on Sintel [15, 128] and FlyingThings3D [71]. These data sets are briefly discussed below. We generated additional motion segmentation ground truth for Sintel to use this data set for evaluation. Both data sets provide camera motion information, which allows us to evaluate the performance of MoA-Net, which requires flow angles of the rotation compensated flow field as input. We compare our


(a) Generating connected object regions


(b) Splitting object regions into motion regions


(c) Assigning 3D translational directions


(d) Smoothing motion boundaries

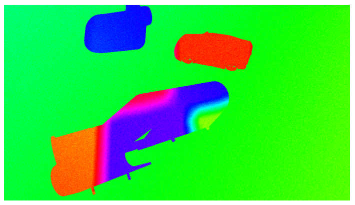

(e) Adding Gaussian noise

Figure 5.2: The process of generating training data for motion segmentation. The abstract object definition reduces an object to *connected image regions* that undergo *independent motion*. (a)-(e) show the process of generating abstract objects for the motion segmentation task.

work to two recently published motion segmentations approaches [111, 42]. Both approaches are learning based approaches that attempt to learn motion patterns given the optical flow as input. In combination with a neural network that produces object segmentations based on appearance, both approaches have shown great results on a variety of different data sets [83, 12, 77, 108, 90, 43, 59]. For comparison purposes, we extract the motion segmentation network of both works and compare their performance on ground truth optical flow with our proposed method. The very modular motion segmentation pipeline of Tokmakov et al. [111] as well as of Jain et al. [42] allows us to analyze their "motion-stream" exclusively.

FlyingThings3D [71] is a relatively large synthetic flow data set comprising 2700 videos, containing 10 stereo frames each. Along with these videos, ground truth optical flow, disparity, intrinsic, extrinsic camera parameters and object instance segmentation masks are provided. However this data set doesn't picture realistic scenarios - random objects like tables, chairs and cars are flying in the 3D world.

Sintel [15, 128] is a well-known optical flow data set, containing 23 video sequences with 20 to 50 frames each. These short video sequences are taken from the computer animated movie *Sintel*. Scenes are relatively realistic simulated. Videos come with ground truth optical flow, depth, intrinsic and extrinsic camera parameters and material segmentation.

Compensating for camera rotation Besides ground truth optical flow, Sintel and FlyingThings3D provide ground truth extrinsic and intrinsic camera matrices. This allows us to compensate the flow for camera rotation. (1) We move image coordinates x_t along the optical flow and obtain new image coordinates x_{t+1}. (2) The new image coordinates x_{t+1} are transformed into 3D camera coordinates X_{t+1}. (3) Given the ground truth camera motion (rotation and translation) between two consecutive frames, we undo the camera rotation in 3D space. (4) The new camera coordinates X_{trans} (after undoing the camera's rotation) are projected back onto the 2D image plane. (5) The rotation compensated flow can be obtained from the pixel displacement between image coordinates x_t and x_{trans}.

5.1.2.1 Evaluation

We use the evaluation scheme of [83]. We show results on two different motion segmentation networks and compare their performance with our motion network on Sintel and the test set of FlyingThings3D.

Jain et. al train a motion segmentation network given rgb-flow images as input. For training, they used estimated optical flow images in rgb-format. Since no motion segmentation are available for ImageNet [98], they propose a procedure to produce (pseudo)-ground truth segmentations based on the provided object bounding boxes, the segmentations of their appearance network and the appearance of the estimated optical flow. Flow images are discarded from the training set, if average rgb-flow inside an object bounding box differs not sufficiently from the background's optical

Motion Segmentation: Sintel						
	Motion					
	J Mean ↑	J Recall ↑	J Decay ↓	F Mean ↑	F Recall ↑	F Decay ↓
Tokmakov et al. [110, 111]	**50.38**	**55.43**	**45.32**	**52.43**	**54.95**	45.58
Jain et al. [42]	30.27	24.78	32.72	28.07	14.02	**31.89**
ours	**55.13**	**55.24**	**26.62**	**59.94**	**61.67**	**16.76**

Table 5.1: Self-supervised motion segmentation: Comparison to state-of-the-art. We compare our motion segmentation network with two recent motion segmentation networks that segment optical flow into static background and independently moving objects. Best viewed in color (**1st-best** , **2nd-best**).

flow. Their segmentations are rather conservative - they often segment just a small portion of the moving object or nothing, which leads to an overall low performance of their motion segmentation network. On both data sets - Sintel and FlyingThings3D - their performance is rather low. One might argue that moving objects in Sintel and FlyingThings3D are quite different from objects that the network trained on ImageNet has seen before. Also, their automatic procedure to generate (pseudo)-ground truth significantly limits the variability of motion fields.

Tokmakov et. al trained their network on ground truth optical flow provided by the FlyingThings3D data set. Each flow vector is represented using polar coordinates (flow magnitude and angle) during training. On Sintel as well as FlyingThings3D they show overall a good performance. If a video scene shows high variance in depth as in the bamboo video sequences of Sintel (Figure 5.3(c) and 5.3(d)), their segmentation is highly depth dependent, which leads to erroneous motion segmentations. Especially in those cases, MoA-Net outperforms both other motion segmentation networks by a large margin.

MoA-Net (ours) is trained purely on translational angle fields, which are generated in a synthetical manner as described in Chapter 5.1.1.2.1. This allows for producing motion segmentations that are completely independent upon the scene depth. Since

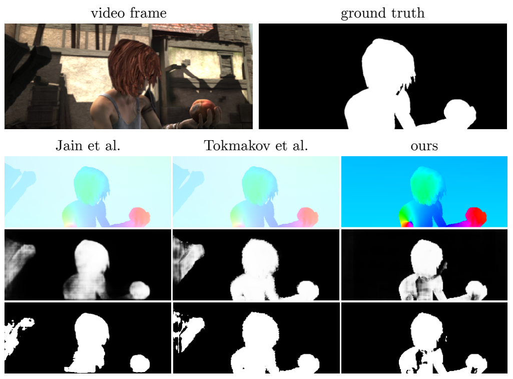
Motion Segmentation: FlyingThings3D-Test						
			Motion			
	J Mean ↑	J Recall ↑	J Decay ↓	F Mean ↑	F Recall ↑	F Decay ↓
Tokmakov et al. [110, 111]	89.13	98.40	**-2.11**	93.55	98.54	**-2.29**
Jain et al. [42]	21.57	6.47	2.51	30.04	8.77	1.85
ours (flow angle FT3D)	**91.12**	**99.78**	-0.02	**94.33**	**99.63**	-0.41
ours (self-supervised)	75.53	95.76	3.55	82.25	97.65	1.68

Table 5.2: Comparison of motion networks trained on different training data and tested on FlyingThings3D-Test. Tokmakov et al. and ours-FT3D are trained using the provided ground truth optical flow of FlyingThings3D, Jain et al. relies on estimated optical flow of a subset of videos from ImageNet, and ours is trained on fully automatically generated training data as described in 5.1.1.2. Best viewed in color (**1st-best**, 2nd-best).

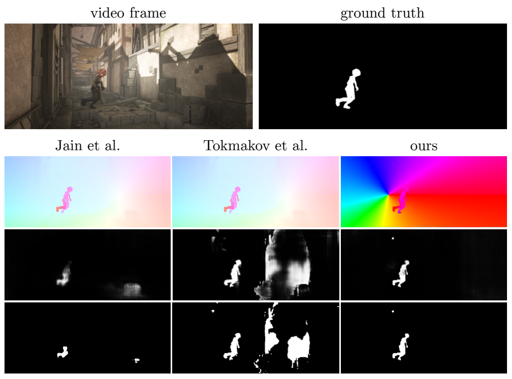
the motion of moving objects is only approximated during training using multiple rigid motion models (3D translational direction) there is some deviation from real world motions, e.g. real world motions might also have sharp motion boundaries within a moving object. This leads to failure cases in certain situations.

5.1.2.2 Results: Binary motion segmentation

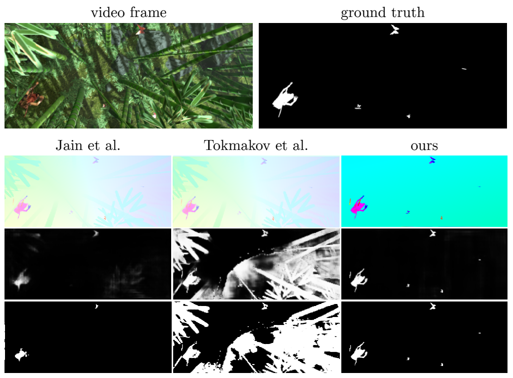
On Sintel we outperform Tokmakov et al. by 4% points using the J-Mean metric and by more than 7% points regarding the F-Mean (see Table 5.1). On FlyingThings3D-test, the motion segmentation network of Tokmakov et al. produces high quality motion segmentation masks. Their accuracy in terms of IoU differs from their performance on Sintel by a large margin (39% points). This significant difference is very likely due to the similar nature of training and test data (their network is trained on FlyingThings3D-train). When our MoA-Net is trained on the same ground truth flow as Tokmakov et al., but using only the optical flow's angle after compensating for camera rotation, we outperform their method (91.12% versus 89.13% - see Table 5.2)). Our proposed motion segmentation network, however, is trained in a self-supervised manner. We show significantly better performance than Jain et al.



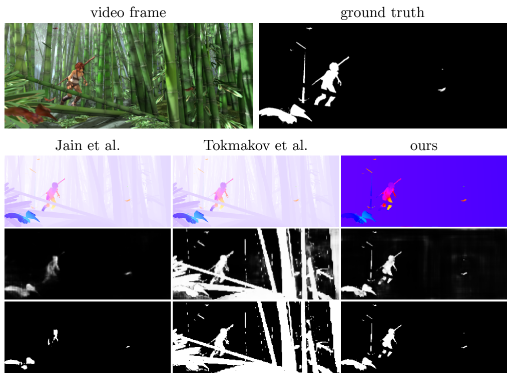
(a) Sintel - alley1



(b) Sintel - alley2



(c) Sintel - bamboo1



(d) Sintel - bamboo2

Figure 5.3: *first row*: input frame and ground truth motion segmentation. *Second row*: input to the motion segmentation network of the two different methods used for comparison an our input - optical flow as rgb image, optical flow in its angle and magnitude representation, angle of the rotation compensated flow. *Third row*: raw motion network output for each method. *Fourth row*: motion segmentation of each method

on Sintel as well as FlyingThings3D. We achieve state-of-the-art results on Sintel, whereas on FlyingThings3D we rank second best after Tokmakov et al.

Tokmakov et al. and Jain et al. do not need any pre-processing of the optical flow, however, here we show that a more analytical approach, which includes a step of pre-processing the optical flow - compensating for camera rotation, has a high potential for further improvements and solving the task of motion segmentation without the need of large training data sets.

5.1.3 Summary

There have been several works for self-supervised video segmentation (and also fully unsupervised approaches) incorporating second order cues and tasks such as colorization or solving jigsaw puzzles for the task of video segmentation [121, 6, 24, 138, 82, 69]. In this Chapter we pretend an alternative form of self-supervised learning attempting to incorporate physical knowledge about the motion field and its formation given a complex scene geometry with high variation in depth. We show promising results on two synthetic (and ideal) data sets with ground truth camera rotation information provided. It remains to be explored to develop an end to end approach that first estimates the camera motion given noisy optical flow and second segments the video into its moving objects and static background. This is subject of the following Chapter 5.2.

5.2 Learning Object Motion from Rotation-Compensated Flow Fields

Humans and animals have developed various approaches to improve their motion perception and detection. One key tool developed in the visual system of humans and animals is motion field stabilization. Humans for example use *smooth pursuit tracking* [122, 54] using eye rotations, to keep the perceived motion field as stable as possible. And some bird species, e.g., chickens, bob their heads to keep the motion field stable for short periods of time. Inspired by these strategies and previous work we propose a two-step approach for motion segmentation that first attempts to stabilize the motion field (compensating for the observer's rotation), and then given the rotation-compensated field, motion patterns are learned using a convolutional neural network.

We combine our geometry-based method for estimating camera rotation, and a CNN framework for learning to segment moving objects. Other than in Chapter 5.1

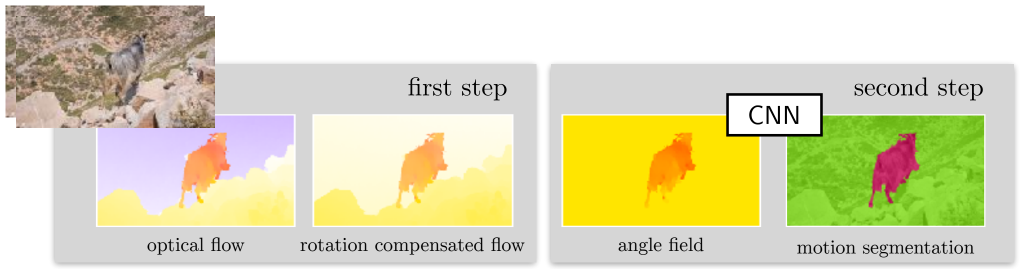


Figure 5.4: An overview: End-to-end approach of learning object motions from optical flow. Inspired by classical approaches for the problem of motion segmentation [5, 40, 72], we first compensate the observed motion field for camera rotation ("first step"), and segment the remaining translational optical flow field using a learning based approach ("second step"). The observed flow field on the left has complex motion patterns: the motion directions of foreground and background are pointing in opposite directions, due to large variance in scene depth, and the combined impact of camera rotation and translation. Estimating the camera rotation ("the right spin"), and compensating the flow field for this rotation simplifies the motion field dramatically, in this case yielding similar motion directions for foreground and background. This provides simpler inputs to our learning based motion segmentation framework.

we develop an end-to-end system that starts from estimating the camera rotation and then segments the rotation compensated (noisy) flow field - rather than the ideal motion field - into static background and independently moving objects.

In Chapter 5.2.1.2 we present a new likelihood for a translational motion field vector. We then introduce an approach for estimating the camera rotation and the translational motion direction using a likelihood maximization approach, given the rotation compensated flow in Chapter 5.2.1.1. We show in Chapter 5.2.2 that the task of learning motion patterns is improved, resulting in an better motion segmentation performance.

5.2.1 Methods

Our estimation of camera rotation depends upon finding the rotation $[A, B, C]$ which maximizes the likelihood of the resulting translation flow field. Our flow likeli-

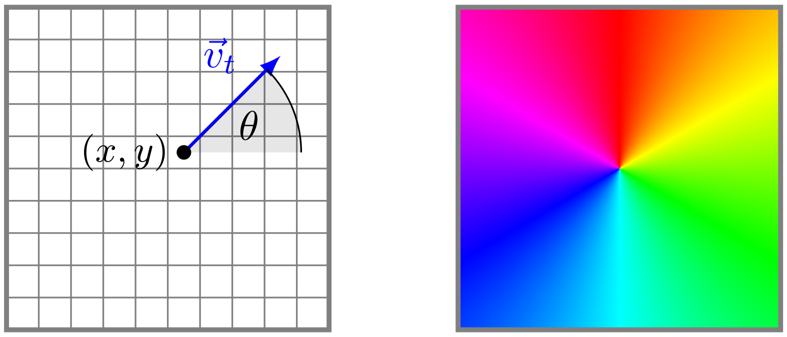


Figure 5.5: The translational motion field vector. Left: motion field vector $\vec{v}_t$ at a particular pixel position (x, y). Right: color coding of the angle field $\theta(x, y)$ at each pixel location for the case of camera translation along the optical axis $[U, V, W] = [0, 0, 1]$.

hood incorporates a model for the optical flow's noise as well as a prior distribution over the *inverse scene depth*.

To address the challenge of estimating camera motion in the presence of moving objects, we weight each pixel using soft object motion masks, which are output of our segmentation method and evolve over time. This way independently moving objects have almost no influence on our optimization for camera motion - the influence of moving objects is suppressed due to a low weight.

In the following, we first introduce the flow likelihood. This likelihood is used as an error function to estimate camera motion. We then describe how camera motion parameters are estimated by minimizing this error.

5.2.1.1 Likelihood of the translational motion field

Let $\vec{o}_t$ be the observed translational flow vector, e.g., flow estimated with [107], at a particular pixel position (x, y). Let the translational 3D motion direction of the camera given by $[U, V, W]$ be a unit vector. The three translational camera parameters $[U, V, W]$ and the pixel position (x, y) define the direction of a motion field vector at a particular pixel location in the image (see Fig. 5.5). As derived in Chapter 4.2 equation 4.35, the probability of observing $\vec{o}_t$ at (x, y) given a motion direction $[U, V, W]$ is given by:

$$p(\vec{o}_t \mid U, V, W, x, y) = \int_0^\infty p(\vec{n})\, p_r(r \mid U, V, W, x, y)\, dr, \tag{5.1}$$

which represents all the ways that the observed flow vector $\vec{o}_t$ could occur as a combination of motion in the direction $[U, V, W]$, a random motion magnitude r, and an optical flow error $\vec{n}$.

The motion field magnitude is not independent of the motion direction. As we can see, the likelihood (5.1) depends on the distribution of motion field magnitudes p_r. This magnitude is challenging to model directly, since it depends on the camera's translational motion direction $[U, V, W]$ and the pixel location. In Chapter 4.2 we modeled p_r by assuming that the motion field magnitude r is independent on the flow direction $[U, V, W]$. However this turns out to be quite inaccurate. Especially in case of strong z-motion (forward motion) the motion field magnitudes close to the focus of expansion are near zero, whereas the motion field magnitudes of a horizontal camera motion are independent of image location. Next we present a new way of modeling the distribution over motion field magnitudes p_r that alleviates these problems.

From flow magnitudes to inverse depth. Given the motion field of a pure translational motion, from the perspective projection equations [14], we can derive the motion field components u_t and v_t as:

$$u_t = \frac{-fU + xW}{Z}, \qquad v_t = \frac{-fV + yW}{Z}. \tag{5.2}$$

Then the motion field magnitude r is:

$$\begin{aligned} r &= \sqrt{u_t^2 + v_t^2}, && (5.3)\\ &= \frac{1}{Z} \cdot \sqrt{(-fU + xW)^2 + (-fV + yW)^2}, && (5.4)\\ &= \frac{1}{Z} \cdot g(f, x, y, U, V, W), && (5.5) \end{aligned}$$

where g is a function that controls all aspects of the magnitude that are *not* related to depth.

Distributions over flow magnitudes. Given the reformulation of magnitude r in terms of $g(\cdot)$ and the inverse depth $\frac{1}{Z}$, we would now like to determine the *induced distribution* on motion field magnitudes, given the distribution on inverse depths, i.e., we aim to compute $p_r(r|g(f,x,y,U,V,W))$ through $p_{\frac{1}{Z}}(\frac{1}{z})$. Using the relation between r and $g(\cdot)$ from (5.5), we can rewrite $p_r(r|g(\cdot))$ as follows

$$p_r(r|g(\cdot)) = \frac{p_{\frac{1}{Z}}(\frac{r}{g(\cdot)})}{g(\cdot)}, \tag{5.6}$$

which is effectively just a change of units. Expressing the distribution over flow magnitudes in terms of the distribution over inverse depth has a significant advantage. This formulation effectively factors motion direction (U, V, W), focal length f and scene depth into the function $g(\cdot)$, and the distribution over depth can be modeled without relying on these dependencies and making further approximations.

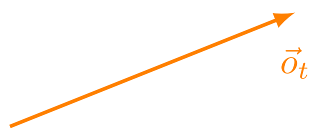
Flow likelihood. Now the likelihood (5.1) can be written using a distribution over inverse depth, rather than flow magnitudes:

$$p(\vec{o_t} \mid U, V, W, x, y) = \int_0^\infty p(\vec{n})\, p_r(r \mid g(\cdot))\, dr, \tag{5.7}$$

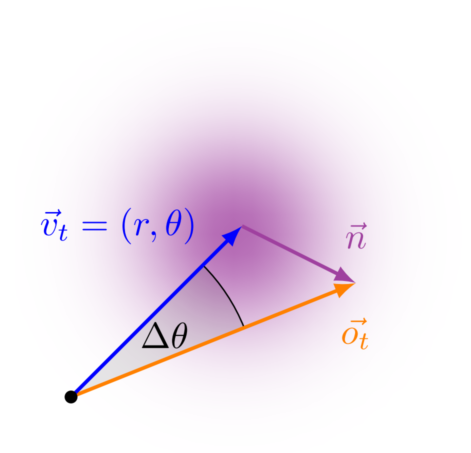
$$= \int_0^\infty p(\vec{n})\, \frac{p_{\frac{1}{Z}}\left(\frac{r}{g(\cdot)}\right)}{g(\cdot)}\, dr. \tag{5.8}$$

The key advantage of this is that while flow magnitudes are not independent of the motion direction, the inverse depths ARE independent of motion direction, and thus the model is more realistic.

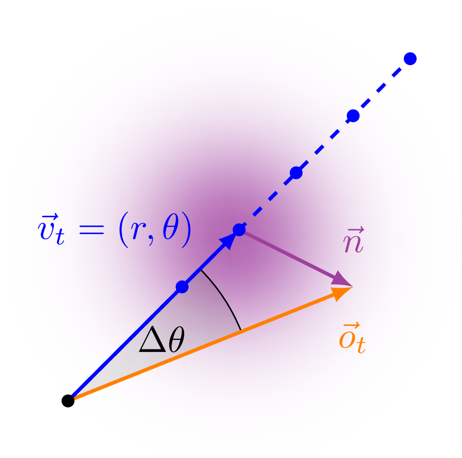
5.2.1.1.1 Implementation details. We model the probability of the flow noise $p(\vec{n})$ as a multivariate normal $p(\vec{n}) \sim \mathcal{N}(\mu, \Sigma)$ and the inverse depth $p(\frac{1}{Z})$ as an exponential distribution $p(\frac{1}{Z}) \sim \text{Exp}(\lambda)$.



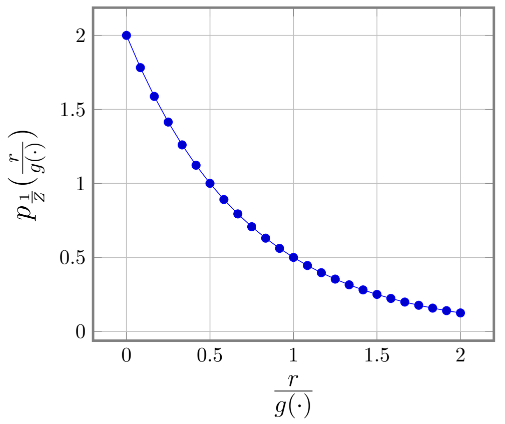
(a) Translational (observed) flow vector $\vec{o}_t$ at pixel location (x, y).



(b) Observed optical flow vector $\vec{o}_t$, is a noisy observation of the motion field vector $\vec{v}_t$: $\vec{o}_t = \vec{v}_t + \vec{n}$.



(c) To compute the flow likelihood, we integrate over the unknown motion magnitude of the motion field vector $\vec{v}_t$.



(d) Probability distribution over inverse depth.

Figure 5.6: Flow likelihood (a)-(c): computation of the probability $p(\vec{n})$ at a particular pixel position (x, y). (d): probability distribution over inverse depth. The flow likelihood is maximal is the observed flow vector $\vec{o}_t$ and the motion field vector $\vec{v}_t$ point into the same direction with similar magnitude - which refers to small flow noise $\vec{n}$.

The noise covariance Σ is assumed to be spherical and is measured using the ground truth flow of Sintel [15] and the corresponding estimated optical flow [107]. We obtain $\sigma = 16.5 \cdot 10^{-5} I$, where I is the identity matrix. λ is the rate parameter of the exponential distribution modeling the inverse depth, and is estimated using ground truth depths from Sintel. We measured $\lambda = 0.64$. The distribution over inverse depth can be seen in Figure 5.6(d).

3D Lookup table. For computational efficiency the integral in equation 5.8 is approximated using a discrete sum over motion field magnitudes r and flow likelihood values are pre-computed for efficiency. Towards the goal of representing the flow likelihood using a 3D lookup table we express the likelihood function with only three degrees of freedom. We start from equation 5.8 and revise each part of the function and reduce its dimensionality.

Probability of the flow noise $p(\vec{n})$. Let $\vec{o}_t$ be the estimated (observed) motion vector with magnitude m and direction α and let $\vec{v}_t$ be the true motion vector with magnitude r and direction θ, then the noise is the difference between those two. We represent those vectors using the exponential form (Euler's formula), which decomposes a vector into its magnitude part and angle part:

$$\vec{n} = \vec{o}_t - \vec{v}_t = m \cdot e^{i\alpha} - r \cdot e^{i\theta} \tag{5.9}$$

$$= e^{i\alpha} \cdot \left(m - r \cdot e^{i(\theta - \alpha)} \right) \tag{5.10}$$

$$= e^{i\alpha} \cdot \left(m - r \cdot e^{i(\Delta\theta)} \right) \tag{5.11}$$

Since the noise covariance Σ is assumed to be spherical, the probability of the noise is independent of its direction. However it is dependent upon the flow noise magnitude. The flow noise magnitude is defined by m, r and $\Delta\theta$.

$$|\vec{n}| = \left| m - r \cdot e^{i(\Delta\theta)} \right| \tag{5.12}$$

Thus one can write the probability of the flow noise $p(\vec{n})$ as the probability of the flow noise vector $\vec{n}$ with magnitude $|\vec{n}|$ and angle of zero degree.

$$p\Big(\vec{n}(|\vec{n}|, 0)\Big) \sim \mathcal{N}\Big(\mu, \Sigma\Big) \tag{5.13}$$

The probability of the flow noise comes with two degrees of freedom, which are $\Delta\theta$ and the observed flow magnitude m.

Probability over inverse depth and scale factor $g(\cdot)$. We will start from equation 5.5. Here the probability over inverse depth initially comes with three degrees of freedom for the camera translation U, V and W, the focal length f and the pixel location x, y. In the following we will reformulate the equation, such that just one additional degree of freedom besides the flow magnitude m and the angle difference $\Delta\theta$ between the two motion vectors $\vec{o}_t$ and $\vec{v}_t$ is required to compute the likelihood of a flow vector at a particular image position.

$$\begin{aligned} g(\cdot) &= \sqrt{(-fU + xW)^2 + (-fV + yW)^2} && (5.14)\\ &= \sqrt{W^2 \cdot \left(\left(-\frac{fU}{W} + x\right)^2 + \left(-\frac{fV}{W} + y\right)^2\right)} && (5.15)\\ &= W \cdot \sqrt{(x - x_0)^2 + (y - y_0)^2} && (5.16)\\ &= W \cdot D && (5.17) \end{aligned}$$

The Eq. 5.16 describes a circle with center coordinates $(x_0, y_0) = \left(-\frac{fU}{W}, -\frac{fV}{W}\right)$. Here the coordinates (x_0, y_0) describe the position of the focus of expansion. D is the distance of a point in the image from the focus of expansion. Consequentially, $g(\cdot)$ is the distance of a point in the image from the focus of expansion scaled by W. One can see that $g(\cdot)$ is only determined by one factor, the scaled distance of a point in the image to the focus of expansion. *Thus, the probability over inverse depth comes* *with just one additional degree of freedom which is $g(\cdot)$, the scaled distance of a point in the image to the focus of expansion.*

Flow likelihood lookup table. Using the reformulation of the probability of the flow noise and the scale factor $g(\cdot)$ allows us to pre-compute the flow likelihood and generate a 3D lookup table to simply lookup the likelihood values during camera motion estimation. The three dimensions of the lookup table are $\Delta\theta$, flow magnitude m and $g(\cdot)$. The integral over motion field magnitudes r is approximated using a discrete sum over r. This leads to the following final Equation:

$$p(\vec{o_t} \mid U, V, W, x, y) = \int_0^\infty p\left(\vec{n}(r, m, \Delta\theta)\right) \frac{p_{\frac{1}{Z}}\left(\frac{r}{g(\cdot)}\right)}{g(\cdot)} \, dr \tag{5.18}$$

$$= \sum_{r=0}^{R} p\left(\vec{n}(r, m, \Delta\theta)\right) \frac{p_{\frac{1}{Z}}\left(\frac{r}{g(\cdot)}\right)}{g(\cdot)} \, \Delta r. \tag{5.19}$$

Our flow likelihood addresses the challenge of estimating the camera's motion in the presence of noisy optical flow. To get an intuition about its behaviour we visualize the structure of our lookup table and show a slice for $g(\cdot) = \text{const.}$ (see Figure 5.7). The color *red* indicates high likelihood values, *dark blue* indicates low likelihood values. The lower the angle difference $\Delta\theta$ between the vectors $\vec{o_t}$ and $\vec{v_t}$, the higher the likelihood. Also please note that for very small flow magnitudes m the flow likelihood is almost the same regardless $\Delta\theta$. This makes sense, since the flow direction tends to be unreliable if its magnitude is close to zero.

5.2.1.2 Camera motion estimation via likelihood maximization

Given an observed optical flow vector $\vec{o}$ we want to find a translational motion direction (U, V, W) and a camera rotation (A, B, C), such that the flow likelihood is maximal or the negative log-likelihood is minimal. Recall $\vec{o_t}$, which is the observed translational flow vector after subtracting off the flow $\vec{v_r}$ due to camera rotation:

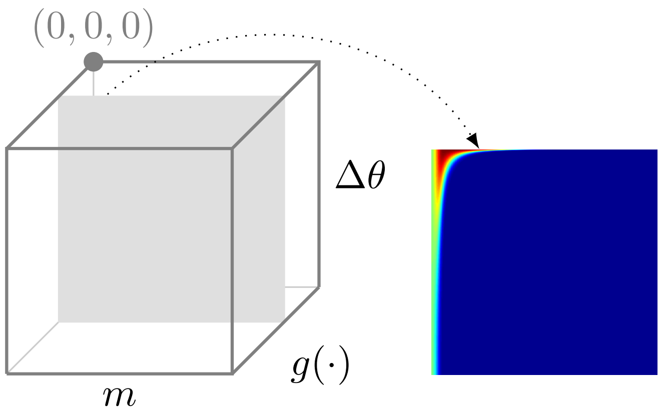


Figure 5.7: Lookup table. Right: 3D lookup table, left: visualization of one *slice* (for $g(\cdot) = \text{const.}$) of the lookup table (Best viewed in pdf.)

$$\vec{o_t} = \vec{o} - \vec{v}_r(A, B, C). \tag{5.20}$$

Given the rotation compensated flow, we minimize the negative log-likelihood as follows:

$$\begin{aligned} & \hat{A}, \hat{B}, \hat{C}, \hat{U}, \hat{V}, \hat{W} \\ = & \underset{A,B,C,U,V,W}{\arg\min} \sum -\log(p(\vec{o_t}(A, B, C)|U, V, W, x, y)). \end{aligned}$$

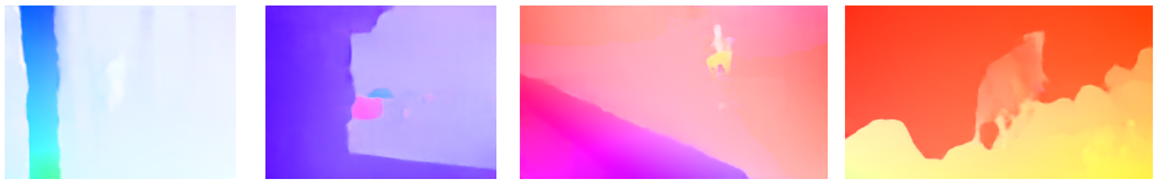
The estimated rotational motion field defined by $[\hat{A}, \hat{B}, \hat{C}]$ is then subtracted off from the original flow field and we obtain the rotation compensated flow field. Local minima are a concern, especially in cases of noisy optical flow, inaccurate estimates of independently moving objects present in the scene or complex scene geometry. To reduce the risk of an unstable optimization, we start from different starting points: (1) camera rotation and translation estimate of the previous frame, (2) camera rotation estimate weighted by depth estimate of previous frame and the translation estimate of the previous frame and (3) camera rotation estimate weighted by depth estimate of previous frame and the translation estimate of the previous frame in opposite direction. Especially for scenes with complex scene geometry and high variation in depth a pre-estimate of the camera rotation alone weighted by the scene's depth improves stability and avoids local minima. At very distant scenes the motion is mainly influenced by the camera rotation and not the camera translation (see Figure 5.8).

5.2.1.3 Object Motion Segmentation

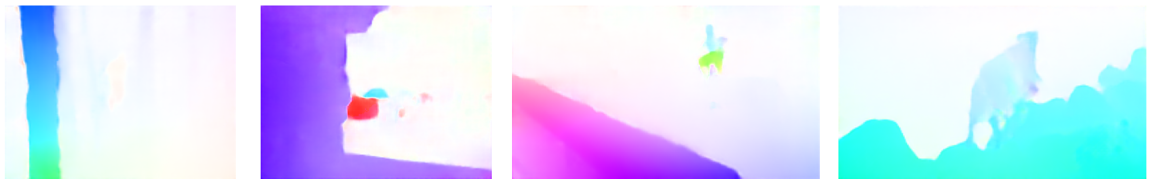
Motivated by the success of CNN based approaches for object motion segmentation, we build our segmentation framework on a state-of-the-art model [110] based on the widely used U-Net architecture [97]. In contrast to the original method, our network takes rotation-compensated flow fields as input to segment independently moving objects. Learning object motion based on rotation-compensated flow field appears to be a task, that is much easier to learn for a network. Since the complexity of optical flow patterns that couple the scene geometry, camera motion (rotation+translation) is dramatically simplified. While our network architecture is similar to [110], we propose important modifications to the training procedure, as described in the following section.

5.2.1.3.1 Incorporating geometric information into training We train our network on estimated translational flow fields. First, we estimate optical flow using [107] on the FlyingThings3D data set [71]. The ground truth camera rotation provided with the data set is subtracted from the flow to obtain a rotation-compensated flow field. This flow field is input to our network as a matrix of size $h \times w \times 3$. Unlike [110] we represent the flow angle using a unit vector representation instead of explicit angles in the range of $[0, ..., 2\pi]$. This avoids segmentation discontinuity in angle at 0 (or 2π respectively). The third component of the input vector is the optical flow's magnitude, which is simply concatenated. An interesting question for training the motion segmentation network without rotation-compensated optical flow

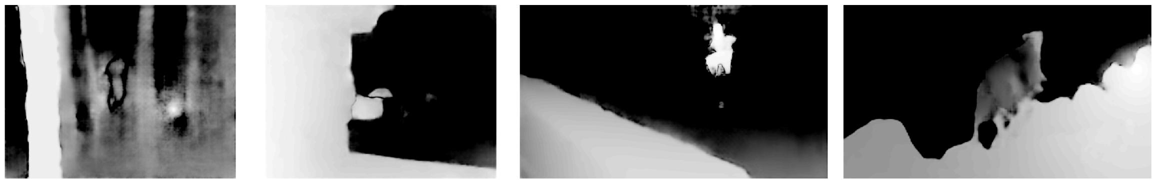
(a) video frame



(b) optical flow



(c) rotation compensated optical flow



(d) depth estimate

Figure 5.8: Flow, rotation compensated flow and the relative depth estimate. We show sample videos from the data set Complex Background (video sequences: traffic, forest) as well as two sample videos from the Davis data set (video sequence: parkour, goat). A comparison of (b) and (d) shows how motion at distant is dominated by camera rotation. After subtracting of the camera's rotation the remaining flow magnitude in these areas is very small (light color). If the flow magnitude is small the motion direction is noisy. This can be seen in (e).

is whether it is worthwhile to incorporate magnitude into the training procedure. We study this question in detail in Section 5.2.2.1.

5.2.2 Experiments

We evaluate our approach on the widely used DAVIS data set [83] and show ablation studies on FlyingThings3D [71]. Different variants of our core network are analyzed using the FlyingThings3D data set [71]. We begin with a brief presentation of the data sets used for training and evaluation, and then discuss the variants of our motion segmentation network, and its evaluation and comparison to the state of the art on DAVIS.

DAVIS [83] (Densely Annotated VIdeo Segmentation) contains 50 video sequences in total. They show different moving objects in various environments. A segmentation of the most prominent moving object is provided for each frame. This data set has been widely used for general video segmentation as well as motion segmentation. In our experiments we show an evaluation comparing motion segmentation methods and general video segmentation methods separately. We use the entire data set (validation+test set) for evaluation, in accordance with previous work [110].

FlyingThings3D [71] (FT3D) is a large optical flow data set, providing ground truth optical flow, the original RGB images, camera motion and depth. It is a synthetic data set showing random objects like chairs, tables, etc. flying in the 3D world along random trajectories. The data set is split into test and training sets. We show experiments using ground truth optical flow and also the estimated optical flow from the RGB images.

5.2.2.1 Ablation study

5.2.2.1.1 Network variants. We trained four variants of our motion segmentation network, with: (1) ground truth optical flow, (2) the ground truth flow after having removed ground truth camera rotation, i.e., with rotation compensated-flow

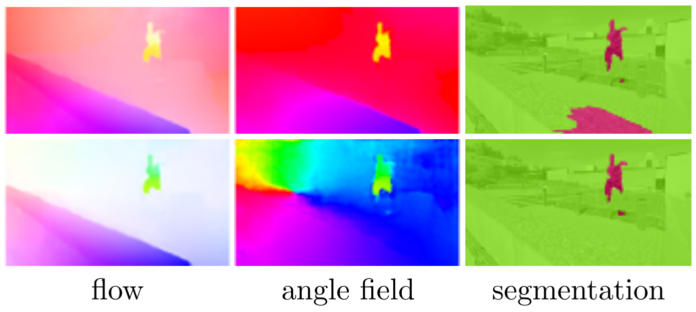


Figure 5.9: Comparison of motion segmentation results based on the original and the rotation-compensated flow field. Top row: motion segmentation with the original flow field that includes camera rotation, translation and object motion. Bottom row: motion segmentation based on *rotation-compensated flow field*. Note that the angle field (middle) of the rotation-compensated flow is entirely depth independent. The angle field is fully determined by the translational camera motion and object motion. In this example one can observe a clear z-motion of the camera, which is shown by the rainbow pattern. The angle field of the original flow containing both camera rotation and translation is depth dependent (top row, middle image). This angle field clearly shows discontinuities in angle at the wall, which is due to significant changes in depth and not because of independent object motion.

fields, (3) estimated optical flow field using PWC-Net [107], and (4) estimated ground truth flow compensated with ground truth camera rotation, i.e., estimated rotation compensated-flow field.

Table 5.3 shows the analysis with these four variants. Training and testing with ground truth optical flow (original: gt FT3D or compensated: gt transFT3D) is significantly better than using estimated optical flow. Segmentation accuracy is about 20% higher on the FT3D test set for ground truth, compared to estimated optical flow. Training on rotation-compensated optical flow consistently leads to improved quality of the final segmentation, e.g., 90.68% vs 93.23%. This motivates us to use rotation-compensated flow fields for training a motion segmentation network. A direct comparison in terms of segmentation quality between using the original optical flow instead of the rotation-compensated optical flow as input is shown in Figure 5.9.

trained with...	tested with...	angle+magnitude
gt FT3D	gt FT3D	90.68
gt transFT3D	gt transFT3D	**93.23**
PWC-Net FT3D	PWC-Net FT3D	77.18
PWC-Net transFT3D	PWC-Net transFT3D	**78.69**

Table 5.3: Ablation study: Network variants. We trained four networks using flow angle and magnitude with: the provided ground truth optical flow of FT3D [71] (gt FT3D), ground truth optical flow after subtracting ground truth camera rotation (gt transFT3D), estimated optical flow using [107] (PWC-Net FT3D), and estimated optical flow after subtracting ground truth camera rotation (PWC-Net transFT3D). Segmentation accuracy is measured on the FT3D test set with intersection over union (IoU) scores.

5.2.2.1.2 Training on flow angle only versus angle+magnitude. As discussed in Chapter 3, rotation-compensated flow comprises all the information about independent object motion and the scene structure (depth). In this context, two interesting questions to tackle are: *how well can one extract information about independent object motion from the angle alone*, and *does including the flow magnitude (training the network on the full optical flow) improve motion segmentation?*. We show this analysis in Table 5.4, with further variants of our network. Using angle and magnitude together (angle+magn in the table) leads to the highest performance. However, note that we achieve reasonable segmentation quality even when using the flow angle alone. The network trained on ground truth optical flow adapts very poorly to estimated optical flow, with the segmentation accuracy dropping from 93.23% to 24.44% for the angle+magn variant.

5.2.2.1.3 Rotation estimation via likelihood maximization. The FT3D data set is not suitable for evaluating the performance of the camera rotation, since pixel displacements are unrealistically large. Instead we show results on the Sintel data set (Tab. 5.5), and compare our new likelihood optimization procedure with Bideau et al. [5]. The ground truth focal length is provided, so an accurate estimate of the

trained with...	tested with...	angle	angle+magn
gt transFT3D	gt transFT3D	77.47	93.23
gt transFT3D	PWC-Net transFT3D	24.06	24.44
PWC-Net transFT3D	PWC-Net transFT3D	77.79	**78.69**

Table 5.4: Ablation study: Training with angle vs angle and magnitude. We trained four variants of our segmentation network with: (1) angle of the rotation-compensated flow of FT3D, (2) angle and magnitude of the rotation-compensated flow of FT3D (angle+magn), (3) angle of the estimated (PWC-Net) rotation-compensated flow, and (4) angle and magnitude of the estimated rotation-compensated flow. We show consistently better performance by including magnitude. The performance is the worst when the network is trained on the angle of the rotation-compensated ground truth flow. Here, the noise in angle leads to a very significant drop on estimated optical flow data. Segmentation accuracy is measured on the FT3D test set with intersection over union (IoU).

	Bideau et al. [1]	ours
gt-flow	0.08 / 0.22 / 0.02	0.03 / 0.06 / 0.01
PWC-flow	0.13 / 0.34 / 0.04	0.05 / 0.11 / 0.03

Table 5.5: Ablation study: Camera rotation estimation. Avg. yaw/pitch/roll error in degrees between 2 consecutive frames. *gt-flow, PWC-flow:* To evaluate rotation estimation we used ground-truth segmentation masks to weight the optim. loss. Thus, errors in the segmentation procedure are not propagated throughout the video.

camera's rotation is possible. If no ground truth focal length is available as in Davis, we use a fixed focal length for all videos. This leads eventually to a wrong estimates of the three camera rotation parameters $[A, B, C]$ (by a fixed offset), however the error of the induced motion field (Equation 3.19) is negligible small. Our camera rotation estimation based on maximizing the flow likelihood shows consistently better results on the Sintel data set. More importantly, the gap in performance gets very significant when using estimated flow (PWC-flow). Our new optimization approach is significantly more robust to noisy flow data, since it incorporates an explicit noise model.

Measure		LMP [110]	TMM [5]	Ours-flow	Ours-flow*
$\mathcal{J}$	Mean ↑	58.4	40.1	59.7	**62.5**
	Recall ↑	67.3	34.3	69.6	**73.8**
	Decay ↓	5.6	15.2	4.3	**3.8**
$\mathcal{F}$	Mean ↑	58.4	39.6	59.5	**61.1**
	Recall ↑	66.0	15.4	66.4	**69.9**
	Decay ↓	7.9	12.7	**5.4**	5.6
$\mathcal{T}$	Mean ↓	87.8	**51.3**	74.5	83.4

Table 5.6: Binary motion segmentation: Comparison to other approaches using only motion cues on DAVIS, i.e., without any appearance. Ours-flow refers to the variant of our model using only motion cues and no appearance terms and Ours-flow* denotes a motion-only upper bound, which uses ground truth segmentation for camera motion estimation. Best viewed in color (**1st-best** , 2nd-best).

5.2.2.2 Results: Binary motion segmentation

5.2.2.2.1 DAVIS: Optical flow only. We begin by comparing our motion segmentation network with other methods that use optical flow as the only cue for segmentation. Table 5.6 shows these results on DAVIS. LMP is a learning based approach that estimates motion cues [110]. This network is trained on ground truth optical flow of FlyingThings3D, ignoring scene geometry, i.e., it does not compensate for camera rotation. TMM [5], on the contrary, compensates flow for camera rotation and attempts to model the motion field using translational motion models. Their translational motion models are quite limited however, and fail to capture the complex motion of certain moving objects, such as a walking person. Our approach (Ours-flow in the table) improves over both these state-of-the-art motion segmentation methods. We also compute an upper bound for our result (Ours-flow* in the table) by masking out independently moving objects, with ground truth segments, for our camera motion estimation procedure. This masking procedure eliminates errors in our camera motion estimation due to outliers in optical flow, such as moving objects.

5.2.2.2.2 DAVIS: Optical flow + Appearance. Table 5.7 compares the results of our complete approach, using motion and appearance cues, with the state of the art.

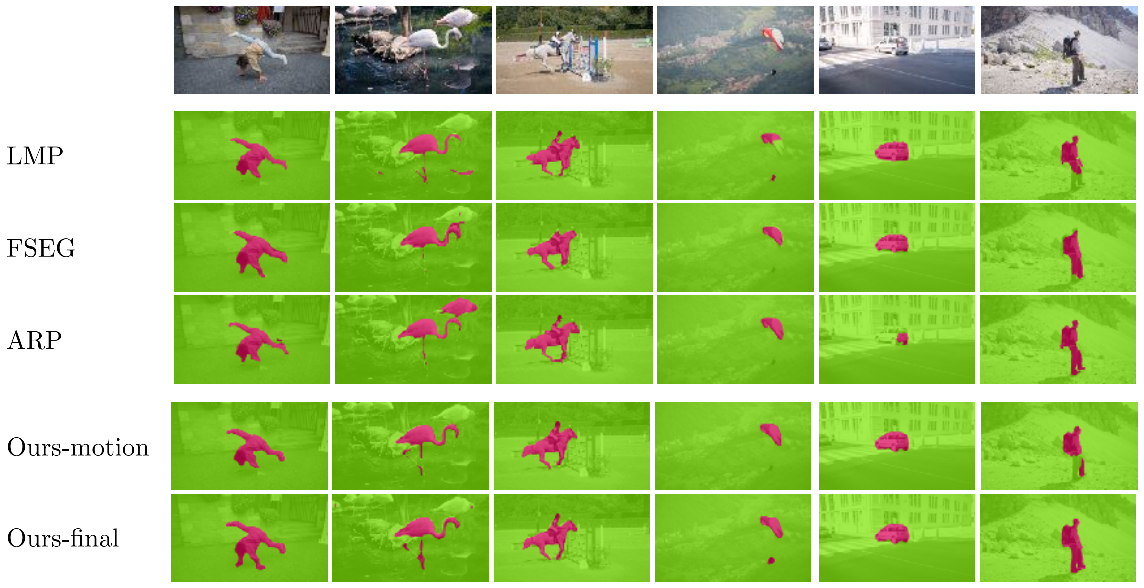


Figure 5.10: Qualitative segmentation results. Qualitative segmentation results on the DAVIS data set, showing a comparison with three other best performing methods. Ours-final denotes our complete method and Ours-flow the variant based on motion cues alone. (Best viewed in pdf.)

Our approach outperforms nearly all the methods. Our performance is comparable to [49] in terms of mean/recall $\mathcal{J}$ and $\mathcal{F}$, while we outperform on the decay measures and $\mathcal{T}$. We also show a qualitative comparison with the best performing methods in Figure 5.10. Since [49] relies on segmenting the primary object(s) in a video, it is biased towards the object's appearance. For example, it only segments a part of the car (2nd column from the right), which moves from the darker (shadow) area to the brighter (sunny) region. It can also incorrectly segment stationary objects, e.g., the flamingo in the background (2nd column from the left), as it matches the primary object in appearance. Our variants, shown in the last two rows in the figure, overcome such errors. We highlight the complementarity of motion and appearance cues in the example shown in the last column, where we miss the hiker's foot when relying on motion alone (Ours-flow), since it is not moving. However, integrating motion with appearance, we segment the entire object accurately.

Measure		[22]	[110]	[55]	[42]	[109]	[49]	[57]	[81]	[111]	Ours
$\mathcal{J}$	Mean ↑	64.1	69.7	68.3	71.6	51.4	**76.3**	56.9	57.5	75.1	**75.8**
	Recall ↑	73.1	82.9	77.7	87.7	58.1	**89.2**	67.1	65.2	87.9	**89.3**
	Decay ↓	8.6	5.6	5.7	**1.7**	12.7	3.6	7.5	4.4	2.0	**1.5**
$\mathcal{F}$	Mean ↑	59.3	66.3	67.2	65.8	49.0	**71.1**	50.3	53.6	70.9	**71.4**
	Recall ↑	65.8	78.3	75.9	79.0	57.8	82.8	53.4	57.9	**82.9**	**83.7**
	Decay ↓	8.6	6.7	7.4	4.3	13.8	7.3	7.9	6.5	**3.3**	**2.2**
$\mathcal{T}$	Mean ↓	36.6	68.6	25.8	29.5	25.6	35.9	**21.0**	29.3	**22.0**	25.6

Table 5.7: Binary motion segmentation: Comparison to state-of-the-art motion segmentation methods on DAVIS. Best viewed in color (**1st-best** , **2nd-best**).

5.2.3 Summary

We proposed a new motion segmentation approach that lies at the intersection of classical methods based on perspective geometry and learning based frameworks. Our approach first estimates camera rotation, and then extracts rotation-compensated flow fields to learn a motion segmentation model. We show that combining the strengths of two motion segmentation paradigms achieves state-of-the-art results on the widely used DAVIS data set.

CHAPTER 6

APPLICATIONS: ROTATION ESTIMATION OF UNMANNED AREA VEHICLES

Relatively inexpensive, high quality and versatile video cameras of various types such as the GoPro action camera or drone-mounted cameras have increased the amount of publicly available video data enourmously [124]. Drone-mounted cameras have enabled citizens to provide video coverage of land areas previously inaccessible. Drones are part of search and rescue missions, they can access areas in cases of natural disasters and can monitor wildlife world wide. This new data source creates a completely new application area for classical computer vision tasks such as object detection [33], semantic segmentation [70, 80], tracking [67, 60] and camera motion estimation [84, 91]. However, this vast amount of unstructured video material suddenly available comes with several uncertainties questioning the trustworthiness of these videos. *Does the meta data coming with the video data actually correspond to the video? How can we ensure that the meta data was not manipulated after the video was taken?* If one wishes to use and analyse these type of new accessible data in a meaningful manner one has to find ways to ensure the trustworthiness of those videos. Most drone videos come with a flight plan and/or meta data defining the motion of the camera, however their trustworthiness is not ensured.

The goal of this work is to verify that the motion depicted in the video is consistent with the set of flight instructions given at the beginning of the flight. For this purpose we use our camera rotation estimation algorithm presented in Chapter 4.2 and aim to verify provided flight instructions using the visual information from the video. Given a video of an arbitrary scene with unconstrained camera motion and the camera's focal length this algorithm estimates the three camera rotation parameters $[A, B, C]$ and the translational motion direction $[U, V, W]$.

Drone. For experiments a DJI Mavic Air drone was used. This drone has a three-axis gimbal, a 4K UHD (3840x2160) 30 fps camera, and capable of speeds of 30 km/h in obstacle avoidance mode. Its camera comes with a 1/2.3" CMOS sensor and a field of view (FOV) of 85°. To estimate the correct camera rotation information about the field of view or the focal length f in pixel is required. One can convert one measure into the other (see Equation 6.1 and Equation 6.2).

$$f[\text{in pixel}] = \text{image width [in pixel]} \cdot \frac{f[\text{in mm}]}{\text{sensor width [in mm]}} \tag{6.1}$$

$$\text{FOV} = 2 \cdot \arctan\left(\frac{1}{2} \cdot \frac{\text{sensor width [in mm]}}{f[\text{in mm}]}\right) \tag{6.2}$$

6.1 Videos of unmanned area vehicles

Given a video taken by an unmanned area vehicle (UAV) we verify its motion pattern (defined by a motion program) by comparing the provided motion instructions with camera motion estimates based on the raw video. There are two general ways to create the motion program: *unconstrained* and *constrained* programs. In a constrained program we restrict the drone to a small number of types of motions in a small geographic area. In an unconstrained program we use any type of motion a drone is capable of, using a hand-crafted series of motions over a wide-area.

6.1.1 Constrained unmanned area vehicle motion

Constrained motions for an UAV can be of various types such as a series of translations (move up/down, left/right, forward/backwards), or rotations (yaw, roll, pitch), and combinations of the two. In our experiments we instruct the drone to follow a series of movements consisting of two parts: one hotpoint motion (combined camera rotation and translation), either clockwise or counter-clockwise, followed by a motion to fly towards the point of interest to an inner radius around the point of interest and moving backwards to the outer circle. A motion type like the hotpoint motion leads to challenging flow fields, where it is hard to estimate the coupled camera rotation and translation. In these cases camera rotation to the left comes with a camera translation to the right (in opposite direction) to keep the object in focus. This opposite characteristics of motion patterns in combination with near planar scene structures lead to almost indistinguishable motion patterns of camera rotation and translation challenging current motion estimation algorithms.

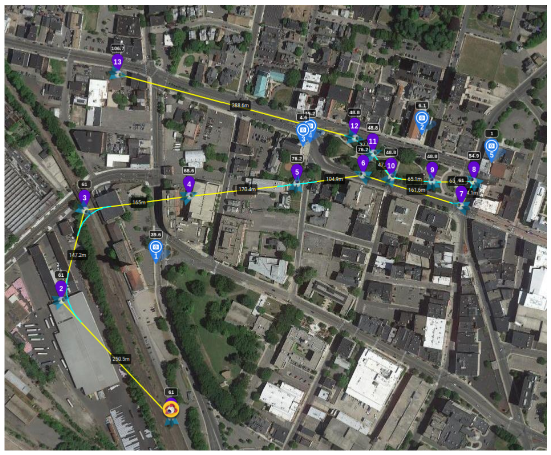
6.1.2 Unconstrained unmanned area vehicle motion

To collect videos with unconstrained motion we take advantage of a popular online drone flight planning and execution application called Litchi[1]. Litchi allows users to publicly post their flight plans and link to videos from the flight hosted by YouTube and Vimeo. We take advantage of these public videos by downloading the flight plans and the video and use them in our evaluation. Using videos such as third party videos helps us to eliminate any potential bias in how we collected videos and gives us access to a large variety of scenes (rural, nature, cities, etc.), lighting conditions, and DJI drone models.

6.2 Experiments

Figure 6.1 shows a flight plan taken from Litchi. Each plain numbered pin represents a *waypoint* for the drone to fly to, and each numbered pin containing a camera icon represents a *point of interest* for the drone to focus on during different parts of the mission. The curves around waypoints represent the actual flight path to be taken to smooth out the drones motion to and from the way point. The drone mo-

[1]`https://flylitchi.com/`


(a) Sample mission 1

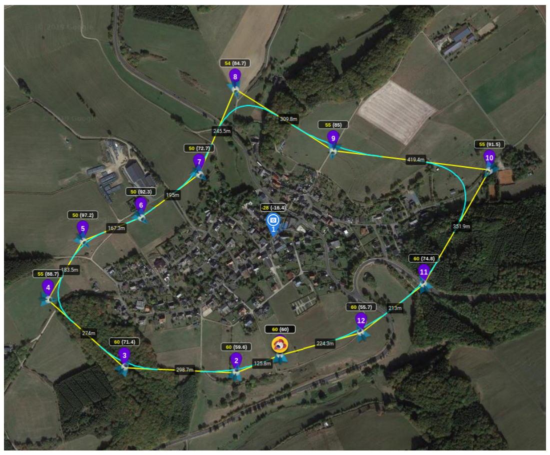

(b) Sample mission 2

Figure 6.1: (a) A sample unconstrained mission from Litchi. `https://flylitchi.com/hub?m=wsi7vg9HQl` and (b) a sample unconstrained mission from Litchi. `https://flylitchi.com/hub?m=bHg7fNYTSW` (accessed November 22nd, 2019)

tion following such a flight plan based on points of interests rather than specific flight instructions comes with unconstrained and very variable motion patterns in 3D.

Drone video sequences. Videos taken by a a drone are usually much longer with higher resolution than typical test video sequences used in standard computer vision data sets. This comes with a challenge for computational speed. The original video taken by a drone and downloaded from Youtube has a frame size of 1920x1080. To speed up the optical flow computation [107] and our motion estimation algorithm the video is first compressed by reducing the frame size to a size of 256x144. Second, the frame rate is reduced which leads to smoother motion and increases the motion between two consecutive frames. Compression and reduced frame rate together lead to a high quality camera motion estimate with minimal noise (see Figure 6.1).

Motion verification. To ensure the trustworthiness of these drone videos online available to the public, we estimate the drone's camera rotation (in degree) and the translational motion direction (a 3D unit vector) using the motion estimation algorithm described in Chapter 4.2. Figure 6.3 shows motion estimates of our algorithm

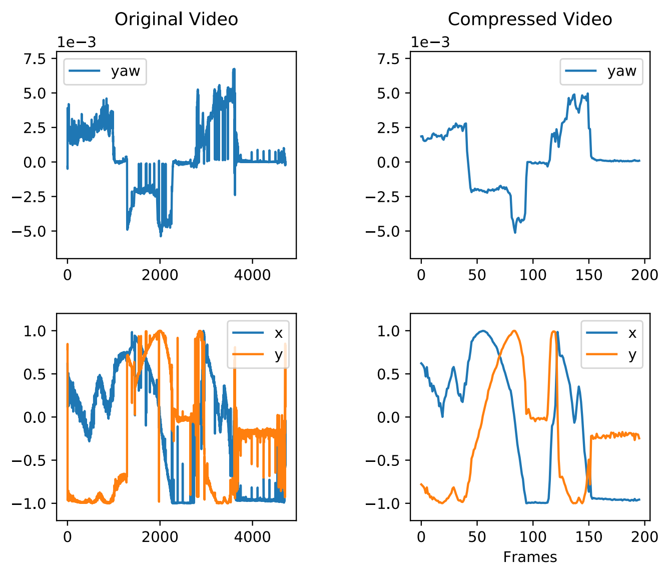
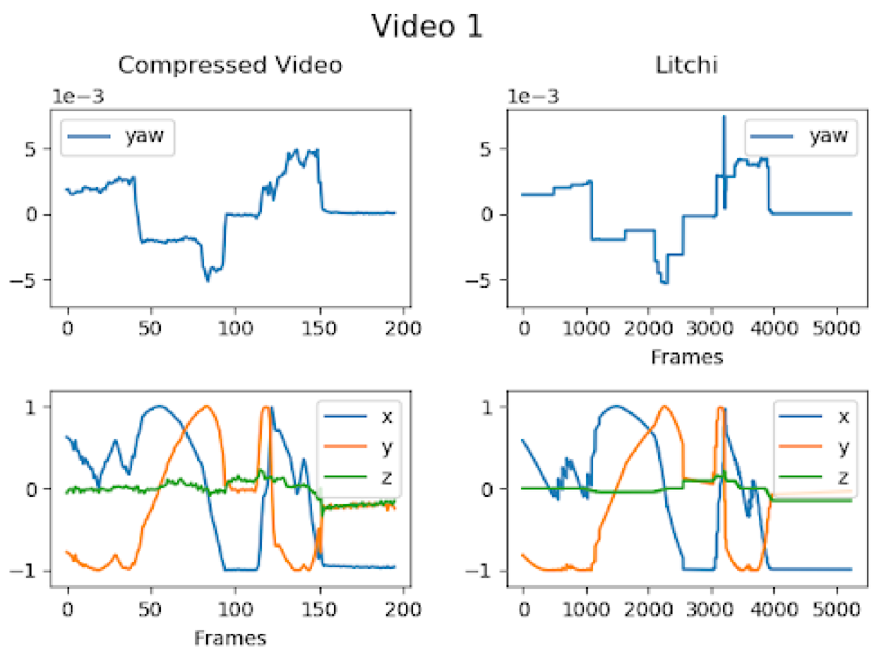


Figure 6.2: Unconstrained motion pattern estimation. Comparison of the camera motion from the original video (left) and the compressed video (right). The camera motion from the compressed video is smoother that the camera motion from the uncompressed video due to the lower frame rate.

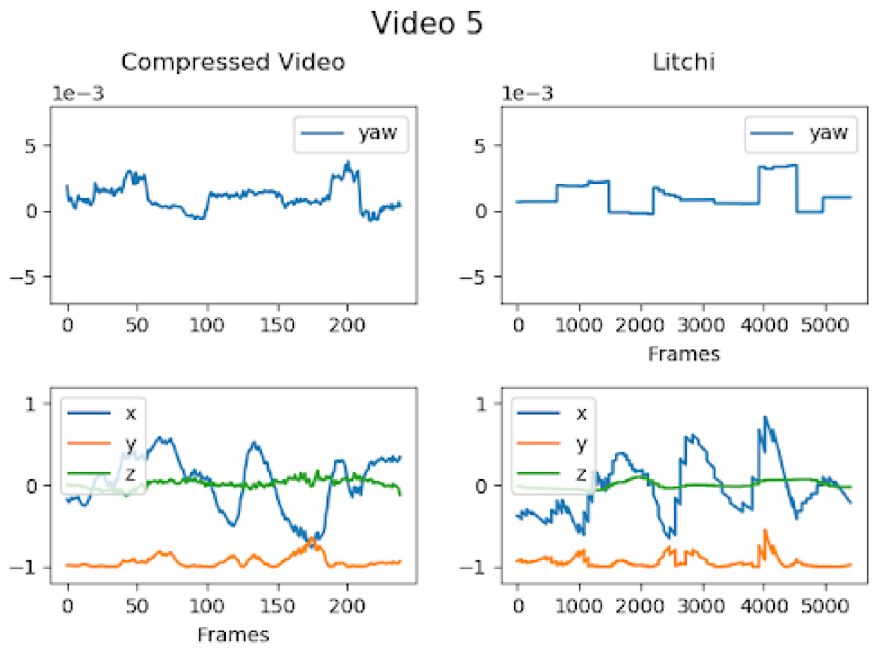
on the left and the motion instructions based on the Litchi flight plan on the right. We are able to match the flight plan quite accurately in both cases. So the flight-plan that comes with the uploaded video sequences matches the flight pattern of the drone in the video.

6.3 Summary

High quality and online available drone videos have opened a area of application for camera motion estimation algorithms. This enormous amount of newly available data rises new questions. What motion pattern does the drone fly? Is the motion pattern consistent with the provided flight instructions? These drone videos come with a high variety of interesting aspects and challenges to be explored. Highly variable camera motion in 3D, mostly planar scenes (due to large distance to the ground) and computational efficiency are aspects to consider in this context. We have been able to show that our camera motion estimation algorithm developed as a



(a) Sample mission 1



(b) Sample mission 2

Figure 6.3: Comparison of the UAV's motion from compressed video (left) and the Litchi flight plan (right).

part of our motion segmentation approach is not only robust on standard computer vision data sets, it also works on diverse real world videos with different motion and scene characteristics such as drone videos presented in this Chapter.

CHAPTER 7
DISCUSSION AND FUTURE DIRECTIONS

7.1 Discussion

In this dissertation we developed an approach to segment independently moving objects combining cues of optical flow, depth and ego-motion. We will discuss the connection of these three closely related areas in further detail and how they can be integrated to benefit from each other (see Chapter 7.1.1). Not always is motion information captured by a single optical flow field sufficiently to segment a frame into its independently moving parts. Challenges arise in cases where (i) an object or just part of an object stands still just for a short amount of time, (ii) its motion doesn't differ significantly from the camera's motion or (ii) optical flow is misestimated. Some of these challenging scenarios and approaches to address cases of weak motion information are discussed in the following (see Chapter 7.1.2).

7.1.1 Interpreting optical flow by integrating motion segmentation, ego-motion estimation and depth estimation

Motion segmentation, ego-motion estimation, and depth estimation are three closely related areas of current research. Motion segmentation attempts to segment all independently moving objects, ego-motion estimation aims to estimate the observer's (camera) motion and methods addressing the problem of depth estimation produce depth estimates of the pictured scene, based on two or more multiple video frames. All three approaches are related to the task of analyzing how an image, the projection of the 3D world on a 2D image plane, changes over time while we as the observers and other objects move. These changes over time are captured by optical flow field. Thus it makes sense to focus on examining all three areas simultaneously rather than in isolation if one want to interpret information captured in optical flow.

Estimating scene depth. Depth can be computed from rotation compensated flow following Equation 3.20. The quality of depth estimation highly depends upon the camera's motion since a change of view point (parallax) is required to form a motion field that captures the depth information of the pictured scene. Without “sufficient” parallax one is not able to obtain a good depth estimate of the scene in a geometrical manner. Since optical flow couples motion information due to camera motion as well as scene depth, depth estimates can be a supportive source of additional information to robustly estimate the cameras motion.

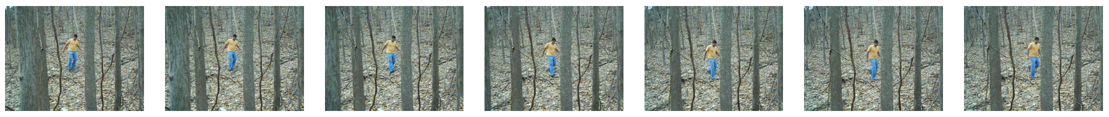
Incorporating scene depth for camera rotation estimation. Our approach incorporates relative depth estimates of the scene to improve camera rotation estimation (Chapter 5.2.1.2). General statistics capturing the distribution over inverse scene depth are included in our flow likelihood presented in Chapter 5.2.1.1. This distribution captures important statistics of inverse depth across multiple different videos (statistics are measured using the Sintel data set [15]). However scene depth varies quite a bit among different videos showing different scenes and can even vary within a single frame, if variation in depth is high. Indoor scenes show a quite different distribution of the inverse depth than an outdoor scene where most objects are at distant. This detailed information about *local relative depth* for unique scene structures is not captured by our flow likelihood function. In future work depth could be modeled more locally by taking depth estimates of the past into account instead of having one global depth distribution for all videos regardless of the pictured scene.

7.1.2 Handling cases of weak motion information in optical flow

Optical flow captures the motion between two consecutive frames. However motion patterns change over time so an object's motion can be temporally hard to separate from static background. These challenging cases can occur, when (i) an object or just part of an object stands still just for a short amount of time, (ii) its motion doesn't differ significantly from the camera's motion or (iii) optical flow is misestimated. In those cases motion segmentation approaches that are solely based on a single optical flow field might fail. Considering long-term motion analysis and object appearance are possibilities to overcome these challenges, which are possible future directions to address in further detail.

Long-term motion analysis for a temporal consistent segmentation. In Chapter 4.1 and 4.2 temporal consistencies are naturally incorporated into our approach. We segment a frame into its different motion components in a Bayesian fashion, where the prior probability of a motion component is developed based on its posterior probability of the previous frame. Our work on learning motion patterns from optical flow however doesn't take any temporal information into account. In Chapter 5.1 motion patterns of objects are learned given a single rotation compensated optical flow field. Since object motion patterns change smoothly over time a long-term motion analysis based on multiple video frames or taking the estimate of the previous frame into account might be helpful to refine those segmentations and to guarantee a temporal consistent segmentation regardless uncertainties in optical flow.

Incorporating object appearance in cases where the object's motion cues alone are not sufficient. We tackle the problem of motion segmentation, thus objects appearance is only a secondary cue to be considered. However object appearance clearly can add additional valuable information to segment moving objects in a more


(a) video frame

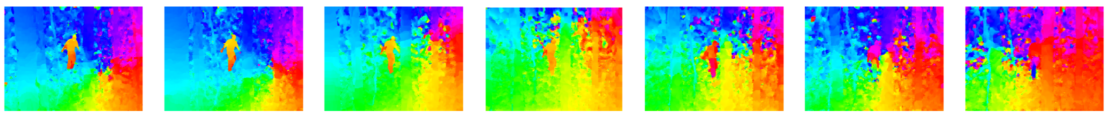

(b) angle field of rotation compensated flow

Figure 7.1: Motion information contained in optical flow is not always sufficient to segment the moving object accurately. We show frame 1,4,7,10,13,16 and 19 of the forest video sequence of the complex background data set [72]. Note that in the first five frames the person is clearly visible in the angle field (second row). In the last two frames the person's motion "melts" with the background motion and is hard to separate based on optical flow information alone. Considering temporal information as well as information about objects appearance are possibilities to address weak motion information in optical flow in this case.

robust manner in several cases. These cases are (i) partial object motion, where part of an object moves while other parts are still or (ii) object motion temporally moves into the same direction as the camera such that their motion patterns "melt" into each other. An example showing these challenging cases is given in Figure 7.1. In the first five frames the person clearly is visible in the angle field of the translational optical flow field. However in the last frames the persons motion melts with the background motion and is hard to segment based on the current optical flow field alone. An appearance model has been incorporated in many previous works addressing motion segmentation [42, 110, 111]. These appearance models are mostly implemented as a separate network stream which connected with the motion segmentation stream in a final step. Rather than learning segmentations based on motion and appearance in isolation, future work might benefit from learning motion patterns using appearance cues and motion cues simultaneously.

7.2 Future directions

Computer vision and human perception are closely related research areas. How can one teach a computer to see and understand the world as we humans do? What are the strengths of a computer vision system compared to a human vision system and what are the weaknesses? We move, we discover new interesting stuff that raises our curiosity – especially if a perceived situation doesn't match certain expectations, and we learn.

Teaching a machine to see, to understand what it sees and setting the perceived information into context is a highly challenging task towards creating the opportunity for smart interactions between humans and machines. Yet we are far away from machines being able to interact successfully with an unstructured environment involving interaction with objects as well as humans. The work of this thesis addressed the challenge of segmenting independently moving objects in an arbitrary environment. Motion - our motion as well as our motion perception - is a key ability that we as living beings have to explore our environment. Our motion for example helps us to perceive depth, and the motion of objects helps us recognizing these objects even if those are unknown to us or due to missing unique appearance cues invisible when they are still. We might not be able to name those objects at that time, but due to their independent motion we perceive them as objects and connect them with important information about their motion characteristics. Solely based on motion we are able to detect objects in complex scenes regardless their appearance, which is a big step towards understanding an unstructured environment.

As a next subsequent step it is important not only to perceive the world as it is, one also has to understand situations, set objects and actions into context with each other to form expectations about what is going to happen next. Then the action has to be performed as a logical consequence based on previously perceived information and its resulting expectations. If we ask a vision system - a machine - what should be expected next, we ask the system to solve multiple standard vision tasks and to connect those simultaneously. Examples of those tasks include detecting actors and analyzing their motions, actions as well as their interactions with each other while *being consistent with the physics of the real world.* If an actor climbed a mountain and is walking towards a cliff he will stop walking as soon as he reached the cliff to enjoy the view. Obviously he will not continue walking since he would fall. We know that, not because we have seen hundreds of people fall off cliffs. Instead, we understand the principles of the physical world and can easily adapt to new scenarios. Creating a common sense understanding (in this context often referred to as intuitive physics) for machines requires consideration of many different aspects, which makes it an especially challenging and interesting task for current vision systems.

In future incorporating an understanding about physical principles into learning based vision systems will be essential to build machines that understand observed scenarios and can adapt learned principles to a never seen situation - a big step towards intelligent interactions.

BIBLIOGRAPHY

[1] Achanta, Radhakrishna, Shaji, Appu, Smith, Kevin, Lucchi, Aurelien, Fua, Pascal, and Süsstrunk, Sabine. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence 34*, 11 (2012), 2274–2282.

[2] Adelson, Edward H, and Movshon, J Anthony. Phenomenal coherence of moving visual patterns. *Nature 300*, 5892 (1982), 523.

[3] Baker, Simon, Scharstein, Daniel, Lewis, JP, Roth, Stefan, Black, Michael J, and Szeliski, Richard. A database and evaluation methodology for optical flow. *International Journal of Computer Vision 92*, 1 (2011), 1–31.

[4] Bideau, Pia, and Learned-Miller, Erik. A detailed rubric for motion segmentation. *arXiv preprint arXiv:1610.10033* (2016).

[5] Bideau, Pia, and Learned-Miller, Erik. Its moving! A probabilistic model for causal motion segmentation in moving camera videos. In *Proc. ECCV* (2016), Springer, pp. 433–449.

[6] Bideau, Pia, Menon, Rakesh R., and Learned-Miller, Erik. Moa-net: Self-supervised motion segmentation. In *The European Conference on Computer Vision (ECCV) Workshops* (September 2018).

[7] Bideau, Pia, RoyChowdhury, Aruni, Menon, Rakesh R, and Learned-Miller, Erik. The best of both worlds: Combining cnns and geometric constraints for hierarchichal motion segmentation. In *CVPR* (2018).

[8] Black, Michael J, and Anandan, Paul. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer vision and image understanding 63*, 1 (1996), 75–104.

[9] Born, Richard T, and Bradley, David C. Structure and function of visual area mt. *Annu. Rev. Neurosci. 28* (2005), 157–189.

[10] Braddick, Oliver J, O'Brien, Justin MD, Wattam-Bell, John, Atkinson, Janette, Hartley, Tom, and Turner, Robert. Brain areas sensitive to coherent visual motion. *Perception 30*, 1 (2001), 61–72.

[11] Brown, Duane. The bundle adjustment-progress and prospect. In *XIII Congress of the ISPRS, Helsinki, 1976* (1976).

[12] Brox, Thomas, and Malik, Jitendra. Object segmentation by long term analysis of point trajectories. In *Proc. ECCV* (2010), Springer, pp. 282–295.

[13] Brox, Thomas, and Malik, Jitendra. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE transactions on pattern analysis and machine intelligence 33*, 3 (2011), 500–513.

[14] Bruss, Anna R, and Horn, Berthold KP. Passive navigation. *Computer Vision, Graphics, and Image Processing 21*, 1 (1983), 3–20.

[15] Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. A naturalistic open source movie for optical flow evaluation. In *Proc. ECCV* (2012), Springer-Verlag, pp. 611–625.

[16] Chen, Liang-Chieh, Papandreou, George, Kokkinos, Iasonas, Murphy, Kevin, and Yuille, Alan L. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *Proc. ICLR* (2015).

[17] Chen, Liang-Chieh, Papandreou, George, Kokkinos, Iasonas, Murphy, Kevin, and Yuille, Alan L. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *arXiv preprint arXiv:1606.00915* (2016).

[18] Cheng, J., Tsai, Y.-H., Wang, S., and Yang, M.-H. Segflow: Joint learning for video object segmentation and optical flow. In *IEEE International Conference on Computer Vision (ICCV)* (2017).

[19] Dave, Achal, Tokmakov, Pavel, and Ramanan, Deva. Towards segmenting everything that moves. *arXiv preprint arXiv:1902.03715* (2019).

[20] Dürer, A., and Strauss, W.L. *The Painter's Manual: A Manual of Measurement of Lines, Areas, and Solids by Means of Compass and Ruler Assembled by Albrecht Dürer for the Use of All Lovers of Art with Appropriate Illustrations Arranged to be Printed in the Year MDXXV.* The literary remains of Albrecht Dürer. Abaris Books, 1977.

[21] Edgerton, Samuel Y. Brunelleschi's mirror, alberti's window, and galileo's' perspective tube'. *História, Ciências, Saúde-Manguinhos 13* (2006), 151–179.

[22] Faktor, Alon, and Irani, Michal. Video segmentation by non-local consensus voting. In *Proc. BMVC* (2014), vol. 2, p. 8.

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

[24] Fragkiadaki, Aikaterini, Seybold, Bryan, Schmid, Cordelia, Sukthankar, Rahul, Vijayanarasimhan, Sudheendra, and Ricco, Susanna. Self-supervised learning of structure and motion from video. In *arxiv (2017)* (2017).

[25] Fragkiadaki, Katerina, Zhang, Geng, and Shi, Jianbo. Video segmentation by tracing discontinuities in a trajectory embedding. In *Proc. CVPR* (2012), pp. 1846–1853.

[26] Fukushima, Junko, Akao, Teppei, Kurkin, Sergei, Kaneko, Chris RS, and Fukushima, Kikuro. The vestibular-related frontal cortex and its role in smooth-pursuit eye movements and vestibular-pursuit interactions. *Journal of Vestibular Research 16*, 1, 2 (2006), 1–22.

[27] Girshick, Ross. Fast R-CNN. In *Proc. ICCV* (2015), pp. 1440–1448.

[28] Girshick, Ross, Donahue, Jeff, Darrell, Trevor, and Malik, Jitendra. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2014), pp. 580–587.

[29] Gopnik, Alison, Meltzoff, Andrew N, and Kuhl, Patricia K. *The scientist in the crib: What early learning tells us about the mind.* William Morrow Paperbacks, 2000.

[30] Granshaw, SI. Bundle adjustment methods in engineering photogrammetry. *The Photogrammetric Record 10*, 56 (1980), 181–207.

[31] Hariharan, Bharath, Arbeláez, Pablo, Girshick, Ross, and Malik, Jitendra. Hypercolumns for object segmentation and fine-grained localization. In *Proc. CVPR* (2015), pp. 447–456.

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

[33] He, Kaiming, Gkioxari, Georgia, Dollár, Piotr, and Girshick, Ross. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2961–2969.

[34] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. In *Proc. CVPR* (2016), pp. 770–778.

[35] Horn, Berthold, Klaus, Berthold, and Horn, Paul. *Robot vision.* MIT press, 1986.

[36] Horn, Berthold KP. Projective geometry considered harmful. *Unpublished Memo* (1999).

[37] Horn, Berthold KP, and Schunck, Brian G. Determining optical flow. *Artificial intelligence 17*, 1-3 (1981), 185–203.

[38] Hur, Junhwa, and Roth, Stefan. Joint optical flow and temporally consistent semantic segmentation. In *European Conference on Computer Vision* (2016), Springer, pp. 163–177.

[39] Ilg, Eddy, Mayer, Nikolaus, Saikia, Tonmoy, Keuper, Margret, Dosovitskiy, Alexey, and Brox, Thomas. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 2462–2470.

[40] Irani, Michal, and Anandan, P. A unified approach to moving object detection in 2D and 3D scenes. *PAMI 20*, 6 (1998), 577–589.

[41] Jaegle, Andrew, Phillips, Stephen, and Daniilidis, Kostas. Fast, robust, continuous monocular egomotion computation. In *Proc. ICRA* (2016), IEEE, pp. 773–780.

[42] Jain, Suyog, Xiong, Bo, and Grauman, Kristen. FusionSeg: Learning to combine motion and appearance for fully automatic segmention of generic objects in videos. In *Proc. CVPR* (2017).

[43] Jain, Suyog Dutt, and Grauman, Kristen. Supervoxel-consistent foreground propagation in video. In *European Conference on Computer Vision* (2014), Springer, pp. 656–671.

[44] Jiang, Huaizu, Larsson, Gustav, Maire Greg Shakhnarovich, Michael, and Learned-Miller, Erik. Self-supervised relative depth learning for urban scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 19–35.

[45] Jin, Yuxin, Tao, Linmi, Di, Huijun, Rao, Naveed I, and Xu, Guangyou. Background modeling from a free-moving camera by multi-layer homography algorithm. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on* (2008), IEEE, pp. 1572–1575.

[46] Ke, Qifa, and Kanade, Takeo. A robust subspace approach to layer extraction. In *Motion and Video Computing, 2002. Proceedings. Workshop on* (2002), IEEE, pp. 37–43.

[47] Keuper, Margret. Higher-order minimum cost lifted multicuts for motion segmentation. In *Proc. ICCV* (2017).

[48] Keuper, Margret, Andres, Bjoern, and Brox, Thomas. Motion trajectory segmentation via minimum cost multicuts. In *Proc. ICCV* (2015), pp. 3271–3279.

[49] Koh, Yeong Jun, and Kim, Chang-Su. Primary object segmentation in videos based on region augmentation and reduction. In *Proc. CVPR* (2017), IEEE, pp. 7417–7425.

[50] Krähenbühl, Philipp, and Koltun, Vladlen. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Proc. NIPS* (2011), pp. 109–117.

[51] Krizhevsky, Alex, Hinton, Geoffrey, et al. Learning multiple layers of features from tiny images. Tech. rep., Citeseer, 2009.

[52] Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. ImageNet classification with deep convolutional neural networks. In *Proc. NIPS* (2012), pp. 1097–1105.

[53] Kundu, Abhijit, Krishna, K Madhava, and Jawahar, CV. Realtime motion segmentation based multibody visual slam. In *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing* (2010), ACM, pp. 251–258.

[54] Land, Michael F. Motion and vision: Why animals move their eyes. *Journal of Comparative Physiology A 185*, 4 (1999), 341–352.

[55] Lao, Dong, and Sundaramoorthi, Ganesh. Extending layered models to 3D motion. In *Proc. ECCV* (2018).

[56] LeCun, Yann, Boser, Bernhard, Denker, John S, Henderson, Donnie, Howard, Richard E, Hubbard, Wayne, and Jackel, Lawrence D. Backpropagation applied to handwritten zip code recognition. *Neural computation 1*, 4 (1989), 541–551.

[57] Lee, Yong Jae, Kim, Jaechul, and Grauman, Kristen. Key-segments for video object segmentation. In *Proc. ICCV* (2011), IEEE, pp. 1995–2002.

[58] Lezama, José, Alahari, Karteek, Sivic, Josef, and Laptev, Ivan. Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *CVPR 2011* (2011), IEEE, pp. 3369–3376.

[59] Li, Fuxin, Kim, Taeyoung, Humayun, Ahmad, Tsai, David, and Rehg, James M. Video segmentation by tracking many figure-ground segments. In *Proceedings of the IEEE International Conference on Computer Vision* (2013), pp. 2192–2199.

[60] Li, Siyi, and Yeung, Dit-Yan. Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models. In *Thirty-First AAAI Conference on Artificial Intelligence* (2017).

[61] Li, Yi, Qi, Haozhi, Dai, Jifeng, Ji, Xiangyang, and Wei, Yichen. Fully convolutional instance-aware semantic segmentation. *arXiv preprint arXiv:1611.07709* (2016).

[62] Lin, Tsung-Yi, Maire, Michael, Belongie, Serge, Hays, James, Perona, Pietro, Ramanan, Deva, Dollár, Piotr, and Zitnick, C Lawrence. Microsoft coco: Common objects in context. In *European conference on computer vision* (2014), Springer, pp. 740–755.

[63] Liu, Ce, Yuen, Jenny, and Torralba, Antonio. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence 33*, 5 (2010), 978–994.

[64] Liu, Wei, Anguelov, Dragomir, Erhan, Dumitru, Szegedy, Christian, Reed, Scott, Fu, Cheng-Yang, and Berg, Alexander C. SSD: Single shot multibox detector. In *Proc. ECCV* (2016), Springer, pp. 21–37.

[65] Long, Jonathan, Shelhamer, Evan, and Darrell, Trevor. Fully convolutional networks for semantic segmentation. In *Proc. CVPR* (2015), pp. 3431–3440.

[66] Longuet-Higgins, Hugh Christopher, Prazdny, Kvetoslav, et al. The interpretation of a moving retinal image. *Proc. R. Soc. Lond. B 208*, 1173 (1980), 385–397.

[67] Loquercio, Antonio, Maqueda, Ana I, Del-Blanco, Carlos R, and Scaramuzza, Davide. Dronet: Learning to fly by driving. *IEEE Robotics and Automation Letters 3*, 2 (2018), 1088–1095.

[68] Lucas, Bruce D, Kanade, Takeo, et al. An iterative image registration technique with an application to stereo vision.

[69] Mahendran, Aravindh, Thewlis, James, and Vedaldi, Andrea. Self-supervised segmentation by grouping optical-flow. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 0–0.

[70] Mattyus, Gellert, Wang, Shenlong, Fidler, Sanja, and Urtasun, Raquel. Enhancing road maps by parsing aerial images around the world. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 1689–1697.

[71] Mayer, Nikolaus, Ilg, Eddy, Hausser, Philip, Fischer, Philipp, Cremers, Daniel, Dosovitskiy, Alexey, and Brox, Thomas. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4040–4048.

[72] Narayana, Manjunath, Hanson, Allen, and Learned-Miller, Erik. Coherent motion segmentation in moving camera videos using optical flow orientations. In *Proc. ICCV* (2013), pp. 1577–1584.

[73] Narayana, Manjunath, Hanson, Allen, and Learned-Miller, Erik G. Background subtraction: separating the modeling and the inference. *Machine vision and applications 25*, 5 (2014), 1163–1174.

[74] Noroozi, Mehdi, and Favaro, Paolo. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision* (2016), Springer, pp. 69–84.

[75] Ochs, Peter, and Brox, Thomas. Object segmentation in video: A hierarchical variational approach for turning point trajectories into dense regions. In *Proc. ICCV* (2011), IEEE, pp. 1583–1590.

[76] Ochs, Peter, and Brox, Thomas. Higher order motion models and spectral clustering. In *Proc. CVPR* (2012), IEEE, pp. 614–621.

[77] Ochs, Peter, Malik, Jitendra, and Brox, Thomas. Segmentation of moving objects by long term video analysis. *PAMI 36*, 6 (2014), 1187–1200.

[78] Ogale, Abhijit S, Fermüller, Cornelia, and Aloimonos, Yiannis. Motion segmentation using occlusions. *PAMI 27*, 6 (2005), 988–992.

[79] Otsu, Nobuyuki. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics 9*, 1 (1979), 62–66.

[80] Pan, Xuran, Gao, Lianru, Marinoni, Andrea, Zhang, Bing, Yang, Fan, and Gamba, Paolo. Semantic labeling of high resolution aerial imagery and lidar data with fine segmentation network. *Remote Sensing 10*, 5 (2018), 743.

[81] Papazoglou, Anestis, and Ferrari, Vittorio. Fast object segmentation in unconstrained video. In *Proc. ICCV* (2013), pp. 1777–1784.

[82] Pathak, Deepak, Girshick, Ross, Dollár, Piotr, Darrell, Trevor, and Hariharan, Bharath. Learning features by watching objects move. In *Proc. CVPR* (2017).

[83] Perazzi, Federico, Pont-Tuset, Jordi, McWilliams, Brian, Van Gool, Luc, Gross, Markus, and Sorkine-Hornung, Alexander. A benchmark dataset and evaluation methodology for video object segmentation. In *Proc. CVPR* (2016), pp. 724–732.

[84] Pinard, Clement, Chevalley, Laure, Manzanera, Antoine, and Filliat, David. Learning structure-from-motion from motion. In *The European Conference on Computer Vision (ECCV) Workshops* (September 2018).

[85] Pinheiro, Pedro O, Collobert, Ronan, and Dollár, Piotr. Learning to segment object candidates. In *Advances in Neural Information Processing Systems* (2015), pp. 1990–1998.

[86] Pinheiro, Pedro O., Lin, Tsung-Yi, Collobert, Ronan, and Dollr, Piotr. Learning to refine object segments. In *Proc. ECCV* (2016).

[87] Pirenne, Maurice Henri. The scientific basis of leonardo da vinci's theory of perspective. *The British Journal for the Philosophy of Science 3*, 10 (1952), 169–185.

[88] Pont-Tuset, Jordi, Perazzi, Federico, Caelles, Sergi, Arbeláez, Pablo, Sorkine-Hornung, Alexander, and Van Gool, Luc. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675* (2017).

[89] Powers, David Martin. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation.

[90] Prest, Alessandro, Leistner, Christian, Civera, Javier, Schmid, Cordelia, and Ferrari, Vittorio. Learning object class detectors from weakly annotated video. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (2012), IEEE, pp. 3282–3289.

[91] Ranjan, Anurag, Jampani, Varun, Balles, Lukas, Kim, Kihwan, Sun, Deqing, Wulff, Jonas, and Black, Michael J. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 12240–12249.

[92] Redmon, Joseph, Divvala, Santosh, Girshick, Ross, and Farhadi, Ali. You only look once: Unified, real-time object detection. In *Proc. CVPR* (2016), pp. 779–788.

[93] Redmon, Joseph, and Farhadi, Ali. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 7263–7271.

[94] Redmon, Joseph, and Farhadi, Ali. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).

[95] Ren, Shaoqing, He, Kaiming, Girshick, Ross, and Sun, Jian. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (2015), pp. 91–99.

[96] Revaud, Jerome, Weinzaepfel, Philippe, Harchaoui, Zaid, and Schmid, Cordelia. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 1164–1172.

[97] Ronneberger, Olaf, Fischer, Philipp, and Brox, Thomas. U-net: Convolutional networks for biomedical image segmentation. In *Proc. MICCAI* (2015), Springer, pp. 234–241.

[98] Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision 115*, 3 (2015), 211–252.

[99] Sevilla-Lara, Laura, Sun, Deqing, Jampani, Varun, and Black, Michael J. Optical flow with semantic segmentation and localized layers. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016).

[100] Sevilla-Lara, Laura, Sun, Deqing, Jampani, Varun, and Black, Michael J. Optical flow with semantic segmentation and localized layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 3889–3898.

[101] Shen, Jianbing, Peng, Jianteng, and Shao, Ling. Submodular trajectories for better motion segmentation in videos. *IEEE Transactions on Image Processing 27*, 6 (2018), 2688–2700.

[102] Shum, Heung-Yeung, Ke, Qifa, and Zhang, Zhengyou. Efficient bundle adjustment with virtual key frames: A hierarchical approach to multi-frame structure from motion. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)* (1999), vol. 2, IEEE, pp. 538–543.

[103] Simonyan, Karen, and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[104] Smith, Linda, and Gasser, Michael. The development of embodied cognition: Six lessons from babies. *Artificial life 11*, 1-2 (2005), 13–29.

[105] Stückler, Jörg, and Behnke, Sven. Efficient dense rigid-body motion segmentation and estimation in rgb-d video. *International Journal of Computer Vision 113*, 3 (2015), 233–245.

[106] Sun, Deqing, Roth, Stefan, and Black, Michael J. Secrets of optical flow estimation and their principles. In *Proc. CVPR* (2010), IEEE, pp. 2432–2439.

[107] Sun, Deqing, Yang, Xiaodong, Liu, Ming-Yu, and Kautz, Jan. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 8934–8943.

[108] Tang, Kevin, Sukthankar, Rahul, Yagnik, Jay, and Fei-Fei, Li. Discriminative segment annotation in weakly labeled video. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2013), pp. 2483–2490.

[109] Taylor, Brian, Karasev, Vasiliy, and Soatto, Stefano. Causal video object segmentation from persistence of occlusions. In *Proc. CVPR* (2015), pp. 4268–4276.

[110] Tokmakov, P., Alahari, K., and Schmid, C. Learning motion patterns in videos. In *Proc. CVPR* (2017).

[111] Tokmakov, P., Alahari, K., and Schmid, C. Learning video object segmentation with visual memory. In *Proc. ICCV* (2017).

[112] Torr, Philip HS. Geometric motion segmentation and model selection. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 356*, 1740 (1998), 1321–1340.

[113] Triggs, Bill, McLauchlan, Philip F, Hartley, Richard I, and Fitzgibbon, Andrew W. Bundle adjustmenta modern synthesis. In *International workshop on vision algorithms* (1999), Springer, pp. 298–372.

[114] Tron, Roberto, and Vidal, René. A benchmark for the comparison of 3-d motion segmentation algorithms. In *2007 IEEE conference on computer vision and pattern recognition* (2007), IEEE, pp. 1–8.

[115] Tung, Hsiao-Yu Fish, Cheng, Ricson, and Fragkiadaki, Katerina. Learning spatial common sense with geometry-aware recurrent networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 2595–2603.

[116] Van den Bergh, Michael, and Van Gool, Luc. Real-time stereo and flow-based video segmentation with superpixels. In *2012 IEEE Workshop on the Applications of Computer Vision (WACV)* (2012), IEEE, pp. 89–96.

[117] Vertens, Johan, Valada, Abhinav, and Burgard, Wolfram. SMSnet: Semantic motion segmentation using deep convolutional neural networks. In *Proc. IROS* (2017).

[118] Vidal, René, and Ma, Yi. A unified algebraic approach to 2-d and 3-d motion segmentation. In *European Conference on Computer Vision* (2004), Springer, pp. 1–15.

[119] Vidal, René, Soatto, Stefano, Ma, Yi, and Sastry, Shankar. Segmentation of dynamic scenes from the multibody fundamental matrix. In *ECCV Workshop on Vision and Modeling of Dynamic Scenes* (2002).

[120] Vijayanarasimhan, Sudheendra, Ricco, Susanna, Schmid, Cordelia, Sukthankar, Rahul, and Fragkiadaki, Katerina. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804* (2017).

[121] Vondrick, Carl, Shrivastava, Abhinav, Fathi, Alireza, Guadarrama, Sergio, and Murphy, Kevin. Tracking emerges by colorizing videos. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 391–408.

[122] Walls, GL. The evolutionary history of eye movements. *Vision Research 2*, 1-4 (1962), 69–80.

[123] Wang, John YA, and Adelson, Edward H. Representing moving images with layers. *IEEE Transactions on Image Processing 3*, 5 (1994), 625–638.

[124] Wang, Shenlong, Bai, Min, Mattyus, Gellert, Chu, Hang, Luo, Wenjie, Yang, Bin, Liang, Justin, Cheverie, Joel, Fidler, Sanja, and Urtasun, Raquel. Torontocity: Seeing the world with a million eyes. *arXiv preprint arXiv:1612.00423* (2016).

[125] Wedel, Andreas, Meißner, Annemarie, Rabe, Clemens, Franke, Uwe, and Cremers, Daniel. Detection and segmentation of independently moving objects from dense scene flow. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition* (2009), Springer, pp. 14–27.

[126] Wei, Donglai, Lim, Joseph J, Zisserman, Andrew, and Freeman, William T. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 8052–8060.

[127] Wolf, Paul R, and Ghilani, CD. Adjustment computations-statistics and least. *squares in Surveying and GIS* (1997).

[128] Wulff, J., Butler, D. J., Stanley, G. B., and Black, M. J. Lessons and insights from creating a synthetic optical flow benchmark. In *ECCV Workshop on Unsolved Problems in Optical Flow and Stereo Estimation* (Oct. 2012), A. Fusiello et al. (Eds.), Ed., Part II, LNCS 7584, Springer-Verlag, pp. 168–177.

[129] Wulff, Jonas, Sevilla-Lara, Laura, and Black, Michael J. Optical flow in mostly rigid scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (July 2017).

[130] Xiao, Jiangjian, and Shah, Mubarak. Motion layer extraction in the presence of occlusion using graph cuts. *IEEE transactions on pattern analysis and machine intelligence 27*, 10 (2005), 1644–1659.

[131] Xu, Xun, Fah Cheong, Loong, and Li, Zhuwen. Motion segmentation by exploiting complementary geometric models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 2859–2867.

[132] Yan, Jingyu, and Pollefeys, Marc. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *European conference on computer vision* (2006), Springer, pp. 94–106.

[133] Yang, Yanchao, Loquercio, Antonio, Scaramuzza, Davide, and Soatto, Stefano. Unsupervised moving object detection via contextual information separation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 879–888.

[134] Zamalieva, Daniya, and Yilmaz, Alper. Background subtraction for the moving camera: A geometric approach. *CVIU 127* (2014), 73–85.

[135] Zhang, Guofeng, Jia, Jiaya, Xiong, Wei, Wong, Tien-Tsin, Heng, Pheng-Ann, and Bao, Hujun. Moving object extraction with a hand-held camera. In *2007 IEEE 11th International Conference on Computer Vision* (2007), IEEE, pp. 1–8.

[136] Zhang, Richard, Isola, Phillip, and Efros, Alexei A. Colorful image colorization. In *European conference on computer vision* (2016), Springer, pp. 649–666.

[137] Zhao, Hengshuang, Shi, Jianping, Qi, Xiaojuan, Wang, Xiaogang, and Jia, Jiaya. Pyramid scene parsing network. *arXiv preprint arXiv:1612.01105* (2016).

[138] Zhou, Tinghui, Brown, Matthew, Snavely, Noah, and Lowe, David G. Unsupervised learning of depth and ego-motion from video. In *CVPR* (2017).