OPTICAL FLOW TEMPLATES FOR MOBILE ROBOT ENVIRONMENT UNDERSTANDING

A Thesis
Presented to
The Academic Faculty

by

Richard Roberts

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Interactive Computing

Georgia Institute of Technology
Dec 2014

OPTICAL FLOW TEMPLATES FOR MOBILE ROBOT ENVIRONMENT UNDERSTANDING

Approved by:

Professor Irfan Essa, Committee Chair
School of Interactive Computing
Georgia Institute of Technology

Professor Frank Dellaert, Adviser
School of Interactive Computing
Georgia Institute of Technology

Professor James M. Rehg
School of Interactive Computing
Georgia Institute of Technology

Professor Panagiotis Tsiotris
School of Aerospace Engineering
Georgia Institute of Technology

Professor Fernando de la Torre
Robotics Institute, Carnegie Mellon University
Georgia Institute of Technology

Date Approved: 8 June 2014

ACKNOWLEDGEMENTS

I dedicate this thesis to my family and friends. Heather, Michael, Mom, your belief in me, love, and support are a vital source of strength, tranquility, and self-worth. David, Miguel, Melissa, Marie, Pablo, Ana, Luca, your love, friendship, and wisdom have been an incredible source of life, happiness, and the most important education. People I met and spent time with along the way made this experience incredible, Jinhan, John, Alex T., Crystal, Alex C., Maya, and many others. I am truly lucky to know all of you and to have gone through these years with you, much more than I can express. Frank, your guidance, wisdom, and unwavering encouragement of me and commitment to the highest standards, have made this work and my experience in grad school great things that they truly would not have been without you. The experiences I've had and lessons I've learned are already proving invaluable and I greatly appreciate the time I spent with you. I can't imagine having this experience with anyone else.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

In this work we develop optical flow templates. In doing so, we introduce a practical tool for inferring robot egomotion and semantic superpixel labeling using optical flow in imaging systems with arbitrary optics. In order to do this we develop valuable understanding of geometric relationships and mathematical methods that are useful in interpreting optical flow to the robotics and computer vision communities.

This work is motivated by what we perceive as directions for advancing the current state of the art in obstacle detection and scene understanding for mobile robots. Specifically, many existing methods build 3D point clouds, which are not directly useful for autonomous navigation and require further processing. Both the step of building the point clouds and the later processing steps are challenging and computationally intensive. Additionally, many current methods require a calibrated camera, which introduces calibration challenges and places limitations on the types of camera optics that may be used. Wide-angle lenses, systems with mirrors, and multiple cameras all require different calibration models and can be difficult or impossible to calibrate at all. Finally, current pixel and superpixel obstacle labeling algorithms typically rely on image appearance. While image appearance is informative, image motion is a direct effect of the scene structure that determines whether a region of the environment is an obstacle.

The egomotion estimation and obstacle labeling methods we develop here based on optical flow templates require very little computation per frame and do not require building point clouds. Additionally, they do not require any specific type of camera optics, nor a calibrated camera. Finally, they label obstacles using optical flow alone without image appearance.

In this thesis we start with optical flow subspaces for egomotion estimation and detection of "motion anomalies". We then extend this to multiple subspaces and develop mathematical reasoning to select between them, comprising optical flow templates. Using these we classify environment shapes and label superpixels. Finally, we show how performing all learning and inference directly from image spatio-temporal gradients greatly improves computation time and accuracy.

CHAPTER I

INTRODUCTION

1.1 Objective

In this thesis we develop and evaluate *optical flow templates*. Our contributions in doing so are: *First*, we introduce a practical tool for tasks such as inferring robot egomotion and semantic superpixel labeling using optical flow, in imaging systems with arbitrary optics. *Second*, we contribute understanding of geometric relationships and mathematical methods that are useful in interpreting optical flow to the robotics and computer vision communities.

1.2 Thesis Statement

Optical flow templates are an effective tool for capturing information on large-scale environment structure and robot egomotion, which is useful for autonomous navigation of a robot, from arbitrary uncalibrated imaging systems on mobile robots.

1.3 Claims

To support the thesis statement we make several claims, each supported by one chapter.

Learned optical flow subspaces are an effective tool for inferring robot egomotion and motion anomalies from cameras with arbitrary optics, in scenarios exhibiting depth regularity. (Chapter 3) Optical flow subspaces are the basic object that we expand on in later chapters.

Selecting among multiple optical flow subspaces allows classification of coarse environment shapes from cameras with arbitrary optics, though this information is of limited usefulness to robot navigation. (Chapter 4) Our first attempt to infer semantic structure from optical flow was to classify coarse environment shapes such as "path curving left" or "wall on right". While effective in restricted scenarios, we do not believe this method is easily applicable to more general environments, nor is the information inferred very useful for autonomous navigation.

Optical flow templates are an effective tool for semantically labeling superpixels as "ground plane", "distant", or "obstacle" from optical flow measured in a camera with arbitrary optics, and this information is useful for autonomous robot navigation. (Chapter 5) Much modern work in autonomous navigation labels image regions according to how traversable they are, then uses that information to plan trajectories or control the robot. Optical flow templates provide similar information directly from optical flow, a signal complementary to image appearance, and with no need for a stereo rig.

Accuracy, versatility, and computational efficiency of optical flow templates is improved by several adaptations informed by our experiences working with optical flow in this thesis. (Chapter 6) We extend our previous work to learn templates from data instead of calculating them, improve the modeling of the "obstacle" class, and perform learning and inference directly from image spatiotemporal gradients instead of from optical flow.

1.4 Motivation

The motivation for this work stems from characteristics of the current state of the art in obstacle detection and scene understanding that we consider limitations for the purpose of autonomous robot navigation.

First, many typical obstacle detection and perception algorithms in use on mobile robots require building 3D point clouds. The limitation here is twofold. Building and storing the point clouds requires significant computation, but also the point clouds themselves are not immediately useful for navigation. Post-processing must be performed to determine the location and extend of obstacles given the point clouds. This processing task becomes increasingly difficult as the density of the point clouds decreases, yet the difficulty of obtaining the point clouds increases with the density of the point cloud. The latter is due to increased computational and storage requirements as well as fundamental limitations of structure-from-motion algorithms to deal with image regions containing little visual texture.

Second, many perception algorithms also require a calibrated camera, particularly those that build point clouds or perform structure from motion as an intermediate step. Camera calibration models impose limits on camera optics. In some calibration models, field of view is limited, and to widen it often requires either multiple cameras or wide-angle lenses that present a calibration challenge due to extreme lens distortion. Further, many useful imaging systems cannot be easily modeled via perspective and parametric distortion model. Examples of these include systems with a rear-view mirror or other non-parabolic mirror, some wide field-of-view lenses, inexpensive lenses with uneven distortion, and multiple cameras with tiled images treated as a single imaging system. Although calibration models vary in generality, no single model covers all optics. Furthermore, difficulty of calibration process, in terms of number of images required, sensitivity to image placement, and need for special equipment, increases considerably with more general models.

Third, many obstacle detection and scene understanding methods that do not build point clouds rely exclusively on image appearance information, without taking optical flow into account. Unlike image appearance, optical flow is directly related to scene structure, and scene structure is often the primary decider of whether a region is an obstacle to a robot. There are a few families of existing obstacle detection methods using optical flow, but they rely either on hand-designed heuristics or again on camera calibration.

1.5 Overview

Chapter 1 is this introduction and Chapter 2 is the literature review. We start in Chapter 3 with optical flow subspaces for egomotion estimation and detection of "motion anomalies".

We then extend this to multiple subspaces and develop mathematical reasoning to select between them, comprising optical flow templates. Using these we classify environment shapes in Chapter 4 and label superpixels in Chapter 5. Finally, in Chapter 6 we show how performing all learning and inference directly from image spatio-temporal gradients greatly improves computation time and accuracy.

CHAPTER II

LITERATURE REVIEW

In Section 2.1 we review methods for estimating motion and/or structure from 2D correspondences or image pairs. In Section 2.2 we review work on generalized imaging systems. In Section 2.3 we review optical flow estimation. In Section 2.4 we survey 2D semantic image labeling, and in Section 2.5 3D scene understanding from images. Finally, in Section 2.7 we review methods for computing traversibility maps from images for mobile robot navigation.

2.1 Estimating Camera Motion and Scene Structure Between Video Frames

In this section we review methods for geometrically estimating camera motion and 3D feature point locations between two video frames, using 2D feature correspondences in those frames. Specifically, we survey epipolar constraints, nonlinear optimization methods, and approximations using subspaces and transformations of the residual function.

Motion and structure estimation has been the focus of much research over the past several decades. The problem is well-posed up to a uniform scale. Because it is also nonlinear, numerous simplifications and approximations have been proposed to allow linear initialization and more efficient but approximate estimation.

2.1.1 The Optical Flow Equation

To show the connection between camera motion and image point motion, we first introduce a general optical flow equation:

$$u(x) = -N_x(d\tau + \Omega \times \tilde{x}),$$

where $x \in \mathbb{R}^3$ is a point on the imaging surface, $\tilde{x} = \|x\|^{-1} x$ is this vector normalized, $u(x) \in \mathbb{R}^3$ is the optical flow vector at that point (always tangent to the imaging surface), d is the *inverse* distance from the camera principal point to the scene point imaged at x, τ is the camera translation vector, and Ω is the camera angular velocity. N_x is the *linear* transformation, a 3×3 matrix, that projects flow on the unit sphere at x to flow on the true imaging surface.

For general motion, a minimum of five points is needed to constrain all of the egomotion and depth variables . The problem is made nonlinear and computationally expensive by the bilinear dependence of flow upon the inverse depth d and the transation τ.

Some early methods of egomotion estimation were based on elimination and nonlinear optimization but considered only a small number of image points [66, e.g.], and some assumed constrained structure or motion [87, 83, e.g.]. These methods were not applicable to the general case of a perspective camera observing many noisy scene points. Generally-applicable methods were developed by simplifying and transforming the problem. As we'll

see these led to biased estimates, so much work went into reducing this bias. To see how these methods evolved, we'll take a brief tour of the history of egomotion estimation.

2.1.2 Algebraic Error Methods

The earliest general-purpose linear egomotion estimation methods used the *epipolar constraint*. This constraint is satisfied when a proposed camera motion is such that the two camera principal points and the projections of the same scene point in each camera are all coplanar. Advantageously, it is linear and yields often unique estimates of camera motion. The main drawback is that this estimate becomes inaccurate and biased under measurement noise. Thus, it may be used as an initialization point for nonlinear egomotion estimation methods, for example by Toscani and Faugeras [89], Weng *et al.* [95], and many others.

2.1.2.1 The Discrete-time Epipolar Constraint

Longuet-Higgins [55] and Tsai and Huang [91] simultaneously derived a linear constraint now known as the essential matrix constraint,

$$x'^{\mathsf{T}} x = 0,$$

meaning that x' in the second image lies on the line Ex, with x in the first image. In fact, Longuet-Higgins originally derived the essential matrix (with different notation) as

$$E = R(\Omega)\,\hat{\tau},$$

where the "hat" operator $\hat{\cdot}$ generates a skew symmetric matrix defined such that $\hat{a}b = a \times b$, and $R(\Omega) = \exp\hat{\Omega}$ is the rotation matrix associated with the axis-angle rotation coordinates Ω. Given this definition, we see that the constraint is that

$$\begin{aligned} 0 &= \left\langle x', R(\Omega)(\tau \times x) \right\rangle \\ &= \left\langle R(-\Omega)x', \tau \times x \right\rangle, \end{aligned}$$

meaning that the point projection in the second camera $R(-\Omega)x'$ lies on the plane spanned by the translation vector τ and the point projection in the first camera x (whose normal is $\tau \times x$). This specifies that both camera principal points as well as both image points all lie on the same plane.

Tsai and Huang [91], as well as many others, used the epipolar constraint to estimate egomotion. The basic constraint, however, minimizes an *algebraic* error, not accounting for noise in image measurements, and as a result becomes inaccurate and biased with noisy measurements. Thus, a number of papers attempt to improve upon the basic constraint. [96] derive a method that is less sensitive to noise but still biased.

2.1.2.2 The Instantaneous-time Epipolar Constraint

A differential epipolar constraint on image and camera velocities, and methods for recovering the camera velocity, was introduced by Zhuang *et al.* [101] and Waxman *et al.* [94]. This constraint arises from the relationship

$$\langle u, v \times x \rangle + \langle x \times \omega, v \times x \rangle = 0,$$

where u is an optical flow vector parallel to the imaging plane and v and ω are the translational and angular velocities, respectively, of the camera. Maybank [62] showed that up to three motion solutions may be consistent with this constraint for some flow fields, and illustrates the conditions under which this ambiguity occurs. Later, Ma *et al.* [59] presented an improved algorithm for noisy flow fields, a unified discussion of epipolar constraint methods with numerous practical and theoretical comparisons between methods, and illustrated connections with the subspace methods of Heeger and Jepson [31, 46] (we discuss these more later). Kanatani [47, 48] later proposed renormalization to reduce bias and noise sensitivity of differential epipolar methods.

2.1.2.3 The Fundamental Matrix Constraint

For uncalibrated cameras, the *fundamental matrix* encapsulates the discrete epipolar constraint [58, e.g.]. Viéville and Faugeras [92] and Brooks *et al.* [8] derived and experimentally verified the differential uncalibrated epipolar constraint.

2.1.3 Least-Squares Methods

2.1.3.1 The Spherical Imaging Surface Approximation

Bruss and Horn [10] developed an egomotion estimation method that is more accurate, but also more computationally expensive, than the epipolar constraint. While the epipolar constraint permits non-rigid scene point motions (as long as the scene points remain in their respective epipolar places), Bruss and Horn's residual enforces rigidity through a least-squares estimate of the depth of each point, via algebraically eliminating the depth. This residual function is linear in the rotational velocity (given an estimate of translation) and nonlinear in the translational velocity. Their residual function is equivalent to

$$\langle u(x), v \times x \rangle + \langle x \times \omega, v \times x \rangle = 0,$$

with variables as above except that here, the image locations x lie on the unit sphere, the optical flow $u(x)$ is tangent to the unit sphere at x, and the translation direction is constrained to $\|v\| = 1$.

Bruss and Horn's method is less computationally expensive than full nonlinear optimization because it employs a particular norm that allows for a linear solution for the camera rotation given translation estimated at each iteration. This norm is in fact equivalent to minimizing error on a spherical imaging surface. Adiv [3] estimates egomotion using the same residual as Bruss and Horn and also segments motion into self-consistent objects. Maybank [63] also uses the same residual and makes explicit the spherical imaging surface. Later comparative surveys by Tian *et al.* [84] and Zhang and Tomasi [100] show that this norm yields significantly biased motion estimates under the presence of noise.

Tomasi and Shi [88] eliminate rotation by measuring the change of angle between scene points from the camera principal point, as these measurements are invariant to rotation. They derive a constraint incorporating these measurements that is bilinear in camera translation and inverse depth. They then eliminate the inverse depth by reformulating the bilinear constraint as a subspace constraint given fixed translation, and nonlinearly optimize

w.r.t. the translation direction. As with previous work, a problem that introduces bias is the transformation of the true measurements without a corresponding transformation of the noise model.

2.1.3.2 Subspace Methods

Heeger and Jepson [31, 46] again minimize the same residual as Bruss and Horn, also making explicit the spherical imaging surface. Their methods are very efficient, however, because they pre-compute the basis of the subspace involved in linearly eliminating inverse scene depth and camera rotation. These "subspace methods" explicitly take advantage of the bilinear dependence on translation and inverse depth. In these they algebraically eliminate the inverse depth z and rotation Ω to derive a non-linear energy function of the translation τ.

Starting again with an equation for optical flow on the sphere,

$$u = \frac{dx}{dt} = -zP^{\perp}(x)\,\tau + \Omega \times x, \tag{1}$$

Heeger and Jepson eliminate the inverse depths z and the camera rotation Ω using the linear relationship between these and the optical flow given fixed camera translation,

$$\begin{array}{ccccc} \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} & = & \begin{bmatrix} P^{\perp}(x_1)\,\tau & & \mathbf{0} & -\hat{x}_1 \\ & \ddots & & \vdots \\ \mathbf{0} & & P^{\perp}(x_n)\,\tau & -\hat{x}_n \end{bmatrix} & \begin{bmatrix} z_1 \\ \vdots \\ z_n \\ \Omega \end{bmatrix} & . \\ & = & C(\tau) & & \\ 3n & & (3n)\times(n+3) & n+3 & \end{array}$$

Here, the columns of $C(\tau)$ span the subspace in which the optical flow is predicted to lie. The vector of inverse depths and camera rotation $[z_1 \cdots z_n\,\Omega]^{\mathsf{T}}$ specify the coordinates in this subspace, and the coordinates of the projection of the flow onto the subspace yields the least-squares optical flow prediction error. Given the optimal estimate of rotation and inverse depth, the distance to the subspace is again the measurement residual. Thus, the optimal estimate for the translation occurs at

$$\min_{\tau} \left\| C^{\perp}(\tau)\,u \right\|^2,$$

where $C^{\perp}(\tau)$, the orthogonal complement of the range (columns) of $C(\tau)$, which is the left nullspace, can be computed with the singular value decomposition. Assuming $C = U\Sigma V^{\mathsf{T}}$, the left nullspace of C is spanned by the columns of U with null singular values, those with index greater than the width of C. Formally this is $C^{\perp} = \left[U^{n+3+1} \cdots U^{3n}\right]$, where the superscript notation indicates a column of the matrix U.

A problem with this method is that it minimizes error under the assumption of a spherical imaging surface. Although with noiseless measurements this is the same as for a planar imaging surface, the transformation distorts the noise and introduces bias. In other

words, the perpendicular distances in the subspaces involved here do not minimize the least-squares measurement error for a planar imaging surface.

In related subsequent work, Jepson and Heeger [44] derive *linear* subspace equations using only the translation, although only using some of the measurement information. This allows direct recovery of the translation direction from linear equations, although this introduces a significant bias in the estimation of translation (which then also biases rotation and depth). The same authors then discuss the source of this bias in [45] and propose a method to remove it.

2.1.3.3 Accuracy and Bias

As we mentioned earlier, transformation of the measurements without a corresponding transformation of the noise model introduces bias into any estimation process. Tian, Tomasi, and Heeger [84] experimentally evaluated the accuracy and bias of the above methods. The Bruss and Horn algorithm, which solves a nonlinear problem (albeit after a transformation of the measurements by the "$ML_{\alpha\beta}$" norm), consistantly performs with the least noise sensitivity and bias. The linear epipolar constraint methods typically perform more poorly in this regard. Renormalization helps significantly, but still does not beat nonlinear optimization.

Zhang and Tomasi [100] investigate the convergence properties of the estimators in several of these algorithms. Algorithms derived using exact algebraic manipulations are unbiased under an infinite number of measurements. Nonetheless, these algorithms are biased under a finite number of measurements and nonzero noise. Experimentally, the authors also compare each algorithm to the method of nonlinear optimization of the basic optical flow equation under the ℓ_2 and $\ell_{1.2}$ (robust) norms.

2.1.4 Modern Visual Odometry

Geometric stereo visual odometry, as described by Matthies [60], relies on finding the camera motion that best explains the motion of image features between cameras and frames in a stereo head. Most current systems employ fast hypothesis testing in a RANSAC harness, followed by pose optimization using inlying feature matches, for example in [69, 68, 4]. Nistér *et al.* report errors of about 12 m over a 360 m course [69]. Others have used a ground plane assumption to compute monocular visual odometry, for example [93, 11]. Geometric methods, while extremely accurate, often assume perspective cameras for which lens distortion can be modeled. Additionally, they are computationally demanding, making them challenging to implement on low-power systems.

2.1.5 Planar-Parallax

Planar-parallax methods estimate scene structure and camera motion relative to a plane in the scene. A key benefit of planar-parallax over general structure from motion is that estimated structure model, because it is more constrained, is less sensitive to noise and semi-textureless image regions. Sawney [78] describes planar-parallax structure from motion for perspective projection as a unification of several past similar methods for orthographic

and weak perspective projection, as well as several application-specific methods. Irani and Anandan [41] extend projective planar-parallax methods to estimate structure and motion directly from the image brightness measurements instead of from point correspondences.

One very relevant domain for planar-parallax methods is in obstacle detection because they allow estimation of structure relative to the dominant ground plane. Carlsson and Eklundh [12] and Enkelmann [18] perform obstacle detection from optical flow by comparing the measured motion parallax to that predicted by a ground plane free of obstacles. To do this they use a calibrated camera and known camera height and attitude relative to the ground plane. Giachetti *et al*. [24] also labels cars using the consistency of observed optical flow with that predicted under a ground plane assumption.

2.2 Generalized Imaging Systems

We are interested in generalized imaging systems with near-arbitrary optics, including multiple viewpoint systems, catadioptric systems, and projective cameras with distortion. Grossberg and Nayar [26] formalize generalized imaging systems in terms of "ray pixels", or *raxels*, and show that a caustic ray surface yields a smooth mapping from pixel locations (u, v) on a 2-dimensional imaging surface to imaged rays.

Pless [71] and Neumann *et al.* [67] derive generalized optical flow and motion constraints for generalized imaging systems. A piece-wise smooth mapping from pixel locations to imaged rays allows optical flow to be well-defined, except at discontinuities in this mapping.

2.3 Optical Flow Estimation

Optical flow estimation in general suffers from the *aperture problem*. Because each neighborhood of pixels can have a different motion in the image, optical flow is typically computed independently on small windows throughout the image (e.g. [57]), but then cannot be evaluated in regions with ambiguous texture. Global smoothness constraints (e.g. [35]) propagate flow to these ambiguous regions, but make strong assumptions about the scene and the imaging system.

More recent work in computing optical flow has applied global constraints that arise from the optics of the imaging system and the structure of the scene. These usually assume a perspective camera, and model motion in the image as an affine or perspective transformation, for example [6, 43]. Some of these methods can estimate differing motion in separate regions of the image, for example [7, 40]. Irani exploits linear subspace constraints that hold over several frames [42]. Others use PCA to find the linear subspace automatically, e.g. [20], but still require optical flow to be computed over entire frames.

2.4 Semantic Image Labeling

Additionally, the coarse environment shape provides prior information for high-level vision tasks that detect and track obstacles and objects such as trees, cars, and pedestrians. This permits application and learning of top-down knowledge such as "pedestrians appear on the ground". This idea has been investigated heavily under scenarios like urban driving and

indoor scene understanding, with information from monocular cues, stereo, and laser point clouds [33, 9, 82].

Brostow *et al*. [9] segment images into relevant regions such as street, sidewalk, car, *etc*. using structure-from-motion cues. Sturgess *et al*. [82] estimate similar segmentations using motion appearance and structure-from-motion information.

2.5 Semantic 3D Scene Understanding

Alternative to these "bottom-up" approaches is the notion of obtaining 3D information for navigation aided by constraints from top-down models. Though not limited to robotics, Hoiem *et al*. [33] use monocular cues to estimate 3D structure. Raza et al. [72] extend semantic labeling to spatiotemporal segments, combining both appearance and motion cues to infer pixel-level semantic structure labels. Geiger *et al*. [23] and Zhang et al. [99] infer 3D street and traffic patterns from video from a moving platform, combining information from vehicle tracking, vanishing points, and image appearance. Recent work in "Manhattan World" environments produces high-quality estimates of large structures like walls and floors, see for example [21, 90]. Mozos *et al*. [65] apply learning to categorize hallways, doorways, and rooms from 2D laser range scans coupled with visual features. Nourani-Vatani *et al*. [70] match optical flow fields by their spatial statistics to a database of locations for topological mapping.

2.6 Navigation Using Optical Flow

Classic work in navigation from optical flow has used heuristics to control robots directly from measured optical flow. This work has its base in animal and human studies of the use of optical flow in navigation [25], [53]. Bio-inspired control laws have been developed for pursuit, escape, hallway following, and obstacle avoidance, for example Duchon *et al*. [17] and as reviewed in Srinivasan *et al*. [81].

More recent work in optical flow navigation has used more informed optical flow patterns beyond heuristics. Conroy *et al*. [13] and Hyslop and Humbert [39] infer similarity of the observed optical flow field to that expected for coarse environment structures such as walls and corridors from a set of template optical flow fields, using "wide-field optic flow integration". They also develop control laws to apply this to autonomous robot navigation.

2.7 Traversability

The main difficulty with applying 3D reconstruction towards the task of autonomous navigation is the further processing required to determine where a robot can and cannot drive. Current standard methods compute a 2D traversability map followed by path planning, using 3D laser range sensors, stereo correspondences, and structure from motion [52, 36]. Additionally, recent methods produce traversability maps using image appearance and learning [64, 49].

2.8 Optical Flow Matching Methods

There have been a number of works focused on matching optical flow fields or other features derived from image motion to classify or infer properties of a video. For example, Davis and Bobick [14] introduce temporal templates, propose motion history images and motion energy images as features for view-specific action recognition. These features encode the image motion over a short video sequence. A. A. Efros [1] introduce optical flow templates (a different concept as ours but with the same name), a spatiotemporal description of the optical flow associated with an action in a video, used to recognize actions of people.

CHAPTER III

GENERAL OPTICAL FLOW SUBSPACES FOR EGOMOTION ESTIMATION AND MOTION ANOMALY DETECTION

This chapter supports the claim that learned optical flow subspaces are effective for inferring robot motion and motion anomalies in cameras with arbitrary optics, in scenarios exhibiting depth regularity. To show this, we present a method for inferring ego-motion and motion anomalies from sparse flow measurements in a generalized imaging system, in the case where the imaging system moves through an environment with some degree of statistical regularity. This method uses learned optical flow subspaces, and we also present a method for learning these subspaces from recorded video data. The work in this chapter was initially presented in [76], and in earlier work where we used a k-nearest-neighbor learner to estimate motion from optical flow [75].

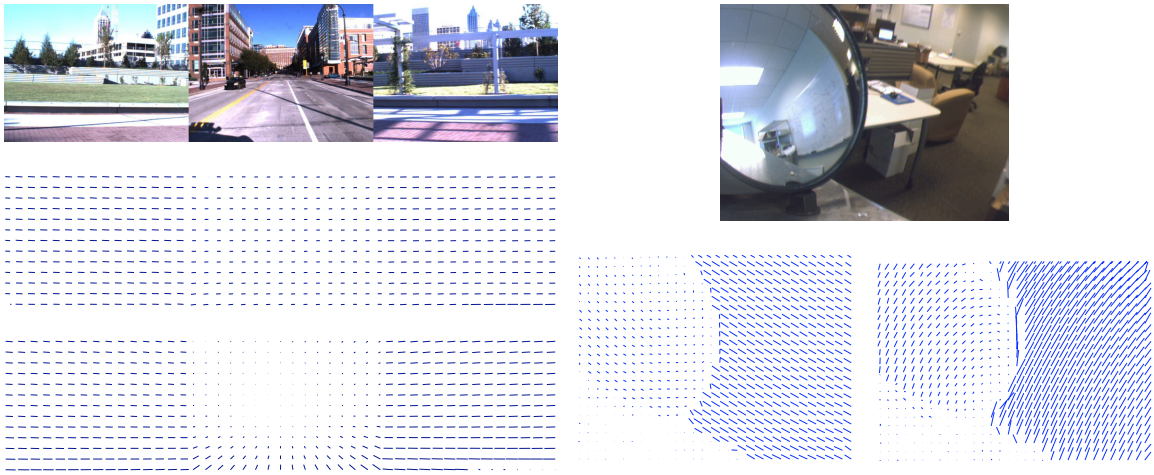
In contrast to work in visual odometry described in Section 2.1, we recover egomotion and dense optical flow directly from sparse optical flow using *learned* robust subspace constraints that hold over extended motion of a generalized imaging system, while simultaneously identifying outliers in the sparse flow. The subspace constraints hold when scene depth in the majority of the image is nearly constant over time, as is typically the case for planar mobile robots in outdoor and urban environments. In these environments, the ground plane, and even buildings and walls, are often at constant depth relative to the robot. Section 2.1.3 describes related work in leveraging subspace constraints in optical flow.

One motivation for this work is in obtaining approximate visual odometry, in a generalized imaging system, as a replacement for more costly or limited methods of obtaining incremental platform motion. Laser scanners, or LIDAR, with which one can perform laser scan matching, tend to be heavy, expensive, and require much power. Wheel odometry, while very accurate indoors, is unreliable due to wheel slippage in outdoor environments, especially on small lightweight robots. Inertial Measurement Units (IMU) accumulate velocity errors, and therefore must be fused with absolute position or velocity measurements.

Another motivation and contribution of this work is in detecting *motion anomalies*. These indicate independently-moving objects in the camera's view, or unexpectedly nearby or faraway scene structure. Motion anomalies are image regions that are not explained by the optical flow subspace model and thus are *salient* regions in need of further visual processing.

To work with generalized imaging systems with nearly-arbitrary optics, we *learn* robust subspace constraints from data. The first learning step is in an unsupervised manner without knowledge of camera motion, in which case motion anomalies may be detected, but egomotion may not be estimated at runtime. A second optional step is a supervised learning step with known camera egomotion that additionally permits online egomotion estimation.

Figure 1a shows the subspace defined by basis flows spanning the 2-dimensional linear optical flow subspace for the 3-camera outdoor driving sequence. The vehicle had 2



(a) 3-camera outdoor driving sequence

(b) "Ad-hoc" catadioptric system

Figure 1: Typical frames and basis flows for the 3-camera outdoor driving sequence (the frame consists of 3 tiled images) and the ad-hoc catadioptric system. The basis flows in the ad-hoc catadioptric system show the irregular optics of the imaging system. Optical flow is coherent both in the mirror and in the view beyond the mirror, but is usually zero or erroneous in the lower-left part of the image where the camera sees only its mounting plate.

degrees-of-freedom, arising from its steering rate and speed. The first basis flow corresponds approximately to yawing, while the second corresponds approximately to driving forwards. In the second basis flow, the flow vectors at the bottom of the image are larger than those at the top because they correspond to points on the ground plane closer to the camera, thus moving more in the image when the vehicle translates. On the other hand, optical flow magnitude is invariant to depth in rotations, yielding the approximately equal-length vectors in the first basis flow.

Figure 1b shows the basis flows for a catadioptric system in which part of the camera's view contains a curved mirror. The lower-left corner of the frame images the plate to which the camera and mirror are rigidly attached. The basis flows show the stark boundaries between these three image regions with differing optics.

3.1 Optical Flow Subspaces in Perspective Cameras

In this section, we first introduce the notion of optical flow subspaces in standard perspective cameras, and then draw a conclusion about the three motion/scene categories for which constant optical flow subspaces exist over extended trajectories. In the following section we extend this to generalized imaging systems.

An optical flow subspace is a linear mapping from low-dimensional set of latent variables $y \in \mathbb{R}^q$ to predicted optical flow

$$u_i = W_i y \tag{2}$$

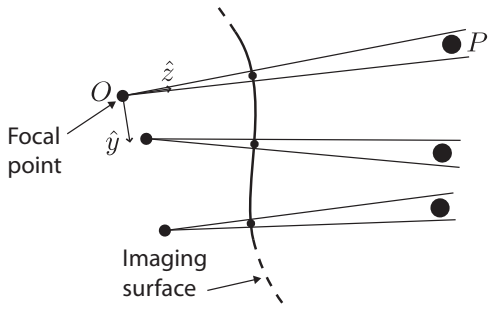


Figure 2: Schematic illustration of a generalized imaging system as a collection of local cameras. Each local camera images a narrow cone of rays on the imaging surface, contributing one pixel to the imaging system. Each local camera is defined by its focal length and its pose with respect to the imaging system.

where $W_i \in \mathbb{R}^{2\times q}$ is the linear mapping to flow corresponding to the i^{th} image location. The columns of $W \in \mathbb{R}^{2wh\times q}$ are *basis flows*, illustrated in Figure 1.

To demonstrate when and why there exists a subspace of optical flow, we now make explicit the linear relationship between camera velocity and optical flow when scene depth at each image location remains constant over time. The optical flow u_i at the i^{th} image location is related to camera velocity $v = [\,\omega_x\,\omega_y\,\omega_z\,v_x\,v_y\,v_z\,]^{\mathsf{T}}$, assuming no noise, according to

$$u_i = V_i(z_i)\,v \tag{3}$$

where $V_i(z_i)$ is an optical flow matrix, which depends on the camera optics and which depends nonlinearly on the scene depth z_i at the i^{th} image location. For a standard perspective camera, the flow matrix is (for example, see [31])

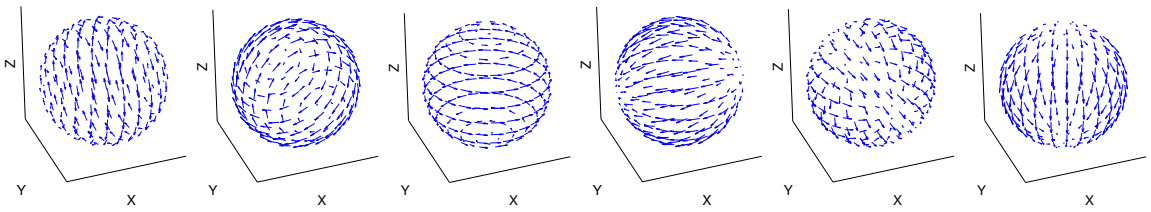
$$V_i(z_i) \triangleq \begin{bmatrix} \frac{x_i y_i}{f} & \frac{-f-x_i^2}{f} & y_i & \frac{-f}{z_i} & 0 & \frac{x_i}{z_i} \\ \frac{f+y_i^2}{f} & \frac{-x_i y_i}{f} & -x_i & 0 & \frac{-f}{z_i} & \frac{y_i}{z_i} \end{bmatrix}, \tag{4}$$

where (x_i, y_i) is the image location at the i^{th} pixel. When the focal length f and the scene depth at each pixel z_i remain constant over time, the flow matrices V_i for each pixel are also constant, and thus V also defines a special linear optical flow template where the velocity components are the latent variables.

Optical flow subspaces due to camera egomotion exist in three primary motion/scene categories. Given that optical flow is linear in rotational velocity, the first category is pure camera rotations, in which there is an optical flow subspace regardless of the scene geometry. Given that optical flow is linear in translational velocity when scene depth remains constant at each pixel, the second category is a scene consisting only of a ground plane with only planar motion on that ground plane. Using the same principle, the third category is linear motion along a constant-cross-section tube or canyon.

3.2 Optical Flow Subspaces in Generalized Imaging Systems

To extend the applicability of optical flow subspaces to generalized imaging systems, we now show that in fact this linearity holds for more general cameras of nearly arbitrary optics



Rotation, $+X$	Rotation, $+Y$	Rotation, $+Z$	Translation, $+X$	Translation, $+Y$	Translation, $+Z$
$v = [100000]^\mathsf{T}$	$v = [010000]^\mathsf{T}$	$v = [001000]^\mathsf{T}$	$v = [000100]^\mathsf{T}$	$v = [000010]^\mathsf{T}$	$v = [000001]^\mathsf{T}$

Figure 3: For illustrative purposes, six basis flows corresponding to rotational and translational camera motion in the canonical camera axes for a spherical imaging surface. These basis flows form the columns of the mapping V of velocity to flow.

for which a parametric calibration is not possible, including distortion, catadioptrics, and multiple viewpoints. Grossberg and Nayar [26] formalize generalized imaging systems in terms of "ray pixels", or *raxels*, and show that a caustic ray surface yields a smooth mapping from 2D image surface locations to imaged rays.

Generalized imaging systems may also be modeled as collections of local perspective cameras, each contributing one pixel to the system, as shown in Figure 2. Each local camera images a single ray, and is defined by its focal length and its pose with respect to the overall imaging system.

To show that the linearity of optical flow with depth regularity holds for generalized imaging systems, we first consider the mapping from velocity to optical flow in a single local camera, which is equivalent to Eq. 4 with the image location $(x_i, y_i) = \mathbf{0}$,

$$V_i(z_i) = \begin{bmatrix} 0 & -1 & 0 & \frac{-f}{z_i} & 0 & 0 \\ 1 & 0 & 0 & 0 & \frac{-f}{z_i} & 0 \end{bmatrix}. \tag{5}$$

Assuming that the optics of the camera do not change, there is a fixed change of frame from the "base frame" of the imaging system to the coordinate frame of the local camera, which is aligned with its local center of projection. Let the constant 6×6 linear mapping F_i change the coordinate frame from the imaging system velocity to the local camera velocity, then the optical flow for every image location in the generalized imaging system is

$$u_i = V_i(z_i) F_i v, \tag{6}$$

which is again linear in the imaging system or platform velocity under the assumption of depth at each pixel constant over time.

To illustrate this linearity, Figure 3 shows the first three columns of a velocity-mapping flow matrix for a spherical imaging surface. In these cases flow matrices W in latent variables, or flow matrices V in platform velocity, may be learned from recorded video using unsupervised and supervised methods, respectively, as we will show.

3.3 Robust Probabilistic Subspace Model

To improve results on real-world data, instead of a deterministic relationship, a probabilistic optical flow subspace defines a probability density on optical flow that is robust to outliers,

$$p\left(u_i \,|\, y, \lambda_i\right) \propto \begin{cases} \mathcal{N}\left(W_i y, \Sigma_u^{\mathrm{v}}\right), & \lambda_i = 1 \\ \mathcal{N}\left(W_i y, \Sigma_u^{\mathrm{f}}\right), & \lambda_i = 0 \end{cases} \tag{7}$$

where $\Sigma_u^{\mathrm{v}} \in \mathbb{R}^{2\times 2}$ is the (small) covariance of an optical flow vector that is an inlier to the subspace, Σ_u^{f} is the (large) covariance of an outlier to the subspace, and $\lambda_i \in \{1,0\}$ indicates a pixel is an inlier or an outlier, respectively. Here we will derive an EM algorithm to bound this likelihood using estimated inlier probabilities, but inlier assignments could also be calculated using other methods, such as RANSAC. Thus an optical flow subspace is $\left(W, \Sigma_u^{\mathrm{v}}, \Sigma_u^{\mathrm{f}}, p\left(\lambda\right)\right)$, where $p\left(\lambda\right)$ is a constant Bernoulli prior probability that any pixel is an inlier to the subspace.

3.4 Estimating Egomotion

Suppose we have now already found some q-dimensional linear subspace (q is near 4 in our experiments) in the optical flow $u_t \in \mathbb{R}^d$ (d typically on the order of 3000) for every t^{th} frame using the method in Section 3.6. To estimate the latent variables for new frames online, we iterate only the E-step, Eqs. 14 and 15 above.

We now turn to estimating the egomotion online, given that we can calculate the latent variables for new frames online. This requires an additional learning step using known platform velocity for some training data. We know from Eq. 2 that there is a linear mapping W from the latent variables y to the flow u, but we also know there there is some other linear mapping V from the true platform velocity v to the optical flow. Thus, there is also a mapping $M \in \mathbb{R}^{6\times q}$ from latent variables to platform velocity, which we learn via iteratively-reweighted least-squares, optimizing

$$M = \arg\min_{\mathring{M}} \sum_t \left\| \mathring{M} y_t - v_t \right\|^2 \tag{8}$$

over the training set with known velocity v_t at each time step. After learning this mapping M from training data, we can estimate the egomotion for new frames by evaluating $v = My$ with the latent variables y estimated for that frame.

We conducted an experiment using a differential-drive robot with two forward facing cameras, which instead of forming a stereo pair, face outwards at angles of approximately 20°. The robot was equipped with a well-tuned pose filter that fused wheel odometry with an inertial measurement unit (IMU). Figure 4 shows the trajectories estimated by this filters, as well as by our method, as described above in this section. We trained our method using pose estimates from the pose filter on the short training sequence shown in the figure, and then switched to estimating pose with our method with no further training. The pose estimated from optical flow is comparable in accuracy to the filtered pose from wheel odometry and IMU.

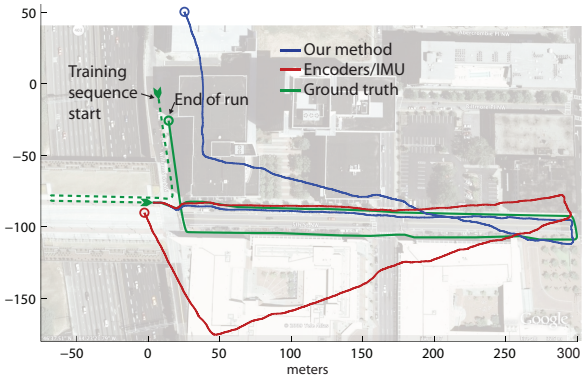


Figure 4: Platform trajectories estimated by our method (blue) and by integrating wheel odometry with an inertial measurement unit (red). We trained our method from ground truth over a 200 m training segment (dashed green), then switched to estimating pose for approximately 675 m with no further training. Each trajectory ends at the circle of the corresponding color.

Figure 5: Detection of outliers in the sparse optical flow in a frame from the 3-camera outdoor driving sequence. Colored lines are sparse optical flow, ranging from green when $p(\text{inlier}) = 1$, to red when $p(\text{inlier}) = 0$. Sparse flow vectors that are inconsistent with the linear flow subspace have low inlier probability. Vectors on the moving pickup truck, in the textureless regions of the road, and on the very close and very far structure of the wall are labelled as outliers.

An informal timing evaluation revealed that our prototype code, with sparse flow extraction in C++ and dense flow and ego-motion estimation in MATLAB, runs faster than 30 Hz (or 33.3 ms per frame) after training. On 1920×480 frames, with 45×13 flow fields, computing sparse flow takes 19.1 ms, subspace coordinates 6.8 ms, and ego-motion 0.2 ms, for a total of 26.1 ms per frame.

3.5 Detecting Motion Anomalies

Figure 5 shows the inliers and motion anomalies (outliers) that are present in a typical frame. Green vectors are inliers, while red vectors are outliers. In the left-facing camera, the vectors on the moving pickup truck are labelled as outliers. In the forward-facing camera, the erroneous vectors on the textureless portion of the road are mostly ignored. In the right-facing camera, the structure of the wall that is very close and very far from the camera are identified as motion anomalies, while the vectors on the structure near the average distance are labeled as inliers. The colors, ranging from green to red, indicate the probability of a flow vector being an inlier.

3.6 Learning Optical Flow Subspaces from Data

To learn the probabilistic optical flow subspaces, we develop a robust extension of probabilistic PCA [85] to find the principal subspace encoding global correlations in optical flow. This robustness is not limited to the traditional case where entire samples (i.e. frames) are discarded as outliers. Instead, we can find dimensions (i.e. individual flow vectors) in each sample whose values are not consistent with the other dimensions. The "intra-sample" robust PCA method developed by De la Torre and Black [15] accomplishes a similar goal by minimizing an energy function that weights each pixel with a per-image-pixel analog outlier process. Here we instead develop a generative model for intra-sample robust PPCA, which models each pixel as a Gaussian mixture of inlier and outlier densities.

3.6.1 Training Data

Given a training video sequence, we compute a down-sampled sparse optical flow field over each pair of frames. We divide each image into a grid of 20×20 pixel cells, then track the strongest Harris corner [29] in each cell using the Lucas-Kanade algorithm [56]. A threshold on corner response that prevents tracking textureless regions makes the flow fields sparse.

Each pair of frames at time t yields an observation vector u_t, filled with the concatenated horizontal and vertical optical flow components from each local camera. For flow fields of size $w \times h$, for example, the length of each observation vector is $d = 2wh$. For each frame there is also a set S_t of the "seen" indices of u_t, where optical flow is available. Indices not in S_t are missing, and we save computation by ignoring them in calculations, as described below.

3.6.2 Maximum Likelihood Formulation for Learning

To learn a linear optical flow subspace, we apply a robust variation on probabilistic principal components analysis [85]. Pertaining to this, the key observations we make about the generative model for optical flow described in Section 3.3 are:

1. The model is similar to PPCA: if all flow vectors are measured and are inliers, the basis flows span the principal subspace, as shown in [85].

2. When conditioned on the parameters and the latent variables, each observation vector is drawn from a mixture model of two *spherical* normal distributions.

3. Although the flow vectors of each frame are correlated through the basis flows and latent variables, all flow components are mutually *independent* when conditioned on the basis flows and latent variables. Thus, we can solve for the basis flows and latent variables while simply ignoring missing data; estimating the missing data is not necessary.

The conditional distribution over a single component of a single flow vector is the mixture model

$$u_{ti} \,|\, y_t, \lambda_{ti}, \theta \sim \mathcal{N}\left(u_{ti}; \bar{u}_{ti}, \sigma_{\mathrm{v}}^2\right)^{\lambda_{ti}} \mathcal{N}\left(u_{ti}; \bar{u}_{ti}, \sigma_{\mathrm{f}}^2\right)^{1-\lambda_{ti}}, \tag{9}$$

where $\bar{u}_{ti} = \mathrm{W}_i y_y$ and $\theta = \left\{\mathrm{W}, \sigma_\mathrm{v}^2, \sigma_\mathrm{f}^2, p(\lambda)\right\}$. Every flow vector in fact has a different density, but they are related through the latent variables y. As with PPCA, there is a Gaussian prior on the latent variables y:

$$y \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}_{q\times q}\right). \tag{10}$$

We are now ready to formulate the maximum likelihood problem of inferring the parameters and hidden variables when conditioned on the observations by maximizing

$$p(\theta, y, \lambda \,|\, u) \propto p(\theta) \prod_t p(x_t) \prod_{i\in S_t} \mathscr{L}\left(u_{ti} \,|\, y_t, \lambda_{ti}, \theta\right) p(\lambda_{ti}), \tag{11}$$

where $\mathscr{L}(\cdot) = \log cp(\cdot)$, denotes the log-likelihood, i.e. c is any constant. Note the product over $i \in S_t$, by which we only consider vectors in the training data that were "seen", and ignore those that were missing.

3.6.3 EM algorithm

The exact maximum-likelihood solution is intractable because of the combinatorial assignments of the indicator variables λ, so we instead apply EM. The complete log-likelihood of (11), using the distributions above is

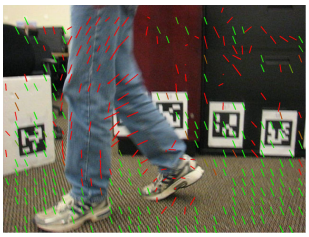
$$\begin{aligned}\mathscr{L}(\theta, y, \lambda \,|\, u) = \sum_t \log \mathcal{N}(y_t; \mathbf{0}, \mathbf{I}) + \sum_{i\in S_t} \lambda_{ti} \left(\mathscr{L}(\lambda_{ti}) + \log \mathcal{N}\left(u_{ti}; \bar{u}_{ti}, \sigma_\mathrm{v}^2\right)\right) \\ + (1-\lambda_{ti}) \left(\mathscr{L}(\lambda_{ti}) + \log \mathcal{N}\left(u_{ti}; \bar{u}_{ti}, \sigma_\mathrm{f}^2\right)\right).\end{aligned} \tag{12}$$

Finding the expectation of this log-likelihood with respect to y and λ simultaneously would be intractable, but we can instead first find the expectation with respect to y, and then with respect to λ, comprising a generalized EM algorithm. We obtain update equations by taking the derivatives of (12) w.r.t. the parameters, and then solving for each parameter:

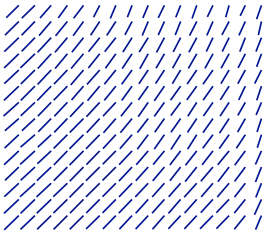
$$\begin{aligned}\sigma_\mathrm{v}^2 &= \left(\sum_{t, i\in S_t} \langle \lambda_{ti} \rangle\right)^{-1} \\ &\quad \sum_{t, i\in S_t} \langle \lambda_{ti} \rangle \left(u_{ti}^2 - 2u_{ti} \mathrm{W}_i \langle y_t \rangle + \mathrm{tr}\left(\left\langle y_t y_t^\mathsf{T} \right\rangle \mathrm{W}_i^\mathsf{T} \mathrm{W}_i\right)\right) \\ \mathrm{W}_i &= \sum_{t\in S_i} \langle \lambda_{ti} \rangle \, \sigma_\mathrm{v}^{-2} \langle y_t \rangle^\mathsf{T} u_{ti} \left(\sum_{t\in S_i} \langle \lambda_{ti} \rangle \, \sigma_\mathrm{v}^{-2} \left\langle y_t y_t^\mathsf{T} \right\rangle\right)^{-1},\end{aligned} \tag{13}$$

where S_t is as above, while S_i is the set of *frame* indices whose i^{th} flow component was not missing. The outlier variance σ_f^2 is estimated in the same way but substituting $\langle 1 - \lambda_{ti} \rangle$ for $\langle \lambda_{ti} \rangle$.

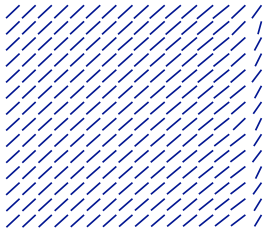
We find the expectation $\langle y_t \rangle$ with respect to its posterior $p(y \,|\, u, \lambda, \theta) \propto p(u \,|\, y, \lambda, \theta)\, p(y)$ using Bayes' law and the distributions (9) and (10). Combining the conditional distributions over each u_{ti} into a multivariate distribution over u_t with diagonal covariance Λ_t^{-1}



(a) Typical frame (640×480) showing outlying sparse flow vectors (in red) found by our method.



(b) One of the two basis flows found by standard PPCA.



(c) The same basis flow found by our method.

Figure 6: a) A pan-tilt camera, viewing a person walking while the camera moves. b) In a basis flow found by standard PPCA, the vectors are not parallel due to outliers (in red), as they should nearly be for this dataset. c) In the same basis flow found by our robust method, the vectors are properly much closer to parallel.

simplifies finding the distribution over y_t. Following, $p(y \mid u, \lambda, \theta)$ has mean and covariance

$$\begin{aligned} \langle y_t \rangle &= \left(\mathrm{W}_{St}^{\mathsf{T}} \Lambda_t \mathrm{W}_{St} + \mathbf{I}_{q \times q} \right)^{-1} \mathrm{W}_{St}^{\mathsf{T}} \Lambda_t y_t, \\ \left\langle y_t y_t^{\mathsf{T}} \right\rangle &= \left(\mathrm{W}_{S_t}^{\mathsf{T}} \Lambda_t \mathrm{W}_{S_t} + \mathbf{I}_{q \times q} \right)^{-1} + \langle y_t \rangle \langle y_t \rangle^{\mathsf{T}}, \\ &\text{where } \Lambda_t = \sigma_{\mathrm{v}}^{-2} \mathbf{I}_{q \times q} \lambda_t. \end{aligned} \tag{14}$$

The expectation $\langle \lambda_{ti} \rangle$ is easier, as it is the simply the proportion of probability of the flow component belonging to the inlier distribution:

$$\langle \lambda_{ti} \rangle = \frac{\mathcal{N}\left(u_{ti}; \hat{u}_{ti}, \sigma_{\mathrm{v}}^2\right)}{\mathcal{N}\left(u_{ti}; \hat{u}_{ti}, \sigma_{\mathrm{v}}^2\right) + \mathcal{N}\left(u_{ti}; \hat{u}_{ti}, \sigma_{\mathrm{f}}^2\right)}, \tag{15}$$

where $\hat{u}_{ti} = \mathrm{W} \left(\mathrm{W}^{\mathsf{T}} \mathrm{W} \right)^{-1} \left(\mathrm{W}^{\mathsf{T}} \mathrm{W} + \sigma_{\mathrm{v}}^2 \mathbf{I}_{q \times q} \right) \langle y_t \rangle$.

3.6.4 Visualizing the Benefits of Robustness

Robustness to outliers improves the quality of the basis flows. Figure 6 illustrates outliers in a sequence from a pan-tilt camera with a person walking in the view. 6a shows sparse flow measurements classified as inliers and outliers. 6b and 6c show, respectively, the second bases found by non-robust PPCA, and by our method, which probabilistically ignores outliers when estimating bases. Because this is a narrow-FOV camera in pure rotation, the vectors in the basis flows should be nearly parallel.

3.6.5 Learned Model Quality and Subspace Dimensionality

Measuring how well the learned subspace model fits the data is important for using optical flow subspaces because it indicates the true subspace dimensionality and whether the optical flow indeed can be modeled with a subspace. The inlier distribution variance σ_v^2 is

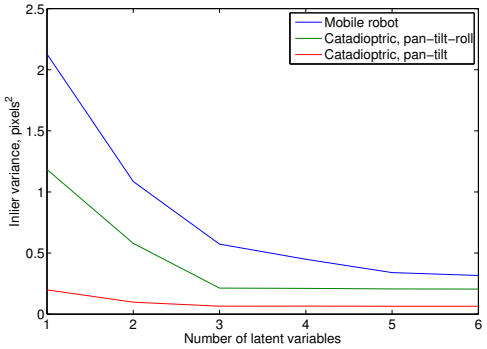


Figure 7: Estimated PPCA inlier distribution variance for models with various numbers of latent variables. Lower variance indicates a better subspace fit to the optical flow. The pan-tilt and pan-tilt-roll catadioptric systems are 2 and 3 DOF, respectively. The mobile robot has 2 controllable DOF's, but additionally pitches and rolls due to sloped ground.

a measure of how well the robust PPCA model fits the training data. Specifically, it is the mean squared error between the predicted and observed optical flow in the training data, weighted by the inlier probability for each flow component. The number of degrees-of-freedom (DOF) of the motion of an imaging system is reflected in the relationship between the number of latent variables (i.e. principal components) in a robust PPCA model and the inlier variance, as shown in Figure 7. The additional decrease in variance in models with more than 4 latent variables in this case suggests to us that there are additional unaccounted but predictable aspects of the optical flow or the depth of the environment that additional latent variables can explain.

CHAPTER IV

RECOGNIZING THE PATH SHAPE AHEAD OF A ROBOT FROM SPATIO-TEMPORAL IMAGE GRADIENTS

In this chapter we use instantaneous spatio-temporal image gradients to recognize path shapes, for example those shown in Figure 8, in front of a mobile robot. We perform this recognition using Bayesian model selection among several pre-learned linear optical flow templates, using the observed image gradients as measurements at each time step. Offline, we learn these linear optical flow templates directly from spatio-temporal image gradients, from video segments labeled with the class of the observed path shape.

Path shape classification has the potential to provide important information for modern high-speed mobile robot controllers that use "motion primitives" [22, 79]. Motion primitives are discrete control laws applied under a specific set of environmental constraints and desired trajectory. A high-level controller then switches between them depending upon conditions. For example, the sliding turn maneuvers common in rally racing provide increased robustness to unknown friction characteristics during sharp turns, while non-sliding controllers are best for navigating nearly-straight path segments. At each time step, the correct controller can be selected depending on the path shape just ahead of the robot.

Additionally, the coarse environment shape provides prior information for high-level vision tasks that detect and track obstacles and objects such as trees, cars, and pedestrians. This permits application and learning of top-down knowledge such as "pedestrians appear on the ground". This idea has been investigated heavily under scenarios like urban driving and indoor scene understanding, with information from monocular cues, stereo, and laser point clouds [33, 9, 82].

Each path shape is associated with one linear optical flow template. A linear optical flow template encodes a probabilistic subspace of optical flow fields that may be observed while the camera observes a particular path shape. The choice of a subspace model makes the template agnostic to the velocity at which the robot moves, and thus only sensitive to the set of depths at each pixel as well as the camera optics and orientation.

A key feature of the subspace model is that the camera needs not be calibrated, nor do its optics need to permit a parametric calibration. The fisheye lens used in our experiments, for example, cannot be calibrated with a radial distortion model. Additionally, the velocity of the robot need not be known or calculated either for template learning or for online model selection.

4.1 Recognizing Path Shapes

The overall goal is to quickly recognize the path shape in the camera's view among several optical flow templates, each of which we have learned for a particular path shape. We rely on optical flow because it is sensitive to the scene structure without being sensitive to appearance such as the locations of edges or textures. Let $k_t \in \{1..K\}$ indicate the path

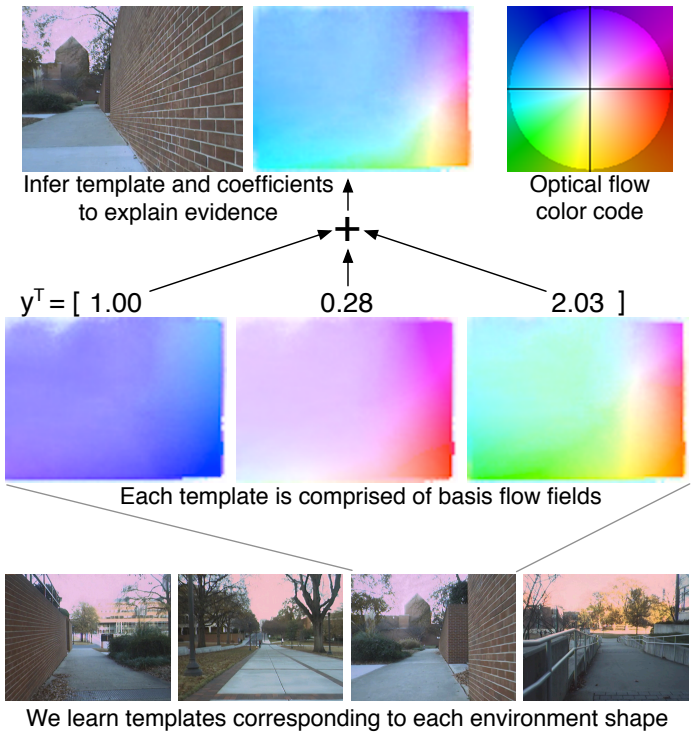


Figure 8: Bottom: We classify large-scale environment shape types such as 'left of path', 'center of path', 'right of path' with approximate model selection over a set of *linear optical flow templates*. **Middle**: A single linear optical flow template comprises a set of *basis flows* that span the subspace of possible optical flow fields resulting from egomotion in the template's environment shape. **Top:** In the illustrated video frame, the image motion is explained by the particular linear combination specified by the latent variable assignment $y = [\,1.00\;0.28\;2.03\,]^{\mathsf{T}}$, which combines forward motion with some camera rotation caused by uneven ground and turning of the platform. Because we learn the templates with an unsupervised method, the basis flows do not correspond to canonical motions such as pure forward motion or pure pitch, and are instead combinations of such motions. The top-right shows the optical flow color code used in the remainder of this thesis. Vector direction is indicated by color hue and vector magnitude is indicated by color saturation.

shape at time step t, and our goal is to find the maximum-likelihood path shape conditioned on the spatial and temporal derivatives D_t of the current image, and on the path shape classification from the previous time step k_{t-1} (to leverage temporal smoothness)

$$\hat{k}_t = \arg\max_{k_t} \sum_{k_{t-1}} p\left(k_t \,|\, D_t, k_{t-1}\right) p\left(k_{t-1}\right). \tag{16}$$

We use the spatial and temporal derivatives as measurements instead of the optical flow because computing optical flow directly is an under-determined and inherently noisy problem. We instead implicitly compute optical flow in the framework of the much more well-determined optical flow template classification problem. To do this, we first apply Bayes' law to obtain a measurement likelihood of the image derivatives given the template,

$$p\left(k_t \,|\, D_t, k_{t-1}\right) \propto p\left(D_t \,|\, k_t\right) p\left(k_t \,|\, k_{t-1}\right). \tag{17}$$

We list our choice of transition model $p\left(k_t \,|\, k_{t-1}\right)$ in Section 4.3.

We factor (16) into the likelihood of the template given the optical flow, and the optical flow given the image derivatives, and integrate out the optical flow field u_t for the t^{th} frame (this integral will be analytical and fast to compute),

$$p(D_t \,|\, k_t) = \int_{u_t} p(D_t \,|\, u_t)\, p(u_t \,|\, k_t)\,. \tag{18}$$

Next, we define models for the image derivatives given the optical flow and the optical flow likelihood given the template. Subsequently, we derive an efficient closed-form expression for evaluating the original template likelihood in (17).

4.1.1 The Constant-Brightness Image Derivative Model

To relate the spatial and temporal derivatives D^x and D^t to the optical flow, we use the classic constant-brightness assumption [34], which states that as an image point corresponding to some point in the physical world moves through the image, it may change location but not brightness. Making the brightness constancy constraint probabilistic with Gaussian noise with variance $\Sigma_I \in \mathbb{R}$, yields a Gaussian conditional density of the temporal derivative D^t_{ti} at time step t at the i^{th} pixel given the optical flow u_{ti} at the i^{th} pixel,

$$p(D_t \,|\, u_t) = \prod_i (2\pi\Sigma_I)^{\frac{-1}{2}} \exp \frac{-1}{2} \left\| D^t_{ti} + D^x_{ti} u_{ti} \right\|^2 \Sigma_I^{-1}. \tag{19}$$

We optimize the covariance parameter Σ_I along with the learned templates as described in Section 4.2.

4.1.2 The Optical Flow Template Model

We choose to model the likelihood of an optical flow field in each template as a probabilistic subspace. In other words, our model assumes that any optical flow field observed while the robot drives on a particular path shape lies close to a subspace, with each path shape exhibiting a different subspace. Just as any subspace is spanned by a set of basis vectors, with an optical flow subspace these basis vectors take the form of "basis flows". An optical flow field u is generated as a linear combination with coefficients $y_k \in \mathbb{R}^q$ of q basis flows in the columns of $W_k \in \mathbb{R}^{2wh \times q}$ plus a small amount of uncorrelated noise with *diagonal* covariance $\Sigma_u \in \mathbb{R}^{2\times 2}$,

$$p(u_t \,|\, y_{tk}) = \prod_i \mathcal{N}(W_{ki} y_{tk}, \Sigma_u)\,, \tag{20}$$

where $W_{ki} \in \mathbb{R}^{2\times q}$ refers to the 2 rows of W_k corresponding to the i^{th} pixel. It is important to note that the dimensionality of the vector space of optical flow fields is $2wh$, twice the number of pixels in the image. A "vector" in the space of optical flow fields is the concatenation of all of the individual optical flow vectors in the frame,

$$u = \begin{bmatrix} u^x_{11} & u^y_{11} & u^x_{21} & \cdots & u^x_{hw} & u^y_{hw} \end{bmatrix}^{\mathsf{T}} \tag{21}$$

Our motivation for choosing a subspace model is that given a fixed set of scene depths at each pixel, the optical flow fields generated by any translational or rotational camera

velocity lie in a subspace. Thus, our optical flow templates respond to a particular set of scene depths at each pixel while remaining agnostic to the camera velocity.

From (20) we obtain an expression for the flow likelihood $p(u_t \,|\, k_t)$ in (18) by integrating out the unknown latent coefficients y_{kt} (we now add a t subscript to denote that these variables are for a single time step as introduced previously)

$$p(u_t \,|\, k_t) = \int_{y_{tk}} \prod_i p(u_{ti} \,|\, y_{tk})\, p(y_{tk}), \tag{22}$$

where the likelihood $p(u_t \,|\, y_{tk})$ of the optical flow u_t given the basis flow coefficients y_{tk} is Gaussian from (20). The integral will again be analytic, as we show later. We optimize the covariance parameter Σ_u along with the learned templates as described in Section 4.2.

4.1.3 Efficiently Evaluating Template Likelihood

We calculate the template likelihood in (17) given the image derivatives efficiently by means of analytic integrals and problem decomposition. The first key is that the optical flow likelihood is independent in each pixel u_{ti} when conditioned on the coefficients y_t. This leads to a per-pixel form of the template likelihood in (18)

$$p(D_t \,|\, k_t) \propto \int_{y_{tk}} \prod_i \int_{u_{ti}} p(D_{ti} \,|\, u_{ti})\, p(u_{ti} \,|\, y_{tk})\, p(y_{kt})\, p(k_t \,|\, k_{t-1}) \tag{23}$$

We compute the integral over the unknown optical flow u_{ti} analytically leveraging the fact that the image derivative likelihood $p(D_{ti} \,|\, u_{ti})$ from (19) and the flow likelihood $p(u_{ti} \,|\, y_{tk})$ from (20) are both Gaussian, to yield a likelihood of the image derivatives given the basis flow coefficients,

$$p(D_{ti} \,|\, y_{tk}) \;=\; \int_{u_{ti}} p(D_{ti} \,|\, u_{ti})\, p(u_{ti} \,|\, y_{tk}) \;=\; (2\pi)^{\frac{-1}{2}} J_i \exp \frac{-1}{2} J_i^2 \left\| D_{ti}^t + D_{ti}^x W_{ki} y_{tk} \right\|^2, \tag{24}$$

$$\text{where } J_i^2 = \left(D_{ti}^x \Sigma_u D_{ti}^{x\mathsf{T}} + \Sigma_I \right)^{-1}$$

Substitution of this more compact image derivative likelihood $p(D_{ti} \,|\, y_{tk})$ into (23) yields

$$p(k_t \,|\, D_t, k) \propto \int_{y_{tk}} \prod_i p(D_{ti} \,|\, y_{tk})\, p(y_{kt})\, p(k_t \,|\, k_{t-1}). \tag{25}$$

Next we integrate out the basis flow coefficients y_{tk} in closed form to produce a new likelihood $p(D_t \,|\, k_t)$ of the image derivatives D_t for *every pixel* given the template k_t,

$$p(D_t \,|\, k_t) = \int_{y_{kt}} \prod_i p(D_{ti} \,|\, y_{tk})\, p(y_{tk}) \tag{26}$$

The key to efficiently evaluating this integral is that although the resulting Gaussian over all of the image derivatives is of high dimensionality, wh, its information matrix takes a

much more compact special form of the sum of a diagonal and an outer product of small matrices. To arrive at its compact factored form, we first write the information form of the joint density on the image derivatives and basis flow coefficients, without yet integrating,

$$\begin{aligned} p(D_t, y_{tk}) &= \prod_i p(D_{ti} \,|\, y_{tk})\, p(y_{tk}) \\ &= \left|\frac{\Lambda}{2\pi}\right|^{\frac{1}{2}} \log \frac{-1}{2} \left[D_t^t y_{tk}\right] \left(\Lambda \left[D_t^t y_{tk}\right]^{\mathsf{T}} + \eta\right) \end{aligned} \tag{27}$$

From the image derivative likelihood (24) and assuming a normal Gaussian prior $p(y_{kt}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ on the latent variable coefficients, we write the blocks of the joint information matrix Λ and information vector η,

$$\Lambda = \begin{bmatrix} \Lambda^{II} & \Lambda^{Iy} \\ \Lambda^{Iy\mathsf{T}} & \Lambda^{yy} \end{bmatrix}, \quad \eta = \begin{bmatrix} \eta^I \\ \eta^y \end{bmatrix} \tag{28}$$

where

- each diagonal entry of the diagonal matrix $\Lambda^{II} \in \mathbb{R}^{wh \times wh}$ is $\Lambda^{II}_{ii} = J_i^2$,
- each row of $\Lambda^{Iy} \in \mathbb{R}^{wh \times q}$ is $\Lambda^{Iy}_i = J_i^2 D^x_{ti} W_i$,
- the matrix $\Lambda^{yy} \in \mathbb{R}^{q \times q}$ is $\Lambda^{yy} = J^2 W_i^{\mathsf{T}} D^{x\mathsf{T}}_{ti} D^x_{ti} W_i + \mathbf{I}$,
- the vector $\eta^I \in \mathbb{R}^{wh}$ is $\eta^I = \mathbf{0}$, and
- the vector $\eta^y \in \mathbb{R}^q$ is $\eta^y = y_{kt}$.

All of these matrices are specific to a single time step *t and* a single template hypothesis k_t, but we have left out these subscripts to avoid further burdening the notation.

To integrate out the unknown basis flow coefficients y_{kt} while obtaining the compact factored form of the marginal information matrix $\bar{\Lambda}^{II}$ and information vector $\bar{\eta}^I$ (the bar differentiates these from the blocks of the joint matrices in the previous expressions), we use the Schur complement,

$$\bar{\Lambda}^{II} = \Lambda^{II} - \Lambda^{Iy} \left(\Lambda^{yy}\right)^{-1} \Lambda^{Iy\mathsf{T}}, \text{ and } \bar{\eta}^I = \eta^I - \Lambda^{Iy} \left(\Lambda^{yy}\right)^{-1} \eta^y \tag{29}$$

with determinant

$$\left|\bar{\Lambda}^{II}\right| = \left|\Lambda^{yy} + \Lambda^{Iy\mathsf{T}} \Lambda^{II} \Lambda^{Iy}\right| \left|\left(\Lambda^{yy}\right)^{-1}\right| \left|\Lambda^{II}\right| \tag{30}$$

The marginal image derivative likelihood is thus

$$p(D_{ti} \,|\, k_t) = (2\pi)^{\frac{-wh}{2}} \left|\bar{\Lambda}^{II}\right|^{\frac{1}{2}} \log \frac{-1}{2} D_t \left(\bar{\Lambda}^{II} D_t^{\mathsf{T}} + \bar{\eta}^I\right) \tag{31}$$

The key with the factored forms of the marginal information matrix (29) and its determinant (30) is that the large dense $wh \times wh$ matrix $\bar{\Lambda}^{II}$ never needs to be formed explicitly. Instead, substituting its factored form into the image derivative likelihood (31) allows efficient evaluation, with no operations on matrices larger than $wh \times q$. Finally, we substitute this efficiently-computed image derivative likelihood into the template likelihood expression (17).

4.2 Learning Optical Flow Templates

To learn the optical flow templates, we first collected training data consisting of video from a front-facing camera on a robot driven through a course comprised of straight and curved segments. We separated the video into clips according to the path shape observed, resulting in videos observing straight course segments, left curves, and right curves. We then learned templates individually for each course shape video segment.

To learn the template W_k for the path shape k, from video frames $t \in T_k$, we wish to maximize the likelihood of the template given the image derivatives D,

$$\hat{W}_k = \arg\max_{W_k} p\left(W_k \,|\, D\right). \tag{32}$$

Using Bayes' law this likelihood is

$$p\left(W_k \,|\, D\right) = \prod_{t \in T_k} p\left(D_t \,|\, W_k\right) p\left(W_k\right) \tag{33}$$

and the image derivative likelihood is equivalent to that derived earlier in (26), i.e. $p\left(D_t \,|\, W_k\right) \equiv p\left(D_t \,|\, k\right)$.

Instead of optimizing the template likelihood (33) directly, we re-introduce the basis flow coefficients y_t as latent variables and alternately update the basis flows W_k and coefficients y_t by conditioning each on the other and the data according to

1. $\hat{W}_k \leftarrow \arg\max_{W_k} p\left(W_k \,|\, D, \hat{y}_k\right)$
2. $\hat{y}_k \leftarrow \arg\max_{y_k} p\left(y_k \,|\, D, \hat{W}_k\right)$.

This method is similar to probabilistic principal components analysis [85]. This method is also similar to the method presented in [76] for learning basis flows, though in the latter sparse optical flow was precomputed whereas here we learn basis flows directly from image derivatives. Each of these maximizations is a least-squares problem, which we solve using Cholesky factorization.

All of the component likelihoods needed to perform the updates on the basis flows and basis flow coefficients have been derived in Section 4.1. To use them, we apply Bayes law to obtain

$$p\left(W_k \,|\, D, \hat{y}_k\right) = \left(\prod_{t \in T_k} \prod_i p\left(D_{ti} \,|\, W_{ki}, \hat{y}_{tk}\right) \right) p\left(W_k\right)$$

and

$$p\left(y_k \,|\, D, \hat{W}_k\right) = \prod_{t \in T_k} \left(\prod_i p\left(D_{ti} \,|\, \hat{W}_{ki}, y_{kt}\right) \right) p\left(y_{kt}\right)$$

with $p\left(D_{ti} \,|\, W_{ki}, y_{tk}\right) \equiv p\left(D_{ti} \,|\, y_{tk}\right)$ from (24). As above, we use a normal prior on the basis flow coefficients $p\left(y_{kt}\right) = \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$. The prior on the basis flows $p\left(W_k\right)$ is a pairwise smoothness model that reduces the effect of noise and makes the optimization fully-constrained,

$$p\left(W_k\right) = \prod_i \prod_{j \in N(i)} \exp \frac{-1}{2} \left\| W_{ki} - W_{kj} \right\|^2_{\Sigma_W},$$

where $N(i)$ is the set of pixels directly adjacent (above, below, left, right) to pixel i, and $\Sigma_W \in \mathbb{R}^{2\times 2}$ is a covariance matrix. In our experiments we use $\Sigma_W = (0.01)^2\mathbf{I}$. We determined this parameter empirically, though it has little effect on the overall accuracy.

We also optimize the covariance parameters for the constant brightness and optical flow subspace models, Σ_I and Σ_u, by maximizing the image derivative likelihood $p\left(D\,|\,\hat{y}_k, \hat{W}_k\right)$. We use Newton's method paired with Cholesky factorization, simplifying the flow covariance $\Sigma_u \in \mathbb{R}^{2\times 2}$ to be uniform diagonal, and subject to the constraint that the variance parameters are greater than a small positive number (10^{-5}) to prevent numerical problems.

4.3 Implementation Details and Parameters

All datasets were collected from a 640×480 30Hz Unibrain Fire-i camera mounted on a wheeled platform. The free parameters of our method are the standard deviation of the image intensity noise for inlier and outlier pixels, for which we used $\sigma^{\mathrm{v}}_{\mathscr{J}} = \frac{1}{255}$ and $\sigma^{\mathrm{f}}_{\mathscr{J}} = \frac{5}{255}$, both in normalized grayscale units, and the per-pixel inlier prior, $p\,(\lambda_{kti}{=}1) = 0.95$. The optical flow covariances Σ^{v}_k and Σ^{f}_k are learned from the data as part of the templates.

We initialize all indicator expectations with $\langle\lambda_{kti}\rangle = 1$. We perform the optimization using the Gauss-Newton method.

Additionally, it is not necessary to perform inference up to the largest pyramid level. In our experiments we stop at level 3, corresponding to 80×60 images and basis flows scaled down from the original 640×480. Additionally, for optimization and inference, we sample only every other pixel, meaning that for the same image size, 1200 pixels are sampled at the largest pyramid level. With these parameters our single-threaded research implementation operates at approximately 15Hz on a 2.2GHz Intel Core i7 laptop.

Our method is highly parallelizable, in that the optimization and likelihood computation described in Section 4.1 may be performed independently and in parallel for each template. Also the image filtering operations such as gradient computations, resampling, and differencing may be threaded or even implemented on DSP, FPGA, or GPU hardware [38].

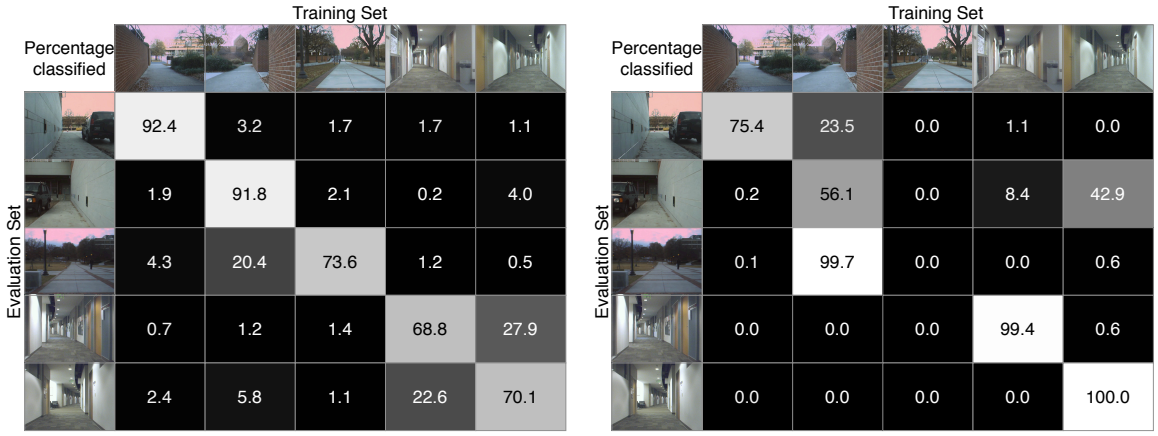
4.4 Experimental Results

In this section we provide experimental evidence to support the claims that *a)* our method efficiently classifies between multiple coarse path shapes and *b)* that to accomplish this it is important to use information from image motion instead of image appearance, to capture differences in structure. To support these claims we perform a quantitative evaluation and comparison with a supervised learning method trained on image appearance. Please see the end of this section for implementation details and parameters.

All datasets contain vibration, yawing, and side-to-side motions of the platform, which violate the *ideal* linearity described in Section 4.1.2, but fit the *approximate* linearity and are thus handled by our method.

4.4.1 Quantitative Evaluation

We learned linear optical flow templates for the environments exemplified by the top row of thumbnails of Figure 9, and performed inference on the environments exemplified by the



(a) Our method, model selection over linear optical flow templates (overall accuracy 74.7%)

(b) Image appearance classification (Neural network, Gist) (overall accuracy 67.3%)

Figure 9: Confusion matrices showing classification results of our method and a learned classifier on image appearance. The images are representative of each environment type in the training and testing datasets. Our higher accuracy on 'left wall', 'right wall', and 'walkway' highlight our use of image motion information versus image appearance. The appearance classifier's higher accuracy in differentiating between 'left curve' and 'right curve' is due to the appearance similarity between the training and testing sets, which were taken on two different floors of the same building. The image motion information, on the other hand, is subtle in these two environments because the hallway curvature is gentle.

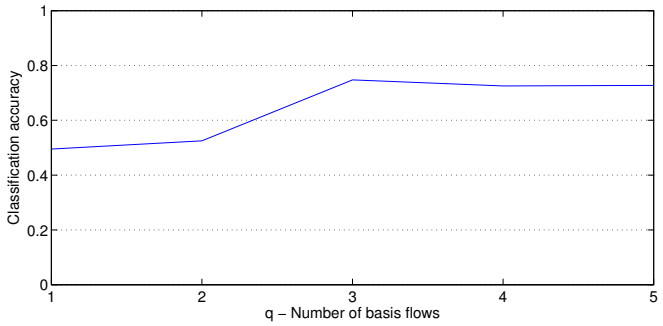


Figure 10: Classification accuracy for linear optical flow templates learned with various numbers of basis flows, i.e. latent variable dimensionalities q, learned and evaluated with the same datasets as in Figure 9.

left column of thumbnails. For the first three environments, the coarse environment shapes are similar between the training and testing datasets, but there are differences such as the height of the wall and distance to it varying. Additionally, the texture and image appearance is quite different. The last 2 environment types are left and right curving hallways (which we consider two different discrete types) with the same curvature. Texture and appearance differ less between training and testing sets in the curved hallways, they are from different building floors. The distance from the robot to the wall varies with side-to-side motions and yawing motions in all videos, and there is also camera vibration. We empirically selected to learn templates with $q = 3$ basis flows as this provided the highest accuracy.

Figure 9 shows the confusion matrices for our method and a learned classifier on image appearance. For the image appearance classifier, we use a neural network learned from hand-labeled data using Gist features. The Gist features were computed by the software[1] accompanying [80]. We chose to use Gist features because they are claimed to capture information about the coarse environment shape and appearance of a scene. We selected the subset of Gist features suggested by [80], and trained a neural network with 200 and 100 node hidden layers for 500 epochs (verifying that error on a holdout set did not increase during training) using Weka [28].

Figure 10 shows the classification accuracy on the same evaluation set for models learned with various numbers of basis flows, i.e. latent variable dimensionality q.

Our higher accuracy on 'left wall', 'right wall', and 'walkway' highlights our use of image motion information versus image appearance. The image appearance classifier's higher accuracy in differentiating between 'left curve' and 'right curve' is due to the appearance similarity between the training and testing sets, which were taken on two different floors of the same building. The key point to note is the difference in the accuracy of the appearance classifier when the training and evaluation images appear similar versus when they appear different. When appearance differs, the accuracy of the appearance-based classifier decreases, where the accuracy of our method remains the approximately the same.

4.5 Summary

In this chapter we presented a method for classifying coarse environment shape from image motion. To do this classification, the method performs approximate model selection over a collection of linear optical flow templates. Each template encodes a coarse environment shape, by means of a set of *basis flows* spanning the subspace of optical flow fields that a moving platform may observe in that environment, under the assumption of per-pixel depth constancy over time. The input is a video stream, and the output is a set of likelihoods for each frame that the image change from the previous frame is explained by each linear optical flow template. Inference takes place directly on spatial image gradients, not requiring optical flow to be computed first. Our results show that our method classifies between training and evaluation datasets whose corresponding environment types are similar in large-scale structure but different in appearance and contain outliers like passing objects.

[1] Available from http://ilab.usc.edu/siagian/Research/Gist/Gist.html

CHAPTER V

OPTICAL FLOW TEMPLATES FOR SUPERPIXEL LABELING IN AUTONOMOUS ROBOT NAVIGATION

For a camera attached to a mobile robot, instantaneous image motion contains rich information about the robot's egomotion and the structure of the environment. In this chapter, our contribution is to present a new framework for capturing this information, in which we attempt to address some shortcomings of previous work, and present an experimental proof-of-concept of this framework.

This framework, which we call *optical flow templates*, illustrated in Figure 11, permits semantically labeling image regions according to their observed optical flow and the predicted optical flow of several templates. Optical flow templates predict the optical flow due to robot egomotion, for a given robot velocity and attitude, and for a particular type of environment structure.

In this work, we use templates for *ground plane* and *distant structure*, and label regions that are not consistent with any other template as possible *obstacles*. Ground plane labels are likely to indicate fairly flat ground that can be driven upon. Far-away objects "at infinity" indicate line-of-sight directions that are likely free of obstacles. Image regions that are not consistent with either of these are likely to be nearby objects or other moving objects to be considered as obstacles.

The work to date towards using image motion for autonomous navigation has several drawbacks that limit its usefulness. While our framework does not completely solve the problem, it is a new way of looking at the problem that addresses some of the drawbacks.

5.1 Optical Flow Templates

An optical flow template encodes the space of possible optical flow fields corresponding to a certain environment structure, invariant to the platform velocity. In turn, optical flow templates also specify a linear mapping from platform velocity $\xi = [\omega_x, \omega_y, \omega_z, v_x, v_y, v_z]^\mathsf{T}$ to optical flow, for a given robot attitude, and for a particular environment structure class. Shown in Figure 11, we specifically work with 3 possible classes: *ground plane*, *distant structure*, and *obstacle*. Let $W^k(\theta) \in \mathbb{R}^{2wh \times q}$ be the optical flow template for environment structure class k, where w and h are the image width and height, and q is the velocity dimension (for our experiments $q = 6$). Optical flow templates are nonlinear functions of the platform attitude θ. Thus they predict a flow field for structure class k as $u^k = W^k(\theta)\,\xi$.

5.1.1 Optical Flow as a Gaussian Mixture of Templates

Because each pixel may be generated from any template, we model the optical flow u_i at each pixel i as a Gaussian mixture of the flow predicted $\hat{u}_i^k$ by each template,

$$p\left(u_i \,|\, \Lambda_i, \xi\right) = \mathcal{N}\left(0, \Sigma^0\right)^{\Lambda_i^0} \prod_{k=1}^{\kappa} \mathcal{N}\left(\hat{u}_i^k\left(\xi\right), \Sigma\right)^{\Lambda_i^k}, \tag{34}$$

where $\Lambda_i^k \in \{0,1\}$ is a binary indicator of 1 if pixel i takes the *discrete* label k, and 0 otherwise. Σ is the covariance of the Gaussian noise on the optical flow consistent with the template, and Σ^0 is the covariance of zero-mean Gaussian noise on optical flow that is not consistent with any template, i.e. is labeled as *obstacle*. $\hat{u}_i^k\left(v\right) = W_i^k\left(\theta\right)\xi$ is the optical flow predicted by template k at pixel i, and we assume that the platform attitude θ is known, as explained in Section 5.1.2 where we calculate the templates. The notation W_i^k selects the pair of rows corresponding to pixel i from the optical flow template W^k. As is typical with notation in Gaussian mixtures, raising each term to the power of the indicator variable effectively makes only one term active for a given labeling.

Because the optical flow templates result in a linear relationship between velocity and optical flow, the measurement likelihood in Eq. 34 is Gaussian, allowing inference to be fast and guaranteeing convergence. Each optical flow template is one instance of the optical flow subspace model introduced in our previous work [76].

5.1.2 Calculating Optical Flow Templates

The optical flow field in a calibrated projective camera is (see e.g. [32] for derivation)

$$u_i = \begin{bmatrix} y_i & \frac{x_i y_i}{f} & -f - \frac{x_i^2}{f} & \frac{x_i}{z_i} & \frac{-f}{z_i} & 0 \\ -x_i & f + \frac{y_i^2}{f} & \frac{-x_i y_i}{f} & \frac{y_i}{z_i} & 0 & \frac{-f}{z_i} \end{bmatrix} \xi, \tag{35}$$

where x_i and y_i are the horizontal and vertical image coordinates (with the origin at the image center) at pixel i, z_i is the depth at pixel i, and f is the focal length. The matrix is arranged such that $\xi = [\omega_x, \omega_y, \omega_z, v_x, v_y, v_z]^\mathsf{T}$ is the rotational and translational velocity in "robot coordinates", where the x-axis points forwards and the z-axis points down.

For a flow field u comprised of the concatenated flow vectors $u_i \in \mathbb{R}^2$, the optical flow template $W^k\left(\theta\right)$ is thus formed by stacking all of the matrices in Eq. 35. The depth z_i at each pixel depends on both the scene structure and the platform attitude θ. Here, we consider templates corresponding to *ground plane* and *distant structure*. For *ground plane*, we compute z_i using plane-line intersection geometry. For brevity, we omit the derivation which is a straightforward application of the tools in Chapter 2 of [30]. For *distant structure*, we use $z_i = \infty$ everywhere. In Figure 11, the ground plane template shows the camera is pitched slightly down – the boundary between the black and the colorful regions is the horizon.

We assume the platform attitude θ is known. In our experiments, we obtain it from an attitude/heading reference system (AHRS). On autonomous ground vehicles and aircraft, this is common equipment, and is typically implemented using measurements from an accelerometer and gyroscope [19].

5.2 *Superpixel Labeling*

Our method assigns a probability that each superpixel of optical flow in a frame of video belongs to each class, with each class represented by an *optical flow template.* Inference is a matter of labeling superpixels such that all superpixels assigned to the same template exhibit consistent optical flow within that template, and that all superpixels across all templates predict a consistent platform velocity. To simplify notation, we do not include frame numbers or time steps, but each frame is labeled independently.

Let $\lambda_j \in \mathbb{R}^\kappa$ be the vector of probabilities that superpixel j belongs to each class, and $\lambda_j^k \in \mathbb{R}$ be the k^{th} element, i.e. the probability that superpixel j belongs to class k. Let $k \in \{0,1,2\}$, where the special class $k=0$ corresponds to *obstacle*, and $\kappa = 3$ is the number of classes (including *obstacle*). Both the superpixel labels λ_j and the velocity estimate ξ are alternatively refined, but for simplicity we omit iteration numbers from the notation.

5.2.1 Superpixel Labeling Given a Velocity Estimate

Our end goal is to label superpixels according to which optical flow template they are consistent with. This requires knowing the optical flow templates themselves (as obtained in Section 5.1.2). Although each optical flow template encodes an optical flow subspace that is invariant to platform velocity, we want to enforce that all superpixels predict a consistent platform velocity. Thus, we will iteratively estimate the platform velocity (as explained in Section 5.2.2), and in this section, take the current velocity estimate ξ.

While in the previous Section 5.1 we defined the density of optical flow at a pixel i predicted by an optical flow template, we now switch to superpixels. All of the inference here may equally be performed at the pixel level, but using superpixels greatly reduces the computational expense. We use the index j to denote a superpixel, and use u_j and Λ_j to denote the flow and labeling of a superpixel, and $W_j^k(\theta)$ to denote the pair of rows corresponding to the pixel at the center of superpixel j.

We estimate the probability λ_j^k of each j^{th} superpixel belonging to each class. These probabilities define a multinomial distribution $p\left(\Lambda_j = e_k\right) = \lambda_j^k$ over the labels (where $e_k \in \mathbb{R}^\kappa$ is a vector with 1 in position k and 0 elsewhere, i.e. an assignment of the discrete indicator variables). Each assignment probability λ_j^k is the normalized likelihood that the superpixel is consistent with template k,

$$\lambda_j^k = \frac{l\left(\Lambda_j = e_k \,|\, \tilde{u}_j, \xi\right)}{\sum_{\bar{k}} l\left(\Lambda_j = e_{\bar{k}} \,|\, \tilde{u}_j, \xi\right)}, \tag{36}$$

where $l(\cdot) \propto p(\cdot)$ denotes a likelihood and $\tilde{u}_j$ is the *average* measured optical flow in superpixel j. We calculate the likelihoods here using Bayes' law,

$$l\left(\Lambda_j \,|\, \tilde{u}_j, \xi\right) = p\left(\tilde{u}_j \,|\, \Lambda_j, \xi\right) p\left(\Lambda_j\right), \tag{37}$$

where the measurement likelihood $p\left(\tilde{u}_j \,|\, \Lambda_j, \xi\right)$ is as in Eq. 34, except that the predicted flow $\hat{u}_i^k(\xi)$ is calculated by choosing pixel i to be at the center of superpixel j. $p\left(\Lambda_j\right)$ is the class prior, which in our experiments is a simple multinomial distribution.

In the next section, we describe refining the velocity estimate ξ given the labeling estimated with Eq. 36 above. As mentioned before, the velocity and labeling are alternatingly refined until convergence.

5.2.2 Refining the Velocity Estimate Given the Labeling

To refine the velocity ξ, we treat the labeling Λ_j as a hidden variable and update the velocity using expectation-maximization (EM),

$$\xi \leftarrow \arg\max_{\xi} \langle \mathscr{L}(\xi \,|\, \tilde{u}, \Lambda) \rangle, \tag{38}$$

where $\mathscr{L}(\cdot) = \log l(\cdot)$ denotes a log-likelihood, $\tilde{u}$ and Λ are the collections of optical flow vectors and indicator vectors for all superpixels, and the expectation $\langle \cdot \rangle$ is with respect to $p(\Lambda \,|\, \tilde{u}, \xi)$. Using Bayes' law and the linearity of the expectation, we have

$$\langle \mathscr{L}(\xi \,|\, \tilde{u}, \Lambda) \rangle = \mathscr{L}(\xi) + \sum_j \left\langle \mathscr{L}\left(\tilde{u}_j \,|\, \Lambda_j, \xi\right) \right\rangle, \tag{39}$$

where the expectation is now with respect to $p\left(\Lambda_j \,|\, \tilde{u}_j, \xi\right)$. To compute the expected log measurement likelihood $\left\langle \mathscr{L}\left(\tilde{u}_j \,|\, \Lambda_j, \xi\right) \right\rangle$, we note that the class probabilities λ_j^k calculated in the previous section comprise the *sufficient statistics* for this EM algorithm, giving everything we need to calculate

$$\left\langle \mathscr{L}\left(\tilde{u}_j \,|\, \Lambda_j, \xi\right) \right\rangle = \sum_k \lambda_j^k \mathscr{L}\left(\tilde{u}_j \,|\, \Lambda_j = e_k, \xi\right), \tag{40}$$

where the the log measurement likelihood is the log of Eq. 34. Note that the velocity ξ used in Eq. 34 is in fact the variable being optimized for in this section, and is not fixed here as it was in Section 5.2.1.

Because the measurement likelihood is Gaussian, the expected log-likelihood is a quadratic, and thus refining the velocity estimate with the maximization in Eq. 38 is a linear least-squares problem that is easily solved using direct methods.

5.2.3 Summary and Implementation of Method

Given each of the components we have introduced in this section, we now summarize the steps that take place for each incoming video frame here in Algorithm 1:

5.3 Experiments

Our experimental platform is a car equipped with a Point Grey Flea3 (gigabit ethernet version) camera running at 1380×480 at 10Hz, and a MicroStrain 3DM-GX3-45 inertial navigation system (INS), from which we use only the AHRS attitude estimate. Both were

[1]This convergence criteria relies on the likelihood normalization constant remaining constant during optimization. To accomplish this, it is sufficient to include in the likelihood calcaulations the normalization constants from all Gaussians involving the velocity v.

Algorithm 1: Superpixel labeling for each frame.

Input: Current and previous video frames.
Input: Platform attitude θ (e.g. from AHRS).
Result: Class label probabilities λ_j^k for each superpixel j and class k.

1 For current frame, compute SLIC superpixels [2].
2 For previous and current frames, compute TV-L^1 [98, 77] optical flow $\tilde{u}$ and average flow $\tilde{u}_j$ in each superpixel.
3 Compute basis flows $W_j^k(\theta)$, at image locations corresponding to superpixel centers.
4 Initialize class label probabilities uniformly, $\lambda_j^k = \frac{1}{\kappa}$.
5 **while** *change in* $\langle \mathscr{L}(\xi \mid \tilde{u}, \Lambda) \rangle$ *is greater than convergence threshold*^{1} ε **do**
  Update $\xi \leftarrow \arg\max_\xi \langle \mathscr{L}(\xi \mid \tilde{u}, \Lambda) \rangle$ (Sec. 5.2.2)
  Update $\lambda_j^k \leftarrow \frac{l(\Lambda_j = e_k \mid \tilde{u}_j, \xi)}{\sum_{\bar{k}} l(\Lambda_j = e_{\bar{k}} \mid \tilde{u}_j, \xi)}$ (Sec. 5.2.1)
end
6 **return** λ_j^k

connected to an Intel Core i7 laptop for data logging, and the computations described were performed on the logged data.

We operated the car in an urban environment. In collecting the dataset, we took care not to follow any car in front too closely. We did not make any other special considerations in driving style while collecting the dataset.

The parameters $\Sigma = \mathrm{diag}(7.0)$ and $\Sigma^0 = \mathrm{diag}(15.0)$ were used for the noise covariances. These values were selected empirically: Too tight covariance causes too many superpixels to be labeled as *obstacle* due to slight noise in optical flow measurements, while too loose covariance causes all superpixels to be labeled only *ground plane* or *distant structure*. For the label priors, we used $p(\Lambda = \textit{ground plane}) = 0.4$ and $p(\Lambda = \textit{distant structure}) = 0.4$ for superpixels below the horizon, and $p(\Lambda = \textit{ground plane}) = 0$ and $p(\Lambda = \textit{distant structure}) = 2/3$ above the horizon (in each case the remaining probability is for *obstacle*). These values were chosen by manual estimation of the image portion occupied by each class in several typical frames, although the algorithm is not very sensitive to this prior.

Our implementation of the algorithms is in C++ using the GTSAM factor graph library [16]. The C++ classes are wrapped in MATLAB using the `wrap` utility included in GTSAM, and we perform data loading, scripting, and visualization in MATLAB. All source code and datasets will be made available online on the authors' web site by the time of publication (link will be provided in camera-ready version).

5.3.1 Qualitative Analysis

In Figure 12, we present examples of the labeling produced by our method that we consider successful. These examples are successful because the information they contain is useful for autonomous navigation. For the most part the major structure above the ground plane and close to the camera is labeled as "*obstacle*" (red), and the *ground plane* (green) and *distant structure* (blue) are mostly correctly labeled. The *obstacle* structure above the ground

plane is often obstacles, road boundaries, or independently-moving objects. Such structure could be avoided by a navigation method or passed on to higher-detail vision processing.

Sometimes the labeling produced by our method contains errors. One type of error arises because the optical flow estimate is incorrect. Most frequently, this occurs in regions with smooth or repetitive texture, as shown in Figure 14a. This type of error points us towards future work because the root cause of this problem is computing optical flow as an *input* that is unaware of the global optical flow constraints provided by our optical flow templates. We expand on this in the discussion. Another type of error occurs when the differences between the predicted optical flow from two templates becomes too small to distinguish from noise, as shown in Figure 14b. In this case, superpixels on the ground plane may be be labeled with 50% probability of belonging to either the *ground plane* or *distant structure* classes, indicated in the figure by the green/blue or turquoise blended color; also, ground plane superpixels may be labeled as *distant structure* if small error in the velocity estimate of our method causes the optical flow on the ground plane to agree more closely with rotational flow than with translational flow.

5.3.2 Quantitative Analysis

We hand-evaluated 200 frames of video and labels produced by our method, the results of which are shown in Figure 15. In each frame, we determined the number of objects or regions of environment structure mislabeled by our method. "Extra obstacles" are image regions that our method labels in the *obstacle* class, yet there is no structure above the ground plane near the camera in that region. "Missed obstacles" are image regions containing structure above the ground plane near the camera but that our method does not label as *obstacle*. For example, in Figure 14a we would count the region mislabeled on the ground plane as one "extra obstacle", and a missed car or pedestrian would count as one "missed obstacle". In hand-labeling, we took into account the smoothing inherent in the optical flow calculation, so several objects close together, for example poles on the side of the road, are only counted as one object.

Because our method estimates superpixel labels jointly with platform velocity, we test whether errors in superpixel labels are coupled with errors in yaw-rate estimation. Figure 15 plots the yaw-rate error of our method as compared with the vehicle's gyroscope. While gross yaw errors would certainly be coupled with many incorrect superpixel labels, the comparison demonstrates that there is no correlation between small yaw errors and errors in superpixel labeling.

5.3.3 Timing

Our research implementation is single-threaded, and takes approximately 20 ms per frame ($\pm$2 ms) to infer superpixel labels and velocity, already having the optical flow and superpixel segmentation available as input. For each frame, the combined time to calculate TV-L^1 optical flow and SLIC superpixels on the CPU is 500 ms per frame, however, there are GPU versions available of both algorithms that achieve real-time performance using CUDA. All timings were measured on an Intel Core i7 3.4 GHz desktop. The nature of the linear least-squares problem makes it adaptable to parallelization and GPU implementation.

5.4 Summary

We have presented a new framework for interpreting optical flow observed by a mobile robot, called *optical flow templates*. Using templates for *ground plane* and *distant structure*, our method labels image regions whose flow is consistent with these templates, as well as labeling flow inconsistent with either as *obstacle*. This latter class comprises objects that occupy space above the ground plane near to the robot, and may be passed on for more detailed and computationally intensive processing.

The key aspects of the superpixel labeling method using this framework, in relation to previous work, include that computational complexity is very low, and geometric models of optical flow remove the need for hand-labeled training data or heuristics. We present an experimental proof-of-concept, labeling video in an urban driving scenario.

We have observed, as shown in Section 5.3.1, that many of the errors made by our method come from errors in optical flow estimation. These errors especially occur in regions of smooth texture, where the task of optical flow is underconstrained and ill-posed without a global optical flow model. Our optical flow templates in fact provide such a model, and joint inference makes much more accurate labels possible, as we show in the next chapter.

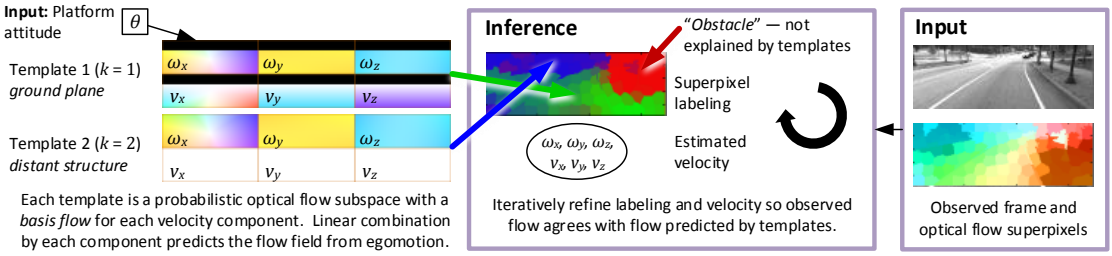
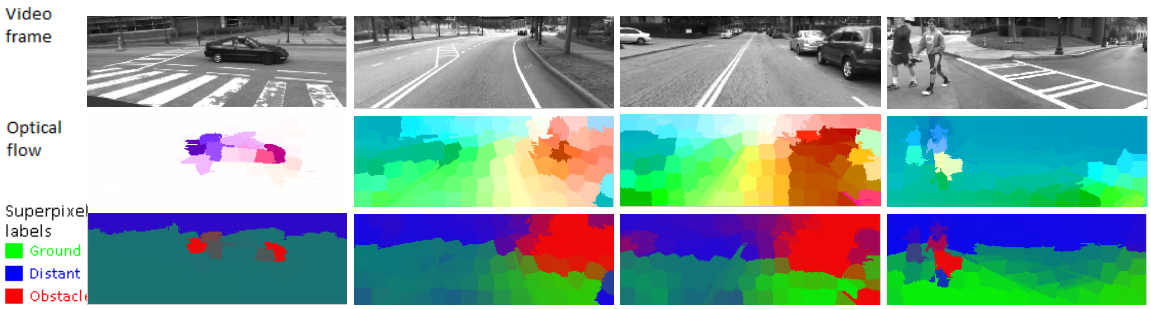


Figure 11: *Optical flow templates* for superpixel labeling. Optical flow templates $W^k(\theta)$, one for each structure class *ground plane* ($k = 1$) and *distant structure* ($k = 2$), predict optical flow u_i at each i^{th} image location, given a platform velocity estimate $\xi = [\omega_x, \omega_y, \omega_z, v_x, v_y, v_z]^\mathsf{T}$ and attitude estimate $\theta \in \mathbb{SO}(3)$. Each template consists of 6 flow fields that combine linearly for each velocity component. For the optical flow color code, see Figure 13. Using observed optical flow, alternatively refine the labeling k_i for each i^{th} superpixel and the estimated velocity ξ. The special class $k = 0$ indicates optical flow pixels that cannot be explained by any template, which we label as *obstacle*.



(a) Car crossing perpendicularly while platform stationary

(b) Passing lamps on right side while driving forwards

(c) Passing cars parked on both sides while driving forwards

(d) Encountering pedestrians during a right turn

Figure 12: Example video frames, superpixel optical flow fields, and superpixel labels by our method.

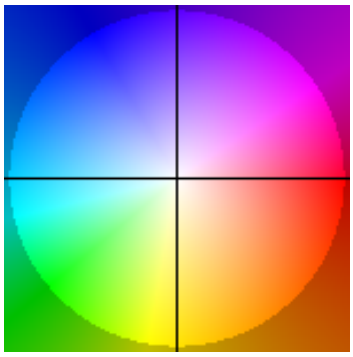
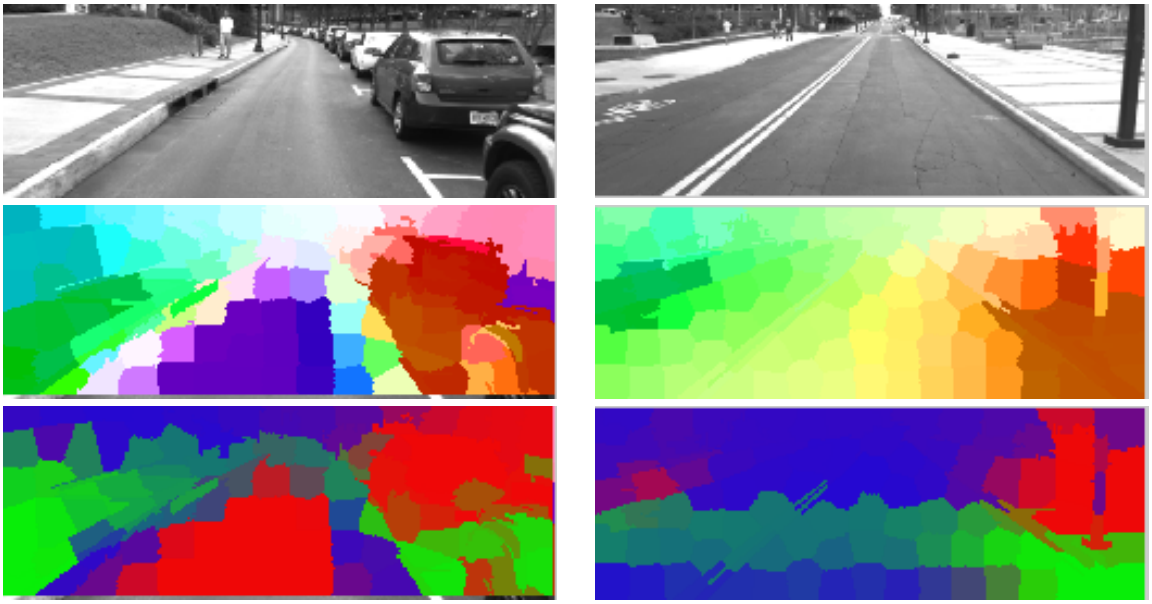


Figure 13: The optical flow color code, as in [5]. Flow fields (e.g. Figure 11) are displayed with a color at each pixel. The hue indicates the direction of the flow and the saturation its magnitude. Here, the center of the black cross (color white) is zero flow. Yellow is downwards flow, red rightwards, etc.



(a) Error in labeling the ground as *obstacle* due to optical flow errors caused by smooth texture.

(b) Error in labeling ground plane as *distant structure* due to similarity between rotational and translational basis flows.

Figure 14: Examples demonstrating errors in labeling by our method.

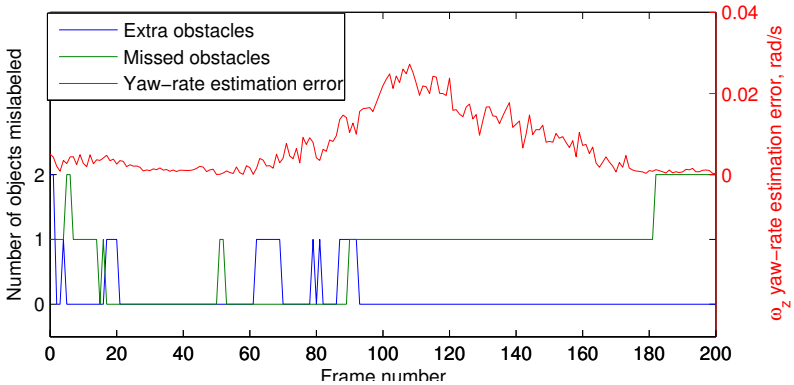


Figure 15: Number of objects mislabeled by our method, evaluated by hand-inspecting 200 frames labeled by our method. We counted "extra obstacles", which are image regions detected by our method as the *obstacle* class when no nearby structure was in fact present, and "missed obstacles", where nearby structure was present but not detected.

CHAPTER VI

DIRECTLY LABELING SUPERPIXELS FROM SPATIOTEMPORAL GRADIENTS

We are interested in autonomous obstacle avoidance for mobile robots using camera sensors. Most of the work to date on this problem has focused primarily on either building and analyzing sparse point clouds, or labeling pixels or superpixels based on image appearance. However, both these families of methods have shortcomings, and optical flow provides an additional powerful source of information to complement point clouds and image appearance.

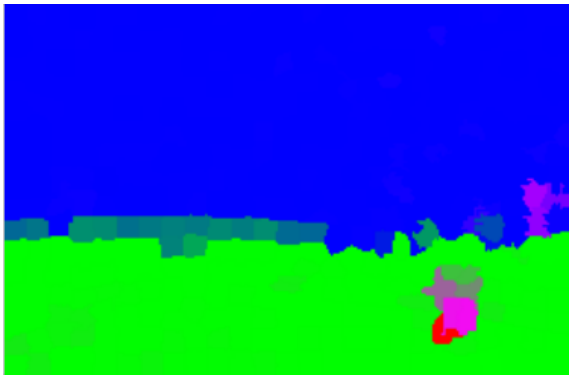
The most ubiquitous methods to date in autonomous robot systems using camera input have focused on two families of methods. In one, stereo or structure-from-motion is used to build point clouds or depth maps, which are analyzed to estimate a ground traversability map, for example [61], [73], [37]. In [27], [51], and [50], image appearance traversability classifiers, combined with a model of the ground plane, are used to propagate stereo-labeled image regions to distances in the traversability map beyond the range of stereo. Many successful methods also use image appearance to propagate traversability but without the use of nearby stereo-informed labels [54], and others use examples provided by a human operator to inform labeling [97]. In most of the working systems described by the preceding papers, stereo and appearance traversability are combined to produce the traversability map used for motion planning.

The above methods are ill-suited for small or inexpensive robots. The point clouds and depth maps of the first family of methods must be fairly dense to support obstacle avoidance, which necessitates expensive and powerful hardware. Additionally, they require a calibrated camera, and calibrating wide-angle lenses, which are certainly useful for obstacle avoidance, is challenging and requires special models and methods. While appearance-based classification methods are computationally fast and do not rely on calibration, they either must be adapted online using information from point clouds or depth maps, or trained offline and thus unable to cope with unexpected appearance changes.

Thus, we argue that optical flow provides a rich source of information to complement image appearance and point clouds in determining traversability. To demonstrate this, we present a method for labeling superpixels in video as *ground*, *distant*, *obstacle*, or *unknown* using as input the observed spatiotemporal image gradients, and a learned optical flow template model, as previewed in Figure 16. We additionally present a method for learning these optical flow templates.

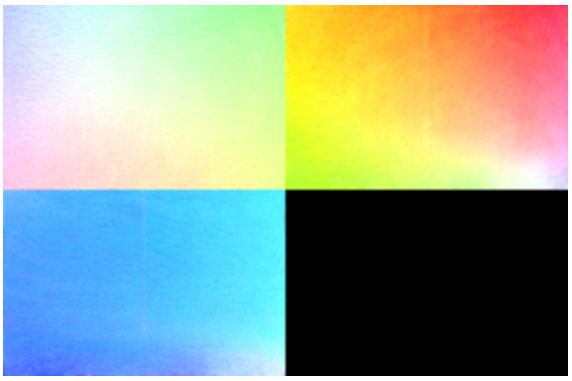
The contributions of this work over previous work in superpixel labeling with optical flow templates [74] are that we *1)* improve labeling accuracy and computational efficiency over prior work by inferring labels *directly from spatiotemporal gradients*, instead of from separately-computed optical flow, *2)* improve labeling accuracy by improved modeling of the *obstacle* class, and *3)* expand applicability to uncalibrated cameras of arbitrary optics, by developing a method for unsupervised learning of optical flow templates from video.

(a) Camera frame



(b) Superpixel label probabilities

(c) Superpixel label legend



(d) Learned rotational optical flow template

(e) Learned translational optical flow template

Figure 16: We are concerned with obstacle avoidance for mobile robots *a)* from a single uncalibrated camera sensor with arbtrary optics. Note the large radial distortion which cannot be undistorted by typical methods. *b)* Towards this goal our method labels superpixels using information from optical flow. *d,e)* Our method learns *optical flow templates* to inform this labeling. Arbitrary camera optics are supported, and there is no need to calibrate the camera or hand-label any training data.

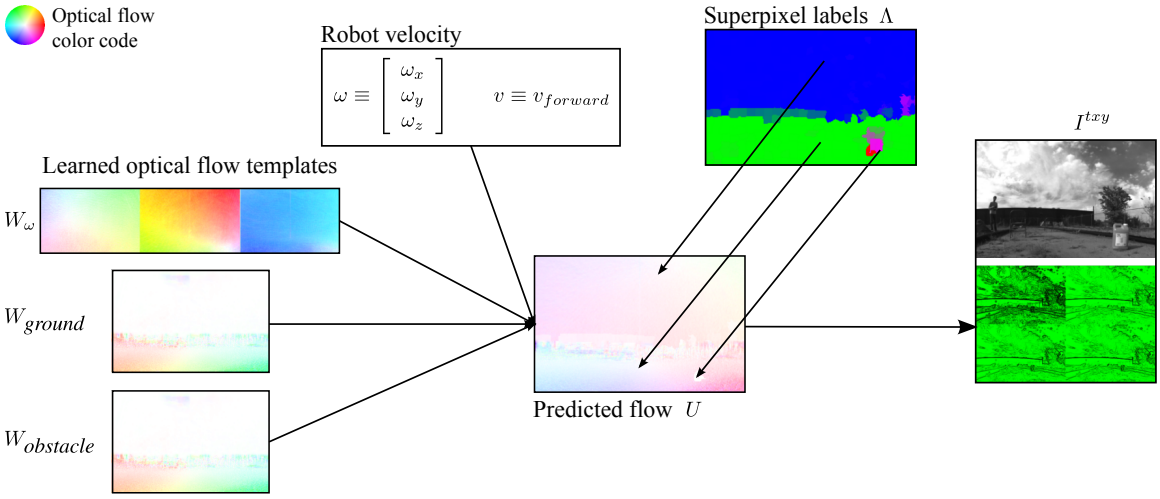


Figure 17: Illustrated Bayes net of the optical flow template model.

6.1 Labeling Superpixels Using Optical Flow Templates

Our goal is to label superpixels according to the scene structure at each superpixel, using as input the spatiotemporal image gradients. To inform this labeling, we use learned models of the optical flow fields due to platform egomotion for a set of possible typical scene structure, called *optical flow templates*. Note that we perform this labeling *without* computing optical flow – dense optical flow calculation is ill-posed and unnecessarily expensive, which is why we instead directly use the spatiotemporal image gradients. Additionally, no velocity information is needed to perform the labeling.

The method explained in this section is the on-line inference component, which is used after the optical flow templates have already been learned off-line using the method explained in Section 6.2.

6.1.1 Labeling Superpixels in Observed Images

We wish to infer superpixel labels $\Lambda_i \in \{0..\kappa\}$ for each i^{th} superpixel in an image, given the image temporal and spatial derivatives I^{txy} of the image and the parameters Θ of the learned optical flow template model. κ is the number of labeling classes other than the *unknown* class.

We will perform inference in the generative model illustrated in Figure 17, whose joint density is

$$p\left(I^{txy}, U, \Lambda, \omega, v \mid \Theta\right) = p\left(I^{txy} \mid U\right) p\left(U \mid \Lambda, \omega, v, \Theta\right) p\left(\Lambda\right) p\left(\omega, v\right). \tag{41}$$

In this model, the unknowns are the superpixel labels Λ, the robot velocity ω, v, and the optical flow U at each pixel, while the observations are the image spatiotemporal gradients I^{txy}. The optical flow templates, encapsulated in the optical flow template parameters Θ, are known from the learning phase to be explained in Section 6.2.

Exact inference of the superpixel labels would require an intractable marginalization of the remaining unknown flow U and velocity ω, v, so we instead employ a variational inference method. Our method alternates between updating the probability of each superpixel label, and a Gaussian density approximation of the robot velocity.

In this variational method, the inference problem is approximated as

$$p\left(\Lambda \mid I^{txy}, \Theta\right) \approx \int_{\omega,v} p\left(\Lambda \mid I^{txy}, \omega, v, \Theta\right) \mathcal{N}\left(\left(\hat{\omega}, \hat{v}\right), \hat{\Sigma}_{\omega,v}\right), \tag{42}$$

where $\hat{\omega}, \hat{v}$ is the mean and $\hat{\Sigma}_{\omega,v}$ is the covariance of a Gaussian density estimate that we will iteratively update. We will derive a simple closed form expression for Eq. 42 at each iteration. The two steps in each iteration are:

6.1.1.1 Label Probability Update

In this step we update a vector of label probabilities $\tilde{\Lambda}_i \in \mathbb{R}^{\kappa+1}$ for each superpixel by evaluating Eq. 42. The first term in Eq. 42 is obtained by applying Bayes' law and marginalizing out the unknown optical flow U in the generative model in Figure 17,

$$\begin{aligned}
p\left(\Lambda \mid I^{txy}, \omega, v, \Theta\right) &= \prod_i p\left(\Lambda_i \mid I_{J_i}^{txy}, \omega, v, \Theta\right) \\
&\propto \prod_i \prod_{j \in J_i} p\left(I_j^{txy} \mid \Lambda_i, \omega, v, \Theta\right) p\left(\Lambda_i \mid \Theta\right) \\
&= \prod_i \prod_{j \in J_i} \int_{U_j} p\left(I_j^{txy}, U_j \mid \Lambda_i, \omega, v, \Theta\right) p\left(\Lambda_i \mid \Theta\right) \\
&= \prod_i \prod_{j \in J_i} \int_{U_j} p\left(I_j^{txy} \mid U_j\right) p\left(U_j \mid \Lambda_i, \omega, v, \Theta\right) p\left(\Lambda_i \mid \Theta\right),
\end{aligned} \tag{43}$$

where J_i is the set of pixels in superpixel i. Note that in the first line, the superpixel labels are independent of each other when conditioned on the robot velocity ω, v.

The first term $p\left(I_j^{txy} \mid U_j\right)$ in Eq. 43 is the image intensity likelihood which, assuming a static and non-occluding scene, depends only on the optical flow. Thus we define it using the brightness constancy constraint,

$$p\left(I_j^{txy} \mid U_j\right) = \mathcal{N}\left(0; I_j^t + I_j^{xy} U_j, \sigma_I\right), \tag{44}$$

where $I_j^{xy} \in \mathbb{R}^{1\times 2}$ is the spatial image gradient at pixel j, and σ_I is the standard deviation of pixel intensity noise.

The second term $p\left(U_j \mid \Lambda_i, \omega, v, \Theta\right)$ is the optical flow likelihood, which in the optical flow template model depends on the superpixel label Λ_i, the robot velocity ω, v, and the template parameters Θ. The form of the optical flow template model is Gaussian, with mean linear in the robot velocity,

$$p\left(U_j \mid \Lambda_i, \omega, v, \Theta\right) = \mathcal{N}\left(W_{\omega,j}\omega + W_{\Lambda_i,j} v, \Sigma_{\Lambda_i}\right) \tag{45}$$

for matrices $W_{\omega,j} \in \mathbb{R}^{2\times 3}$, $W_{\Lambda_i,j} \in \mathbb{R}^{2\times q}$, $\Sigma_{\Lambda_i} \in \mathbb{SPD}^{2\times 2}$, which will be fully explained in Section 6.1.2.

The label prior $p(\Lambda_i \mid \Theta)$ is a categorical distribution. The outlier class *unknown* is assigned a small probability ($p\left(\Lambda_j{=}\textit{unknown}\right) = 0.05$ in our experiments). Then, the region of pixels close to the horizon receives equal probability *distant* and *ground*, and slighly lower probability *obstacle*. The pixels above that region receive zero *ground* probability, and the pixels below that region receive zero *distant* probability.

Since all terms in Eq. 43 are Gaussian as just explained, the integral in Eq. 43 can be computed analytically as

$$\begin{aligned} p\left(I_j^{txy} \mid \Lambda_i, \omega, v, \Theta\right) &= \int_{U_j} p\left(I_j^{txy} \mid U_j\right) p\left(U_j \mid \Lambda_i, \omega, v, \Theta\right) \\ &= \mathcal{N}\left(I_j^t; -I_j^{xy}\left(W_{\omega,j}\hat{\omega} + W_{\Lambda_i,j}\hat{v}\right), \rho_j\right), \end{aligned} \tag{46}$$

with

$$\rho_j = \sigma_i^2 + I_j^{xy}\left(\Sigma_{\Lambda_j} + \left[W_{\omega,j}\, W_{\Lambda_j,j}\right]\hat{\Sigma}_{\omega v}\left[W_{\omega,j}\, W_{\Lambda_j,j}\right]^{\mathsf{T}}\right) I_j^{xy\,\mathsf{T}},$$

where $\hat{\omega}, \hat{v}$ and $\hat{\Sigma}_{\omega v}$ are the statistics of the Gaussian density estimate on velocity from the previous iteration, whose calculation we will now explain.

6.1.1.2 Velocity Gaussian Density Update

As mentioned, we update the mean $\hat{\omega}, \hat{v}$ and covariance $\hat{\Sigma}_{\omega v}$ of the estimated velocity density using the label probability vectors $\tilde{\Lambda}_i$ from the previous iteration. The mean is the maximization of the expected log-likelihood with respect to the label distribution from the previous iteration,

$$\begin{aligned} \hat{\omega}, \hat{v} &\leftarrow \arg\max_{\omega,v} \left\langle \log p\left(\omega, v \mid I^{txy}, \Lambda, \Theta\right)\right\rangle \\ &= \arg\max_{\omega,v} \sum_i \sum_{j\in J_i} \left\langle \log p\left(I_j^{txy} \mid \Lambda_i, \omega, v, \Theta\right)\right\rangle + \log p(\omega, v) \\ &= \arg\max_{\omega,v} \sum_i \sum_{j\in J_i} \sum_{k\in\{0..\kappa\}} \tilde{\Lambda}_{i,k} \log p\left(I_j^{txy} \mid \Lambda_i, \omega, v, \Theta\right) + \log p(\omega, v) \end{aligned} \tag{47}$$

where the prior $p(\omega, v)$ is assumed to be uninformative and

$$p\left(I_j^{txy} \mid \Lambda_i, \omega, v, \Theta\right) = \mathcal{N}\left(I_j^t; -I_j^{xy}\left(W_{\omega,j}\omega + W_{\Lambda_i,j}v\right), \eta_j\right), \text{ with} \tag{48}$$

$$\eta_j = \sigma_I^2 + I_j^{xy}\Sigma_{\Lambda_i} I_j^{xy\,\mathsf{T}}.$$

The maximization is a linear least-squares problem. The definition in Eq. 48 is obtained by computing the integral $\int_{U_j} p\left(I_j^{txy}, U_j \mid \Lambda_i, U_j, \Theta\right)$ in closed-form.

The covariance estimate $\hat{\Sigma}_{\omega v}$ is that of the Gaussian defined by the exponential of the expected log-likelihood in Eq. 47, $\exp\left\langle \log p(\omega, v \mid I^{txy}, \Lambda, \Theta)\right\rangle$, and is obtained efficiently as $\hat{\Sigma}_{\omega v} = \left(R^{\mathsf{T}}R\right)^{-1}$, where $R \in \mathbb{R}^{(3+q)\times(3+q)}$ is the Cholesky factor obtained in optimizing Eq. 47.

6.1.2 Optical Flow Templates

An *optical flow template* models the optical flow at each pixel resulting from typical environment structure, for any possible platform velocity. This model defines the optical flow likelihood $p\left(U_j \mid \Lambda_i, \omega, v, \Theta\right)$ introduced in Section 6.1.1. The explanation here is self-contained, but [74] may be referenced for additional information.

The key to superpixel labeling using optical flow templates is that the label Λ_i determines which typical environment structure predicts the optical flow U_j, and thus the labeling problem is simply model selection as derived in Section 6.1.1. Here, Λ_i=*ground* predicts optical flow consistent with a ground plane, Λ_i=*distant* predicts optical flow from distant structure where flow is determined only by camera rotation, Λ_i=*obstacle* predicts obstacle flow for structure nearer to the camera than that expected for the ground plane, and Λ_i=*unknown* predicts zero-mean wide-variance optical flow to identify superpixels that cannot be explained by the learned templates.

Two intuitive points regarding optical flow templates are:

- Optical flow due to rotation is independent of the scene structure, the contribution due to platform translation is the only one that informs the pixel labels.
- For any *constant* environment structure relative to the robot, implicitly captured in an optical flow template, the optical flow is indeed linear in the platform translational velocity v as modeled in Eq. 49. Optical flow is also linear in the platform rotational velocity ω.

With these two points in mind, the optical flow template model predicts the optical flow U_j as

$$U_j = \begin{cases} \varepsilon_{unknown}, & \Lambda_i\text{=}unknown \\ W_{\omega,j}\omega + W_{\Lambda_i,j}v + \varepsilon_{\Lambda_i}, & \Lambda_i\text{=}ground, obstacle \\ W_{\omega,j}\omega + \varepsilon_{distant}, & \Lambda_i\text{=}distant, \end{cases} \tag{49}$$

where $\omega \in \mathbb{R}^3$ is the platform rotational velocity, $v \in \mathbb{R}^q$ is the platform translational velocity, $W_{\omega,j} \in \mathbb{R}^{2\times 3}$ is the optical flow template that generates the optical flow due to platform rotation, and $W_{\Lambda_i} \in \mathbb{R}^{2\times q}$ is one of κ templates that generates the optical flow due to platform translation for a particular environment structure. ε_x denotes Gaussian noise with covariance Σ_x (with 'x' representing one of the labeling classes). The optical flow template model parameters are thus comprised of $\Theta = (W_\omega, W_1, \Sigma_1, \ldots, W_\kappa, \Sigma_\kappa)$. Here, q is the dimensionality of translational velocity accounting for non-holonomic constraints. Thus for the car-steering robot used in our experiments, $q = 1$.

6.2 Learning Optical Flow Templates from Unlabeled Video

We now wish to learn the optical flow templates used for labeling superpixels in Section 6.1.1 from recorded video. No velocity estimates, camera calibration, or hand-labeling are required to learn the templates – the learning algorithm is almost entirely unsupervised.

The input to the learning algorithm is two videos – one while arbitrarily rotating the camera (hand-held is sufficient) in various directions, and one while the robot drives safely

in its typical environment. In practice, using the separate rotation video improves the quality of the learned templates by preventing translational flow from appearing in the rotational flow template W_ω, and *vice versa*. Even without the separate rotation video, with increasing amounts of training data that is unbiased in turning direction, this confusion vanishes.

The learning algorithm is also a variational method that updates, in turn, the probability vectors of the pixel labels $\tilde{\Lambda}_{t,j}$ for every *pixel* (not superpixel) j in frame t, the Gaussian velocity density on ω_t, v_t for every frame, and the optical flow templates W_ω and W_k for each template k.

The optical flow templates are updated by maximizing the expected log-likelihood of the templates in the generative model in Figure 17, conditioning on all other unknowns fixed at their estimates from the previous iteration. Unlike in the inference algorithm, in learning it suffices to condition on fixed values for ω, v instead of using the Gaussian density,

$$\begin{aligned} W_{\omega,1..\kappa} &= \arg\max_{W_{\omega,1..\kappa}} \left\langle \log p\left(W_{\omega,1..\kappa} \mid I^{txy}_{1..T}, (\Lambda, \hat{\omega}, \hat{v})_{1..T}\right)\right\rangle \\ &= \arg\max_{W_{\omega,1..\kappa}} \left\langle \sum_t \log p\left(I^{txy}_t \mid (\Lambda, \hat{\omega}, \hat{v})_t, \Theta\right)\right\rangle + \log p\left(W_{\omega,1..\kappa}\right), \end{aligned} \tag{50}$$

where $W_{\omega,1..\kappa}$ is all optical flow templates W_ω and W_k for $k \in \{1..\kappa\}$, and T is the number of frames in the training dataset. The term

$$p\left(I^{txy}_t \mid (\Lambda, \hat{\omega}, \hat{v})_t, \Theta\right) = \prod_j \mathcal{N}\left(I^t_{t,j}; -I^{xy}_{t,j}\left(W_{\omega,j}\hat{\omega}_t + W_{\Lambda_j,j}\hat{v}_t\right), \eta_j\right) \tag{51}$$

is identical to Eq. 48 just with appropriate subscripts.

The updates for the label probabilities $\tilde{\Lambda}_{t,j}$ and velocity mean $\hat{\omega}_t, \hat{v}_t$ are identical to those for the inference algorithm in Eqs. 43 and 47 (the notation of those update equations may then be interpreted with each superpixel containing only one pixel each), except that the priors for learning are defined as follows:

- The label priors $p\left(\Lambda_{t,j} \mid \Theta\right)$ are almost the same as during inference, Section 6.1.1.1, except that for the frames that are part of the rotation-only video *ground* receives zero probability, and the *obstacle* class probability is always zero for learning.

- For the velocity prior $p(\omega, v)$ in Eq. 47, we use a Gaussian with fairly large diagonal covariance $\operatorname{diag}\left(\sigma^2_{\omega v}\right)$, with $\sigma_{\omega v} = 10^4$. This constrains the scale ambiguity between velocity magnitude and optical flow magnitude.

- For the template prior $p\left(W_{\omega,1..\kappa}\right)$ in Eq. 50, we use a gentle 4-connected smoothness prior:

$$p\left(W_{\omega,1..\kappa}\right) = \prod_{x \in \{\omega,1..\kappa\}} \prod_j \prod_{n \in N(j)} \mathcal{N}\left(W_{x,j} - W_{x,n}, \Sigma_b\right),$$

  where $N(j)$ is the two neighboring pixels of j to the right and below, and $\Sigma_b = \operatorname{diag}\left(\sigma^2_b\right)$, with $\sigma_b = 2$ in our experiments.

The learning is not highly sensitive to the parameters, $p\left(\Lambda_{t,j} \mid \Theta\right)$, $\sigma_{\omega v}$, σ_i, and σ_b, the values we used here should also be appropriate for most other datasets. A summary of all parameter values used in our experiments is provided in Table 1 of the Experiments and Results section. While in Chapter 3 we estimated the optical flow standard deviations, in this case, the marginalization of the optical flow and presence of the pixel intensity noise with standard deviation σ_i would make such an estimation more complicated, and we were not able to find a straightforward solution.

Over multiple iterations, W_ω converges to the rotational flow fields and W_{ground} to the translational flow fields. Intuitively, the following points are responsible for the maximum-likelihood state of this optimization being the desired solution:

- Within each label class, the templates identify a subspace $[W_\omega\, W_\Lambda]$ in the training data, represented as a rank-deficient Gaussian. Thus, the maximum-likelihood solution occurs when the Gaussian is indeed aligned with this subspace in the data.

- The separation of the rotational and translational flow fields occurs because the rotational flow fields contribute to all label classes whereas the translational flow fields contribute to only non-*distant* classes. This means that the Gaussian prediction of the optical flow U_j has smaller variance and thus higher probability if the *distant* class is chosen to label pixels that are well-explained by rotation only, as opposed to labeling pixels as *ground* class whose flow is explained equally well by only rotational components. Due to this effect, while the prior for *distant*-only labeling in the rotation-only video ensures clean separation of rotational and translational flow, it is unnecessary for unbiased training data.

The learning does not estimate a template $W_{obstacle}$ for the obstacle class. Instead, we leverage the fact that the contribution of translational flow of static obstacles is in the same *direction* as translational flow for any other structure (and still includes the same rotational flow component as any other structure), and that we are interested in identifying obstacles that are closer to the robot than the ground plane. Thus, we set the obstacle template to be the same as the ground plane template but with slightly larger magnitude, $W_{obstacle} = \alpha W_{ground}$, with $\alpha = 1.1$ in our experiments.

One important implementation note is that when solving the least-squares update for the templates, the full system Jacobian is too large to fit in memory. Thus instead, during each iteration we accumulate the system Hessian, updating it with each successive frame. In our implementation each learning iteration thus requires one pass through the data on disk, with label updates, velocity updates, and template Hessian updates all performed as each frame is read.

6.3 Experiments and Results

6.3.1 Sensitivity to Parameters

Most of the parameters in our method do not affect the results much and are thus require little to no tuning, though the inference algorithm is fairly sensitive to appropriate choices of the optical flow covariance Σ_k and the pixel intensity stardard deviation σ_I. We tuned

Figure 18: Our platform is a high-speed mobile robot, a modified 1/8-scale radio-controlled car approximately 50 cm in length with all power, computing, computing, data storage, and sensors on-board. For our experiments we use the front-looking camera sensor, which has a high-distortion 110° FOV lens. The robot was operated at approximately 2m/s during data collection, and the camera framerate is 100Hz.

these by running the inference algorithm several times with a range of values and choosing those that yielded the best results according to visual inspection.

The parameters we list here are typically usable on any other dataset without further tuning, though one needs to scale the flow covariance Σ_k according to resolution and field-of-view. Table 1 lists all of the parameters used in our experiments.

6.3.2 Convergence

Our method uses two iterative algorithms as described in Sections 6.1 and 6.2, so convergence speed and susceptibility to local minima are of interest. We were not able to mathematically prove any bounds or convergence guarantees, but in practice both algorithms converge quickly and in all our experiments have not been susceptible to local minima.

A convergence proof to probabilistic principal components analysis may be found in Tipping and Bishop [86], and our algorithms are built off of this. The key differentiating aspect related to convergence is the introduction of per-pixel indicator variables in our method. In practice, our algorithms converge unless there is an overwhelming portion of outliers in a frame, in which case the indicator variables converge on indicating that the outliers are in fact the inliers, and vice versa.

The inference algorithm that labels superpixels online (Section 6.1) converges very quickly. The labels typically reach very close to their final values within 3 iterations, and fully converge to tight tolerance within 10 iterations. The learning algorithm takes longer, but as it is an offline step, this is not a concern. It typically converges within 30 iterations.

6.3.3 Experimental Platform and Setup

We collected learning and evaluation videos from the front-looking camera of a small, high-speed mobile robot, shown and described in Figure 18, with which to evaluate our method. Each video was collected while driving the robot manually around an oval test track. In the *training* videos the track was free of hand-placed obstacles, though the track has tufts of grass, and there were structures and objects near to the track that we expected to be outliers for the learning algorithm. In the *evaluation* videos, we placed objects on the track with

Description	Symbol	Value
Prior for *unknown* class	$p\left(\Lambda_{t,j}=unknown\right)$	0.05
Pixel intensity noise (img. intensities in $[0,1]$)	σ_I	0.02
Optical flow prediction noise for *unknown* class	$\Sigma_{unknown}$	$\mathrm{diag}\left(1.0\,\mathrm{pix}\right)^2$
Optical flow prediction noise for other classes	$\Sigma_{\omega,1..\kappa}$	$\mathrm{diag}\left(0.35\,\mathrm{pix}\right)^2$
Learning velocity prior	$\sigma_{\omega v}$	10^4
Learning template smoothness	σ_b	2
Obstacle template scaling	α	1.1
Superpixel average size		$100\,\mathrm{pix}^2$
Iterations for inference		3

Table 1: Parameters selected for our experiments.

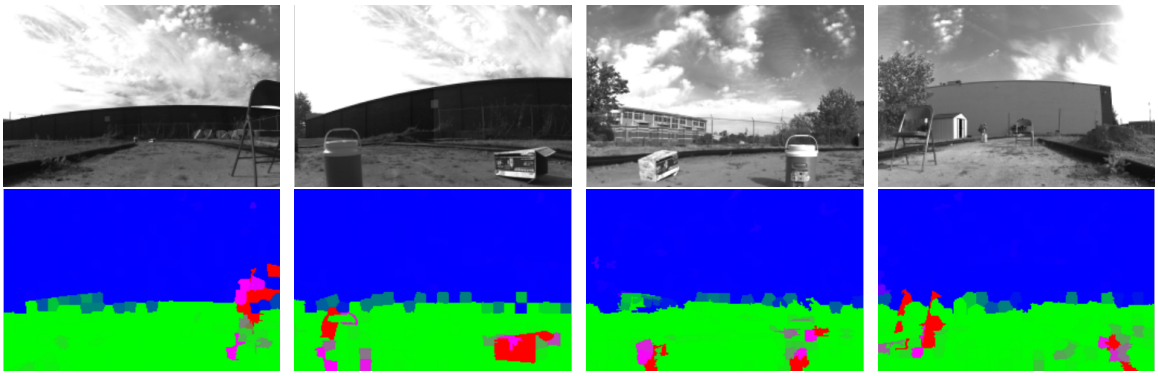


Figure 19: Successful labeling results, see Section 6.3.4 for observations and explanations.

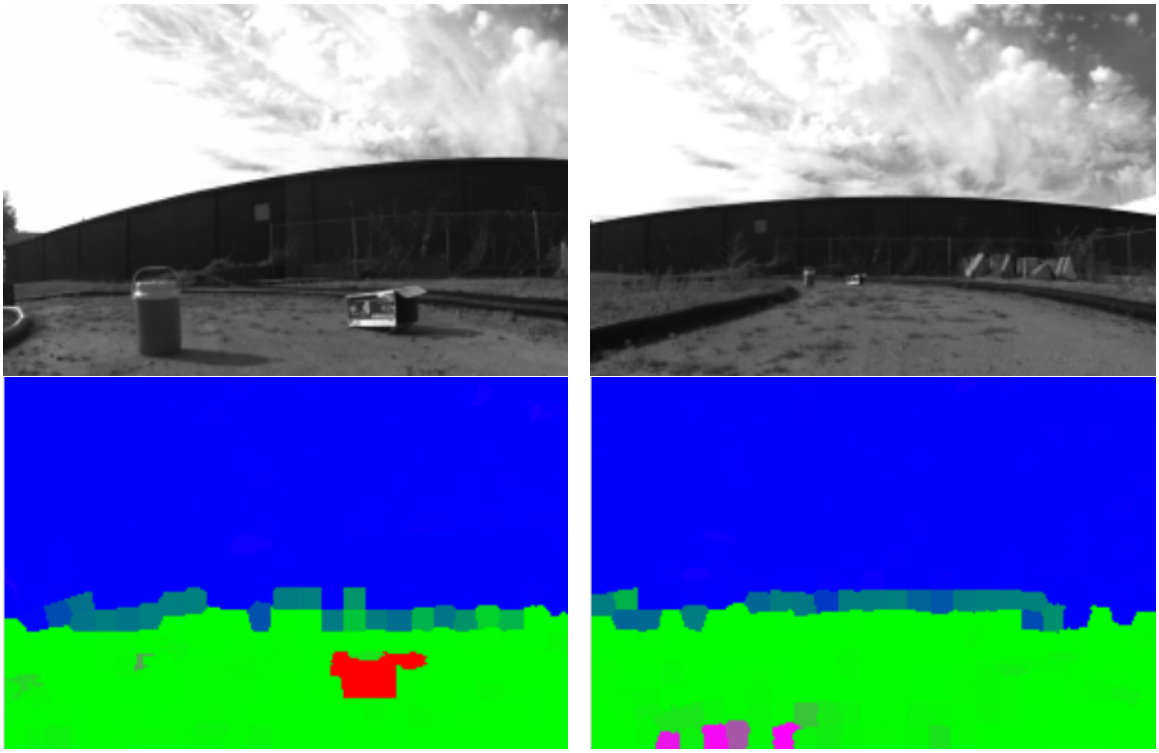


Figure 20: Two types of failure for our method, see Section 6.3.4 for observations and explanations.

the intent that they should be detected as obstacles by our inference algorithm.

6.3.4 Qualitative Results

The optical flow templates learned by our method are shown in Figures 16d and 16e. The rotational template is shown as three flow-fields, whose linear combinations under the robot rotational velocity form the contribution to optical flow from rotation. The translational template is a single flow-field because our non-holonomic robot has only one degree-of-freedom for velocity, "forward velocity" in the body frame. The use of these templates for labeling and their learning was described in Sections 6.1 and 6.2.

Examples of successful labeling results are shown in Figure 19. Typically, boundaries of objects are detected more strongly than smooth-texture regions within objects, a result that is expected since smooth-texture regions provide little-to-no information to optical flow. These results also show that *ground* and *distant* structure are well-distinguished, with distant structure often being sky.

Another observation to make is that obstacles are sometimes classified as *unknown*. The underlying reason for this phenomenon is that the definition of the *obstacle* template as a scaling in magnitude of the *ground* template is exact only for a particular depth-change due to an obstacle. The *unknown* class thus captures regions whose flow is not explained by the *ground* model or the particular choice of *obstacle* model. Improving the modeling of the *obstacle* template is part of our future work, but presently we note that the union of the *obstacle* and *unknown* labels is a good indicator of the presence of obstacles.

Figure 20 shows the types of failure cases that occur with our method. The first shows

Superpixel labeling rate	*obstacle* ∪ *unknown* as obstacle	Only *obstacle* as obstacle
True positive rate	0.4195	0.1385
False positive rate	0.0170	0.0118

Table 2: Quantitative results using hand-labeled ground truth. See Section 6.3.5 for explanation of these statistics.

that small objects directly at the focus-of-expansion of the image motion cannot be detected because image motion at that point is very close to zero. During forward camera motion, the focus-of-expansion is at the image center, but in this example the robot is turning left, which moves the focus-of-expansion towards the left. The second example shows a few superpixels of false positives detected at the bottom of the image. This occurs when the robot speed is very large relative to the camera frame-rate, and the image spatio-temporal gradients are no longer a good linear approximation of the video. The image motion is largest, of course, at the bottom of the image where the ground is close to the camera and the translation vector is less co-incident with the camera rays. One remedy for this is to warp the image successively to improve the linear approximation, and another remedy is simply to increase the frame rate if the camera is capable.

An important metric for robot navigation is the range at which obstacles are detected. Most obstacles are initially detected several meters from the robot. Large obstacles are typically detected up to 10m and small or thin obstacles may be detected only 2m away. We were unfortunately unable to gather more detailed quantitative data on the detection range. The detection range depends upon a number of factors, and optimizing the detection range is a topic of our future work.

Our current implementation in C++ runs at $30\,\text{frames/s}$ on 256×256 images and at $100\,\text{frames/s}$ on 128×128 images, though there is currently obvious room for code optimization. This is several times faster than [74] due to our direct inference from spatiotemporal gradients. Our code and datasets are available online at [will be uploaded in time for camera-ready version].

6.3.5 Superpixel Labeling Accuracy

To quantitatively our method, we measured superpixel true positive and false negative labeling rates against hand-labeled ground truth on the dataset described in Section 6.3.3, as shown in Table 2. These rates are calculated as

$$\text{True positive rate} = \frac{\text{Superpixels correctly labeled obstacle}}{\text{Superpixels that are actually obstacle}}$$

$$\text{False positive rate} = \frac{\text{Superpixels incorrectly labeled obstacle}}{\text{Superpixels that are actually clear}}$$

Intuitively, the true positive rate is the fraction of obstacle superpixels correctly labeled, and the false positive rate is the fraction of clear superpixels incorrectly labeled.

Additionally, each rate in Table 2 interprets the output of our method in two ways. "*obstacle* $\cup$ *unknown* as obstacle" counts both *obstacle* and *unknown* labels as obstacle, as is motivated by the explanation in Section 6.3.4, while "Only *obstacle* as obstacle" counts only *obstacle* labels as obstacle.

While superpixel labeling accuracy rates around 40% may seem low, this does not mean that only half of the obstacles are detected. Smooth texture regions on objects are often not detected while boundaries and textured regions on the same objects are. In fact, all obstacles in the dataset are detected but at varying distances, as described in Section 6.3.4.

6.4 Summary

Towards the goal of autonomous obstacle avoidance for mobile robots, we have presented a method for superpixel labeling using optical flow templates. The argument we make is that optical flow provides a rich source of information that complements image appearance and point clouds in determining traversability. While much past work uses optical flow towards traversability in a heuristic manner, the method we present here instead classifies according to several optical flow templates that are specific to the typical environment shape. We make large improvements over prior work using optical flow templates in accuracy and efficiency due to labeling directly from spatiotemporal gradients and improved modeling of optical flow from obstacles. We also extend optical flow template methods to arbitrary camera optics without the need to calibrate the camera, by learning these templates from unlabeled video. Our results demonstrate successful obstacle detection in an outdoor mobile robot dataset.

CHAPTER VII

CONCLUSIONS

In this thesis we have introduced optical flow templates, which encode a learned relationship between robot motion, scene structure, and optical flow. Optical flow templates combine classical optical flow linearity relationships with modern probabilistic reasoning, allowing them to be applied to long, fairly unconstrained outdoor mobile robot sequences to semantically label pixels and image regions.

In contrast to state-of-the-art methods that require a calibrated camera, the methods we develop here apply to uncalibrated imaging systems with arbitrary optics. Requiring a camera that can be calibrated introduces calibration challenges and places limitations on the types of camera optics that may be used. Wide-angle lenses, systems with mirrors, and multiple cameras all require different calibration models and can be difficult or impossible to calibrate at all.

The applications we consider and evaluate in this thesis are egomotion estimation and semantic superpixel labeling from optical flow. We develop practical algorithms for both that are computationally efficient and able to run in real time. These algorithms exhibit accuracy comparable to other algorithms that are already used for autonomous robot navigation.

In closing we provide a very high-level description of the lessons learned in this thesis. First, we are able to leverage 3D information without building point clouds, by instead considering the linear relationships that exist because we are working with motion though a 3D scene. Second, optical flow works well for egomotion estimation and superpixel labeling, but not for environment shape classification.

We now review the claims and contributions of this thesis.

7.1 Review of Claims

Here we will briefly review the claims of this thesis and the experimental results presented in earlier chapters that support them:

Learned optical flow subspaces are an effective tool for inferring robot egomotion and motion anomalies from cameras with arbitrary optics, in scenarios exhibiting depth regularity. (Chapter 3) We applied our method for learning optical flow subspaces to an urban outdoor mobile robot dataset and successfully used the learned subspace to estimate egomotion with comparable accuracy to wheel odometry.

Selecting among multiple optical flow subspaces allows classification of coarse environment shapes from cameras with arbitrary optics, though this information is of limited usefulness to robot navigation. (Chapter 4) We learned multiple subspaces corresponding to environment shapes on indoor and outdoor mobile robot datasets, and were

able to successfully classify between these shapes using observed optical flow. However, we also found that the results were extremely sensitive to camera orientation, and ran into considerable challenges extending classification to continuously-varying environment shapes. Thus we decided to instead move ahead with superpixel labeling.

Optical flow templates are an effective tool for semantically labeling superpixels as "ground plane", "distant", or "obstacle" from optical flow measured in a camera with arbitrary optics, and this information is useful for autonomous robot navigation. (Chapter 5) We introduced optical flow templates as an optical flow model that encodes multiple types of environment structure and allows labeling pixels or image regions according to which structure they exhibit. Specifically, we label superpixels as "ground", "distant", and "obstacle" on an outdoor urban driving dataset. We found that results were usually accurate for obstacles outside of the optical flow focus of expansion, but depended heavily on the accuracy of the computed optical flow.

Accuracy, versatility, and computational efficiency of optical flow templates is improved by several adaptations informed by our experiences working with optical flow in this thesis. (Chapter 6) We learned optical flow templates from video data instead of calculating them, which extends our superpixel labeling method to work with uncalibrated imaging systems with arbitrary optics. Additionally, we perform both learning and inference directly from spatiotemporal gradients instead of precomputing optical flow, which improves labeling accuracy and improves computational efficiency.

7.2 Future Work

In order to quantitatively evaluate the usefulness of optical flow templates for autonomous navigation, an autonomous navigation system that is informed by our superpixel labels is needed. Doing so would open up research directions into the practical application of fast optical flow methods with generalized optics into autonomous navigation. Furthermore, it would evaluate the utility of our methods in real-world navigation tasks.

Additionally, while the goal of thesis was to understand what information can be obtained using optical flow alone, the best possible scene understanding system should leverage both optical flow information and image appearance information. We believe that optical flow is a strong complementary source of information to image appearance. Ideally, combining optical flow and image appearance would produce more accurate results than either method can produce alone. As reasoning for this, consider that optical flow labels are most accurate close to the robot where motion parallax is largest, whereas image appearance can propagate the information from nearby structure further into the distance where appearance is the same but parallax is imperceptible.

Another way in which image appearance may compliment optical flow is via recent methods (reviewed in Chapter 2) that classify scene depth using image appearance. This scene depth estimate could inform the calculation or learning of optical flow templates, which themselves depend on scene depth. Yet another possibility would be to train a joint

classifier to label image regions on both image appearance and optical flow or even image spatiotemporal gradients.

Finally, given that optical flow template methods are capable of interpreting video data where the scene structure is consistent with that of the learned templates but otherwise identifies structure as "unknown", optical flow template methods could be used as part of a two-tier perception system. Most of a scene can usually be understood using optical flow templates and regions that cannot be understood can be passed onto a more computationally intensive system, which for example could build 3D point clouds only for those small regions.

Bibliography

[1] G. Mori J. Malik A. A. Efros, A. C. Berg. Recognizing Action at a Distance. In *Intl. Conf. on Computer Vision (ICCV)*, pages 726–733, 2003.

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–82, November 2012. ISSN 1939-3539.

[3] G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:384–401, 1985.

[4] M. Agrawal and K. Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive gps. In *IEEE International Conference on Pattern Recognition (ICPR'06)*, 2006.

[5] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A Database and Evaluation Methodology for Optical Flow. *International Journal of Computer Vision*, 92(1):1–31, November 2010. ISSN 0920-5691.

[6] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *European Conference on Computer Vision*, volume 588, pages 237–252. Springer, 1992.

[7] Michael J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, January 1996.

[8] M.J. Brooks, W. Chojnacki, and L. Baumela. Determining the egomotion of an uncalibrated camera from instantaneous optical flow. *Journal of the Optical Society of America A*, 14(10):2670–2677, 1997.

[9] Gabriel J. Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. Segmentation and recognition using structure from motion point clouds. In *Proceedings of the European Conference on Computer Vision*, 2008.

[10] Anna R. Bruss and Bert-hold K. P. Horn. Passive navigation. *Computer Vision, Graphics, and Image Processing*, 21:3–20, 1983.

[11] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa. A robust visual odometry and precipice detection system using consumer-grade single vision. In *IEEE International Conference on Robotics and Automation, Proceedings*, 2005.

[12] Stefan Carlsson and Jan-Olof Eklundh. Object Detection Using Model Based Prediction and Motion Parallax. *European Conference on Computer Vision*, pages 297—-306, 1990.

[13] Joseph Conroy, Gregory Gremillion, Badri Ranganathan, and J. Sean Humbert. Implementation of wide-field integration of optic flow for autonomous quadrotor navigation. *Autonomous Robots*, 27(3):189–198, August 2009. ISSN 0929-5593.

[14] James W Davis and Aaron F Bobick. The Representation and Recognition of Action Using Temporal Templates. *IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[15] F. De la Torre and M.J. Black. Robust principal component analysis for computer vision. *International Conference on Computer Vision (ICCV'01)*, 1:362–369, 2001.

[16] Frank Dellaert. Factor graphs and GTSAM: A hands-on introduction. Technical Report GT-RIM-CP&R-2012-002, Georgia Institute of Technology, 2012.

[17] Andrew Duchon, William H Warren, and Leslie Pack Kaelbling. Ecological Robotics: Controlling Behavior with Optic Flow. In *International Joint Conference on Artificial Intelligence*, 1995.

[18] W Enkelmann. Obstacle Detection by Evaluation of Optical Flow Fields from Image Sequences. *European Conference on Computer Vision*, pages 134—-138, 1990.

[19] Jay Farrell. *Aided Navigation: GPS with High Rate Sensors*. McGraw Hill Professional, 2008. ISBN 0071642668.

[20] D.J. Fleet, M.J. Black, Y. Yacoob, and A.D. Jepson. Design and Use of Linear Models for Image Motion Analysis. *International Journal of Computer Vision*, 36 (3):171–193, 2000.

[21] Alex Flint, David Murray, and Ian Reid. Manhattan scene understanding using monocular, stereo, and 3d features. In *Proceedings of the International Conference on Computer Vision*, 2011.

[22] E. Frazzoli, M.A. Dahleh, and E. Feron. A hybrid control architecture for aggressive maneuvering of autonomous helicopters. In *Proceedings of the IEEE Conference on Decision and Control*, volume 3, pages 2471–2476, 1999.

[23] Andreas Geiger, Martin Lauer, and Raquel Urtasun. A generative model for 3D urban scene understanding from movable platforms. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1945–1952, June 2011. ISBN 978-1-4577-0394-2.

[24] A. Giachetti, M. Campani, and V. Torre. The use of optical flow for road navigation. *IEEE Transactions on Robotics and Automation*, 14(1):34–48, 1998. ISSN 1042296X.

[25] James J Gibson. Visually controlled locomotion and visual orientation in animals. *British journal of psychology*, 49:182–194, April 1958. ISSN 0007-1269.

[26] Michael D. Grossberg and Shree K. Nayar. The raxel imaging model and ray-based calibration. *International Journal of Computer Vision*, 61(2):119–137, 2005.

[27] Raia Hadsell, Pierre Sermanet, Jan Ben, Ayse Naz Erkan, Jeff Han, Matthew K Grimes, Sumit Chopra, Yury Sulsky, Beat Flepp, Urs Muller, and Yann LeCun. Online Learning for Offroad Robots: Using Spatial Label Propagation to Learn Long-Range Traversability. *Robotics: Science and Systems (RSS)*, 11:32, 2007.

[28] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[29] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the Fourth Alvey Vision Conference*, volume 15, 1988.

[30] Richard I. Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000. ISBN 0521623049.

[31] David Heeger and Allan Jepson. Subspace methods for recovering rigid motion I: Algorithm and implementation. *International Journal of Computer Vision*, 7(2): 95–117, 1992.

[32] DJ Heeger and Allan Jepson. Visual Perception of Three-Dimensional Motion. *Neural Computation*, 2:127–137, 1990.

[33] D. Hoiem, A.A. Efros, and M. Hebert. Geometric context from a single image. *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 654–661 Vol. 1, 2005.

[34] Berthold K. Horn and Brian G. Schunck. Determining Optical Flow. In James J. Pearson, editor, *1981 Technical Symposium East*, pages 319–331. International Society for Optics and Photonics, November 1981.

[35] B.K.P. Horn and B.G. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17(1-3):185–203, 1981.

[36] A. Huertas, L. Matthies, and A. Rankin. Stereo-based tree traversability analysis for autonomous off-road navigation. In *IEEE Workshops on Application of Computer Vision, WACV/MOTIONS'05*, volume 1, pages 210–217, 2005.

[37] A. Huertas, L. Matthies, and A. Rankin. Stereo-Based Tree Traversability Analysis for Autonomous Off-Road Navigation. *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*, pages 210–217, January 2005.

[38] B. Hutchings, B. Nelson, S. West, and R. Curtis. Optical flow on the ambric massively parallel processor array (MPPA). In *IEEE Symposium on Field Programmable Custom Computing Machines (FCCM)*, pages 141–148, 2009.

[39] Andrew M. Hyslop and J. Sean Humbert. Autonomous Navigation in Three-Dimensional Urban Environments Using Wide-Field Integration of Optic Flow. *Journal of Guidance, Control, and Dynamics*, 33(1):147–159, January 2010. ISSN 0731-5090.

[40] M. Irani, B. Rousso, and S. Peleg. Detecting and tracking multiple moving objects using temporal integration. In *European Conference on Computer Vision*, pages 282–287. Springer, 1992.

[41] M. Irani, P. Anandan, and M. Cohen. Direct recovery of planar-parallax from multiple frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 (11):1528–1534, November 2002. ISSN 0162-8828.

[42] Michal Irani. Multi-frame correspondence estimation using subspace constraints. *International Journal of Computer Vision*, 48(3):173–194, 2002.

[43] Michal Irani and P. Anandan. Direct recovery of planar-parallax from multiple frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(11): 1528–1534, November 2002.

[44] A.D. Jepson and D.J. Heeger. A fast subspace algorithm for recovering rigid motion. In *Visual Motion, 1991., Proceedings of the IEEE Workshop on*, pages 124–131, Oct 1991.

[45] A.D. Jepson and D.J. Heeger. Linear subspace methods for recovering translational direction. In *Proceedings of the 1991 York Conference on Spatial Vision in Humans and Robots*, 1993.

[46] Allan D. Jepson and David J. Heeger. Subspace methods for recovering rigid motion II: Theory. Technical Report RBCV-TR-90-36, University of Toronto, 1990.

[47] K. Kanatani. 3-d interpretation of optical flow by renormalization. *International Journal of Computer Vision*, 11(3):267–282, 1993.

[48] K. Kanatani. Renormalization for motion analysis: Statistically optimal algorithm. *IEICE Transactions on Information and Systems*, 77(11):1233–1239, 1994.

[49] Yasir Khan, Philippe Komma, and Andreas Zell. High resolution visual terrain classification for outdoor robots. In *IEEE ICCV Workshop on Challenges and Opportunities in Robot Perception*, 2011.

[50] Yasir Niaz Khan, Philippe Komma, and Andreas Zell. High resolution visual terrain classification for outdoor robots, 2011.

[51] Dongshin Kim, Sang Min Oh, and James M. Rehg. Traversability classification for UGV navigation: a comparison of patch and superpixel representations. *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3166–3173, October 2007.

[52] J.F. Lalonde, N. Vandapel, D.F. Huber, and M. Hebert. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of Field Robotics*, 23(10):839–861, 2006.

[53] M. Lehrer, M. V. Srinivasan, S. W. Zhang, and G. A. Horridge. Motion cues provide the bee's visual world with a third dimension. *Nature*, 332(6162):356–357, March 1988. ISSN 0028-0836.

[54] David Lieb, Andrew Lookingbill, and Sebastian Thrun. Adaptive Road Following using Self-Supervised Learning and Reverse Optical Flow. *Robotics: Science and Systems (RSS)*, pages 273—-280, 2005.

[55] HC Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[56] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, volume 3, 1981.

[57] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, volume 3, 1981.

[58] Q.T. Luong and O.D. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1):43–75, 1996.

[59] Y. Ma, J. Košecká, and S. Sastry. Linear differential algorithm for motion recovery: A geometric approach. *International Journal of Computer Vision*, 36(1):71–89, 2000.

[60] Larry Matthies. *Dynamic Stereo Vision*. PhD thesis, Carnegie Mellon University, 1989.

[61] Larry Matthies. Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation. *International Journal of Computer Vision*, 8(1):71–91, July 1992. ISSN 0920-5691.

[62] SJ Maybank. The angular velocity associated with the optical flowfield arising from motion through a rigid environment. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 401(1821):317–326, 1985.

[63] Stephen John Maybank. Algorithm for analysing optical flow based on the least-squares method. *Image and Vision Computing*, 4(1):38 – 42, 1986. ISSN 0262-8856.

[64] J. Michels, A. Saxena, and A.Y. Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proceedings of the 22nd International Conference on Machine learning*, pages 593–600. ACM, 2005.

[65] Oscar Mozos and Wolfram Burgard. Supervised Learning of Topological Maps using Semantic Information Extracted from Range Data. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2772–2777. IEEE, October 2006. ISBN 1-4244-0258-1.

[66] H.H. Nagel and B. Neumann. On 3-d reconstruction from two perspective views. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 2. Citeseer, 1981.

[67] J. Neumann, C. Fermuller, and Y. Aloimonos. Polydioptric camera design and 3d motion estimation. In *Computer Vision and Pattern Recognition, Proceedings of the IEEE Computer Society Conference on*, volume 2, 2003.

[68] Kai Ni and Frank Dellaert. Stereo tracking and three-point/one-point algorithms – a robust approach in visual odometry. In *International Conference on Image Processing*, 2006.

[69] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *IEEE Conference on Computer Vision and Patterns Recognition*, volume 1, pages 652–659, 2004.

[70] Navid Nourani-Vatani, Paulo V. K. Borges, Jonathan M. Roberts, and Mandyam V. Srinivasan. Topological localization using optical flow descriptors. In *Proceedings of the 1st IEEE Workshop on Challenges and Opportunities in Robotic Perception, with ICCV'2011*, 2011.

[71] Robert Pless. Discrete and differential two-view constraints for general imaging systems. In *Omnidirectional Vision, 2002. Proceedings. Third Workshop on*, pages 53–59, 2002.

[72] S. Hussain Raza, Matthias Grundmann, and Irfan Essa. Geometric Context from Videos. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3081–3088, June 2013.

[73] A. Rieder, B. Southall, G. Salgian, R. Mandelbaum, H. Herman, P. Rander, and T. Stentz. Stereo perception on an off-road vehicle. *IEEE Intelligent Vehicle Symposium*, pages 221–226, 2002.

[74] Richard Roberts and Frank Dellaert. Optical Flow Templates for Superpixel Labeling in Autonomous Robot Navigation. *5th Workshop on Planning Perception and Navigation for Intelligent Vehicles (PPNIV13)*, 2013.

[75] Richard Roberts, Hai Nguyen, Niyant Krishnamurthi, and Tucker Balch. Memory-based learning for visual odometry. Submitted to the International Conference on Robotics and Automation, 2008.

[76] Richard Roberts, Christian Potthast, and Frank Dellaert. Learning general optical flow subspaces for egomotion estimation and detection of motion anomalies. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[77] Javier Sánchez, Enric Meinhardt-Llopis, and Gabriele Facciolo. TV-L1 Optical Flow Estimation. *Image Processing On Line*, 2012.

[78] HS Sawhney. 3D geometry from planar parallax. *Conference on Computer Vision and Pattern Recognition*, 1994.

[79] T. Schouwenaars, B. Mettler, E. Feron, and J.P. How. Robust motion planning using a maneuver automation with built-in uncertainties. In *Proceedings of the American Control Conference*, volume 3, pages 2211–2216, 2003.

[80] C. Siagian and L. Itti. Rapid biologically-inspired scene classification using features shared with visual attention. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):300–312, 2007.

[81] Mandyam Srinivasan, Saul Thurrowgood, and Dean Soccol. Competent Vision and Navigation Systems. *IEEE Robotics & Automation Magazine*, 16(3):59–71, September 2009. ISSN 1070-9932.

[82] P. Sturgess, K. Alahari, L. Ladicky, and P. Torr. Combining appearance and structure from motion features for road scene understanding. In *Proceedings of the British Machine Vision Conference*, 2009.

[83] Muralidhara Subbarao and Allen M. Waxman. Closed from solutions to image flow equations for planar surfaces in motion. *Computer Vision, Graphics, and Image Processing*, 36(2-3):208 – 228, 1986. ISSN 0734-189X.

[84] Tina Tian, Carlo Tomasi, and David Heeger. Comparison of approaches to egomotion computation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 315–320, 1996.

[85] Michael E. Tipping and Christopher M. Bishop. Mixtures of Probabilistic Principal Component Analyzers. *Neural Computation*, 11(2):443–482, 1999. ISSN 0899-7667.

[86] Michael E. Tipping and Christopher M. Bishop. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, August 1999. ISSN 1369-7412.

[87] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[88] C. Tomasi and J. Shi. Direction of heading from image deformations. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 422–422. Citeseer, 1993.

[89] G. Toscani and O. Faugeras. Structure and motion from two noisy perspective views. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 221 – 227, mar 1987.

[90] Grace Tsai, Changhai Xu, Jingen Liu, and Benjamin Kuipers. Real-time indoor scene understanding using bayesian filtering with motion cues. In *Proceedings of the International Conference on Computer Vision*, 2011.

[91] R.Y. Tsai and T.S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(1):13–27, 1984.

[92] T. Viéville and O. D. Faugeras. The first order expansion of motion equations in the uncalibrated case. *Computer Vision and Image Understanding*, 64(1):128 – 146, 1996. ISSN 1077-3142.

[93] Hui Wang, Kui Yuan, Wei Zou, and Qingrui Zhou. Visual odometry based on locally planar ground assumption. In *Information Acquisition, 2005 IEEE International Conference on*, page 6, 2005.

[94] A.M. Waxman, B. Kamgar-Parsi, and M. Subbarao. Closed-form solutions to image flow equations for 3d structure and motion. *International Journal of Computer Vision*, 1(3):239–258, 1988.

[95] J. Weng, N. Ahuja, and T.S. Huang. Closed-form solution+maximum likelihood: a robust approach to motion and structure estimation. In *Computer Vision and Pattern Recognition, 1988. Proceedings CVPR '88., Computer Society Conference on*, pages 381 –386, jun 1988.

[96] J. Weng, TS Huang, and N. Ahuja. Motion and structure from two perspective views: algorithms, error analysis, and error estimation. *IEEE transactions on pattern analysis and machine intelligence*, 11(5):451–476, 1989.

[97] Anqi Xu and Gregory Dudek. Trust-driven interactive visual navigation for autonomous robots. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3922–3929, May 2012.

[98] C Zach, T Pock, and H Bischof. A duality based approach for realtime TV-L 1 optical flow. *Ann. Symp. German Association Patt. Recogn*, 2007.

[99] Hongyi Zhang, Andreas Geiger, and Raquel Urtasun. Understanding High-Level Semantics by Modeling Traffic Patterns. *International Conference on Computer Vision (ICCV)*, 2013.

[100] T. Zhang and C. Tomasi. On the consistency of instantaneous rigid motion estimation. *International Journal of Computer Vision*, 46(1):51–79, 2002.

[101] X. Zhuang, T.S. Huang, N. Ahuja, and R.M. Haralick. A simplified linear optic flow-motion algorithm. *Computer Vision, Graphics, and Image Processing*, 42(3): 334–344, 1988.