Inaugural-Dissertation

zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich-Mathematischen Gesamtfakultät
der
Ruprecht–Karls–Universität
Heidelberg

vorgelegt von
Master of Mathematics and Computer Science Andreas Neufeld
aus Ramenskoje (Russland)

Tag der mündlichen Prüfung: .....

Variational Approaches for Motion and Structure from Monocular Video

Betreuer: Prof. Dr. Christoph Schnörr
Prof. Dr. Björn Ommer

Zusammenfassung

Das Schätzen von Bewegung und räumlicher Struktur aus Bildfolgen sind grundlegende Probleme in der Bildverarbeitung. Diese Problemstellungen sind noch aktuell, obwohl die ersten Methoden bereits vor mehreren Jahrzehnten veröffentlicht wurden. Wir präsentieren neue Verfahren zur Bewegungs- und Strukturschätzung für das autonome Fahren. Ein autonomes Auto braucht genaue Kenntnis über seine Umgebung, da eine Fehleinschätzung gravierende Konsequenzen haben kann. Speziell behandeln wir monokulare Verfahren, bei denen nur eine Kamera auf dem Fahrzeug verfügbar ist.

Bildfolgen aus dem Straßenverkehr sind für das Schätzen von Bewegung besonders herausfordernd. Durch die hohe Geschwindigkeit ist die Bewegung sehr groß, die Lichtverhältnisse sind nicht stabil und es kann Verfälschungen geben durch Reflektionen und wetterbedingte Störungen. Wir stellen neue diskrete Verfahren zur Berechnung des optischen Flusses vor, welche probabilistische graphische Modelle für den optischen Fluss definieren.

In dem ersten Ansatz wählen wir einige Stellen im Referenzbild aus, und vergleichen diese mit dem zweiten Bild. Die besten Korrespondenzen, welche auch zu einer monokularen Bewegung passen, werden als Kandidaten für ein graphisches Modell ausgewählt. In einem weiteren Verfahren, vergleichen wir alle Stellen im Referenzbild, lösen das graphische Modell jedoch nicht direkt, sondern approximieren es mit einer Sequenz von kleineren Modellen.

Da wir eine monokulare Bildsequenz haben, müssen neben der Szene auch die Kamerapositionen rekonstruiert werden. Bedingt durch die projektive Geometrie gibt es blinde Flecken auf den Bildern und Mehrdeutigkeiten bezüglich der Skalierung, welche es bei einem System mit mehreren Kameras nicht gibt.

Wir stellen zwei Verfahren für die Strukturschätzung vor. Das erste Verfahren bestimmt den optimalen Weg einer Kamera bezüglich einer Energiefunktion aus optischem Fluss und Tiefenschätzungen. Das zweite Verfahren schätzt die Bewegung der Kamera und planare Szenenbeschreibung aus einem einzigen Flussfeld.

Wir evaluieren die Verfahren auf verschiedenen realen und künstlichen Daten. Für die Evaluation der planaren Rekonstruktionen haben wir einen eigenen Datensatz erstellt, welcher Tiefen- und Ebeneninformationen enthält.

Abstract

Motion and structure estimation are elementary problems of computer vision. These are active areas of research, even though the first methods were proposed several decades ago. We develop new approaches for motion and structure estimation for autonomous driving. An autonomous vehicle requires an accurate model of its environment, wrong decisions made by an autonomous car can have severe consequences. We assume the monocular setup, where only a single camera is mounted on the car.

Outdoor traffic sequences are challenging for optical flow estimation. The high speed of the car causes large displacements in the optical flow field, the lighting conditions are unstable and there can be strong distortions due to reflections and difficult weather conditions. We propose new discrete methods, which determine optical flow as optimal configuration of probabilistic graphical models.

The first approach selects sparse locations in the reference frame, and matches them across the second image. The best correspondences, which match constraints from a multiple view configuration, are considered motion vectors in a graphical model. In a second approach, we solve for dense optical flow by approximating the original infeasible graphical model with a sequence of reduced models.

The monocular configuration poses challenges to the estimation of scene structure, camera positions and scene parameters need to be estimated jointly. The geometry of multiple views creates blind spots in the images, and adds a scale ambiguity, which both to not exist in a setup with multiple cameras.

We propose two methods for structure estimation. The first approach determines the energy optimal camera track, given optical flow and depth observations. A further approach estimates camera motion and a piecewise planar scene description jointly from a single optical flow field. The scene description contains depth and plane normal information.

We evaluate our approaches for motion and structure estimation on different real world and rendered datasets. In addition to evaluation on publicly available evaluation data, we evaluate on a new rendered dataset with ground truth plane normals.

Acknowledgments

During the PhD there were many challenging and uncertain ways to explore, of which a few lead to a dead end. Completing this work would not have been possible without the support and advice by many people, to whom I am very grateful.

The first person to mention is my main supervisor Prof. Christoph Schnörr, who introduced me to the field of image processing, and in large parts to mathematical modeling and optimization in general. Through his supervision I gained very much personal experience during my PhD, in addition to the subject specific knowledge.

Furthermore, I would like to thank my colleagues at Heidelberg University, in particular the members of the Image & Pattern Analysis Group. Special thanks go to Florian Becker and Frank Lenzen, who proposed interesting new ideas and were always prepared for questions and discussions. And I thank Jörg Kappes and Bogdan Savchynskyy for the discussions and very helpful feedback, as well as Evelyn Wilhelm and Barbara Werner for helping me with administrative issues.

Finally, I would like to thank my parents, Ludmila and Alexander Neufeld and my sister Helene Gehrmann for their support and encouragement.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1. Motivation

Autonomous vehicles are about to revolutionize individual mobility in the near future. Prototypes are driving autonomously already, these are equipped with multiple cameras and other sensors such as GPS, laser, radar and ultrasound sensors. Detection of drivable area, traffic signs, traffic lights, other cars and pedestrians is vitally important for autonomous driving, which is why many redundant sensors are applied. Cameras have the advantage that they are passive, they record light from the environment without any active interaction, while radar and ultrasonic sensors emit their own signal and have difficulties with interference and other noise.

A configuration with multiple cameras compared to a single camera provides more information, but needs to be calibrated. If the calibration breaks down due to deformation, or outage of individual cameras, the whole system does not operate correctly anymore. Additionally, any part inside the car comes at a cost, consumes power and space. Therefore, we investigate the monocular setup, which consists of a single forward facing camera.

Human drivers proof that car driving based on visual cues is possible, but computer algorithms still are much worse in image abstraction than humans. Motion and structure estimation are intuitive and instantaneous operations for us, whereas these have been active computer vision research areas in the last decades. Algorithms, however, have the advantage that they can make faster and more accurate decisions than humans, and are not influenced by distraction. We believe that autonomous cars will outperform humans as drivers in the future, and reduce the number of car accidents drastically while increasing comfort for the passengers at the same time.

This work consist of two major parts, motion estimation under realistic outdoor conditions and structure from motion estimation. Outdoor traffic sequences are challenging for motion estimation, due to difficult lighting conditions and fast vehicle motion. The large range of depth in an outdoor environment induces a non linearly curved motion field, even if the underlying scene is perfectly flat and simple. Traditional motion estimation methods struggle with this type of motion, which leads to errors in scene reconstruction.

The monocular camera setup provides less information than a calibrated stereoscopic or multiple view camera systems. We investigate the limits, and propose new methods for monocular structure estimation.

1.2. Related Work

There is a large collection of literature about motion and structure estimation, which are fundamental problems of computer vision. Point correspondences provide an abstraction of raw images, which is required by many computer vision systems. Motion can be defined differently, depending on the task it is designed to solve. The *tracking* problem consists of the motion of a small set of interest points (or *keypoints*) over several frames. If only two frames are used, tracking is usually referred to as *feature matching*, while pixel wise dense correspondence between two frames is denoted *optical flow*.

Similarly, the formulation of structure estimation is application dependent. Early algorithms for monocular structure estimation operate on a small set of keypoints, and aim at accurate camera localization. Later methods introduces dense depth maps for each view, yielding three dimensional models of the environment.

1.2.1. Feature Matching and Optical Flow. While both feature matching and optical flow relate to image correspondence, they are addressed with very different means. Feature matching is performed only on few locations in the image, therefore rich features can be computed and matched efficiently. One of the most widely used feature descriptor is the *scale invariant feature transform (SIFT)* [**55**], which computes image keypoints consisting of location, scale and orientation, as well as a 128 dimensional feature descriptor. Figure 1.1 shows SIFT keypoints for two example images from a traffic sequence. An approximate version of SIFT is given by the *speeded-up robust features (SURF)* [**8**]. SIFT and SURF have motivated further image descriptors, for example *binary robust independent elementary features (BRIEF)* [**19**] and *oriented fast and rotated brief (ORB)* [**75**]. *Histogram of oriented gradients (HOG)* features [**22**] were originally proposed for object recognition, but have been applied in the context of point correspondence as well.

In contrast to sparse feature matching, dense optical flow cannot rely on distinct keypoints with precomputed scale and orientation, but has to use a simpler matching criterion. The first optical flow methods were proposed by Horn and Schunck [**40**] and Lucas and Kanade [**56**], and both are based on the *brightness constancy assumption (BCA)*, assuming that image intensity does not change between two frames along the flow field. This condition alone does not define a unique optical flow field, since it only provides one constraint on two variables.

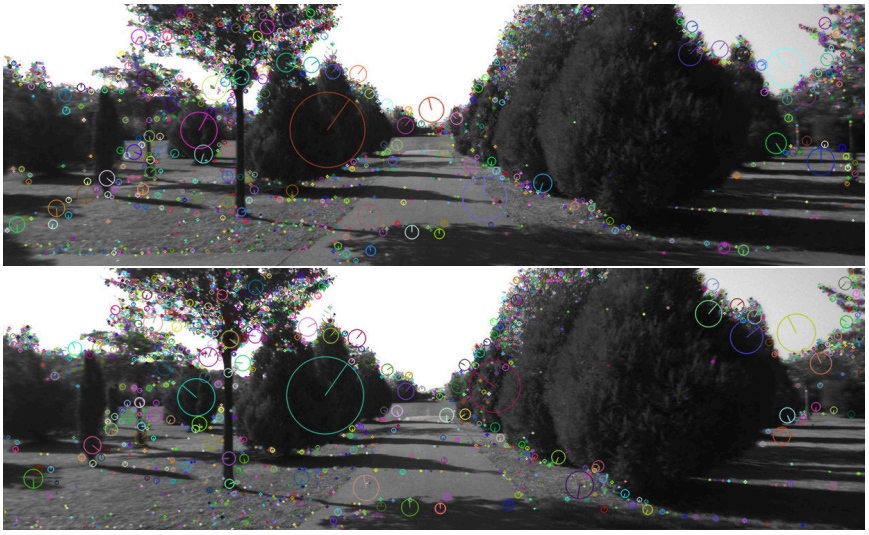
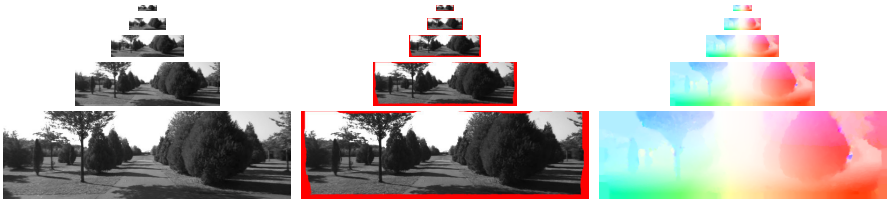


FIGURE 1.1. Detected SIFT keypoints on two consecutive frames from the KITTI dataset. The keypoints provide orientation and scale, which allows the feature extractor to compensate distortions which are due to rotation and scaling. The SIFT keypoint detector is very robust in practice, and can compensate strong perspective distortions. The approach in [**2**] aligns random tourist pictures using SIFT and computes a three dimensional model of the sights, which shows the robustness of the SIFT matching algorithm. Since the number of keypoints is small, compared to the number of pixels in the image, rich features can be extracted and matched in real time on a single core CPU, or on a mobile device.

Therefore, Horn and Schunck add a quadratic regularity term, which yields an energy function with unique minimum. Lucas and Kanade impose implicit regularity through the assumption, that neighboring pixels will have similar optical flow, which does not define a dense flow field in general.

In practical outdoor conditions, the BCA does not hold in many cases due to the complex lighting conditions in outdoor environments and fast movement of the camera. Additionally, the quadratic regularity term does not preserve sharp motion discontinuities and tends to oversmooth the flow field. The original formulation of Horn and Schunck has been basis for many later approaches, where data and regularity terms have been replaced by more robust energy terms [**73, 90**] or data terms based on feature descriptors such as SIFT and HOG [**22, 16, 52**]. The authors of [**25**] estimate illumination changes explicitly based on training data.

A Further limitation of the BCA is, that it can only be enforced if the optical flow field is very small, ideally on a subpixel level, which is called the *aperture problem* [**10**]. Therefore, an image pyramid with



(a) reference frame (b) warped frame (c) optical flow

FIGURE 1.2. Pyramid setup for dense optical flow computation. The optical flow field is scaled up and refined when proceeding to the next layer in the image pyramid. The difference between the reference and the warped frame is used for the data energy, and there usually is a regularity energy of the optical flow field as well. The red parts of the warped frame are undefined, because the flow vectors point outside of the second image, these regions are either ignored, or extrapolated during optical flow estimation. Due to the hierarchical nature of the approach, objects may be lost on a small scale, and cannot be recovered at a later stage.

hierarchical refinement of the estimated optical flow field is required, as depicted in Figure 1.2. There is no guarantee that the optical flow field minimizes the respective energy on full image scale. Some methods improve the initialization by patch matching [**93, 6**] and by additional information, such as edge detection [**74**].

Several optical flow methods have been proposed, which avoid the image pyramid by modeling optical flow as discrete optimization problem, where the labels are given by displacement vectors in each pixel. The main drawback of discrete optical flow lies in the large label space, depending on the maximum flow vector norm there may be thousands of displacement candidates, combined with the large number of pixels in the image, we get an intractable graphical model. The *DiscreteFlow* approach by Menze et al. [**60**] reduces the number of displacement candidates using patch matching by two to three orders of magnitude, which turns the discrete model into tractable size. Other methods extend the optical flow problem with occlusion setection [**5, 3**] or segmentation of the flow field into layers with independent motion [**79, 78, 84, 85**].

Even though continuous methods using the heuristic image pyramid do not guarantee global optimality, they perform very well in practice. However, these methods have been optimized over several decades, while discrete optical flow methods are relatively new. With increasing computational resources, and improvements in discrete inference methods, we see potential in discrete optical flow methods, and focus on them.

1.2.2. Structure Estimation. Monocular structure estimation, in particular *simultaneous localization and mapping (SLAM)* consist of estimation of camera poses and three dimensional scene parameters, which are determined simultaneously. Several approaches to monocular structure computation have been presented, which address the task with different techniques and scene representations.

The *parallel tracking and mapping (PTAM)* method [**43**] tracks a set of keypoints over several frames, and computes the camera location as well as point dpeth values for each view. In the *Dense Tracking and Mapping (DTAM)* approach [**67**], camera locations are computed first, followed by a dense depth map estimation and camera location refinement for each frame. While PTAM and DTAM are primarily designed for reconstruction of small objects in a confined space, *Large Scale Direct Monocular SLAM (LSD-SLAM)* [**28**] performs localization and mapping in a large scale outdoor environment. Loop closure ensures accurate alignment of the point cloud, if the same location is visited repeatedly.

The authors of [**46**] use a voxel representation in a sparse octree datastructure, allowing them to represent large outdoor environments. They combine their approach with a semantic assignment of the scene into *road*, *car*, *building*, *pedestrian* and similar classes. Recent approaches apply *convolutional neural networks (CNN's)* for monocular scene reconstruction. The CNN-SLAM approach [**87**] integrates depth predictions from a single view into a SLAM architecture, jointly with applying a CNN for semantic labeling. Similarly, the authors of [**27**] estimate depth and plane normal information from a single view using a pre-trained CNN.

A stereo setup with two calibrated cameras provides more information than a single camera system. In the monocular case depth information needs to be accumulated over time, while stereo matching corresponds to direct depth observation. The approach presented in [**68**] fits *stixels* to the estimated depth map. Stixels are vertical segments, which cover the entire vertical span of an object, they reflect the assumption that objects in a traffic scene most likely are positioned on the ground. Similarly, [**53**] assigns the classes *sky*, *building*, *vehicle or pedestrian* and *street* from top to bottom in fixed order for each pixel column of the reference image. The approach in [**47**] performs stereo matching and semantic labeling jointly with a graphical model.

Related to stereo estimation is the problem of *scene flow* estimation, where both camera and scene movement are reconstructed from a sequence of stereo image pairs. In a static scene, the scene movement is zero, or inverse to the camera movement, depending on the reference coordinate system. In scenes with dynamic objects, however, scene flow defines the motion of each object in the scene. The approach by Vogel et al. [**91**] estimates rigid motion and plane parameters on image

superpixels. Menze et al. [**59**] compute a segmentation of dynamic objects. Cars are recognized by comparison with detailed computer aided design (CAD) models.

1.3. Contribution

We present new approaches to discrete optical flow estimation and structure computation from monocular video. Both tasks are addressed with variational methods, we define probabilistic graphical models for discrete optical flow, and apply optimal control and continuous energy minimization to the structure computation task.

Our proposed discrete optical flow approaches are based on exhaustive matching in a search window, which yields very large, intractable label spaces. In a sparse approach, we only match distinct regions in the reference frame, and keep very few displacement candidates for consideration in the graphical model. This strategy reduces the size of the graphical model, but it also removes valuable information from the system. Therefore, we present a dense approach, which reduces the large dense model by clustering of flow vectors, and applying a reduction technique based on probabilistic divergence. The reduction procedure is iterated, until a flow field is defined.

For the task of structure computation, we propose two approaches with different objectives. We present a new minimum energy filter, which computes the energy optimal trajectory of the camera from optical flow and depth observations. Additionally, we present an energy minimization framework, which computes camera motion and a scene description using planar segments from two frames only. In order to evaluate the piecewise planar scene description, we generated a dataset with ground truth normal information available.

1.4. Organization

In chapter 2, we present the mathematical principles behind optimization. Starting with an introduction to convex analysis, we proceed to continuous optimization algorithms for convex functions. A brief introduction to optimal control concludes the chapter. Chapter 3 on mathematical models introduces probabilistic graphical models and manifolds. We discuss different approaches to inference on graphical models, and demonstrate how discrete inference relates to convex optimization. We define manifolds, tangent spaces and optimization on functions, which are defined on manifolds.

Chapter 4 explains the concepts of computer vision, that we need for our algorithms. We introduce epipolar geometry, which explains the geometry of point correspondences between multiple views of the same scene, and how point correspondences are used for structure computation.

We explain variational optical flow estimation, in particular discrete optical flow in chapter 5. Discrete optical flow methods mainly struggle with the large label space of optical flow models. We present related methods, and propose new strategies for reducing the models to a tractable size.

In chapter 6, we present filtering techniques for monocular scene reconstruction, and compare the advantages of filtering to batch approaches. We present our minimum energy filter framework, and the two frame scene estimation approach. Chapter 7 concludes the thesis.

CHAPTER 2

Mathematical Optimization

Physical processes obey the principle of minimum potential energy. Similarly, mathematical problems can be expressed as energy minimization or variational problems in many cases. Depending on the formulation of the objective function, different techniques need to be applied, of which we present a brief overview in this chapter.

A major property of an objective function lies in convexity. Convex optimization problems are convenient from the viewpoint of optimization, since there is no risk of converging to a local instead of a global minimum. From duality theory, we get optimality bounds on the current state. There are many optimization methods available for convex problems, which can incorporate constraints and do not require the energy to be continuous or differentiable. However, many practical optimization tasks are non convex. Those can be relaxed to a convex problem, or approximated by a sequence of convex optimization problems.

Optimal control describes the scenario, where we cannot influence the state directly, but only through control variables which satisfy a system of differential equations.

We start with an introduction to convex analysis in section 2.1, in particular linear programming, followed by a survey of optimization algorithms suitable for convex minimization in section 2.2. We distinguish continuous optimization techniques ad splitting methods, which do not require smoothness of the objective function. In section 2.3 we present Pontryagin's maximum principle, defining optimality conditions of the control variables in optimal control problems.

2.1. Convex Analysis

We will briefly summarize the key concepts of convex analysis, that provide the basis for optimization of convex functions. Convex functions generally provide convergence to a global optimum, while for non convex functions convergence can only be achieved to a local optimum, if convergence can be achieved at all.

Before we define convex functions, we define convexity for sets.

Definition 2.1.1 (convex set). A set $C \in \mathbb{R}^n$ is convex, if the line connecting any two points in C is fully contained in C as well,

$$(1-\lambda)x + \lambda y \in C, \quad \forall x, y \in C, \quad \forall \lambda \in [0,1].$$

EXAMPLE 2.1.1. Basic convex sets.

halfspaces: A hyperplane divides $\mathbb{R}^n$ into two halfspaces

$$H^+_{p,\alpha} = \{x \in \mathbb{R}^n \ : \ \langle p, x\rangle \geq \alpha\}$$
$$H^-_{p,\alpha} = \{x \in \mathbb{R}^n \ : \ \langle p, x\rangle \leq \alpha\}$$

polyhedral set: The intersection of finitely many halfspaces is called polyhedral set. The halfspace equations can be written in a matrix A and vector b,

$$P = \{x \in \mathbb{R}^n \ : \ Ax \leq b\}$$

Bounded polyhedral sets are called *polytopes*. Note that an equality constraint can be expressed as the intersection of two inequality constraints.

convex hull: Given a set of points $\{x_i \in \mathbb{R}^n\}$, the set of all linear combinations forms a convex set,

$$H = \sum_{i=1}^{d} \lambda_i x_i, \quad \sum_{i=1}^{d} \lambda_i = 1, \quad \lambda_i \geq 0, \quad \forall i.$$

probability simplex: The $(n-1$ dimensional) probability simplex represents the set of all discrete probability distributions in $\mathbb{R}^n$,

$$\Delta_n = \{x \in \mathbb{R}^n \ : \ x \geq 0, \quad \langle \mathbb{1}, x\rangle = 1\}.$$

convex cones: A set $K \in \mathbb{R}^n$ is called a convex cone, if

$$\lambda x \in K, \quad \lambda \in \mathbb{R}^+, \quad \forall x \in K, \quad \text{and } K \text{ is convex.}$$

The polar cone K^* of cone K is given by

$$K^* = \{y \in \mathbb{R}^n \ : \ \langle y, x\rangle \leq 0, \quad \forall x \in K\}. \tag{2.1}$$

The normal cone $N^C(x)$ to a convex set C is defined as

$$N^C(x) = \{v \in \mathbb{R}^n \ : \ \langle v, y - x\rangle \leq 0, \quad \forall y \in C\}.$$

The normal cone represents all hyperplanes intersecting with C in point x, but no other point in C. The polar cone can be expressed as normal cone of K at zero, $K^* = N^K(0)$.

DEFINITION 2.1.2 (convex function). A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex, if and only if $\forall x, y \in \mathbb{R}^n$

$$f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y), \quad \forall \lambda \in [0, 1].$$

There is a direct connection between convex sets and convex functions, a function $f : \mathbb{R}^n \to \mathbb{R}$ is convex, if and only if its *epigraph*

$$\operatorname{epi} f = \{(x, \alpha) \in \mathbb{R}^n \times \mathbb{R} : f(x) \leq \alpha\}$$

is a convex set. *Level sets* of convex functions are convex sets as well,

$$\operatorname{lev}_\alpha f = \{x \in \mathbb{R}^n : f(x) \leq \alpha\}.$$

The *directional derivative* of function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as

$$Df(x)[v] = \lim_{h \to 0} \frac{f(x + hv) - f(x)}{h}.$$

The gradient $\nabla f(x)$ is given by

$$Df(x)[v] = \langle \nabla f(x), v \rangle, \quad \forall v \in \mathbb{R}^n.$$

For any convex and differentiable function f, the affine approximation around x_0 provides a lower bound of f,

$$f(x_0) + \langle \nabla f(x_0), x - x_0 \rangle < f(x). \tag{2.2}$$

If f is convex, the lower bound is valid on the entire domain, while in the non convex case it holds only in a small neighborhood of x_0.

Equation (2.2) motivates the definition of the subdifferential for non differentiable functions

$$\partial f(x_0) = \{p \in \mathbb{R}^n \ : \ \langle p, x - x_0 \rangle \leq f(x) - f(x_0)\}. \tag{2.3}$$

the elements of the subdifferential are called *subgradients*. Subgradients can be interpreted as slopes of affine functions supporting the original function f. If f is differentiable at x_0, then $\partial f(x_0) = \{\nabla f(x_0)\}$.

THEOREM 2.1.1 (Fermat's rule). *A convex function f is minimized at $\hat{x}$, if and only if*

$$0 \in \partial f(\hat{x}).$$

PROOF. From the definition of the subgradient in eq. (2.3), we see that $0 \in \partial f(x_0)$ if and only if $f(x_0) < f(x), \ \forall x \in \mathbb{R}^n$. □

2.1.1. Duality. A convex function can be described as a set of points, or as supremum of a set of hyperplanes. This dual perspective on the same function has many applications in mathematical optimization. The Fenchel conjugate of function $f : \mathbb{R}^n \to \mathbb{R}$ is given by

$$f^* : \mathbb{R}^n \to \mathbb{R} : \quad f^*(p) = \sup_{x \in \mathbb{R}^n} \{\langle p, x \rangle - f(x)\}.$$

The *biconjugate* of f is given by

$$f^{**}(x) = (f^*)^*(x) = \sup_{p \in \mathbb{R}^n} \{\langle p, x \rangle - f^*(p)\}.$$

The dual function f^* is always convex, hence f^{**} is convex as well. f^{**} is the tightest convex approximation of f.

LEMMA 2.1.2 (Fenchel's inequality). *From the definition of the conjugate function, we see*

$$f^*(p) + f(x) \geq \langle p, x \rangle.$$

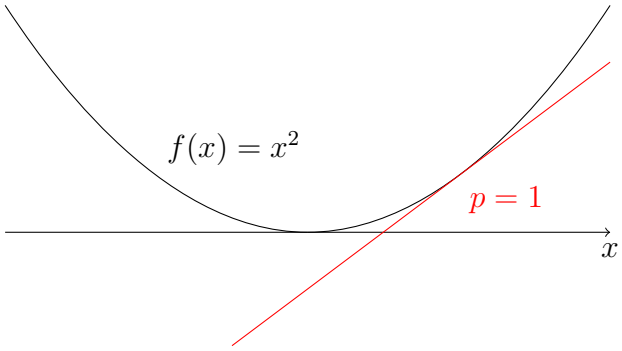


FIGURE 2.1. The dual function $f^*(p)$ represents the optimal (negative) shift of hyperplane $\langle p, x\rangle$, so that the epigraph of f is supported. Any convex epigraph can be described as infinite set of points, or as infinite intersection of hyperplanes.

Furthermore

$$f^{**}(x) \leq f(x),$$

with equality if and only if f is convex. Hence, there is a one to one correspondence between f and its dual f^* for convex functions, which is called the *Legendre-Fenchel Transform*.

The duality between points and bounding planes can be illustrated by looking at the dual of the *indicator function* on a convex set $C \subset \mathbb{R}^n$,

$$\delta_C(x) = \begin{cases} 0 & x \in C \\ \infty & \text{otherwise} \end{cases}.$$

The dual is given by

$$\delta_C^*(p) = \sup_{x \in \mathbb{R}^n} \{\langle p, x\rangle - \delta_C(x)\} = \sup_{x \in C} \langle p, x\rangle =: \sigma_C(p),$$

which is the *support function*. $\sigma_C(p)$ represents the bounding hyperplane in direction of p,

$$C \subset \{x \in \mathbb{R}^n \mid \langle p, x\rangle \leq \sigma_C(p)\}.$$

The dual of the support function is again the indicator function $\delta_C(p)$,

$$\sigma_C^*(p) = \sup_{x \in \mathbb{R}^n} \{\langle p, x\rangle - \sigma_C(x)\} = \begin{cases} 0 & p \in C \\ \infty & \text{otherwise} \end{cases}.$$

If $C = K$ is a cone, recalling the definition of the polar cone (2.1), we observe that the support function on the polar cone K^* is identical to the indicator function on K $\delta_K(x)$,

$$\sigma_{K^*}(x) = \sup_{y \in K^*} \langle y, x\rangle = \begin{cases} 0 & x \in K \\ \infty & \text{otherwise} \end{cases}$$

Vector norms can be expressed as support functions, i.e. the 1-norm can be expressed as

$$\|x\|_1 = \sigma_{[-1,1]^n}(x).$$

Hence the dual is given by

$$\|.\|_1^*(p) = \delta_{[-1,1]^n}(p)$$

2.1.2. Lagrangian Multipliers. Many function we want to minimize in practice are of the form

$$E(x) = f(x) + g(G(x)), \tag{2.4}$$

where $f(x) : \mathbb{R}^n \to \mathbb{R}$ may represent the actual model, and $g : \mathbb{R}^m \to \mathbb{R}$, $G : \mathbb{R}^n \to \mathbb{R}^m$ can enforce constraints or assumptions on regularity.

We perturb eq. (2.4) with auxiliary variables u,

$$\phi(x, u) = f(x) + g(G(x) + u),$$

and introduce

$$v(u) = \inf_x \phi(x, u).$$

Now we form the dual

$$\begin{aligned} v^*(p) &= \sup_{u\in\mathbb{R}^m} \langle p, u\rangle - \inf_{x\in\mathbb{R}^n} f(x) + g(G(x) + u) \\ &= \sup_{u\in\mathbb{R}^m, x\in\mathbb{R}^n} -f(x) + \langle p, u\rangle - g(G(x) + u) \\ &= -\inf_{x\in\mathbb{R}^n} f(x) + \langle p, G(x)\rangle - g^*(p) \end{aligned}$$

Forming the biconjugate yields

$$\begin{aligned} v^{**}(u) &= \sup_{p\in\mathbb{R}^m} \langle p, u\rangle - v^*(p) \\ &= \sup_{p\in\mathbb{R}^m} \langle p, u\rangle + \inf_{x\in\mathbb{R}^n} f(x) + \langle p, G(x)\rangle - g^*(p) \end{aligned}$$

At $u = 0$, we have

$$v^{**}(0) = \sup_{p\in\mathbb{R}^m} \inf_{x\in\mathbb{R}^n} f(x) + \langle p, G(x)\rangle - g^*(p)$$

From the original objective and the biconjugate, we get the following system of primal and dual objectives,

$$\begin{aligned} (P) \quad & \inf_{x\in\mathbb{R}^n} \sup_{p\in\mathbb{R}^m} f(x) + \langle p, G(x)\rangle - g^*(p) \\ (D) \quad & \sup_{p\in\mathbb{R}^m} \inf_{x\in\mathbb{R}^n} f(x) + \langle p, G(x)\rangle - g^*(p) \end{aligned} \tag{2.5}$$

The function

$$L(x, p) = f(x) + \langle p, G(x)\rangle$$

is called the *Lagrangian* with multiplier p. In many practical cases, $g^*(p)$ will be an indicator function, and can be transformed into constraints on p.

2.1.3. Optimality Conditions. We can obtain verified global optimality from duality theory, since the duality gap between the primal and the dual energies vanishes only at the global optimal solution. Subgradients of primal and dual conjugate functions can be related to each other.

THEOREM 2.1.3 (Inversion Rule for Subgradients). *Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex, then*

$$\hat{p} \in \partial f(\hat{x}) \quad \Leftrightarrow \quad \langle \hat{p}, \hat{x} \rangle = f(\hat{x}) + f^*(\hat{p}). \tag{2.6}$$

Furthermore, if strong duality holds

$$\hat{p} \in \partial f(\hat{x}) \quad \Leftrightarrow \quad \hat{x} \in \partial f^*(\hat{p}). \tag{2.7}$$

PROOF. In order to show (2.6), we rearrange eq. (2.3)

$$\begin{aligned} \hat{p} \in \partial f(x) \quad &\Leftrightarrow \quad f(x) \geq f(\hat{x}) + \langle \hat{p}, x - \hat{x} \rangle, \quad \forall x \in \mathbb{R}^n \\ &\Leftrightarrow \quad f(x) - \langle \hat{p}, x \rangle \geq f(\hat{x}) - \langle \hat{p}, \hat{x} \rangle, \quad \forall x \in \mathbb{R}^n \end{aligned}$$

We may take the infimum with respect to x, remembering that equality can be achieved at $x = \hat{x}$,

$$\Leftrightarrow \quad -f^*(\hat{p}) = f(\hat{x}) - \langle \hat{p}, \hat{x} \rangle,$$

which shows eq. (2.6).

Applying (2.6) to the dual $f^*(p)$, assuming that $f^{**} = f$ yields

$$\begin{aligned} \hat{x} \in \partial f^*(\hat{p}) \quad &\Leftrightarrow \quad \langle \hat{p}, \hat{x} \rangle = f^*(\hat{p}) + f^{**}(\hat{x}) \\ &\Leftrightarrow \quad \langle \hat{p}, \hat{x} \rangle = f^*(\hat{p}) + f(\hat{x}). \end{aligned}$$

The RHS is identical to (2.6), which proofs eq. (2.7). □

Applying the subgradient inversion rule to our system of primal and dual objectives in (2.5), we get the following optimality conditions on x^* and p^*,

$$x^* \in \arg\min_x L(x, p^*), \quad p^* \in \partial g(G(x^*)).$$

COROLLARY 2.1.4 (KKT conditions). *With these optimality conditions, we can derive the* Karush-Kuhn-Tucker (KKT) *conditions. We look at minimization problems of the form*

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} f(x) \\ \textit{subject to } &G(x) \leq 0, \\ &H(x) = 0. \end{aligned}$$

The Lagrangian reads

$$L(x, p, q) = f(x) + \langle p, G(x) \rangle + \langle q, H(x) \rangle,$$

and we have the optimality conditions

$$\begin{aligned} &x^* \in \arg\min_x L(x, p^*, q^*),\ G(x^*) \leq 0,\ H(x^*) = 0 \\ &p^* \geq 0,\ \langle p^*, G(x^*)\rangle = 0. \end{aligned} \tag{2.8}$$

PROOF. The subdifferential of the indicator function on a cone K is the normal cone N^K. In the case above we have

$$\partial \delta_K(G(x)) = N^K(G(x)).$$

Since both $\mathbb{R}_-$ and $\{0\}$ are cones with normal cones $\mathbb{R}_+$ and $\mathbb{R}$, respectively, the result (2.8) follows. □

2.1.4. Linear Programming. We introduce linear programming briefly, since we will encounter it in the context of inference in graphical models. Linear Programming consist of minimizing a linear objective under linear constraints.

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \langle c, x\rangle \\ &\text{s.t. } Ax \leq b \end{aligned}$$

The constraints limit x to a polyhedral set, we define the Lagrangian

$$L(x, p) = \min_{x \in \mathbb{R}^n} \max_{p \in \mathbb{R}^m_-} \langle c, x\rangle + \langle p, b - Ax\rangle.$$

By exchanging the min and max operations and rearranging, we get

$$\max_{p \in \mathbb{R}^m_-} \min_{x \in \mathbb{R}^n} \langle p, b\rangle + \langle c - A^\top p, x\rangle,$$

which corresponds to the dual LP

$$\begin{aligned} &\max_{p \in \mathbb{R}^m} \langle b, p\rangle \\ &\text{s.t. } A^\top p = c,\ p \leq 0. \end{aligned}$$

It can be shown that if the primal LP is feasible and bounded, so is the dual and there is no duality gap. Let x^* and p^* denote the optimal primal and dual solutions, then $\langle p^*, b - Ax^*\rangle$ and $\langle x^*, c - A^\top p^*\rangle$ both have to vanish, which is called *complementarity slackness*.

There are two main approaches to solve linear programs in practice, the simplex and the interior point methods. The simplex algorithm searches the vertices of the feasible set until an optimal solution is found, while the interior point method performs updates within the feasible set. Even though the number of vertices is exponential in the number of defining equations, the simplex method shows similar performance in practice as polynomial-time interior point methods. If in addition the variables have to be integer, we have an integer linear program (ILP), which is an NP-hard problem.

2.2. Continuous Optimization Algorithms

In the following we present different algorithms for different kinds of minimization problems. The algorithms presented in this section all operate in the primal domain without evaluating the dual function and are iterative, variabel x is updated repeatedly until convergence to x^*.

2.2.1. Differentiable Functions. In addition to differentiability, we assume our function $f : \mathbb{R}^n \to \mathbb{R}$ to have a Lipschitz continuous gradient with constant L satisfying

$$|\nabla f(x) - \nabla f(y)| \leq L|x - y|, \quad \forall x, y.$$

Assuming that we know the gradient of the function we aim to minimize, we may move along the direction of maximum descent, which is along the negative gradient, as shown in Algorithm 1. This algorithm

Input : step size $\alpha \in \mathbb{R}$, accuracy ϵ
Output: x^* satisfying $\nabla f(x^*) < \epsilon$
while $\nabla f(x) \geq \epsilon$ **do**
  $x \leftarrow x - \alpha \nabla f(x)$;
end

Algorithm 1: Gradient descent with fixed stepsize.

is simple and straightforward to implement, but also shows slow convergence. Since the stepsize is fixed, the updates are either too small in the beginning, or to large when x is close to the optimum x^*.

Applying a line search method will result in a better choice of updates, and accelerate convergence significantly. The Armillo point is the most distant intersection of the function with a linear expansion around the current position,

$$f_{\text{Armillo}}(y) = f(x) + \langle \alpha \nabla f(x), x + y \rangle, \quad \alpha \in \left(0 \quad \tfrac{1}{2}\right).$$

The Armillo line search method is shown in Algorithm 2. Figure 2.2 illustrates the approach.

Input : $x \in \mathbb{R}^n$, $y \in \mathbb{R}^n$, $\alpha \in (0,\ \frac{1}{2})$, $\beta \in (0,\ 1)$
Output: stepsize t
$t \leftarrow 1$;
while $f(x + ty) > f(x) + \langle \alpha \nabla f(x), x + ty \rangle$ **do**
  $t \leftarrow \beta t$;
end

Algorithm 2: Backtracking Line Search in direction y.

The gradient may be more sensitive to changes in particular directions, so that the steepest descent is not necessarily optimal. But the update direction y needs to be *gradient related*, in order to decrease the value of f in each iteration, $\langle \nabla f, y \rangle < 0$. If we have the (positive

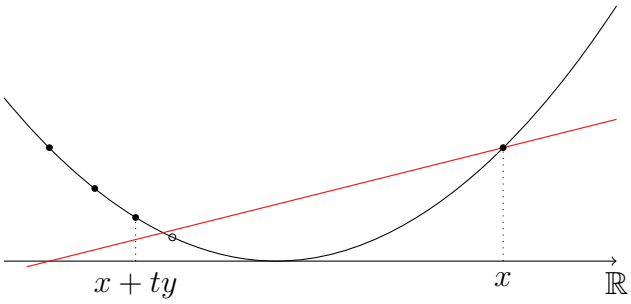


FIGURE 2.2. Armillo Line Search for $f(x) = x^2$ around $x = 1$ with $\alpha = 0.25$ and $\beta = 0.9$. The search direction is $y = -1$, the calculated stepsize is $t = 0.81$.

definite) Hessian available, Newton's method can be applied, as shown in Algorithm 3.

If the second order derivative is available as well, optimization can be improved further. Even if f is twice differentiable, storing the $n \times n$ elements of the Hessian matrix in computer memory may not be feasible. For non-convex f, the Hessian may not be positive definite, so that the update direction is not guaranteed to be gradient related. Negative components of the Hessian would need to be removed, in order to guarantee convergence to a local minimum.

Input : accuracy ϵ
Output: x^* satisfying $\|\nabla f(x^*)\| < \epsilon$
while $\|\nabla f(x)\| \geq \epsilon$ **do**
  $y \leftarrow -(\nabla^2 f(x))^{-1} \nabla f(x)$;
  determine optimal step size t;
  $x \leftarrow x + ty$;
end

Algorithm 3: Newton method.

If we assume the function f to be a quadratic vector norm $\frac{1}{2}\|F(x)\|^2$ (assuming that $F(x)$ is once differentiable), and apply the Newton method, we have the Levenberg-Marquardt algorithm as presented in Algorithm 4. We can find first and second order derivatives, $\nabla F(x) = J$, $\nabla f(x) = J^\top F(x)$ and $\nabla^2 f(x) = J^\top J$. Levenberg-Marquardt is a special case of the *Gauss-Newton* method, where the Hessian matrix is approximated with the Jacobian, $H = J^\top J$.

The Newton method fits a quadratic model to f, and minimizes the model function for the update. But it may be desirable to restrict the maximum update step to radius τ, since the model loses accuracy with increasing distance. The quadratic model function reads

$$m_x(y) = f(x) + \langle \nabla f, x \rangle + \frac{1}{2} \langle x, \mathrm{Hess}(f)x \rangle.$$

Input : accuracy ϵ
Output: x^* satisfying $\|\nabla f(x^*)\| < \epsilon$
while $\|\nabla f(x)\| \geq \epsilon$ **do**
  $y \leftarrow -(J^\top J)^{-1} J^\top F(x)$;
  determine optimal step size t;
  $x \leftarrow x + ty$;
end

Algorithm 4: Levenberg-Marquardt

The trust region size τ is updated at each iteration based on the quotient of the actual and the model energy difference. At iteration k, this quotient is given by

$$\rho = \frac{f(x^k) - f(x^{k+1})}{m_x(0) - m_{x^k}(x^{k+1} - x^k)}. \tag{2.9}$$

ρ determines whether the trust region should be increased or decreased. If ρ is small, the model is not accurate, and the region should be decreased. On the other hand, if ρ is close to one, the model is accurate, and the region should be expanded in order to allow larger updates. Additionally, depending on ρ, an update may be rejected. This approach is called the *trust-region method*, shown in Algorithm 5. Instead of the

Input : accuracy ϵ, threshold ρ_{accept}, maximum size $\hat{\tau}$
Output: x^* satisfying $\|\nabla f(x^*)\| < \epsilon$
while $\|\nabla f(x)\| \geq \epsilon$ **do**
  $y \leftarrow \arg\min_y m_x(y) + \delta_{\|y\|<\tau}(y)$;
  calculate ρ as in eq. (2.9);
  if $\rho < \frac{1}{4}$ **then**
    $\tau \leftarrow \frac{\tau}{4}$;
  end
  else if $\rho > \frac{3}{4}$ *and* $\|y\| = \tau$ **then**
    $\tau \leftarrow \min(2\tau, \hat{\tau})$;
  end
  if $\rho > \rho_{accept}$ **then**
    $x \leftarrow x + y$;
  end
end

Algorithm 5: Trust-Region Method

hard constraint $\|y\| < \tau$, a soft constraint penalizing $\kappa\|y\|^2$ may be utilized.

2.2.2. Splitting Methods. All of the approaches in the previous section require a Lipschitz continuous gradient, thus cannot be used if f is not differentiable or not continuous. Non-differentiable functions

appear very often in image processing, for example as indicator function or *total variation (TV)* term,

$$\mathrm{TV}(u) = \int_{x\in\Omega} \|\nabla u(x)\|_1.$$

Splitting methods can minimize these functions effectively.

We will need the proximal operator, which is defined as

$$\mathrm{Prox}_{\lambda f}(x) = \arg\min_y \left\{ \frac{1}{2}(x-y)^2 + \lambda f(y) \right\}.$$

If f is not continuous, the proximal mapping may still be feasible to evaluate in closed form. For example, if $f = \delta_C$ is the indicator function on convex set C, the proximal map is the orthogonal projection on C.

The basic forward backward splitting, shown in Algorithm 6 method can minimize functions of form

$$f(x) = g(x) + h(x).$$

With g and h both being proper and convex, but only g being differentiable with Lipschitz continuous gradient with constant L. An

Input : $\tau \in (0, 2/L)$
Output: approximate minimizer x^*
repeat
  $x \leftarrow \mathrm{Prox}_{\tau h}(x - \tau\nabla g(x))$;
until *convergence*;

Algorithm 6: Forward Backward Splitting

extension to minimizing a sum of non-differentiable functions $f(x) = g(x) + \sum h_i(x)$ is presented in [**70**], called generalized forward-backward splitting.

If f has the form

$$f(x) = h(x) + g(Ax),$$

the fast primal dual algorithm can be used, shown in Algorithm 7. $h(x)$ needs to be differentiable with Lipschitz continuous gradient with constant L, $g(Ax)$ only needs to be proper and convex. As shown in section 2.1.2, minimizing f corresponds to the saddle point problem

$$\min_{x\in\mathbb{R}^n} \max_{p\in\mathbb{R}^m} \langle Ax, p\rangle + h(x) - g^*(p).$$

For guaranteed convergence, we require $\tau_1\tau_2 L^2 < 1$. If $h(x)$ is strongly convex, and accelerated version is available, where τ_1 and τ_2 are updated on each iteration, thus achieving quadratic instead of linear convergence. Details can be found in [**20**].

If we have the more general problem formulation

$$\begin{aligned} \min_{x,z} \quad & g(x) + h(z) \\ \text{s.t.} \quad & Ax + Bz = c, \end{aligned} \tag{2.10}$$

Input : $\tau_1, \tau_2 > 0$
Output: primal minimizer x^* and dual maximizer p^*
$z \leftarrow x$;
repeat
  $p \leftarrow \text{Prox}_{\tau_1 g^*}(p + \tau_1 A z)$;
  $\tilde{x} \leftarrow x$;
  $x \leftarrow \text{Prox}_{\tau_2 h}(x - \tau_2 A^\top p)$;
  $z \leftarrow 2x - \tilde{x}$;
until *convergence*;

Algorithm 7: Fast Primal Dual Algorithm

we introduce the *augmented Lagrangian*

$$L_\rho(x, z, y) = g(x) + h(z) + \langle y, Ax + Bz - c \rangle + \frac{\rho}{2} \|Ax + Bz - c\|^2. \quad (2.11)$$

The augmented Lagrangian can be minimized using the *Alternating Direction Method of Multipliers (ADMM)*, shown in Algorithm 8. While

Input : $\rho > 0$
Output: approximate optimal point (x^*, z^*) of eq. (2.10)
repeat
  $x \leftarrow \arg\min_x L_\rho(x, z, y)$;
  $z \leftarrow \arg\min_z L_\rho(x, z, y)$;
  $y \leftarrow y + \rho(Ax + Bz - c)$;
until *convergence*;

Algorithm 8: Alternating Direction Method of Multipliers (ADMM)

ADMM can be slow in obtaining the exact solution, it may find a good approximation very quickly [**13**].

2.3. Elements of Optimal Control

In the previous sections on continuous and discrete energy minimization, we assumed that the variable can be freely chosen, and the associated energy is given immediately. In practice we may not be able to choose the state arbitrarily, but only change it according to control variables. Such an optimization task is called *optimal control problem*. We present an optimal control formulation for visual odometry in section 6.1.2.

If we need to optimize the optimal path of a car, the path is subject to acceleration and steering, which are constrained as well. Simplifying the setup to one dimension (the rocket car), the state is given by distance s and velocity v, the control variable is acceleration u. We have state constraints $s(0) = a$, $v(0) = 0$, $s(T) = b$, $v(T) = 0$, where start time $t_0 = 0$ and end time T is unknown. The control variable u is

constrained to the interval $[-1,\ 1]$. The optimal trajectory of the car depends on the criterion we minimize, two possibilities are

$$\min_{u(t)} T \qquad \text{(time optimal)} \tag{2.12a}$$

$$\min_{u(t)} \int_0^T \frac{1}{2} u(t)^2 dt. \qquad \text{(energy optimal)} \tag{2.12b}$$

Definition 2.3.1 (Optimal Control Problem). Find control function $u : [t_0,\ T] \to \mathcal{U} \subset \mathbb{R}^k$ and corresponding state variables $x : [t_0,\ T] \to \mathbb{R}^n$, that satisfy an (ordinary) system of differential equations (ODE)

$$\dot{x} = f(t, x(t), u(t)).$$

There may exist restrictions on $u(t)$ and $x(t)$. In many cases the initial state may be fixed, $x(0) = x_0$, and u may be restricted to an interval, or to a discrete label set. The control variable u needs to be optimized subject to an objective function

$$\mathcal{J}(u) = \Phi(T, x(T)) + \int_{t_0}^T L(x(t), u(t)) dt.$$

Pontryagin's maximum principle [**4**] states the optimality conditions on optimal control variable u, which also depend on the constraints of u and x. Let the constraints be given by

$$\begin{aligned} r(T, x(T)) = 0, \quad & r :\ \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^{m_1} \\ s(T, x(T)) \geq 0 \quad & s :\ \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^{m_2}. \end{aligned}$$

Then with the adjoint variables $\lambda :\ [t_0,\ T] \to \mathbb{R}^n$, $\alpha \in \mathbb{R}^{m_1}$, $\beta \in \mathbb{R}^{m_2}$, we can define the *Hamiltonian* and the *augmented cost function*

$$\begin{aligned} H(x(t), u(t), \lambda(t)) &= -L(x(t), u(t)) + \lambda(t)^\top f(t, x(t), u(t)) \\ \Psi(T, x(T), \alpha, \beta) &= \Phi(T, x(T)) - \alpha^\top r(R, x(T)) - \beta^\top s(T, x(T)). \end{aligned}$$

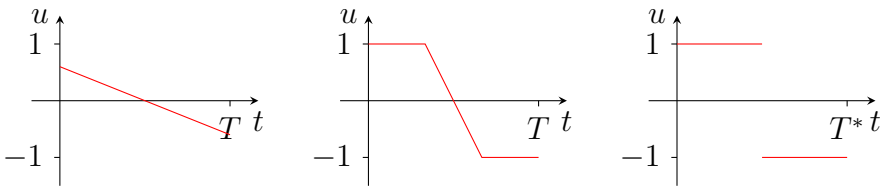
Let (x^*, u^*) be feasible local minimum of the objective function, then

- $\exists$ adjoint variables $\lambda^* :\ [t_0,\ T] \to R^n$, $\alpha^* \in \mathbb{R}^{m_1}$, $\beta^* \in \mathbb{R}^{m_2}$.
- λ^* satisfies the adjoint differential equation

$$\begin{aligned} \dot{\lambda^*} &= -f_x(t, x^*(t), u^*(t)) + L(x^*(t), u^*(t))^\top \\ &= -H_x(x^*(t), u^*(t)). \end{aligned}$$

- The Hamiltonian is constant,

$$H(x^*(t), u^*(t), \lambda^*(t)) \equiv C.$$



(a) large T (b) small T (c) minimum T

FIGURE 2.3. Optimum acceleration $u(t) \in [-1,\ 1]$ with respect to the minimum energy objective (2.12b) and constraints $s(0) = a$, $s(T) = b$, $v(0) = v(T) = 0$. If we choose $T = T^*$ time optimal, there is only one feasible trajectory, which is maximum acceleration followed by maximum deceleration.

- The *transversality conditions* hold,

$$\begin{aligned} \lambda^*(T^*)^\top &= -\Psi_x(T^*, x^*(T^*), \alpha^*, \beta^*) \\ &= -\Phi_x(T^*, x^*(T^*)) + \beta^* s_x(T^*, x^*(T^*)) \\ H(x^*(t), u^*(t), \lambda^*(t)) &= \frac{\partial}{\partial T}\Psi(T^*, x^*(T^*), \alpha^*, \beta^*). \end{aligned}$$

  It follows that $H(x^*(t), u^*(t), \lambda^*(t)) \equiv 0$ if T is not free.

Furthermore, u^* maximizes the Hamiltonian,

$$H(x^*(t), u^*(t), \lambda^*(t)) \geq H(x^*(t), v, \lambda^*(t)), \quad \forall v : [t_0,\ T] \to \mathcal{U}.$$

From these optimality conditions, we can infer the optimal solution of the one dimensional car trajectory mentioned above. Depending on the value of T, compared to the distance $d = b - a$, we get different solutions of the energy optimal case (2.12b), as depicted in Figure 2.3. In most practical examples, we do not get a closed form solution, but need to integrate the corresponding variables numerically.

CHAPTER 3

Mathematical Models

In the previous chapter, we assumed a function or control variable, which are either free or constrained by equality or inequality constraints. Some mathematical models assume the corresponding variables to lie in particular non Euclidean spaces. In particular, we present discrete graphical models and smooth manifolds in this chapter.

Discrete graphical models define a probability for a configuration of a graph with vertices and edges, where each vertex can be assigned a finite number of values. Discrete optimization exploits the structure of the underlying graph, because the discrete minimization problem is NP hard in general. However, particular types of discrete optimization problems can be solved efficiently.

In some cases, our variables cannot have arbitrary values, but are restricted to a manifold- Manifolds define their own differential geometry, which influence the optimization strategy.

We introduce graphical models and the inference problem in section 3.1. Since general inference on graphical models is an NP-hard problem, we present different methods for different types of graph, utilizing the respective graph structure. In some cases, discrete inference can be reformulated as a convex continuous optimization problem. In section 3.2, we introduce manifolds with their respective differential geometries, in order to perform optimization in non Euclidean spaces.

3.1. Probabilistic Graphical Models

Many real world problems are structured. There may be a hierarchical structure between *latent* and *observed* variables, or they are structured as a collection of smaller problems, such as an image segmentation problem, which consists of many individual pixel segmentation problems depending on each other. Probabilistic graphical models can represent such dependencies explicitly, hence offering wide flexibility. However, performing inference on graphical models can be challenging in some cases.

3.1.1. Exponential Models. A probabilistic graphical model assigns a probability to each possible configuration of the variables. For example, the *multivariate Gaussian distribution* $\mathcal{N}(x \mid \mu, \Sigma)$ with mean

μ and covariance Σ assigns a probability to each $x \in \mathbb{R}^n$,

$$p_1(x) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma)}} \exp\left(-\frac{1}{2}\langle x-\mu, \Sigma^{-1}(x-\mu)\rangle\right). \tag{3.1}$$

If we assume $\mu = (\mu_1,\ \mu_2)$ and $\Sigma = \operatorname{diag}(\sigma_1^2,\ \sigma_2^2)$ to be diagonal, we may rewrite the probability above as product of two univariate normal distributions,

$$p_2(x) = \mathcal{N}(x \mid \mu_1, \sigma_1^2)\mathcal{N}(x \mid \mu_2, \sigma_2^2). \tag{3.2}$$

$p_1(x)$ and $p_2(x)$ differ in the structure of the underlying graph, p_2 factorizes into two distributions of simpler form. This factorization into simpler parts is utilized when performing inference on graphical models, direct inference on the large model is typically not feasible.

The matrix multiplication in equation (3.1) can be expanded, and inserted into eq. (3.2), yielding

$$p_2(x) = \frac{1}{Z} \exp - \left\langle \begin{pmatrix} \frac{1}{2\sigma_1^2} & \frac{-\mu_1}{\sigma_1^2} & \frac{\mu_1^2}{2\sigma_1^2} & \frac{1}{2\sigma_2^2} & \frac{-\mu_2}{\sigma_2^2} & \frac{\mu_2^2}{2\sigma_2^2} \end{pmatrix}, \phi(x) \right\rangle,$$

with $\phi(x) = \begin{pmatrix} x_1^2 & x_1 & 1 & x_2^2 & x_2 & 1 \end{pmatrix}$. The Gaussian distribution is a special case of an *exponential model*, which is defined as

$$p(x|\theta) = \frac{1}{Z(\theta)} \exp\left(\langle \theta, \phi(x)\rangle\right), \tag{3.3}$$

where $Z(\theta)$ is the partition function ensuring normalization,

$$Z(\theta) = \int_{\mathcal{X}} \exp(\langle \theta, \phi(x)\rangle).$$

We may rewrite eq. (3.3) as

$$p(x|\theta) = \exp(\langle \theta, \phi(x)\rangle - \psi(x)), \quad \psi(\theta) = \log(Z(\theta)).$$

An energy function can be associated with the probability defined in eq. (3.3) by applying the negative logarithm,

$$E(x) = -\log p(x) = \psi(\theta) - \langle \theta, \phi(x)\rangle. \tag{3.4}$$

The normalizing constant $\psi(\theta)$ does not depend on state x, and hence can be omitted in the context of inference.

Another important instance of the exponential family are *discrete graphical models*. The probability of configuration x in any (second order) graphical model with vertices $\mathcal{V}$ and edges $\mathcal{E}$ is given by

$$p(x|\theta) = \frac{1}{Z(\theta)} \exp\left(\sum_{v\in\mathcal{V}} \theta_v(x_v) + \sum_{(v,w)\in\mathcal{E}} \theta_{v,w}(x_v, x_w)\right)$$

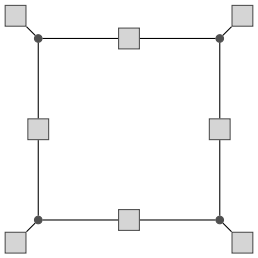


FIGURE 3.1. Example factor graph, vertices are shown as black dots, and factors as gray squares. This example shows a 2×2 grid graph, where factors are added to each vertex, and on horizontal and vertical edges. Note the bipartite graph structure, no two vertices or factors are connected.

The sum can be defined as an inner product between model parameters θ and indicator vector $\phi(x)$ given by

$$\phi_v(x, l) = \begin{cases} 1 & \text{if } x_v = l \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_{v,w}(x_v, x_w, l_v, l_w) = \begin{cases} 1 & \text{if } x_v = l_v, x_w = l_w \\ 0 & \text{otherwise} \end{cases}$$

A discrete graphical model can be represented by a *factor graph*, which is a bipartite graph separating variables and factors. A factor node is connected to a set of variable nodes, and assigns a local probability to each possible configuration.

DEFINITION 3.1.1 (Factor Graphs). A Factor graph (V, F, E) consists of vertices V, factors F and edges E, Figure 3.1 shows a simple example. The graph is bipartite, there are no edges between any two vertex or factor nodes. Let $a \in F$ denote a factor node, then $x(a)$ denotes the connected vertices and $f_a(x(a))$ the probabilities contained in factor a. The probability of configuration x is given by

$$p(x) = \frac{1}{Z} \prod_{a \in F} f_a(x(a)).$$

In order to express a factor graph in form of (3.3), we introduce $\phi(x)$ as a vector of indicator variables for each factor component. The model parameters θ contain the negative log of the corresponding factor probabilities,

$$p(x) = \frac{1}{Z(\theta)} \exp\left(\sum_{a \in F} \langle \theta_a, x_a \rangle \right). \tag{3.5}$$

Given a graphical model, we may be interested in determining it's optimal configuration, where optimality can be defined in different ways.

3.1.2. Marginalization. One way is to maximize the *marginal probability* for each label at each vertex, which is defined as

$$p(x_n = l) = \sum_{x, x_n = l} p(x).$$

Marginals can be computed in linear time with the *sum product algorithm*, if the underlying graphical model is a tree. The algorithms consists of sending messages between variable and factor nodes. Let $a \in F$ denote a factor, and $v \in V$ a vertex, then the corresponding messages are given by

$$n_{v \to a}(x_v) = \prod_{c \in N(v) \setminus a} m_{c \to v}(x_v) \tag{3.6a}$$

$$m_{a \to v}(x_v) = \sum_{x \in x(a) \setminus x_v} f_a(x) \prod_{w \in N(a) \setminus v} n_{w \to a}(x_w) \tag{3.6b}$$

Initially, all messages can be set to 1. This algorithm forms the basis for the *belief propagation* class of inference algorithms, which will be described in more detail below. If the model does have cycles, the algorithm is also called *loopy belief propagation* and may not converge. It is not possible to determine whether loopy belief propagation will converge on a given graphical model.

3.1.3. Maximum A-Posteriori Inference. Choosing the label with largest marginal for each vertex does not necessarily give the optimal configuration, it may not even be feasible. Thus, we may prefer the *maximum a-posteriori (MAP)* inference,

$$x^* = \arg\max_{x \in \mathcal{X}} p(x). \tag{3.7}$$

Note that the optimum of eq. (3.7) does not depend on the scale of $p(x)$, normalization is not required for MAP inference.

If the model is cycle-free, a changing the sum into a max operation in eq. (3.6b) yields the *max product algorithm*, performing MAP estimation.

$$n_{v \to a}(x_v) = \prod_{c \in N(v) \setminus a} m_{c \to v}(x_v) \tag{3.8a}$$

$$m_{a \to v}(x_v) = \max_{x \in x(a) \setminus x_v} f_a(x) \prod_{w \in N(a) \setminus v} n_{w \to a}(x_w) \tag{3.8b}$$

As in marginalization, there is no convergence guarantee on cyclic graphs. The max product algorithm may diverge in practice, but it has been applied successfully in optical flow and stereo matching models [**31**].

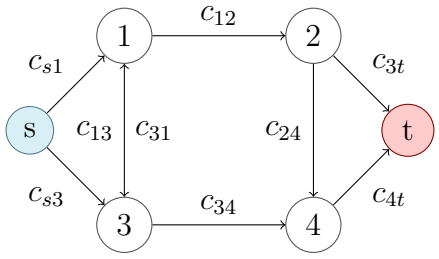


FIGURE 3.2. Example network for the maximum flow problem. The flow is emitted from node s and absorbed by node t. Each edge can only transmit a limited capacity, and flow has to be preserved at each vertex.

3.1.4. Submodular Graphical Models: Graph Cuts. While inference in graphical models computationally hard in general, it can be simpler for specific types of models. One example are *submodular* graphical models, where inference can be related to the maximum flow problem. An example maximum flow graph is shown in Figure 3.2.

DEFINITION 3.1.2 (Maximum Flow Problem). Given a network graph (V, E, c, s, t), we want to maximize the flow subject to constraints. V and E represent graph vertices and (undirected) edges, c represents a positive costs on each edge and s, t are the source and target vertices. The flow has to satisfy the following two constraints.

capacity constraint: $f_{uv} \leq c_{uv}$. The flow along edge (u, v) must not exceed the edge capacity.

flow conservation: $\sum f_{\cdot u} = \sum f_{u\cdot}$, $\forall u \in V \setminus \{s, t\}$. For each vertex (apart of s and t) the incoming and outgoing flows have to be equal.

DEFINITION 3.1.3 (Graph Cut). A cut is a partition of a graph with vertex set V into two subsets $A, B \subset V$. The sum of costs of the edges crossing A and B is denoted the cost of the cut.

THEOREM 3.1.1 (Max Flow Min Cut Theorem (Ford & Fulkerson 1956)). *The maximum flow of a graph equals its minimum graph cut. The maximum flow problem can be modeled as linear program, of which the dual can be interpreted as solving for the minimum cut.*

Assuming that we have a binary graphical model, we may reformulate the inference problem as a maximum flow problem on the graph, an illustration is shown in Figure 3.3. The maximum flow problem is well understood and a variety of polynomial time algorithms exist to solve it, thus enabling fast and exact inference.

For each vertex v, we define variable $x_v \in [0, 1]$ relaxed labeling variable, allowing soft assignments. Let $J_v(x), x \in \{0, 1\}$ denote the

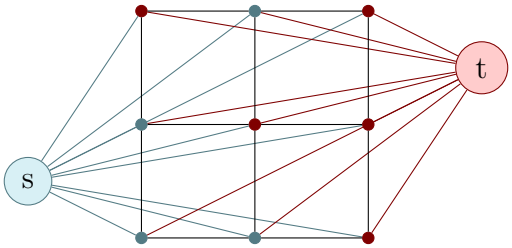


FIGURE 3.3. Reformulating the inference problem as graph cut. The flow from s to t has to pass the graphical model, where s and t represent the labels. Unary costs are represented in the edges between the original graph and s and t, while pairwise costs are represented in the graph edges.

unary costs for vertex v, then the relaxed cost reads

$$\begin{aligned}\tilde{J}_v(x_v) &= (1-x_v)J_v(0) + x_v J_v(1)\\ &= x_v(J_v(1) - J_v(0)) + J_v(0)\\ &= (1-x_v)(J_v(0) - J_v(1)) + J_v(1).\end{aligned}$$

Depending on the sign of $J_v(1) - J_v(1)$, the value has to be added as cost to the edge connecting s or (with reverse sign) t. The constant part $J_v(0)$ or $J_v(1)$ (depending on the sign of the difference) does not affect the optimal configuration and is ignored.

The pairwise terms can be inserted into the network in a similar way. The cost is given by

$$\begin{aligned}\tilde{J}_{vw}(x_v, x_w) = &\begin{pmatrix}1-x_v & x_v\end{pmatrix}\begin{pmatrix}A & B\\ C & D\end{pmatrix}\begin{pmatrix}1-x_w\\ x_w\end{pmatrix}\\ &= A + (D-B)x_v + (B-A)x_w\\ &\quad + (B+C-A-D)x_v(1-x_w).\end{aligned}$$

This expression factorizes into constant part A, which can be removed, two unary parts $D-B$ and $B-A$, which need to be processed as described above, and a pairwise part $B+C-A-D$. Since a flow network requires positive edge costs, the term $J(0,1)+J(1,0)-J(0,0)-J(1,1)$ has to be positive. This property is called *submodularity*, which also extends to the non-binary case.

DEFINITION 3.1.4 (Submodularity). A function $f : [N] \times [M] \to \mathbb{R}$ is submodular, if for every $1 \le i < i' \le N$ and $1 \le j < j' \le M$ the following property holds.

$$f(i,j) + f(i',j') \le f(i,j') + f(i',j).$$

Note that submodularity may depend on the order of labels, there is also a notion of permuted submodular functions [**77**].

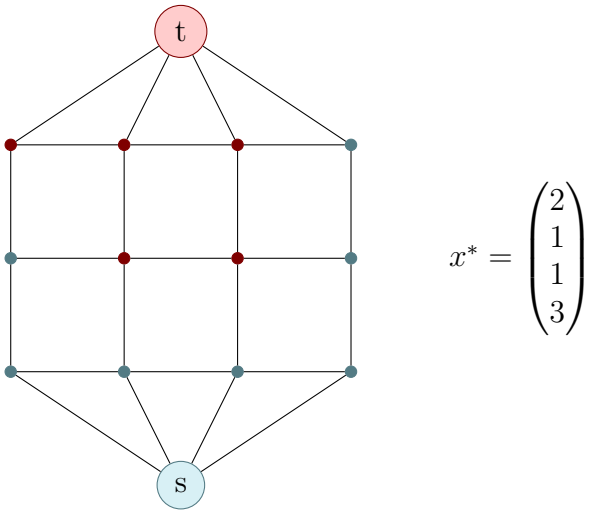


FIGURE 3.4. Ishikawa's method for MAP inference in a chain graph with four vertices and three labels, counting vertices from left to right. The original graph is replicated for each label, the edge weights are determined by the unary and pairwise costs of the original graphical model. The labels need to be ordered, and the pairwise term has to be a convex function of the label difference. Inference is performed by applying graph cuts to the extended graph.

If the graphical model is binary, but not submodular, *Quadratic Pseudo-Boolean Optimization (QPBO)* [**35**] can be applied, but is not guaranteed to find a label for each vertex.

Submodularity may be viewed as a correspondence of convexity in discrete optimization, since submodular functions usually can be solved to global optimality. There are extensions of the procedure above for non-binary submodular models, which solve a sequence of binary submodular function. Either by expanding a certain label α (α-expansion) or by swapping two labels α and β (α-β-swap), as described in [**14**].

In [**41**], Ishikawa presents inference on graphical models with more than two labels as graph cut formulation. The labels have to be linearly ordered, and the pairwise term has to be a convex function of the label difference,

$$E(x) = \sum_{v \in V} D(x_v) + \sum_{(v,w) \in E} P(x_v - x_w).$$

$D(.)$ is the unary energy, which is arbitrary, and $P(.)$ is the convex pairwise term. An illustration of the graph construction is shown in Figure 3.4. The flow network replicates the original graph for each label, and edges have to be stored explicitly when solving the max flow problem, which causes a large memory requirement of the approach.

3.1.5. The Local Polytope Relaxation. Rather than maximizing the probability of a certain state, we may instead minimize the corresponding energy,

$$\arg\min_{x\in\mathcal{X}}\langle\theta,\phi(x)\rangle.$$

In order to simplify the expression further, we may replace the function ϕ with vector μ, which is constrained to all possible vectors $\phi(x)$, that correspond to a configuration $x \in \mathcal{X}$,

$$\arg\min_{\mu\in\mathcal{M}}\langle\theta,\mu\rangle. \tag{3.9}$$

Each component of the unary and pairwise factors has a corresponding component in vector μ. $\mathbb{M}$ is called the *marginal polytope*, since it also represents all possible marginals of the original graphical model with varying parameters θ. While the objective function is simple, the constraint set $\mathbb{M}$ is complicated and contains the computational difficulty of the original inference problem. $\phi(x)$ is of higher dimension than x, and more importantly, it is subject to interdependent constraints that arise from the graph structure.

Minimizing eq. (3.9) is not easier than maximizing the original probability as in eq. (3.7), but the highly complicated polytope $\mathbb{M}$ can be replaced by a simpler set $\mathbb{L}$ called the *local polytope*. Instead of modeling the entire graph structure, the local polytope only imposes constraints that arise on each factor independently. The local polytope is a superset of the marginal polytope, since those local constraints are satisfied by any valid marginal. Elements of the local polytope are also referred to as *pseudomarginals* [**92**].

Let f_a be a unary factor, and v_a be the corresponding vertex. Then the corresponding part of μ is subject to the following constraints.

$$\mu_a(i) \in [0,1],\ \forall i \in v_a$$
$$\sum_{i\in v_a}\mu_a(i) = 1$$

These constraints imply that μ_a is a valid probability distribution on vertex v_a. In case of second order factor f_{ab}, with corresponding vertices v_a and v_b, the constraints read as follows.

$$\mu_{ab}(i,j) \in [0,1],\ \forall i \in v_a, j \in v_b$$
$$\sum_{j\in v_b}\mu_{ab}(i,j) = \mu_a(i)$$
$$\sum_{i\in v_a}\mu_{ab}(i,j) = \mu_b(j)$$

The constraints ensure that label selections based on first order factors are consistent with selections in second order ones. Higher order factors can be handled similarly.

The local polytope provides a relaxation of the marginal polytope which can be feasible in many even high dimensional cases. While the extreme points of the marginal polytope are always binary vectors, the local polytope may contain fractional extreme points, resulting in not well defined optimal states. If the underlying graph is a tree, the local polytope is identical to the marginal polytope, and exact inference is feasible.

3.1.6. Dual Decomposition. We may write the local polytope relaxation as linear program,

$$\begin{aligned}
&\min_{\mu} \langle \theta, \mu \rangle \\
&\text{s.t. } \mu(i) \in [0,1], \ \forall i \\
&\qquad \text{for all second order factors } f_{ab}\text{:} \\
&\qquad \sum_{j \in v_b} \mu_{ab}(.,j) = \mu_a \\
&\qquad \sum_{i \in v_a} \mu_{ab}(i,.) = \mu_b
\end{aligned}$$

Rather than listing constraints factor-wise, we may change to edges in the factor graph.

$$\begin{aligned}
&\min_{\mu} \langle \theta, \mu \rangle \\
&\text{s.t. for all edges } (v,f),\ v \in V \text{ and } f \in F\text{:} \\
&\qquad \mu_f(i) \in [0,1], \ \forall i \\
&\qquad A^f \mu_f = \mu_v
\end{aligned}$$

Note that unary factors do not provide restrictive constraints. Since μ_v is defined by its neighboring higher order factors, it is restricted to $[0,1]$ implicitly. Also, for any unary factor f, $A^f = I$, so that the consistency constraint reads $\mu^f = \mu_v$, thus μ^f may simply be removed from the LP.

In order to solve the local polytope relaxation, we introduce Lagrangian variables λ^{vf},

$$\begin{aligned}
\mathcal{L}(\mu, \lambda) &= \langle \theta, \mu \rangle + \sum_{f \in F} \delta_{\Delta_{|f|}}(\mu_f) + \sum_{(v,f) \in E} \langle \lambda^{vf}, A^f \mu_f - \mu_v \rangle \\
&= \langle \theta, \mu \rangle + \sum_{f \in F} \delta_{\Delta_{|f|}}(\mu_f) + \sum_{(v,f) \in E} \langle \lambda^{vf}, A^f \mu_f \rangle - \langle \lambda^{vf}, \mu_v \rangle
\end{aligned}$$

Since μ_v is unconstrained, we need to impose constraints on $\lambda^v f$ for a well defined Lagrangian,

$$\sum_{(v,\cdot) \in E} \lambda^{v\cdot} = 0, \ \forall v \in V.$$

The Lagrange term can now be optimized with any suitable algorithm, such as ADMM, leading to the dual decomposition approach. In [**42**], the Lagrangian is optimized with a bundle method, where a simple approximation of the energy function is maintained and updated during optimization.

3.1.7. Tree Based Belief Propagation Algorithms. As we have seen many times in previous sections, inference on trees is much simpler than inference on cyclic graphs. Therefore, it appears plausible to approximate a cyclic graph by a collection of trees. This approach has been introduced in [**92**], presenting the *tree-reweighted belief propagation (TRBP)* algorithm, and later extended to the *sequential tree-reweighted (TRW-S)* algorithm in [**44**].

Instead regarding the model parameters as fixed values, we introduce the MAP estimation problem as function of θ as variable,

$$\Phi(\theta) := \min_{x \in \mathcal{X}} \langle \theta, \phi(x) \rangle.$$

$\Phi(\theta)$ is convex, hence Jensen's inequality can be applied,

$$\Phi(\theta) \leq \sum_i \alpha_i \Phi(\theta^i), \text{ where } \sum_i \alpha_i \theta^i = \theta, \ \alpha_i > 0 \ \forall i.$$

Since inference on trees is inherently simpler than inference on cyclic models, we may choose the θ^i vectors to represent trees. Components of θ^i, that correspond to factors which are not contained in the respective tree, are set to zero. The θ^i vectors are stacked to matrix $\Theta \in \mathbb{R}^{d \times |\phi(x)|}$, and the following problem needs to be minimized,

$$\begin{aligned} &\max_{\Theta} \sum_i \alpha_i \min_x \langle \theta^i, \phi(x) \rangle \\ &\text{s.t. } \sum_i \alpha_i \theta^i = \theta. \end{aligned}$$

In order to find the dual, we set up the Lagrangian,

$$\begin{aligned} \mathcal{L}(\Theta, \tau) &= \sum_i \alpha_i \Phi(\theta^i) + \langle \tau, \theta - \sum_i \alpha_i \theta^i \rangle \\ &= \sum_i \alpha_i (\Phi(\theta^i) - \langle \theta^i, \tau \rangle) + \langle \tau, \theta \rangle \end{aligned}$$

The dual function $\min_\Theta \mathcal{L}(\Theta, \tau)$ of τ is precisely the LP relaxation over the local polytope as presented in section 3.1.5, a proof is shown in [**92**]. This holds independently of the choice of trees, as long as each graph edge is covered by some tree. The paper presents a belief propagation approach over the tree decomposition, which is guaranteed to converge, but the duality gap may not be tight at the optimum. If a set of vertices can be found, for which the optimal configurations agree in all trees, then this configuration is guaranteed to be globally optimal.

The same graphical model can be represented with different vectors θ, i.e. by shifting unary energies into the pairwise terms. TRW-S generalizes eq. (3.1.7) by allowing any reparameterization of θ. Through this flexibility, TRW-S achieves monotonicity in its updates, while TRBP may increase the energy.

3.1.8. Continuous Labeling. We can simplify the local polytope relaxation as in section 3.1.5 further, if the pairwise energy satisfies certain conditions. We introduce simplex variable $s(v) \in \Delta_{|L(v)|}$ for each vertex v, which represents the probabilities over all labels $L(v)$. The unary energy can be written as inner product,

$$E_{\text{unary}} = \sum_{v \in V} \langle F(v), s(v) \rangle,$$

where $F(v)$ contains the unary energies. If the pairwise term of two neighboring vertices v and w is a convex function of $s(v) - s(w)$, the overall minimization is convex in the simplex variables, and we do not need variables for the pairwise factors.

The authors of [**49**] use this formulation for image labeling with TV regularization. We stack the simplex variables and corresponding unaries to vectors s and f, and let A denote the vector valued discrete gradient operator. The overall energy reads

$$E(s) = \langle s, f \rangle + \alpha \|As\|_1 + \delta_{\Delta_L}(s). \tag{3.10}$$

Rewriting the $L1$ norm as inner product with dual variable p,

$$\alpha \|As\|_1 = \sup_{p \in [-\alpha,\ \alpha]^{|p|}} \langle p, As \rangle,$$

we can minimize energy (3.10) with splitting algorithms from convex optimization. Projection on the simplex, which most likely will be required, can be computed in a finite number steps as shown in [**61**].

3.2. Elementary Manifolds

In the previous chapter on mathematical optimization, we assumed the variable $x \in \mathbb{R}$ to be free in the Euclidean space. Constraints can be imposed by an indicator function on a set $C \subset \mathbb{R}^n$, but the structure of set C is not represented adequately with this approach. For example, if we restrict our variables to the unit sphere S^n, the distance between any two points should be given by the arc between them rather than the straight line.

For a formal definition of manifolds, we first need to introduce *charts* and *atlases*, as described in [**1**].

Definition 3.2.1 (Chart). Let $\mathcal{M}$ be a set in $\mathbb{R}^n$. Then a bijection ϕ between a subset $\mathcal{U} \subset \mathcal{M}$ and $\mathbb{R}^d$ is called a chart, denoted $(\mathcal{U}, \phi)$. $\phi(x)$ are the *local coordinates* of $x \in \mathcal{M}$, and the inverse mapping ϕ^{-1} is called a *local parameterization* of $\mathcal{M}$.

DEFINITION 3.2.2 (Atlas). An Atlas is a set of charts $\mathcal{U}_\alpha$, covering the entire set $\mathcal{M}$, $\bigcup_\alpha \mathcal{U}_\alpha = \mathcal{M}$. Also, the given charts need to to be compatible on their intersections. For any α and β with $\mathcal{U}_\alpha \cap \mathcal{U}_\beta \neq \emptyset$, the sets $\phi_\alpha(\mathcal{U}_\alpha \cap \mathcal{U}_\beta)$ and $\phi_\beta(\mathcal{U}_\alpha \cap \mathcal{U}_\beta)$ are open subsets of $\mathbb{R}^d$, and the function $\phi_\beta \circ \phi_\alpha$ is smooth.

An Atlas defines a differentiable structure on $\mathcal{M}$. The set $\mathcal{M}$ together with a differentiable structure is called a manifold.

3.2.1. Differential Geometry. If we have a real valued function f defined on a manifold $\mathcal{M}$, we need to incorporate the geometry of $\mathcal{M}$, in order to define derivatives. Let $\gamma(t) : \mathbb{R} \to \mathcal{M}$ denote a curve on $\mathcal{M}$. Then the Euclidean derivative

$$\dot{\gamma}(0) = \lim_{h \to 0} \frac{\gamma(h) - \gamma(0)}{h}$$

may not well defined, since $\mathcal{M}$ is not necessarily a vector space, which is required for evaluating the difference $\gamma(h) - \gamma(0)$. However, if we compose γ with function $f : \mathcal{M} \to \mathbb{R}$, we can construct the derivative

$$\frac{d}{dt}(f \circ \gamma) = Df(\gamma(0))[\gamma'(0)],$$

which is the derivative of f at $\gamma(0)$ in direction $\gamma'(0)$. $\gamma'(0)$ is a *tangent vector* to $\mathcal{M}$ at $\gamma(0)$. The span of all tangent vectors that arise from different paths γ with identical starting point $\gamma(0)$ is called the *tangent space* of $\mathcal{M}$ at $\gamma(0)$, denoted $T_{\gamma(0)}\mathcal{M}$. Unlike the manifold itself, the tangent space is vector space by construction. The union of all tangent spaces is denoted *tangent bundle*,

$$T\mathcal{M} = \bigcup_{x \in \mathcal{M}} T_x\mathcal{M}. \tag{3.11}$$

A function $\xi : \mathcal{M} \to T\mathcal{M}$ is called a *vector field* on $\mathcal{M}$, the set of all smooth vector fields is denoted with $\mathfrak{X}(\mathcal{M})$. Let $f : \mathcal{M} \to \mathbb{R}$ be a smooth real valued function on $\mathcal{M}$, then the derivative of f along vector field ξ is given by $\xi f : \mathcal{M} \to \mathbb{R}$,

$$(\xi f)(x) = \xi_x(f) = Df(x)[\xi_x].$$

The multiplication of ξ by function f is given by

$$(f\xi)_x = f(x)\xi_x,$$

and we may add two vector fields ξ and ζ,

$$(\xi + \zeta)_x = \xi_x + \zeta_x.$$

In section 2.2, we have seen second order optimization methods, for example the Newton method. Defining the second order derivative of a function on a manifold $\mathcal{M}$ encounters a similar issue as in the first order case. Tangent vectors in different locations live in different tangent spaces, hence they are not directly comparable. In case of the

Euclidean space, the classical directional derivative of vector field ξ with respect to vector field ζ is given by

$$(\nabla_\zeta \xi)_x = \lim_{h\to 0} \frac{\xi_{x+h\zeta_x} - \xi_x}{h}. \tag{3.12}$$

$\nabla_\zeta \xi$ is called the *covariant derivative* of ξ with respect to ζ.

On non-Euclidean manifolds, covariant derivatives can be established through *affine connections*. An affine connection $\nabla : \mathfrak{X}(\mathcal{M}) \times \mathfrak{X}(\mathcal{M}) \to \mathfrak{X}(\mathcal{M})$ defines a derivative of a vector field on $\mathcal{M}$, is not unique, and has to satisfy the following properties,

$$\begin{aligned} \nabla_{f\zeta+g\chi}\xi &= f\nabla_\zeta \xi + g\nabla_\chi \xi && \text{(linearity in } \zeta) \\ \nabla_\zeta(a\xi + b\eta) &= a\nabla_\zeta \xi + b\nabla_\zeta \eta && \text{(linearity in } \xi) \\ \nabla_\zeta(f\xi) &= (\zeta f)\xi + f\nabla_\zeta \xi && \text{(Leibnitz' law)} \end{aligned}$$

The affine connection for the Euclidean case (3.12), where tangent vectors to not need to be transformed, is called *Euclidean connection* or *canonical connection*.

3.2.2. Riemannian Manifolds. While the tangent space of general manifolds is a vector space, and hence can be equipped with the Euclidean inner product, this inner product may not reflect the local geometry well. Riemannian manifolds are equipped with a Riemannian metric, which is an inner product $\langle \cdot, \cdot \rangle_x$ on the tangent space at $x \in \mathcal{M}$. The corresponding norm is given by

$$\|\xi\|_x = \sqrt{\langle \xi, \xi \rangle_x}.$$

If $\mathcal{M}$ is a submanifold of another Riemannian manifold $\overline{\mathcal{M}}$, then the Riemannian metrics are identical. Most our manifolds are submanifolds of $\mathbb{R}^{n\times p}$ with its standard Riemannian metric

$$\langle A, B \rangle = \operatorname{tr}(A^\top B).$$

We now define the *Riemannian gradient*,

$$\langle \nabla_{\mathcal{M}} f, \xi \rangle_x = Df(x)[\xi],$$

and the Riemannian connection

$$\nabla_{\zeta_x} \xi = P_x(D\xi(x)[\zeta_x]).$$

P_x denotes orthogonal projection into the tangent space at x. The derivative $D\xi(x)[\zeta_x]$ is well defined in the embedding manifold $\mathbb{R}^{n\times p}$. The Riemannian connection defines the Hessian operator

$$\operatorname{Hess} f(x)[\xi_x] = \nabla_{\xi_x} \nabla_{\mathcal{M}} f.$$

With gradient and Hessian operators available, we can now formulate second order optimization algorithms on general Riemannian manifolds.

In contrast to the Euclidean space, the update $x^{k+1} = x^k + y$ may not be well defined if x lies in a manifold $\mathcal{M}$. Therefore we need

retractions, which map a tangent vector back onto the manifold. At point $x \in \mathcal{M}$, the retraction $R_x : T_x\mathcal{M} \to \mathcal{M}$ has to satisfy the following properties,

$$\begin{aligned} R_x(0) &= x \\ DR_x(0)[\xi] &= \xi, \ \forall \xi \in T_x\mathcal{M} \end{aligned} \tag{3.13}$$

The first condition ensures that if update step $y = 0$, then x does not change after the update. The second conditions is called local rigidity, ans ensures that the direction of the update is (approximately) preserved.

We can now define the *pullback* of f through retraction R,

$$\hat{f}_x = f \circ R_x.$$

If an algorithm requires to evaluate $f(x+ty)$, as for example line search does, we need to pull back the function f from the manifold into the tangent space $T_x\mathcal{M}$.

There can be many different retractions for the same manifold that satisfy the conditions in (3.13). For a unique definition, we may request that the induced curve on the manifold has zero acceleration,

$$\frac{d^2}{dt^2} R_x(ty) = 0.$$

With this retraction, distance is preserved and there is a unique solution, at least for a particular choice of affine connection, which is required for the second order derivative. This particular retraction is called the *exponential map*, and forms a local bijection between $T_x\mathcal{M}$ and $\mathcal{M}$.

The *unit sphere* S^{n-1} forms a submanifold of $\mathbb{R}^n$, and is defined by

$$S^{n-1} = \{x \in \mathbb{R}^n \mid x^\top x = 1\}.$$

Let $\gamma(t) \in S^{n-1}$ be a curve on the sphere. Then the derivative of $\gamma(t)^\top \gamma(t)$ has to vanish,

$$\frac{d}{dt}\gamma(t)^\top\gamma(t) = 0 \ \Leftrightarrow \ \dot{\gamma}(t)^\top\gamma(t) + \gamma(t)^\top\dot{\gamma}(t) = 0.$$

This has to hold for all paths $\gamma(t)$, hence $\dot{\gamma}(t)$ has to satisfy $\dot{\gamma}(t)^\top\gamma(t) = 0$ and the tangent space of S^{n-1} is given by

$$T_x(S^{n-1}) = \{z \in \mathbb{R}^n \mid z^\top x = 0\}.$$

The orthogonal projection into the tangent space is given by the Euclidean projection of $\mathbb{R}^n$,

$$P_x = I - xx^\top.$$

The exponential map is given by

$$\mathrm{Exp}_x(t\xi) = x\cos(\|\xi\|t) + \frac{\xi}{\|\xi\|}\sin(\|\xi\|t).$$

We can check that the second derivative of the curve $\gamma(t) = \operatorname{Exp}_x(t\xi)$ vanishes,

$$\begin{aligned}\frac{D^2}{dt^2}\operatorname{Exp}_x(t\xi) &= P_x \frac{d^2}{dt^2} x(t) \\ &= (I - \gamma(t)\gamma(t)^\top)(-\|\xi\|^2)\gamma(t) = 0\end{aligned}$$

The Orthogonal Stiefel manifold is defined as

$$\operatorname{St}(p,n) = \{X \in \mathbb{R}^{n\times p} \mid X^\top X = I_p\}.$$

We can form the derivative of $X(t)^\top X(t)$ for $X(t) \in \operatorname{St}(p,n)$, and find the tangent space of the Stiefel manifold,

$$T_X \operatorname{St}(p,n) = \{Z \in \mathbb{R}^{n\times p} \mid X^\top Z + Z^\top X = 0\}. \tag{3.14}$$

In order to simplify the expression, we decompose $\dot{X}(t)$ into the pat spanned by $X(t)$, and the orthogonal part,

$$\dot{X} = X\Omega + X_\perp K.$$

Inserting into eq. (3.14) yields

$$\begin{aligned} & X^\top(X\Omega + X_\perp K) + (X\Omega + X_\perp K)^\top X = 0 \\ \Leftrightarrow \quad & \Omega + \Omega^\top = 0\end{aligned}$$

Hence, $\Omega \in \mathcal{S}_{\text{skew}}(p)$ and $K \in \mathbb{R}^{(n-p)\times p}$. The dimension of $\operatorname{St}(p,n)$ is

$$\begin{aligned}\dim \operatorname{St}(p,n) &= \frac{1}{2}p(p-1) + (n-p)p \\ &= np - \frac{1}{2}p(p+1).\end{aligned}$$

3.2.3. Lie Groups. Before we define Lie groups, we need to define groups first. Let the set G be equipped with operation $\cdot$ satisfying

$$\begin{aligned}\text{closure:} &\quad \text{Let } a, b \in G, \text{ then } a \cdot b \in G \\ \text{associativity:} &\quad a \cdot (b \cdot c) = (a \cdot b) \cdot c \\ \text{identity element:} &\quad \exists e \in G,\ e \cdot a = a \cdot e = a,\ \forall a \in G \\ \text{inverse element:} &\quad \forall a \in G,\ \exists a^{-1} \in G \text{ s.t. } a^{-1} \cdot a = a \cdot a^{-1} = e.\end{aligned}$$

Then $(G, \cdot)$ is a group. Lie groups are groups, which are also differentiable manifolds. We assume that the group operation is given by matrix multiplication, the following sets are examples of Lie groups,

- The general linear group $\operatorname{GL}(n) = \{X \in \mathbb{R}^{n\times n} \mid \det X \neq 0\}$, which contains all invertible matrices.
- The special linear group $\operatorname{SL}(n) = \{X \in \mathbb{R}^{n\times n} \mid \det X = 1\}$.
- The orthogonal group $\operatorname{O}(n) = \operatorname{St}(n,n) = \{X \in \mathbb{R}^{n\times n} \mid X^\top X = I_n\}$, representing all isometries in $\mathbb{R}^n$.
- The special orthogonal group $\operatorname{SO}(n) = \{X \in \mathbb{R}^{n\times n} \mid X^\top X = I_n,\ \det(X) = 1\}$, representing all rotations in $\mathbb{R}^n$.

Three dimensional rotations are Lie groups regarding matrix multiplication, and $\mathbb{R}^3$ is a Lie group regarding addition, we define the *special Euclidean group* describing all rigid motions,

$$\mathrm{SE}(3) = \left\{ \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \;\middle|\; R \in \mathrm{SO}(3),\ t \in \mathbb{R}^3 \right\}.$$

The composition of two elements of SE(3) is given by matrix multiplication, therefore the inverse rigid transform is given by the matrix inverse,

$$\begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} R^\top & -R^\top t \\ 0 & 1 \end{pmatrix}. \tag{3.15}$$

In the following, we introduce the general *algebra* and the *Lie algebra*, which is closely related to a Lie group.

Definition 3.2.3 (Algebra). An *algebra* $\mathcal{A}$ is a vector space, which is extended with a product operation, written uv for $u, v \in \mathcal{A}$. The product has to satisfy the following conditions, with $u, v, w \in \mathcal{A}$ and $\lambda \in \mathbb{R}$,

$$\begin{aligned} \lambda(uv) &= (\lambda u)v = u(\lambda v) \\ (u+v)w &= uw + vw \\ u(v+w) &= uv + uw \end{aligned}$$

If in addition $u(vw) = (uv)w$ holds, $\mathcal{A}$ is an associative algebra.

The *Lie bracket of two vector fields* $[X, Y] \;:\; \mathfrak{X}(\mathcal{M}) \times \mathfrak{X}(\mathcal{M}) \to \mathfrak{X}(\mathcal{M})$ is again a vector field satisfying

$$[X, Y]f = X(Yf) - Y(Xf).$$

This vector field is also called the *commutator product* of X and Y.

Definition 3.2.4 (Lie Algebra). A Lie algebra is a vector space $\mathfrak{g}$, equipped with a *Lie bracket* operation $[.,.] \;:\; \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$ satisfying

$$\begin{aligned} [\alpha u + \beta v, w] &= \alpha[u, w] + \beta[v, w] \\ [u, \alpha v + \beta w] &= \alpha[u, v] + \beta[u, w] \\ [u, v] &= -[v, u] \\ 0 &= [u, [v, w]] + [v, [w, u]] + [w, [u, v]], \end{aligned}$$

where $u, v, w \in \mathfrak{g}$ and $\alpha, \beta \in \mathbb{R}$. The last condition is called the *Jacobi identity*. Any associative algebra with the *matrix commutator*

$$[X, Y] = XY - YX \tag{3.16}$$

defines a Lie algebra.

The tangent space at the identity element forms a Lie algebra [**48**]. The corresponding Lie algebras for the example Lie groups above are given by

- $\mathfrak{gl}(n) = \mathbb{R}^{n \times n}$.

- $\mathfrak{sl}(n) = \{X \in \mathbb{R}^{n\times n} \mid \operatorname{tr}(X) = 0\}$.
- $\mathrm{O}(n)$ and $\mathrm{SO}(n)$ share the same tangent space at the identity, which is given by the skew symmetric matrices, $\mathfrak{so}(n) = \{X \in \mathbb{R}^{n\times n} \mid X^\top = -X\}$. $\mathrm{O}(n)$ consists of $\mathrm{SO}(n)$, and a second connected component with $\det(X) = -1$, representing reflections.

If the group operation is matrix multiplication, the exponential map is given by the matrix exponential. The matrix exponential may not be efficient to calculate, but in some cases we get closed form solutions. For example in case of three dimensional rotations, the matrix exponential is given in closed form by *Rodrigues' formula*, which can be shown by expanding and re-arranging the exponential series,

$$\exp([a]_\times) = I_3 + \frac{\sin(\|a\|)}{\|a\|}[a]_\times + \frac{1-\cos(\|a\|)}{\|a\|^2}[a]_\times^2.$$

3.2.4. Gradient Flows on Manifolds. Many algorithms we presented in 2.2 can be extended to functions over manifolds [**1**]. Details for the Stiefel and quotient manifolds are shown in [**26**]. In this section we show that matrix factorizations, including eigenvalue computation and singular value decomposition, can be reformulated as gradient flows on matrix manifolds.

We will briefly present the *double bracket isospectral flow* presented in [**38**], which diagonalizes a matrix under preservation of its eigenvalues.

Let $Q = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ be any diagonal matrix. Then the manifold of orthogonally equivalent matrices is given by

$$\mathcal{M}(Q) = \{\Theta^\top Q \Theta \mid \Theta \in \mathrm{O}(n)\}.$$

Any matrix in $\mathcal{M}(Q)$ has eigenvalues $\lambda_1, \ldots, \lambda_n$, hence any gradient flow on the manifold will be isospectral. Let $H \in \mathcal{S}_{\mathrm{sym}}(n)$ be a symmetric matrix, then there exists diagonal matrix Q such that $H \in \mathcal{M}(Q)$. The tangent space of $\mathcal{M}(Q)$ at H is given by

$$T_H\mathcal{M}(Q) = H\Omega - \Omega H = [H, \Omega],\ \Omega \in \mathcal{S}_{\mathrm{skew}}(n).$$

The tangent space can be found by differentiating $\Theta^\top H \Theta$ at $\Theta = I_n$. Let $N \in \mathbb{R}^{n\times n}$ be a diagonal matrix, then the double bracket isospectral flow

$$\frac{d}{dt}H(t) = [H(t), [H(t), N]]$$

converges to a diagonal matrix. If $N = \operatorname{diag}(1, \ldots, n)$, the limit will contain the sorted eigenvalues of H on its diagonal. Note that $[A, B]$ is skew symmetric, if both A and B are symmetric. In [**38**], the authors show that this gradient flow corresponds to the Riemannian gradient of the function

$$\min_{H\in\mathcal{M}(Q)} \frac{1}{2}\|N - H\|^2 = \min_{H\in\mathcal{M}(Q)} \frac{1}{2}\|N\|^2 + \frac{1}{2}\|H\|^2 - \operatorname{tr}(NH)$$

for a suitable choice of the Riemannian metric. The terms $\|N\|^2$ and $\|H\|^2$ are constant, because N is constant and H has constant eigenvalues.

Similar to the manifold of isospectral matrices, the matrices of constant rank form a smooth manifold as well. The following parametrization of the fixed rank manifold is given in [**89**],

$$\begin{aligned}\mathcal{M}_k &= \{X \in \mathbb{R}^{m\times n} \mid \operatorname{rank}(X) = k\} \\ &= \{U\Sigma V^\top \mid U \in \operatorname{St}(k,m), V \in \operatorname{St}(k,n), \Sigma = \operatorname{diag}(\sigma_1,\dots,\sigma_k)\},\end{aligned}$$

with $\sigma_1 > \cdots > \sigma_k > 0$. Rank constraints appear in multiview geometry, but also in other areas of computer vision [**30**].

CHAPTER 4

Multiple View Geometry

In contrast to laser and radar sensors, cameras cannot measure depth directly. The scene can only be estimated by aligning images from different cameras, whose positions are known. In our setup, the exact camera locations are not known a priori, but need to be estimated es well based on point correspondences. In this chapter, we present the geometry of a multiple camera system, where in the monocular case the different views are generated by the same camera in different locations.

We introduce projective geometry in section 4.1, where we demonstrate the camera model, how homographies can represent planar motion between two views, and epipolar geometry. Sections 4.2 introduces the stereo matching problem, where point correspondences between two calibrated cameras are estimated. Multiple frame scene reconstruction is presented in section 4.3. Section 4.4 gives an overview of monocular reconstruction approaches.

4.1. Projective Geometry

Estimation of point correspondences can be simplified, if the images are taken by cameras in different, ideally known locations. We introduce the camera model in seciton 4.1.1. If the scene consists of a plane, the induced motion can be calculated by multiplication with a homography matrix, which is described in section 4.1.2. In case of a general scene with unknown depth, there is no direct point correspondence, but the motion is constrained to *epipolar lines* by the Essential matrix. Epipolar geometry and estimation of the Essential matrix are described in sections 4.1.3 and 4.1.4.

4.1.1. The Camera Model. We assume the *pinhole camera model*, where viewing rays are captured on a planar image plane at $z = 1$. Figure 4.1 shows the pinhole camera setup and the projection of a point into the image plane. Since the pinhole camera cannot distinguish different scales of the same point, any point represents its entire equivalence class, which we refer to as *homogeneous coordinates*. The projection π onto the image plane, where each viewing direction is represented uniquely, is given by

$$\pi \begin{pmatrix} x & y & z \end{pmatrix}^\top = \begin{pmatrix} \frac{x}{z} & \frac{y}{z} & 1 \end{pmatrix}^\top .$$

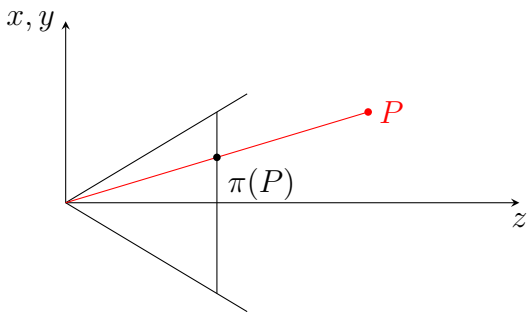


FIGURE 4.1. The pinhole camera. Points along viewing rays cannot be distinguished, hence they are identified through homogeneous coordinates. The projection π maps each point onto its unique representative on the image plane at $z = 1$.

Points and lines on the projective plane can be viewed as dual objects [**36**]. The roles of points and lines can be exchanged in any theorem of 2-dimensional projective geometry. As an example, any two points x_1 and x_2 define a line l with line equation $l^\top x = 0$, and any two lines l_1 and l_2 define a point x. Both relations can be expressed as outer products,

$$l = x_1 \times x_2 \tag{4.1a}$$

$$x = l_1 \times l_2. \tag{4.1b}$$

The duality principle holds for any statement of points and lines in three dimensions.Homogeneous coordinates enable us to represent infinite points compactly as $x = (x_1\ x_2\ 0)^\top$. The points at infinite lie on the line at infinity given by $l_\infty = (0\ 0\ 1)^\top$.

The standard camera described above needs to be calibrated before it can be applied to a three dimensional scene. Focal lengths and principal point are denoted *intrinsic*, and the camera position and orientation *extrinsic parameters*. The intrinsic parameters are given by camera matrix

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix},$$

where f_x and f_y denote the focal lengths (in pixels) and $(c_x\ c_y\ 1)^\top$ is the principal point, which is the pixel position corresponding to point $(0\ 0\ 1)^\top$. Inverting a camera matrix results in another camera,

$$K^{-1} = \begin{bmatrix} \frac{1}{f_x} & 0 & -\frac{c_x}{f_x} \\ 0 & \frac{1}{f_y} & -\frac{c_y}{f_y} \\ 0 & 0 & 1 \end{bmatrix},$$

The extrinsic parameters are given by rotation R and translation t, which need to be applied (inversely) to the scene points,

$$x = K \begin{bmatrix} R^\top & -R^\top t \end{bmatrix} \begin{pmatrix} X \\ 1 \end{pmatrix}.$$

Inversion of egomotion parameters R and t is given by eq. (3.15). In later sections, we will assume the intrinsic parameters to be known and perform calculations in the 3D model, rather than the rasterized image. We ignore other effects of physical cameras, such as lens distortion.

4.1.2. Homographies. Assuming that we know the camera locations, and the viewed scene consists of a single plane, point correspondences can be computed by applying a homography matrix $H \in \mathbb{R}^{3\times 3}$. The representation with homographies is convenient, since it replaces explicit modeling of camera and scene parameters with a simple matrix multiplication. The homography matrix is scale invariant, hence it has eight degrees of freedom, and can be estimated uniquely from four correspondences.

We assuming that the extrinsic parameters of the two cameras are given by

$$P_1 = \begin{bmatrix} I_3 & 0 \end{bmatrix}, \quad P_2 = \begin{bmatrix} R & t \end{bmatrix},$$

and the plane is defined by parameters v satisfying

$$v^\top X = 1.$$

For point x in the image plane of the first camera, we can compute depth $d(x)$ with $X = d(x)x$ by

$$v^\top X = v^\top (d(x)x) = 1 \;\Rightarrow\; d(x) = \frac{1}{v^\top x}.$$

From scene point X, we can find the corresponding point x',

$$x' = \pi(R^\top (X - t)).$$

Using scale invariance of the projection π, we can reformulate this equation to $x' = \pi(R^\top (I_3 - tv^\top)x)$. We observe that point correspondences on a plane can be calculated with homography $H = R^\top (I_3 - tv^\top)$. Again, we have eight degrees of freedom, three for each of R, t and v minus one for the scale ambiguity between the translation vector and the plane parameters.

The homography can be extended with camera intrinsics, in order to map in the pixel domain by

$$H_a = K_2 H K_1^{-1} = A + av^\top. \tag{4.2}$$

The components A and a depend on the camera motion, and v defines the plane. In [**29**], the *manifold of multiple homographies* is presented,

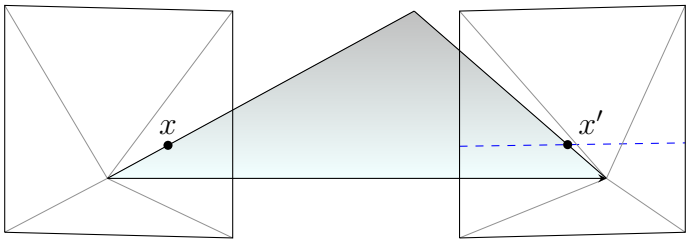


FIGURE 4.2. The viewing ray of the first camera along pixel x and the second camera center define a plane, which intersects with the second image plane and defines the corresponding epipolar line. The Fundamental matrix F provides immediate access to the corresponding line in the second camera $x'^{\top} l = 0$ of point x in the first camera by $l = Fx$.

where multiple homographies between the same frames are assumed to share the same camera motion. We can vectorize eq. (4.2),

$$\vec{H}_a = \vec{A} + (I_3 \otimes a)v = \begin{bmatrix} (I_3 \otimes a) & \vec{A} \end{bmatrix} \begin{pmatrix} v \\ 1 \end{pmatrix}.$$

Extending the plane vector to a matrix of multiple planes yields a matrix of the corresponding homographies, where all share the same components A and a. There are ambiguities in the representation in eq. (4.2), which can be utilized to reduce the dimension to $4n + 7$, where n is the number of homographies, see [**29**] for details. Hence, the factorization reduces the degrees of freedom already for two homographies.

While the calculation of the homography given camera motion and plane parameters is straightforward, the inverse relation is not. There are different approaches to factorize a given homography into its components, see [**57**] for a survey. The decomposition is inherently not unique, there exist two different decompositions (plus scale ambiguities) that describe the same homography matrix.

4.1.3. Epipolar Geometry. Assuming a setup of two cameras with known positions and calibration, epipolar geometry describes how point correspondences relate to 3D objects. The viewing ray from the camera center through pixel x in the first camera renders itself as a line in the image plane of the second camera, as shown in Figure 4.2. This point to line mapping can be represented by a single matrix F, called the *Fundamental matrix*, which depends on relative camera pose and the intrinsic camera parameters. The corresponding line in the second view can be calculated by $l = Fx$, hence any two corresponding points x and x' in the two cameras have to satisfy

$$x'^{\top} F x = 0.$$

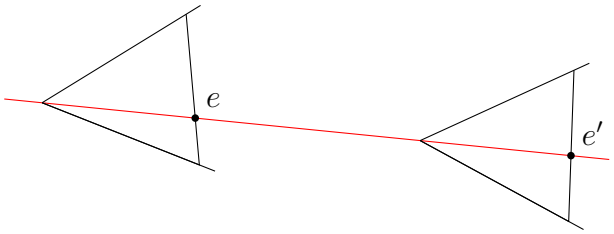


FIGURE 4.3. The epipoles are given by intersecting the line connecting both camera centers with the respective image plane. Since the camera center is part of every viewing ray, its point correspondence has to lie on every epipolar line.

Conversely, $F^\top$ describes the mapping from the second into the reference camera, $xF^\top x' = 0$ holds as well.

An important concept of epipolar geometry is the location of the epipoles in both cameras. The epipole is the intersection of the line connecting the two camera centers with the image planes. Figure 4.3 depicts where epipoles e in the reference frame, and e' in the second frame are located. If relative camera pose R and t are known, we can derive both epipoles immediately,

$$e = t$$
$$e' = R^\top t.$$

Since the epipole is contained in all epipolar lines, it must hold that

$$e'^\top Fx = 0 \; \forall x \;\; \Rightarrow \;\; e'^\top F = 0.$$

Therefore, the Fundamental matrix has rank two.

The Fundamental matrix can be calculated from intrinsics K_1 and K_2 and the relative rotation R and translation t by

$$F = K_2 R^\top [t]_\times K_1^{-1}. \tag{4.3}$$

This expression can be derived using the expression for the epipole and eq. (4.1a).

Since we assume intrinsic parameters to be known and compensated for, we may delete the camera matrices from expression (4.3), and are left with the *Essential matrix*

$$E = R^\top [t]_\times. \tag{4.4}$$

R and t can be recovered from a given Essential matrix using the SVD, but the solution is not unique. There are two distinct solutions for R, and a sign ambiguity for t, which yields four possible solutions of the decomposition. In later chapters, we will only refer to the Essential matrix, and not model the camera parameters explicitly.

There are degenerate configurations where the Essential matrix cannot be defined. The first case is when all points lie on a scene plane,

then there exists a homography H such that $x'_i = Hx,\ \forall i$. If we then choose $E = SH$, for any skew-symmetric matrix S, we have

$$x_i'^\top Ex = x_i^\top H^\top SHx_i = 0$$

There are infinitely many Essential matrices describing the point correspondences perfectly. The second degenerate case is at $t = 0$. Eq. (4.4) would simply yield $E = 0$, which is a trivial solution to the equation $x'^\top Ex = 0$ and does not satisfy the rank two constraint. Scene depth cannot be recovered from a stationary camera.

4.1.4. Essential Matrix Estimation. So far, we assumed the relative camera pose to be known and derived the Essential matrix, but in practice this is rarely the case. Usually, the Essential matrix needs to be estimated from point correspondences (x_i, x'_i), where the rank 2 constraint needs to be enforced during estimation. We can rewrite the Essential matrix constraint in linear form,

$$x_i'^\top Ex_i = 0 \ \Leftrightarrow\ (x_i^\top \otimes x_i'^\top)\,\mathrm{vec}(E) = 0.$$

We then define A to contain the Kronecker products of all corresponding points,

$$A = \begin{pmatrix} x_1^\top \otimes x_1'^\top \\ \dots \\ x_N^\top \otimes x_N'^\top \end{pmatrix}, \quad A\,\mathrm{vec}(E) = 0. \tag{4.5}$$

The Essential Matrix has eight degrees of freedom. Since each point correspondence introduces a linear constraint, we need at least eight points for estimating the Essential matrix.

The baseline approach is the *normalized 8-point algorithm*, which is based on the SVD. The algorithm does not require normalization to Essential matrices, but can estimate arbitrary Fundamental matrices. The 8-point algorithm is fast, since no iterative updates are required,

Input : Point correspondences $\{(x_i, x'_i)\}$
Output: Fundamental matrix F
Find normalizing transforms T and T', where $\hat{x}_i = Tx_i$, $\hat{x}'_i = Tx'_i$;
Construct A as in eq. (4.5) in terms of $\hat{x}_i$ and $\hat{x}'_i$. Define $\hat{F}$ as the vector corresponding to the smallest singular value of A;
Enforce rank constraint on $\hat{F}$ by setting the smallest singular value to zero;
De-normalize the Fundamental matrix, $F = T'^\top \hat{F} T$;
Algorithm 9: Normalized 8-point algorithm

but may not be accurate. The choice of normalization influences the outcome, and it is not clear which normalization should be utilized. The rank constraint is enforced by projecting the estimated Essential

matrix onto the manifold of rank two matrices, it is not clear how much accuracy is lost at that step.

Another approach is to perform algebraic minimization. In [**88**], the energy $\Psi(A\,\mathrm{vec}(E))$ is minimized subject to $\|\mathrm{vec}(E)\| = 1$, where Ψ is a convex error function. The norm one constraint avoids the trivial solution $E = 0$, but does not enforce the correct rank, hence the matrix needs to be projected again after iterative optimization.

In contrast to the above methods, the approach presented in [**37**] introduces the manifold of essential matrices $\mathcal{E}$, and performs the estimation task as energy minimization on the manifold. The Essential matrix can be factorized as

$$E = UE_sV^\top,\ E_s = \mathrm{diag}(s, s, 0),\ U, V \in \mathrm{SO}(3),\ s > 0.$$

Since scale cannot be determined from point correspondences, we restrict ourselves to *normalized Essential matrices* with $s = 1$. Differentiating with respect to U and V, with respective tangent spaces $U\Omega^U$ and $V\Omega^V$ and $\Omega = [(\omega_1\ \omega_2\ \omega_3)^\top]_\times$, we find the tangent space

$$\begin{aligned} T_E\mathcal{E} &= \{U(\Omega^U E_1 - E_1\Omega^V)V^\top\} \\ &= \left\{ U \begin{bmatrix} 0 & \omega_3^V - \omega_3^U & -\omega_2^V \\ \omega_3^U - \omega_3^V & 0 & \omega_1^V \\ -\omega_2^U & \omega_1^U & 0 \end{bmatrix} V^\top \right\}. \end{aligned}$$

We observe that ω_3^U and ω_3^V appear only in difference, hence we have an over-parametrization with the six components of ω^U and ω^V. The Essential manifold is only five dimensional. This matches the geometric interpretation, where we would expect three degrees of freedom from the rotation and three from the translation part minus one for scale ambiguity. Given vector $x \in \mathbb{R}^5$, we define tangent vector $\xi_E(x)$ by

$$\xi_E(x) = U \begin{bmatrix} 0 & -x_3 & -x_5 \\ x_3 & 0 & x_4 \\ -x_2 & x_1 & 0 \end{bmatrix} V^\top.$$

We can translate the tangent vector back on Ω^U and Ω^V, in order to update U and V accordingly during optimization,

$$\Omega^U = \left[\begin{pmatrix} x_1 \\ x_2 \\ \frac{x_3}{\sqrt{2}} \end{pmatrix} \right]_\times, \quad \Omega^V = \left[\begin{pmatrix} x_4 \\ x_5 \\ \frac{-x_3}{\sqrt{2}} \end{pmatrix} \right]_\times.$$

The authors derive Riemannian gradient and Hessian of the quadratic energy

$$J(E) = \frac{1}{2n} \sum_{i=1}^{n} (x_i'^\top E x)^2,$$

and apply the Newton method for minimization, under consideration of different retractions. The energy function is convex, but the set of valid Essential matrices is not. Hence, we are not guaranteed global

minimization, but the algorithm turns out to be fast and accurate in practice.

In [**18**], the problem of estimating the fundamental matrix is lifted into the set of probability measures on all valid Essential matrices. Even though this set is infinite dimensional, the authors propose a method to minimize the algebraic error as a sequence of convex polynomial optimization problems, and demonstrate state of the art performance.

4.2. Stereo Matching

In a camera setup, where two cameras are positioned horizontally, and facing in the same direction and take pictures simultaneously, we have a stereo camera system. Due to epipolar geometry, the optical flow field between both images is restricted to be horizontal, which reduces the stereo matching problem to one dimension, rather than two. As in the monocular case, the horizontal epipolar lines intersect in the epipole, which lies at infinity. Such a horizontal optical flow field is called a *disparity map*, and the disparities are connected to depth by

$$\text{depth} = \frac{\text{baseline}}{\text{disparity}}.$$

The baseline has to be measured in pixels, thus depends on the intrinsic camera parameters.

Even though stereo is similar to optical flow in its formulation, it is addressed with very different methods. The search space is smaller, and the labels are naturally ordered, which simplifies the application of discrete approaches to stereo vision. Early papers in computer vision [**31, 86**], as well as recent ones [**95, 96, 12**] solve the stereo estimation problem by discrete MAP inference. A continuous approach to the stereo estimation as labeling problem is presented in [**72**]. Other approaches are local (independent at each pixel) [**76**], or semiglobal [**39**], which means that an energy is formulated, but not minimized to global optimality.

We present the semiglobal matching in more detail, it is a well established stereo matching algorithm due to its fast runtime and high accuracy. The approach does not require rectified stereo images, but the camera poses need to be known in order to have a one dimensional correspondence problem. The energy of disparity field D is given by

$$\begin{aligned} E(D) = &\sum_{x\in\Omega} C(x, D(x)) + \\ &\sum_{y\in\mathcal{N}(x)} p_1 \mathbb{1}_{D(x)-D(y)=1} + p_2 \mathbb{1}_{D(x)-D(y)>1}. \end{aligned} \tag{4.6}$$

The parameters p_1 and p_2 penalize disparity differences of 1 and > 1, respectively, where $p_1 \leq p_2$ should hold. Parameter p_1 represents small

changes in disparity due to slanted or curved surfaces, while p_2 models depth discontinuities.

The data energy is based on the information theoretic *mutual information*, which measures how well the first patch predicts the second one. The mutual information of images I and the warped image I^w is given by the entropy of the images minus their joint entropy

$$\mathrm{MI}(I, I^w) = H(I) + H(I^w) - H(I, I^w).$$

Two images get high matching score if they contain much information, and are well predictable from each other. The entropy is given by

$$H(I) = \sum_c P_I(c) \log P_I(c),$$

where $P_I(c)$ denotes the probability of gray value c in image I. Similarly, the joint entropy is given by gray value pairs (c_1, c_2) as they occur in I and I^w. Note that the data energy is a function of the entire labeling, and cannot be evaluated pixel wise. The unary cost $C(x, D(x))$ in eq. (4.6) is defined based on an initial disparity map.

Exact inference on the energy (4.6) is computationally non trivial, since the problem is NP hard. However, if the vertices were linearly ordered, fast exact inference would be possible with dynamic programming. Therefore, semiglobal matching approximates the original problem on the two dimensional grid with a sum of one dimensional problems with different slopes.

The stereo setup comes with advantages in three dimensional scene reconstruction as well. Since the camera positions are known, the depth can be estimated in correct scale, and the stereo camera positions provide more reconstruction accuracy, than a monocular camera traveling into the scene. With the epipole at infinity, it does not have any influence on the depth reconstruction accuracy, there are no blind spots in the image. In addition to this, both stereo cameras take pictures at the same time, thus the static scene assumption is valid, while it may not hold in a monocular setting. The stereo setup is inherently better posed for reconstruction than the monocular one.

4.3. Structure Estimation with Bundle Adjustment

Given a set of images, bundle adjustment consist of estimating the corresponding scene geometry. Depending on the scene setup, the variables may differ, but the set of images is assumed to be fixed, bundle adjustment is a *batch approach*. Typically, point correspondences on the images will be computed first, which are then explained by camera poses and 3D scene points.

A slightly simplified scenario lies in the assumption that all cameras have the same center point, and only differ in orientation, as for

FIGURE 4.4. Panorama stitching with bundle adjustment. The 3D orientations and intrinsic camera parameters need to be estimated, in order to render the pictures into a common artificial view (top row). The bottom image shows the final panorama, which is the result of further postprocessing. This particular panorama illustrates the difficulty that arises with dynamic objects in the scene. A car appears twice, because it was photographed in different locations in the input images.

example in panorama stitching [**15**]. Due to the lack of camera translation, the induced motion is depth invariant, and no 3D points need to be computed. Each input image is equipped with a rotation R_i and intrinsic camera parameters K_i. The homography mapping view j into view i is given by

$$H_{ij} = K_i R_i R_j^\top K_j^{-1}.$$

Given point correspondences $\{(u_i^k, u_j^k)\}$, where i and j define an image pair and k enumerates the respective matches, bundle adjustment lies in minimization of the energy

$$E = \sum_{i,j,k} \frac{1}{2} \|u_i^k - \pi(H_{ij} u_j^k)\|^2$$

over image rotations and intrinsic camera parameters. The energy can be minimized with Levenberg-Marquardt. While the energy is convex, the set of rotation matrices is not, and we are only guaranteed convergence to a local minimum. Furthermore, no global orientation is enforced, rotating all cameras jointly does not change the objective, since the homography H_{ij} only depends on relative orientation between any two views.

Figure 4.4 shows an example panorama image, where postprocessing has to been applied after bundle adjustment. Differences in brightness need to be reduced (gain compensation), and the transition between the different views needs to be blurred.

4.4. Monocular Localization and Mapping

Structure from Motion for monocular videos is referred to as *simultaneous localization and mapping (SLAM)*, where the camera position inside a virtual world is computed, while the world map is updated simultaneously. In the following, we present a short survey of monocular SLAM approaches.

An early work in the field is the *parallel tracking and mapping (PTAM)* approach [**43**], where the video is split into segments sing keyframes, and bundle adjustment is performed on each segment. The tracking and mapping parts are handled in parallel in two independent threads. The method tracks a very sparse set of feature points, and is targeted towards small closed environments, such as small objects in an indoor environment. A similar approach, which does not split intow two parallel threads, and has a different set of image features, is presented in [**23**].

Large Scale Direct Monocular SLAM (LSD-SLAM) [**28**] focuses on accurate odometry and semi-dense depth maps for long video sequences in outdoor environments. The keyframes are aligned according to the estimated camera parameters and depth values, minimizing the photometric error yields accurate poses and depths, and reduces sensitivity to scale drift. Loop closure detects previously visited locations, and corrects drift errors in previous keyframes. The approach runs in real time on a single core CPU.

A dense approach is given by the *Dense Tracking and Mapping (DTAM)* method [**67**], which still runs in real time, but on parallel GPU hardware. Each keyframe is equipped with a dense inverse depth map ξ. Auxiliary variables α are introduced, which are coupled to ξ,

$$E_{\xi,\alpha} = \int_\Omega g(u)\|\nabla\xi\|_\epsilon + \lambda C(u,\alpha) + \frac{1}{2\theta}(\xi-\alpha)^2 du. \tag{4.7}$$

$g(u)$ denotes a weighting term, which depends on the image gradient. The ϵ norm is defined similarly to a scaled version of the Huber norm (5.3),

$$\|x\|_\epsilon = \begin{cases} \frac{\|x\|_2^2}{2\epsilon} & \text{if } \|x\|_2 < \epsilon \\ \|x\|_1 - \frac{\epsilon}{2} & \text{otherwise} \end{cases},$$

but the choice between L1 and L2 norm depends on the respective case. Selecting the L2 norm for small values of $\|x\|$ reduces the staircasing effect, which tends to occur with TV regularization. $C(u,\alpha)$ is the photometric error term, which depends on α and the camera parameters, which are estimated in a separate tracking method. Utilizing the dense depth estimates improves pose estimation compared to the sparse PTAM tracking algorithm. Energy (4.7) is minimized with the primal-dual method. The non convexity of the dataterm $C(u,\alpha)$ is handled by discretizing the inverse depth interval $[\xi_{\min}, \xi_{\max}]$ into equally spaced

segments, and performing exhaustive search in order to evaluate the proximal operator. More recent similar approaches are presented in [**82, 54, 24**].

An alternative scene representation is given by voxels, which can either be empty or solid. Each keyframe is equipped with a depth map, which are then fused into a global voxel representation. The following energy is presented in [**99**] and [**98**] for depth map integration,

$$E(u) = \int_\Omega \sum_{i \in \mathcal{I}} w_i(x)|u(x) - f_i(x)|dx + \alpha \, \mathrm{TV}(u).$$

$f_i(x)$ denotes the vote of frame i for voxel x, which is 1 for air and -1 for solid, the object surface is given by the isosurface $u \equiv 0$. Occluded voxels are ignored by setting $w_i(x) = 0$ for a certain depth behind the observed object. The TV term is three dimensional, including x, x and z derivatives. The approach in [**94**] describes a setup, where a camera is mounted to a flying drone, while a compute server estimates a voxel reconstruction simultaneously. The dense voxel representation is limited to small range objects and grows cubic with increased resolution.

CHAPTER 5

Variational Discrete Optical Flow Estimation

The original optical flow methods based on the brightness constancy assumption are successful on many optical flow benchmarks, but they also have limitations. The brightness constancy assumption does not always hold in outdoor scenarios, especially in the case of a camera mounted on a car. Modifying the optical flow method to minimize a more robust data term may not be trivial. The warping scheme adds a further source of estimation error. Small objects tend to disappear on a small scale, recovering an object later on a finer scale is unlikely.

Discrete optical flow methods can overcome these limitations. The data term is evaluated for each pixel independently, hence there is no requirement on convexity or differentiability, and there is no requirement of a warping scheme, However, depending on the search space of optical flow candidates in each pixel, the model may become infeasible. Another drawback of discrete optical flow methods is the regular grid structure of the displacement labels. Discrete optical flow methods usually apply a continuous postprocessing, in order to achieve subpixel accurate optical flow.

In this chapter, we summarize continuous and discrete optical flow methods with their respective advantages. We propose two new approaches to discrete to discrete optical flow, a sparse approach with filtered flow candidates, and a dense one. The sparse version is particularly targeted to the static outdoor environment, and making use of epipolar geometry. The dense version approximates the intractably large flow model by sequence of smaller models, where the approximation strategy is justified by probabilistic means. We choose discrete optical flow models, because we think that the limitations of discrete approaches will become less important in the future, where computers will have more memory and computing resources available, and discrete inference algorithms are likely improve as well.

5.1. Optical Flow Estimation

The classical optical flow estimation methods are presented by Horn and Schunck in [**40**] and Lucas and Kanade [**56**]. Both approaches are based on the assumption, that the image brightness does not change over time along any track in the sequence,

$$\frac{dI}{dt} = 0.$$

The time derivative decomposes into x, y and t,

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\underbrace{\frac{dx}{dt}}_{:=u} + \frac{\partial I}{\partial y}\underbrace{\frac{dy}{dt}}_{:=v} + \frac{\partial I}{\partial t} = 0. \tag{5.1}$$

The partial derivatives of the intensity is given by the respective image gradients. At each location, we get a single constraint on two variables, thus the problem formulation is underdetermined. The brightness constancy assumption can only estimate the part of optical flow, which lies along the image gradient, the orthogonal part on constant brightness cannot be recovered from image data alone. Therefore, regularity between neighboring pixels is imposed by adding the squared optical flow gradient to the objective, which reads

$$\begin{aligned} E_{\mathrm{HS}}(u) = \int_{x\in\Omega} & (I_x(x)u + I_y(x)v + I_t(x))^2 \\ & + \alpha\left((\nabla u_x)^2 + (\nabla u_y)^2\right) dx. \end{aligned} \tag{5.2}$$

From the Euler-Lagrange equation, we get optimality conditions on the optical flow field,

$$\begin{aligned} I_x^2 u + I_x I_y v &= \alpha^2 \Delta u - I_x I_t \\ I_x I_y u + I_y^2 v &= \alpha^2 \Delta v - I_y I_t \end{aligned}$$

An iterative scheme solves for u and v satisfying the Euler-Lagrange equations, using a discrete approximation to the Laplace operator. The classical model by Horn and Schunck has been basis for several optical flow methods, which extended the original formulation to different data and regularity terms, e.g. [**16, 83**].

The approach by Lucas and Kanade avoids the requirement of an explicit regularization by assuming that neighboring pixels $\{x^i\}$ will have very similar optical flow. This assumption generates several brightness constancy equations, where the exact number depends on the choice of neighborhood, such that the resulting system is overdetermined.

$$\underbrace{\begin{bmatrix} I_x(x^0) & I_y(x^0) \\ I_x(x^1) & I_y(x^1) \\ \dots & \\ I_x(x^n) & I_y(x^n) \end{bmatrix}}_{:=A} \begin{pmatrix} u \\ v \end{pmatrix} = \underbrace{\begin{bmatrix} -I_t(x^0) \\ -I_t(x^1) \\ \dots \\ -I_t(x^n) \end{bmatrix}}_{:=b}$$

We compute u and v using the least squares solution

$$A^\top A \begin{pmatrix} u \\ v \end{pmatrix} = A^\top b \;\Rightarrow\; \begin{pmatrix} u \\ v \end{pmatrix} = (A^\top A)^{-1} A^\top b.$$

We may multiply the image patch with a weighting matrix W,

$$\begin{pmatrix} u \\ v \end{pmatrix} = (A^\top W A)^{-1} A^\top W b.$$

The matrix $(A^\top W A)$ is also called the *structure tensor*, which we will encounter again in the section sparse optical flow estimation. In homogeneous regions, however, the resulting linear system may not be invertible. There is no guarantee for estimating a dense optical flow field. In contrast to the method by Horn and Schunck, there is no need for an iterative update scheme.

The drawback of both approaches is that the relation in eq. (5.1) only holds if the motion is very small compared to the image resolution. Displacements over several pixels can only be estimated by refining the optical flow field estimated over different scales of the image. Since details are lost after blurring and downscaling, motion of small objects may not be estimated correctly.

5.1.1. Robust Optical Flow. The traditional approaches to optical flow estimation as presented in the previous section lack robustness, of both the data and the regularity term. The gray value constancy assumption is not robust against changes in illumination, and the quadratic smoothing prior (5.2) does not preserve sharp motion discontinuities.

Robustness against changes of illumination can be achieved by *normalized cross correlation (NCC)* [**90**] on image patches. Given two vectors of gray values, the NCC energy reads

$$E_{\text{NCC}}(p, q) = 1 - \frac{(p - \bar{p})^\top (q - \bar{q})}{\|p - \bar{p}\| \|q - \bar{q}\|},$$

$\bar{p}$ and $\bar{q}$ denote the average grey values of the respective patch. We observe that NCC is invariant against additive and multiplicative changes in p or q. However, the energy is not differentiable and not convex.

Another robust matching term is based on the CENSUS transform [**97, 34**], which lists the sign differences of the respective pixels in comparison to the patch center x^*,

$$\text{CENSUS}(p, x) = \text{sign}_\epsilon(p(x) - p(x^*)),\ \text{sign}_\epsilon(\alpha) = \begin{cases} -1 & \text{if } \alpha < -\epsilon \\ 0 & \text{if } |\alpha| \leq \epsilon \\ 1 & \text{if } \alpha > \epsilon \end{cases}.$$

This is a modified version of CENSUS, which is also used in [**90**], the original transform does not have the ϵ ball around zero. The CENSUS matching of patches p and q consists of counting different signs in the CENSUS transformed patches,

$$E_{\text{CENSUS}}(p, q) = \sum_x \mathbb{1}_{CENSUS(p,x) \neq CENSUS(q,x)}$$

The CENSUS transform is robust against both additive and multiplicative changes as well, it reduces the gray value information into a binary or ternary sign function, thus removing most of the information. The energy is piecewise constant in terms of the center pixel gray value.

The quadratic regularity term (5.2) can be replaced by the more robust $L1$ norm or total variation,

$$\mathrm{TV}(u) = \|\nabla u_x\|_1 + \|\nabla u_y\|_1.$$

The $L1$ norm is more robust, since outliers get a linear rather than quadratic weight, and convex, but not differentiable. Therefore, it may be approximated by a the Huber norm,

$$\|\alpha\|_\delta^H = \begin{cases} \frac{1}{2}\alpha^2 & \text{if } |\alpha| < \delta \\ \delta(|\alpha| - \frac{1}{2}\delta) & \text{otherwise} \end{cases}, \tag{5.3}$$

which is locally quadratic around zero, but grows linearly for large values of α. A similar approximation is the *Charbonnier functional* [**17**], which does not require a piecewise definition and has a continuous second derivative,

$$\|\alpha\|_\delta^C = \sqrt{\alpha^2 + \delta}.$$

This approach has been applied in [**16, 83**].

The non-differentiability of the $L1$ regularity term can be addressed by applying optimization techniques, which do not require a smooth objective. E.g. the DataFlow approach [**90**] computes the optical flow field using the fast primal dual method for the dataterms mentioned above and total generalized variation (TGV),

$$E_{\mathrm{TGV}}(u, w) = E_{\mathrm{data}}(u) + \alpha_1\|\nabla u - w\|_1 + \alpha_0\|\nabla w\|_1. \tag{5.4}$$

The auxiliary variable w contains affine parameters for the optical flow field, whose gradient is assumed to be sparse. Thus, TGV regularization encourages piecewise affine optical flow, while TV regularization assumes a piecewise constant flow field u. The fast primal dual method requires to evaluate the proximal operator for the data term, which is possible for NCC and CENSUS matching. Also, the proximal operator of the data term can be evaluated for each pixel independently, which makes the approach parallelizable.

A similar approach is presented in [**71**], but the TGV term is extended to a non local neighborhood,

$$\begin{aligned}\mathrm{NLTGV}(u, w) = &\sum_{x\in\Omega}\sum_{y\in\Omega} \alpha_1(x, y)|u(x) - u(y) - \langle w(x), x - y\rangle| \\ &+ \sum_{x\in\Omega}\sum_{y\in\Omega} \alpha_0(x, y)\|w(x) - w(y)\|_1.\end{aligned}$$

The Census matching is evaluated across different scales, in order to increase robustness against scale changes as well. Energy minimization is performed over the image pyramid in a fast primal dual framework as well.

5.1.2. Discrete Approaches. The methods for optical flow estimation we presented in the previous section require the creation of a multiscale pyramid, in order to warp and refine the flow field from coarse to fine. Small objects, which cannot be detected at small scale, will not be rediscovered at a finer level due to the greediness of the approach. Different approaches to avoid the coarse to fine scheme have been proposed, which operate on the reference images in full size.

The approach of Steinbrücker [**80**] separates the data and regularity energies by introducing auxiliary variables u and v,

$$E(u, v) = E_{\text{data}}(u) + E_{\text{reg.}}(v) + \frac{1}{2\theta}(u - v)^2.$$

The energy is minimized with respect to u and v alternatingly until convergence. The regularity plus coupling term is convex, and can be minimized for example with the fast primal dual method. The data plus coupling term can be minimized by exhaustive search. The method does not have any requirement on the dataterm, Drawback of this approach is that it requires a long sequence of independent optimization tasks, and parameter θ needs to be decreased over time in order to couple u and v.

Estimating optical flow by means of discrete optimization is difficult, due to large number of labels. The approach by Mozerov [**65**] filters optical flow candidates by the symmetric phase only filter (SPOF), a frame correlation method based on the Fourier transform. The constrained search area is input to a graphical model with gray value and gradient constancy as data energy, and a truncated TV regularity term, followed by a (discrete) subpixel refinement.

The approach by Menze et al. [**60**] "DiscreteFlow" addresses this issue by reducing the optical flow labels to a small set of proposals. These proposals are generated by a nearest neighbor search in a feature space, similar to the PatchMatch approach [**7**]. The nearest neighbor search is implemented efficiently on a randomized k-d tree datastructure. Optical flow proposals are also shared among neighbors, since they are likely to have similar motion. Inference on the graphical model with on optical flow proposals is implemented with block coordinate descent (BCD) [**21**], which is a fast but approximate inference method.

FusionFlow [**50**] takes a similar approach, by fusing the outcome of different continuous methods into a single optical flow field. The labels are generated by the Horn and Schunck and Lucas and Kanade approaches with different parameters, the problem is then reformulated into a binary graphical model and optimized with graph cuts. The final optical flow field is postprocessed with a continuous method, which is very common in discrete optical flow estimation.

5.1.3. Functional Lifting and Convex Relaxation. As in the discrete approaches we presented so far, the problem of optical flow

estimation is viewed as labeling problem, where labels correspond to displacement vectors. This strategy increases the problem size, the unary energies have to be precomputed for all possible displacements. Inference may increase memory requirement further, e.g. if Ishikawa's method is applied, to which we introduce a more memory efficient alternative in the following.

We may interpret the (second order) labeling problem as continuous energy minimization task, assuming that the labels are ordered and the label assignment can change continuously between them. If all unary and pairwise energies were convex, the overall energy is convex, and can be minimized to global optimality. However, the assumption, that all unary energies are convex is unrealistic in the context of optical flow estimation, but the convexity assumption on the pairwise term is satisfied by an $L2$ or $L1$ penalty term.

We now shift from the discrete to the continuous problem formulation, in order to reformulate the labeling problem into a convex energy at the expense of increasing the dimension. Let the (one dimensional) optical flow energy be given by

$$E(u) = \int_{x\in\Omega} E_{\text{data}}(x,u)dx + \int_{x\in\Omega} |\nabla u| dx. \tag{5.5}$$

Let Γ denote the possible range of optical flow. We now introduce the lifted function $\phi : [\Omega \times \Gamma] \to \{0,1\}$,

$$\phi(x,\gamma) = \begin{cases} 1 & u(x) > \gamma \\ 0 & \text{otherwise} \end{cases},$$

with the constraint that $\phi(x,\gamma_{\min}) = 1$ and $\phi(x,\gamma_{\max}) = 0$, $\forall x \in \Omega$. The valid constraint set is denoted with Ω_Γ. Then we can reformulate energy (5.5) in terms of ϕ,

$$E'(\phi) = \int_{(x,\gamma)\in\Omega_\Gamma} |\nabla\phi(x,\gamma)| + E_{\text{data}}(x,\gamma)|\partial_\gamma\phi(x,\gamma)| d(x,\gamma). \tag{5.6}$$

A proof is presented in [**69**]. However, we do not yet have a convex energy in eq. (5.6), since Ω_Γ is not convex. We need to relax the binary set $\{0,1\}$ to the interval $[0,1]$, in order to get a convex relaxed functional in ϕ, which can be optimized with the primal dual algorithm.

The approach requires an ordered label set, hence it is evaluated on stereo cases only. The authors of [**33**] present an extension, where the general flow problem can be modeled with a similar strategy, under the assumption that the pairwise term is separable (e.g. the $L1$ norm),

$$E_{\text{reg.}}(u,v) = E_{\text{reg.}}(u) + E_{\text{reg.}}(v).$$

The data energy needs to be computed for all possible displacements, which limits the approach to small motions.

In the following sections, we describe two approaches of describing optical flow as labeling problem, which address the problem of large displacements.

5.2. Sparse Discrete Optical Flow

We express optical flow estimation as a labeling problem, where labels correspond to precomputed sparse optical flow candidates. The flow candidates have to satisfy several properties, they need to match well according to visual features, and they need to meet constraints that arise from epipolar geometry. The workflow of our algorithm is presented in algorithm 10. In the following sections, we explain the

Input : reference and secondary images I_1 and I_2
Output: sparse flow field u
learn patch characteristics on I_1;
determine distinct locations in I_1;
perform matching, keep a limited number of matches per pixel;
estimate epipolar geometry and filter matches;
infer optical flow by means of discrete optimization;
Algorithm 10: Our sparse discrete optical flow approach.

individual steps in detail.

5.2.1. Learning Patch-based Matching. Given a patch p, we subtract the mean gray value $\overline{p}$ to gain robustness against illumination changes and define the vector

$$f(p) := p - \overline{p}.$$

Given a set of training patches $P = \{p_1, ..., p_n\}$ extracted from the reference frame at each pixel (sufficiently distant from the image border), we apply Principal Component Analysis (PCA) by computing the respective empirical mean μ_P and covariance matrix C_P

$$\mu_P = \frac{1}{|P|} \sum_{p \in P} f(p),$$

$$C_P = \frac{1}{|P|} \sum_{p \in P} (f(p) - \mu_P)(f(p) - \mu_P)^\top.$$

Since C_P is a covariance matrix, it has only real and non-negative eigenvalues. We perform the spectral decomposition

$$C_P = B^\top \Lambda B,$$

where B contains the eigenvectors as its rows, and the diagonal matrix $\Lambda = \mathrm{diag}(\lambda)$ contains the eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots$. We only keep

the first N eigenvectors which represent the components that capture most variance of the data,

$$B' := (b_{ij})_{i=1,\dots,N,j=1,\dots,D},$$
$$\Lambda' = \operatorname{diag}(\lambda_1, \dots, \lambda_N).$$

We can reduce the dimension of our features by projecting onto the subspace of N components which capture most variance,

$$f' = B' f(p),$$

and compute the data energy, e.g. NCC, on the projected features.

The a-contrario model [**76**] is based on the principal components as well, but takes a probabilistic approach to the matching problem. Image statistics are calculated on the reference frame by finding the histograms of the respective principal components. The cumulative histogram of the i'th principal component of patch q is denoted $H_i(q)$. The resemblance probability $\widehat{p^i}_{qq'}$ represents the probability, that the i'th principal components of q and q' occur by chance,

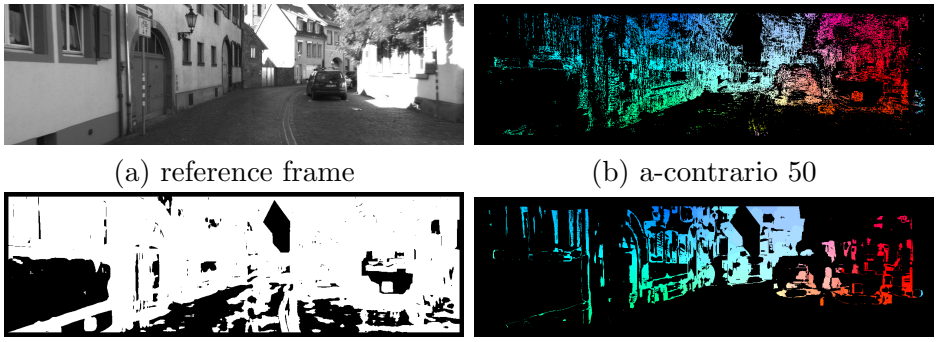
$$\widehat{p^i}_{qq'} = \begin{cases} H_i(q') & \text{if } H_i(q') - H_i(q) > H_i(q) \\ 1 - H_i(q') & \text{if } H_i(q) - H_i(q') > 1 - H_i(q) \\ 2|H_i(q) - H_i(q')| & \text{otherwise} \end{cases}$$

If both components are equal, the resemblance probability is zero, since equality cannot happen by chance. The mismatch probabilities are computed for each component, quantized into a non decreasing sequence of negative powers of two, and multiplied to a single mismatch probability for the pair of patches. The quantization step ensures that mismatches in early components get a higher weight than later ones, since these are visually more perceivable. The full method is presented in algorithm 11.

Input : principal components q and q' and cumulative histograms H_i
Output: mismatch probability p
permute q and q' jointly, so that q is sorted in decreasing order;
compute $\widehat{p^i}_{qq'}$ for each component;
define $p^i_{qq'}$ as quantization of $\widehat{p^i}_{qq'}$;
$p = \prod_i p^i_{qq'}$;

Algorithm 11: a-contrario matching

Discrete patch matching can only be successfully applied in distinct or feature-rich regions. We select the most distinct points in the reference frame and only apply matching on them. Our model can directly evaluate the distinctness of a patch by using its Euclidean distance to



(a) reference frame (b) a-contrario 50

(c) distinct patches (d) NCC 100

FIGURE 5.1. Comparison of the ground truth (c) with our approach and different data terms. Our method recovers large scale motion in distinct areas.

FIGURE 5.2. Matches found by our NCC with 100 principal components. Multiple matches which differ by only one pixel are joined to one here for illustration. In two of three cases, the correct correspondence is included in the candidate list. The example, where the correct match is not found, is a case of disocclusion. Figure 5.3 shows a more detailed view on the patches.

the average patch, this is similar to distinctness as defined in [**58**]. An example of distinct patches is shown in Figure. 5.1.

Figures 5.2 and 5.3 show example matches found by NCC and the a-contrario model. We can observe that the correct match is found, if it is not occluded, and that the wrong matches are similar in appearance.

5.2.2. Graphical Model Construction and Inference. We estimate optical flow by defining a graphical model on optical flow candidates, which are found by matching the patch around the pixels in the

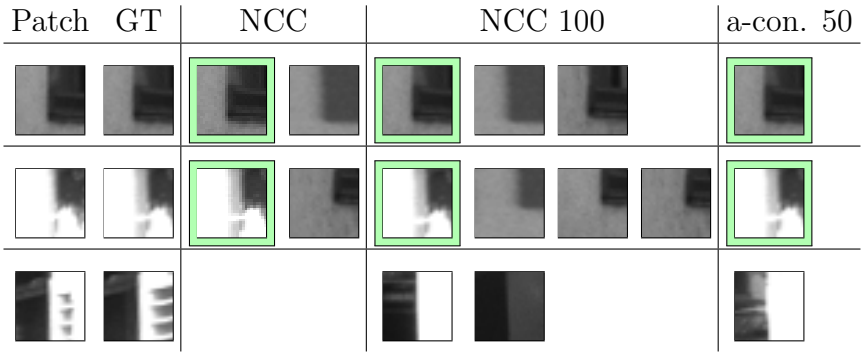


FIGURE 5.3. Close up view of the example matches shown in Figure 5.2. All matchings find the correct correspondence for the first two patches, which are highlighted in green. The third patch is changing due to a disocclusion, so that we cannot expect a correct match. We can see that the original (full rank) NCC finds a subset of the approximate NCC dataterm. The a-contrario model finds exactly the correct match on those two by looking at only the first 50 of 625 principal components, but returns a wrong match on the third patch. Also we can see that patch-wide changes in illumination are correctly ignored by all matchings.

reference image with the second one. In order to accelerate the matching, we only match distinct locations in the first image, and restrict the search space to a maximum displacement. The resulting matches are filtered by the matching score, and limited in number for each pixel. A distinct patch may not appear in the graphical model, if no suitable match can be found.

In addition to the appearance based filtering, we assume a monocular scene setup, and filter matches based on epipolar geometry. We calculate the Essential matrix with the Helmke algorithm [**37**] on keypoint matches found by SIFT [**55**]. For increased robustness against outliers, the Essential matrix estimation is inside a *random sample consensus (RANSAC)* framework. The Essential matrix is estimated on a small sample from the input data, and evaluated against the whole dataset. If a sample describes many input points well, the current estimate is returned, otherwise the procedure is repeated. Even though this approach appears to be heuristic, it has been applied successfully in many contexts in computer vision [**36**]. If the epipolar line constraint is violated, we delete the respective match from the list of candidates.

The epipolar line constraint alone is not strong enough to filter out wrong matches effectively, since only a small portion of the epipolar line contains matches with positive depth. In order to find scene depth, we factorize the Essential matrix into rotation R and translation t, under the assumption that the camera moves forward, and has only small rotation. For a given match, we calculate the respective depth d at

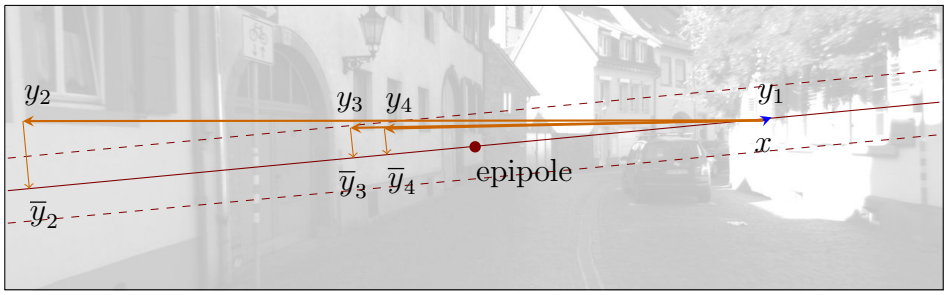


FIGURE 5.4. The candidates of the middle patch in Figure 5.2, x is the patch location in the reference frame. Candidates which are further away from the epipolar line than a given threshold are not admitted by our model. Also, the depth corresponding to a candidate must be positive, where the projected points $\overline{y}_i$ are used for depth computation. Candidate y_2 fails in both these tests, y_3 and y_4 are within the margin but do not correspond to a positive depth. Only the correct match y_1 is admitted for further consideration in the graphical model.

location x in the first image, and depth d' for the corresponding point x' by solving a linear system

$$dx = t + d'x' \;\Leftrightarrow\; \begin{bmatrix} x & -x' \end{bmatrix} \begin{pmatrix} d \\ d' \end{pmatrix} = t.$$

Since the viewing rays do not have to intersect, but can be skew, the system of equations is overdetermined. We find d and d' by calculating the least squares minimum. If depth d is negative, the match is deleted from the list of candidates. Figure 5.4 illustrates the match filtering. Since the Essential matrix estimate is noisy, a patch is only deleted with a large deviation from the epipolar line.

The remaining matches form the labels of a graphical model, whose optimal state is our estimated flow field. We define two pixels as neighbors, if they are closer than a certain radius r. Since we only have labels on a subset of pixels, choosing a local neighborhood may result in many disconnected components. Also, a non local neighborhood reduces the staircasing effect, TV regularization is known for. Neighboring pixels , which are further apart, get a lower weighting. The weight for pixels x and y is given by

$$w(x, y) = \exp\left(-\tfrac{1}{2}\sigma^{-2}\|x - y\|\right).$$

The overall energy is given by

$$E(l) = \sum_{x \in \Omega} E_{\text{match}}(l(x)) + \sum_{(x,y) \in \mathcal{N}} w(x, y)\|u_{l(x)} - u_{l(y)}\|.$$

The label at pixel x is denoted $l(x)$, and u_l denotes the respective displacement vector. The corresponding unary and pairwise factors are

	a-contrario 50			NCC 100		
seq.	dens.	C++	ours	dens.	C++	ours
11	5.28	**13.33**	21.01	2.39	10.78	**9.91**
12	6.69	**4.73**	9.52	5.18	**3.14**	5.53
49	5.99	**4.99**	6.59	1.63	**0.68**	1.63
74	4.96	44.27	**33.91**	0.32	55.56	**54.91**

TABLE 5.1. Sparse optical flow evaluation of the 3px error on training sequences of the KITTI benchmark versus the Classic++ approach. We evaluated in the image regions, where both ground truth flow and estimated flow are available. The data term of the a contrario model gives a denser solution than the NCC but is inferior in accuracy.

constructed as tables, and inference is performed using Tree Reweighted Belief Propagation (TRBP) [**92**].

5.2.3. Evaluation. We test our method by evaluating the optical flow field to the Classic++ estimation method by Sun et al [**83**], which is a highly ranked method on the KITTI benchmark. The patch size is 25×25 pixels. Figure 5.5 shows four example outputs on the KITTI flow benchmark. Our optical flow fields are very sparse, homogeneous regions are not matched due to the low distinctness value.

Table 5.1 shows a comparison of our approach to Classic++. We observe that NCC yields larger accuracy while the a-contrario model returns denser flow fields. In three cases, our approach achieves a lower error, but we should note that the evaluation scheme has a slight favor of our method. Our approach can decide the locations which are evaluated, while the Classic++ method has to find a point correspondence for each pixel.

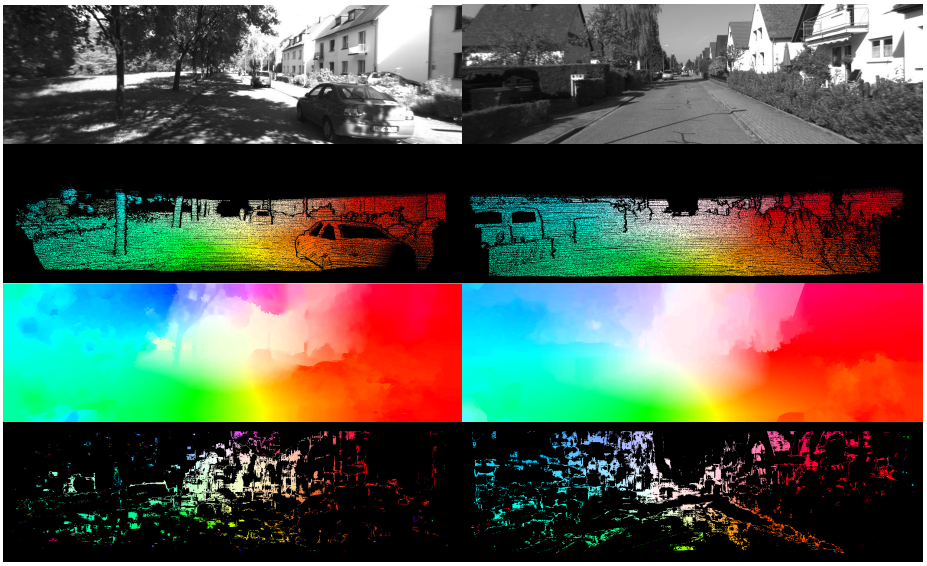
Figure 5.5 shows the output of our method versus Classic++. We see that our flow fields are very sparse, and do not represent the scene structure well. Our matching term is robust against changes in illumination, but does not model changes in scale or orientation, which appear frequently in outdoor traffic sequences.

In the next section, we propose a dense framework, which does not require the filtering of sparse matches, but instead approximates the infeasible dense model by a sequence of small graphical models.

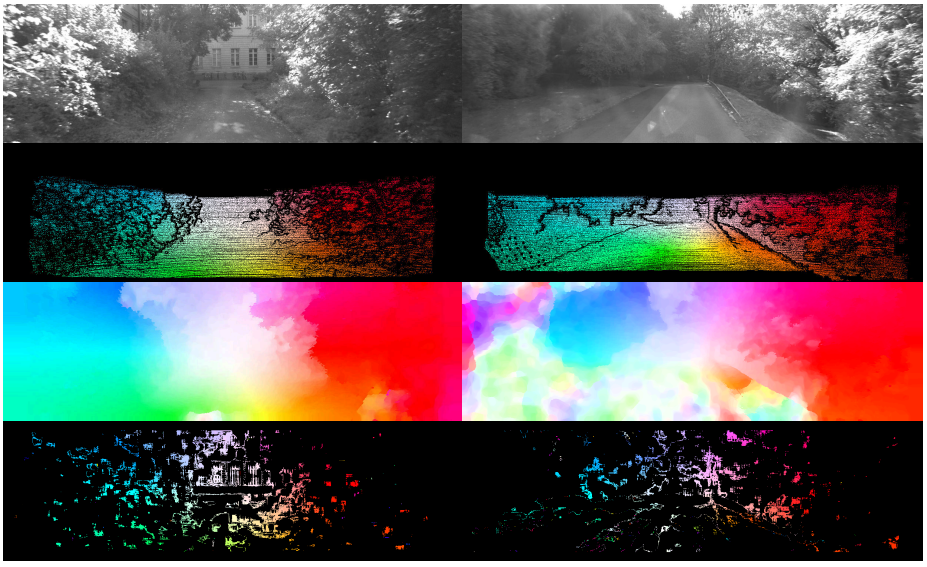
5.3. Dense Hierarchical Label Reduction

In the sparse optical flow framework, we inferred optical flow by minimizing an energy function over a small set of filtered optical flow candidates. For dense output, we wish to minimize the same energy with a dense label set on the pixel grid. Current optical flow methods are required to estimate accurate flow with large motion. Discretizing

optical flow color encoding: 



(a) sequence 11 (b) sequence 12



(c) sequence 49 (d) sequence 74

FIGURE 5.5. Comparison of our sparse optical flow approach to dense variational method C++ [**83**]. For each example we show from top to bottom the reference frame, (sparse) ground truth optical flow, the output of Classic++ and our approach with data term NCC and 20 principal components. There are difficult lighting conditions, especially in sequence 74, where the Classic++ method produces erroneous optical flow in large regions. Our method produces very sparse optical flow fields.

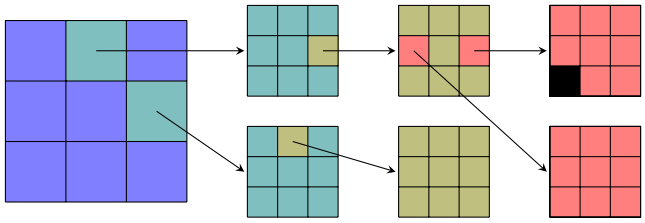


FIGURE 5.6. Label selection mechanism for a 2D optical flow field. The square represents possible displacement vectors for a fixed pixel. In the first iteration the displacement grid is partitioned into a 3×3 grid (3-tiling). Inference in the reduced graphical model is performed, the best two labels are chosen and refined. This procedure iterates until an individual label has been selected, here filled in black.

FIGURE 5.7. A graphical model with two vertices and an edge between them. The first vertex has three possible labels, and the second five. The labels are clustered into a blue and an orange set. The reduction is performed independently on each factor.

the search space with pixelwise resolution generates large graphical models, where already the unary energy values will not fit into the memory of a modern computer.

We aim at approximating an infeasible model with a sequence of small models.LogCut [**51**] achieves logarithmic complexity by solving for the variable bitwise, where each bit is found by inference on a binary graphical model with graph cuts. Similarly, we define a hierarchy of labels, and find a path through the tree by solving a small graphical model at each step. The reduction is based on divergence between the respective probability distributions.

In case of optical flow, there is a natural clustering of labels into a regular grid structure, we refer to the grid size as tiling. Figure 5.6 shows a hierarchical path through label clusters, until a single optical flow label is selected. Instead of selecting the single optimal cluster in each step, we allow selection of multiple clusters, in order to track multimodal distributions more accurately. In our experiments, we perform inference with TRW-S, which calculates approximate marginals, that allow us to select multiple best labels. Figure 5.7 shows an example reduction into a binary graphical model.

5.3.1. Reduced Graphical Model Parameters. In this part we derive the corresponding energy function for the proposed label reduction technique. We define that clustering $C = \{c_k : k = 1, ..., K\}$ to be the collection of all clusters. To each of these clusters, we associate a probability denoted by q_k. Each factor of the given graphical model is reduced separately.

Let p_l denote the probability of label l in the respective factor. A suitable measure of distance is given by the Kullback-Leibler (KL) [**45**] divergence,

$$KL_p(q) = \sum_{c_k \in \mathcal{C}} \sum_{l \in c_k} p_l \log \left(\frac{p_l}{q_{c_k}} \right) \tag{5.7a}$$

$$= - \sum_{c_k \in \mathcal{C}} a_{c_k} \log q_{c_k} + \text{const.}, \quad a_{c_k} = \sum_{l \in c_k} p_l. \tag{5.7b}$$

The second equality follows from properties of the logarithm and that the probability summed over the clusters are constant.

In order to minimize the KL distance, we need to differentiate (5.7b), and solve for a vanishing gradient. However, since q represents a probability distribution, the derivative needs to be projected to the tangent space of the probability simplex. The projection is given by

$$P_{\Delta_K} = I - \frac{\mathbb{1}\mathbb{1}^\top}{\mathbb{1}^\top \mathbb{1}}, \quad \mathbb{1} = \begin{pmatrix} 1, & \dots, & 1 \end{pmatrix}^\top \in \mathbb{R}^K,$$

where I is the identity matrix. Let $a = (a_1, ..., a_K)$ the vector of a_{c_k} introduced in (5.7b), then differentiating (5.7b) w.r.t. q results in

$$\nabla_q KL(p \| q) = P_\Delta \left(\frac{a}{q} \right),$$

where the division is done element-wise. We note that the stationary point is obtained if and only if

$$\frac{a}{q} = \mathbb{1}.$$

Or equivalently, we obtain the *exact* reduced labeling scheme if the cluster's labels correspond to the probabilities of the attaining a label in the original label space, *i.e.*,

$$q_{c_k} = \sum_{l \in c_k} p_l, \quad \forall c_k. \tag{5.8}$$

Furthermore, we see that the reduction scheme is invariant to addition on model energies. We move from the probabilistic viewpoint to the energy formulation. We get the following reduced energy for a cluster c

$$E_{c_k} = - \log \left(\sum_{l \in c_k} \exp(-E_l) \right).$$

Adding a constant to each energy will increase the reduced energy E_{red} by the same constant. However, the reduced energies are *not* invariant to multiplicative scaling of the energies. The identification of the proposed reduction scheme as an instance of the generalized α-divergence, presented next, gives an explicit interpretation of the scale dependency.

5.3.2. Label Reduction Based on the α-Divergence. The introduced label reduction technique can be viewed as an instance of the general α-divergence [**62**]. Let $\alpha \in \mathbb{R} \setminus \{0, 1\}$, then the α-divergence is defined as,

$$D_\alpha(p \parallel q) = \frac{1}{\alpha(1-\alpha)} \sum_l \alpha p_l + (1-\alpha) q_{c(l)} - p_l^\alpha q_{c(l)}^{1-\alpha}.$$

Note that in the limit $\alpha \to 1$, D_α corresponds to the standard KL divergence measure between p and q. Assuming we have found the optimal labeling in the original label space, we need to find the corresponding optimal labeling of the clusters. We minimize $D_\alpha(p \parallel q)$ over the probabilities q. There exist a closed from solution for the global minimizer with vanishing gradient,

$$P_\Delta \left(\frac{1}{\alpha(1-\alpha)} \sum_l (1-\alpha) - (1-\alpha) p_l^\alpha q_{c(l)}^{-\alpha} \right) = 0,$$

with the projection as in (5.3.1). In case of optimality, the projection is invariant under additive and multiplicative constants, therefore optimality is given by

$$\sum_l p_l^\alpha q_{c(l)}^{-\alpha} \propto \mathbb{1}. \tag{5.9}$$

Equation (5.9) implies the following relation between the cluster probabilities and the label space probabilities. For all clusters $c_k \in C$, we have

$$\left\{ q_{c_k}^{-\alpha} \sum_{l \in c_k} p_l^\alpha \stackrel{!}{=} 1, \ \forall c_k \in C \right\} \quad \Rightarrow \quad q_c = \left(\sum_{l \in c} p_l^\alpha \right)^{\frac{1}{\alpha}}. \tag{5.10}$$

Expression (5.10) matches our previous result (5.8) with $\alpha \to 1$. Since the probability values p_i are calculated from the energies, as given by (3.4) p_i^α corresponds to the probability of the scaled energy αE_l. Scaling factor α can be viewed a normalization constant, *i.e.*, the operation in (5.10) scales the objective function, and after performing the KL reduction it restores the original scale. As $\alpha \to \infty$, the expression (5.10) converges to the maximum norm.

As $\alpha \to 0$, the α-divergence converges to the KL divergence between q and p, $\lim_{\alpha \to 0} D_\alpha(p \| q) = \mathrm{KL}(q \| p)$. The divergence measure and its

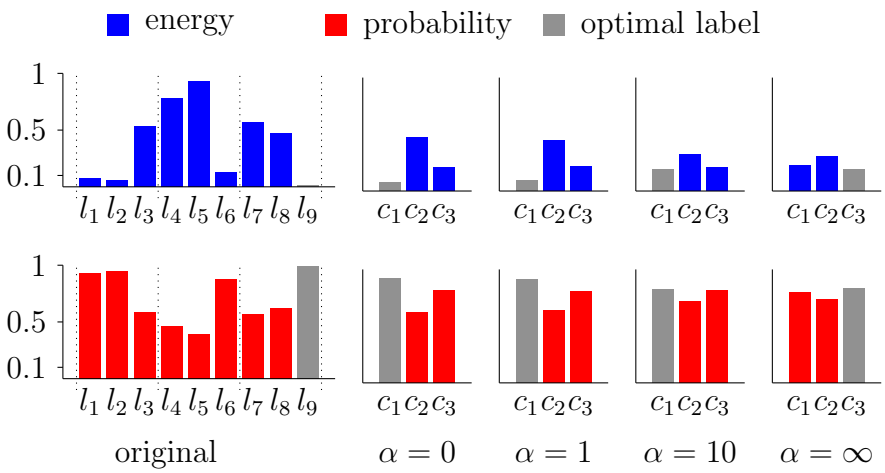


FIGURE 5.8. Reduction output on nine random labels, which are clustered into three clusters of equal size. We minimize the energy, which corresponds to a maximization of the probability. In this case, the optimal label shares the same cluster with low-energy labels, so that the correct cluster is found only for $\alpha = \infty$.

minimum with respect to the cluster probabilities is given by

$$KL(q\|p) = \sum_{c\in\mathcal{C}} \sum_{l\in c} q_c(\log q_c - \log p_l),$$
$$\nabla_q KL(q\|p) = \sum_{l\in c_i} \log q_{c_i} - \log p_l + \frac{q_{c_i}}{q_{c_i}},$$

whereby we get, after solving $\nabla_q KL(q\|p) = 0$,

$$q_{c_i} = \exp\left(\frac{\sum_{l\in c_i} \log p_l}{|c_i|}\right) = \exp\left(-\frac{\sum_{l\in c_i} -\log p_l}{|c_i|}\right).$$

In this case, the minimum value corresponds to averaging the respective energies.

Based on the interpretation of (5.10), we have shown that minimizing the α-divergence between given label probabilities and cluster probabilities correspond to an energy mean for $\alpha = 0$, a soft-min for positive α and the hard min for $\alpha = \infty$. Figure 5.8 shows the reduction output on a small example of nine random labels. These labels are merged into three clusters of equal size. The top row shows the energy values and the the corresponding probabilities for different values of α. As clearly seen, only $\alpha = \infty$ corresponds to the correct labeling in this unary case. This example illustrates that single labels with low energy clustered together with high energy labels are lost with a small value for α.

		global	3-tiling		5-tiling	
			inf.	2-marg.	inf.	2-marg.
EPE	GT noc	1.24	2.23	2.05	**2.04**	2.13
	GT occ	1.38	2.80	2.40	2.51	**2.39**
	global	0	4.66	3.98	4.12	**3.74**
2px	GT noc	9.22	11.94	**11.19**	11.69	14.37
	GT occ	10.64	13.82	**12.91**	13.54	15.97
	global	0	17.28	13.93	16.14	**12.68**

TABLE 5.2. KITTI stereo errors. The 2-marginals version achieves lower error compared to the global optimal solution, but does not necessarily outperform the single cluster selection when compared against ground truth.

Choosing a strictly positive value for α introduces a scale dependency, scaling the energy function can be compensated by scaling adjusting α accordingly. By selecting $\alpha = \infty$, we eliminate this additional degree of freedom and do not need to rescale our energy values.

5.3.3. Evaluation. We evaluate different aspects of the proposed dense flow computation method. Starting with the choice for α and the first aspect is runtime, we proceed to stereo and optical flow estimation performance. All optical flow and stereo models we evaluate use the NCC dataterm and TV as pairwise energy.

In Figure 5.9, we see the impact of changing α on a small stereo model. We observe that an increased tiling improves the result for all values of alpha. Increasing α generally improves the accuracy of the reconstruction. At $\alpha = 0$, the reduction is the mean operation on the energies inside the cluster, the single correct label inside a cluster does not have enough influence on the mean energy. However, at $\alpha = \infty$, the reduction is the min operation, so that the energy of the best label will be selected. We see that following two peaks using the marginals instead of applying inference at each level introduces jitter in the labeling.

Figure 5.10 shows the inference runtimes of direct inference and the proposed dense hierarchical approach as a function of label size. We observe that our algorithm has logarithmic increase in runtime, while the runtime is worse for small models. The global model uses an optimized version for the truncated $L1$ pairwise term, because global inference would not be feasible for the large models otherwise. Even though the global model is using a dedicated solver, the reduced model performs better in terms of runtime.

In case of stereo estimation, inference on the global model is feasible, and we can compare our method to both ground truth and the global model optimum. In Table 5.2, we can see that the finer 5-tiling

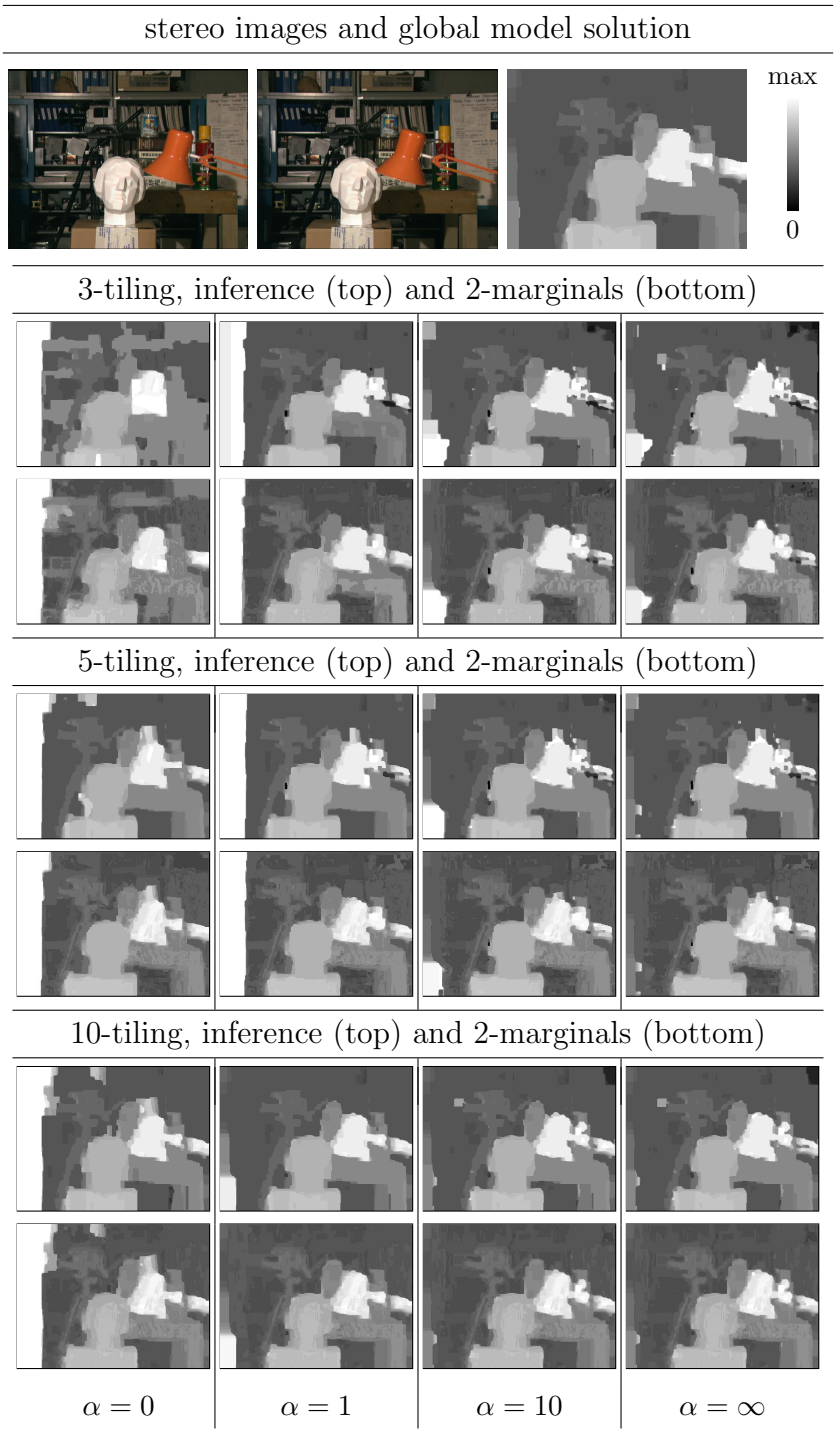


FIGURE 5.9. Results on the Tsukuba dataset for different values of α. We can observe that larger values of α and a finer tiling improve the accuracy. Especially with $\alpha = 0$ and the 3-tiling the scene structure is lost to a large extent, with increasing α we capture increasingly more details of the scene. There are visible artifacts introduced by using the 2-marginals selection.

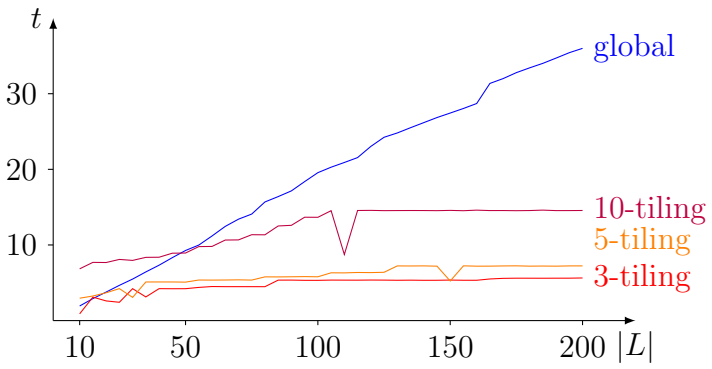


FIGURE 5.10. Inference runtime as function of the number of labels for the proposed reduction scheme and direct inference. While the original model shows linear runtime, our approach is logarithmic. We make use of an optimized version of TRW-S for the truncated L1 penalty function for the global model, but not for the reduced models. Therefore the 10-tiling performs worse for ten labels, even though the reduced model is identical to the global one. If the tiling happens to align well with the number of labels, the reduced method can perform better, which explains the positive outliers for the reduced models.

and the tracking of two clusters rather than one increases accuracy when comparing to the global model, but not against the ground truth data. Jitter artifacts are introduced by the multimodal tracking of two clusters, as we can also see in figures 5.11 and 5.12. The greedy reduced method performs better compared to ground truth, than to the global model. The TV model does not necessarily reflect the scene structure well, assuming a piecewise affine rather than piecewise constant scene would probably perform better. The greedy selection based on matching score improves on the end point error score.

In case of optical flow, the global model is infeasible for direct inference, and we can compare our reduction approach only to the ground truth flow. Figures 5.13 and 5.14 show example results on the sintel optical flow benchmark. Due to the large label space and the previously seen jittering effects, we do not evaluate the 2-marginal reduction, but only inference on the reduced models. Our approach has difficulties with string motion blur or low texture.

Tables 5.3 and 5.4 show optical flow errors in comparison to the DiscreteFlow method by Menze et al. [**60**] for each sequence of the sintel training set. The nearest neighbor flow proposals of DiscreteFlow run on multiple scales, and thus achieve a better unary energy, which leads to lower error in both measures. Postprocessing improves the estimated flow field slightly by subpixel refinement and removal of outliers.

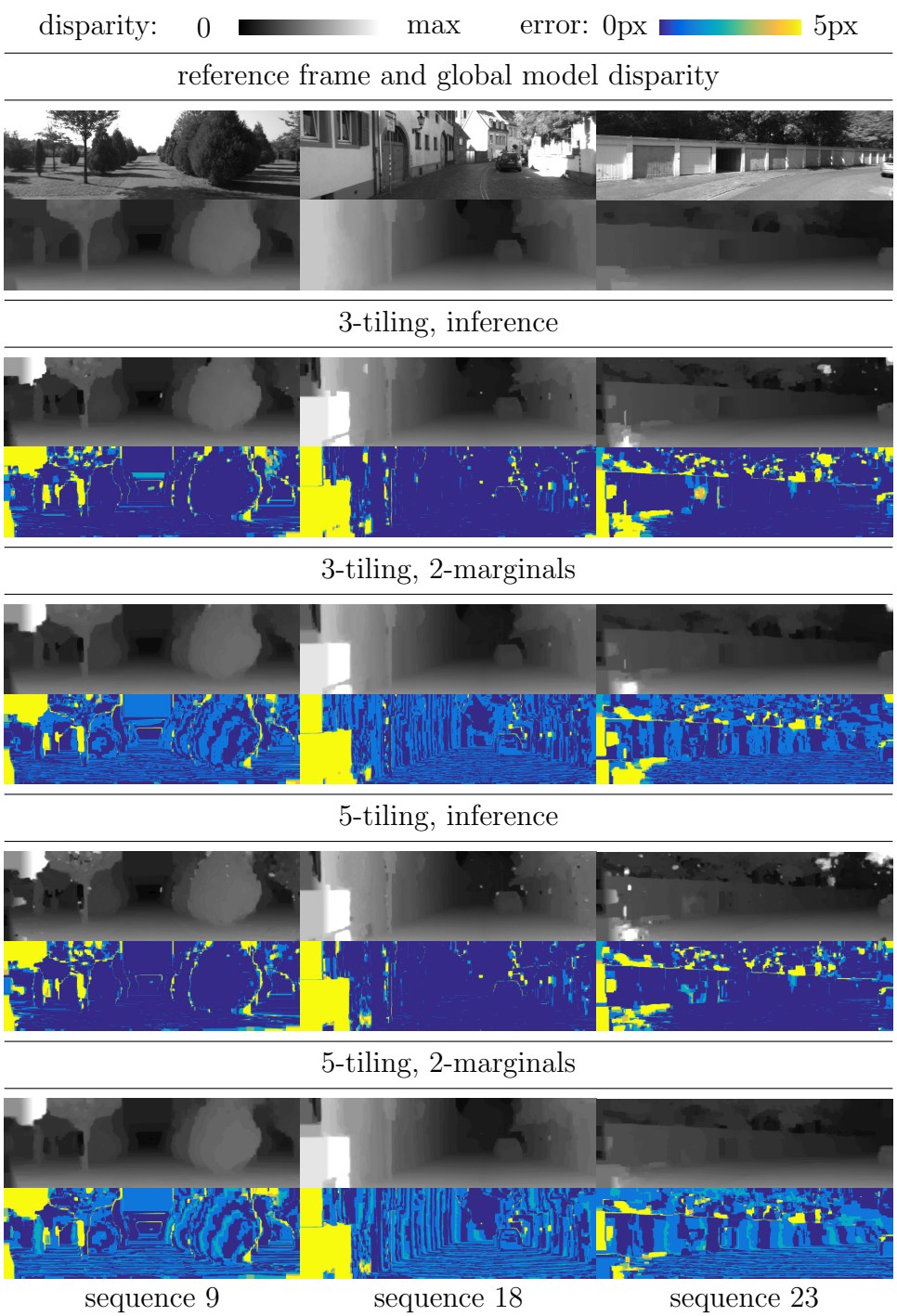


FIGURE 5.11. KITTI stereo examples with computed disparity and error in comparison to the global model. The 2-marginals strategy introduces jitter artifacts, as we can see in the errors. Most reconstruction errors occur in the lower left part of the images, where no correspondence exists, especially when the region has low texture.

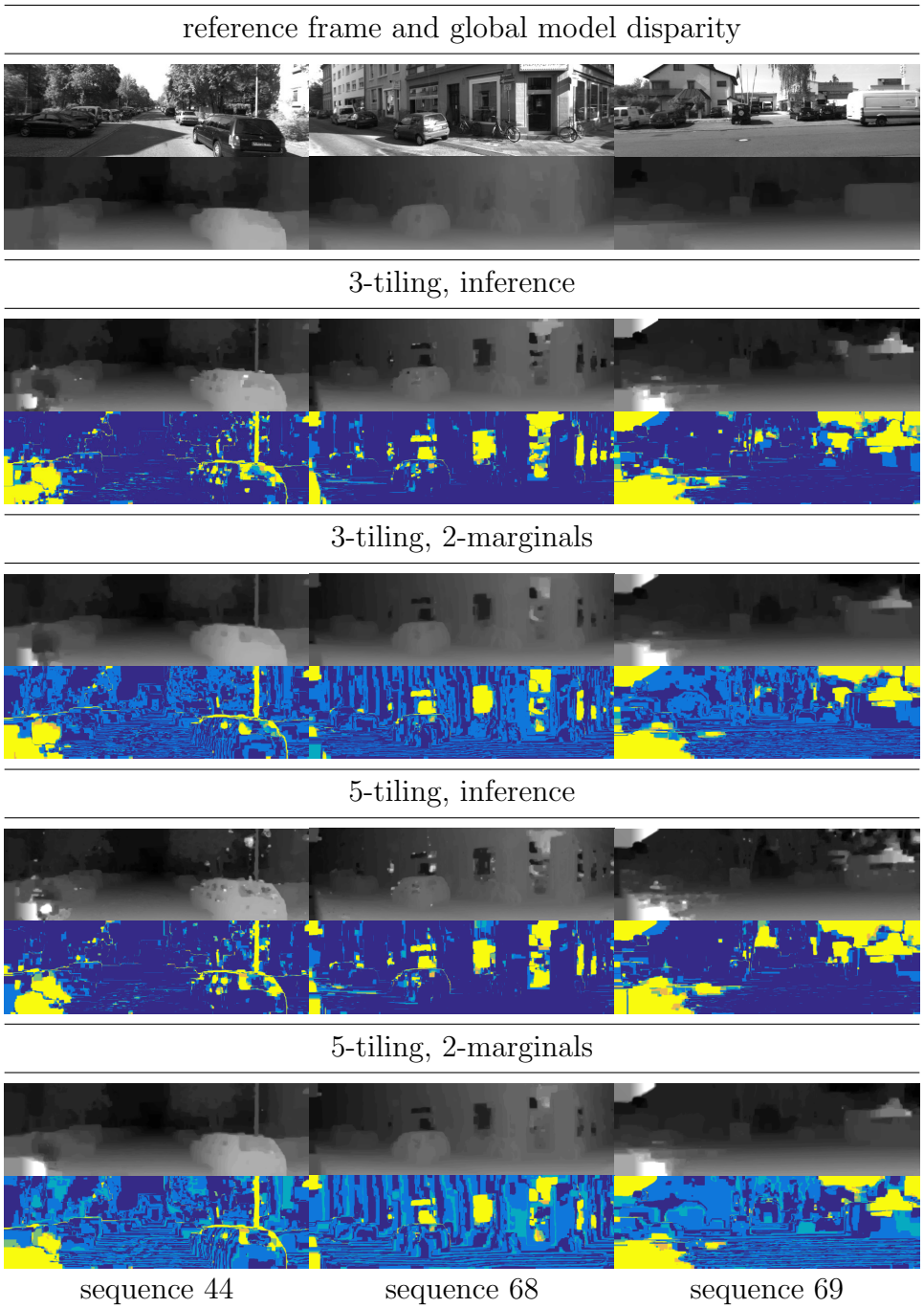


FIGURE 5.12. Further KITTI stereo examples. In some cases, the greedier reduction approach outperforms the global TV based model. The pole in the left image is reconstructed by the reduced, but not by the global model. The global model can fill regions with low unary information, while the reduced model filters out the correct label cluster too early, as we can see at the windows in seq. 68.

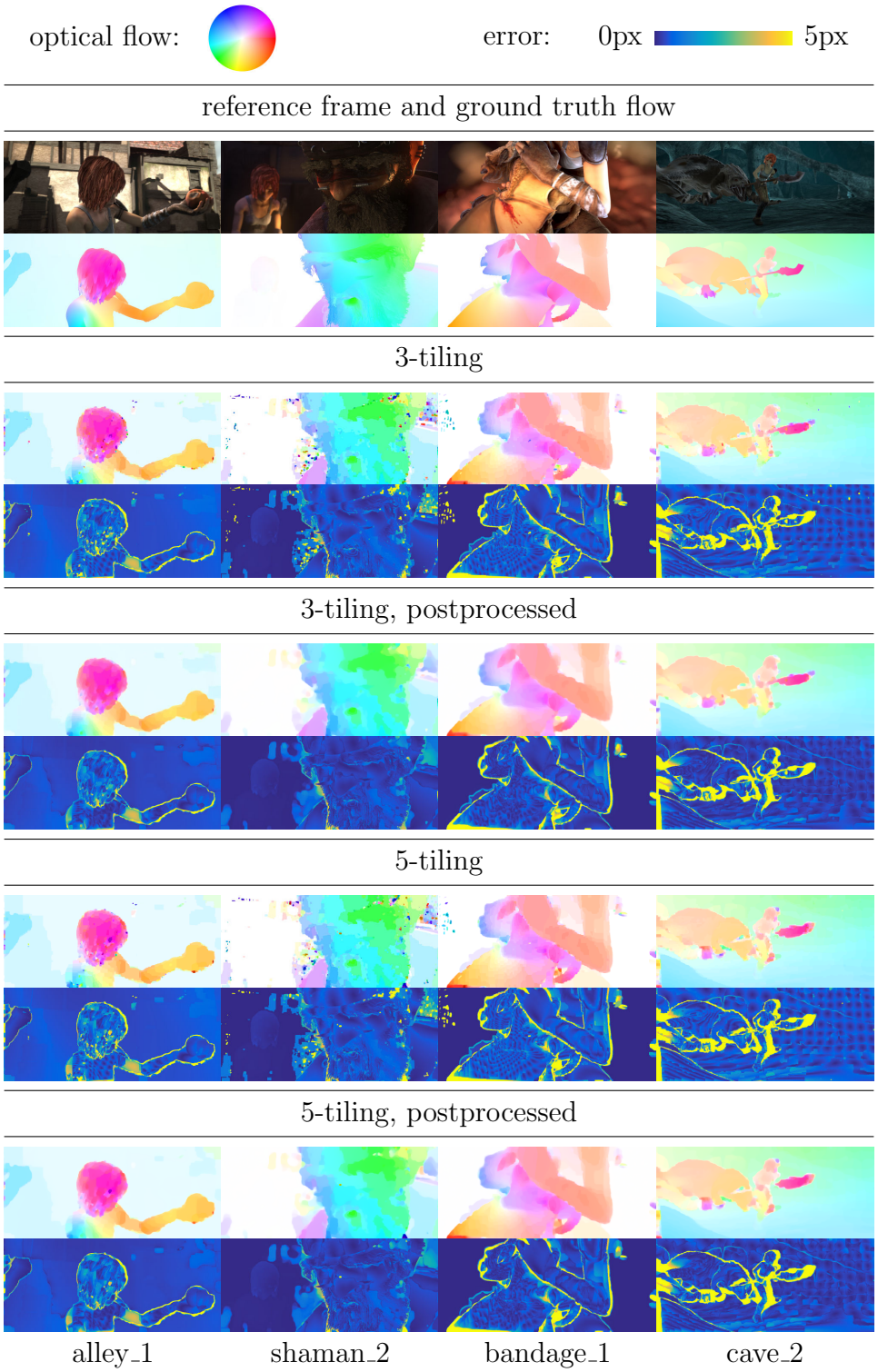


FIGURE 5.13. Sintel optical flow examples with error map. We can see that most errors occur in occluded regions and grid artifacts in the bandage_1 and cave_2 sequences due to label discretization.

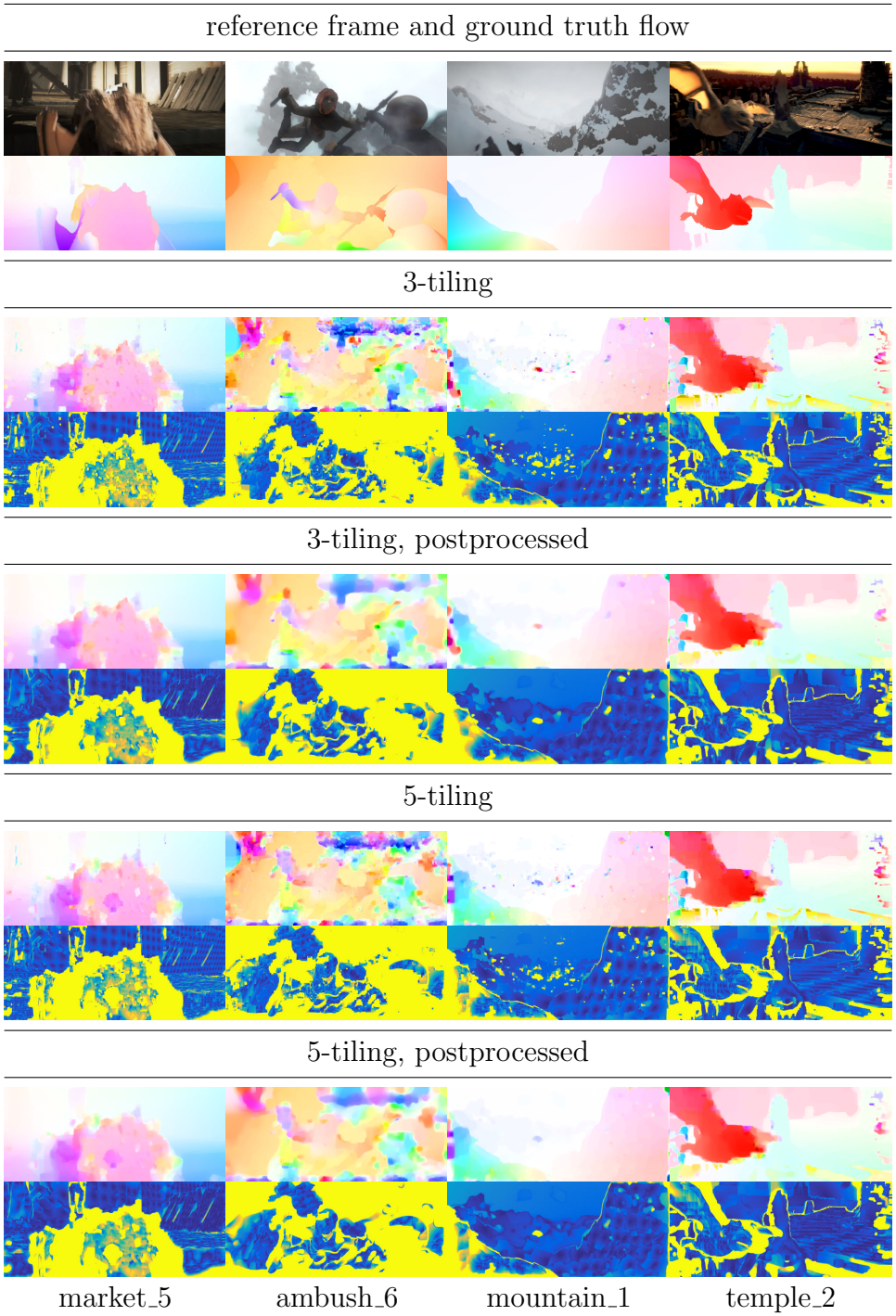


FIGURE 5.14. Further sintel optical flow examples. In the market_5 sequence, we have strong motion blur, which leads to an unreliable dataterm. Similarly, low texture leads to optical flow errors, as seen in the ambush_6 example. The mountain_1 sequence has low texture, but very uniform motion, which simplifies optical flow estimation.

	3-tiling		5-tiling		DF
postproc.	off	on	off	on	
alley_1	0.67	0.58	0.65	0.57	0.17
alley_2	0.74	0.60	0.68	0.54	0.13
ambush_2	44.27	44.20	42.67	42.85	27.04
ambush_4	26.99	25.60	26.52	25.35	21.09
ambush_5	2.33	2.18	2.13	2.01	1.08
ambush_6	24.07	21.44	22.52	20.36	11.32
ambush_7	1.03	0.86	0.93	0.78	0.32
bamboo_1	0.67	0.55	0.65	0.54	0.24
bamboo_2	0.62	0.52	0.59	0.50	0.34
bandage_1	0.98	0.82	0.93	0.79	0.52
bandage_2	0.99	0.94	1.00	0.97	0.65
cave_2	3.84	3.70	3.43	3.32	1.43
cave_4	3.85	3.69	3.60	3.48	2.21
market_2	1.43	1.20	1.41	1.20	0.70
market_5	20.69	19.78	19.45	18.76	8.13
market_6	4.23	3.94	3.33	3.12	1.23
mountain_1	1.80	1.42	1.64	1.34	0.52
shaman_2	0.62	0.41	0.59	0.41	0.19
shaman_3	0.91	0.74	0.84	0.69	0.28
sleeping_1	1.31	0.79	1.29	0.81	0.10
sleeping_2	0.46	0.34	0.44	0.32	0.06
temple_2	5.28	3.40	5.35	3.64	1.06
temple_3	3.12	2.84	3.01	2.79	1.00

TABLE 5.3. Sintel optical flow end point errors in pixels. Postprocessing improves the optical flow field, but the DiscreteFlow method has lower errors in almost all cases. It has the advantage of a multiresolution dataterm, which is more robust against motion blur and other deformations. Due to the filtering of optical flow candidates, the DiscreteFlow method also runs faster than our approach.

	3-tiling		5-tiling		DF
postproc.	off	on	off	on	
alley_1	3.50	2.85	3.38	2.75	1.35
alley_2	2.90	2.55	2.61	2.26	0.81
ambush_2	88.24	87.67	87.12	86.32	71.53
ambush_4	62.81	61.56	61.08	59.75	49.26
ambush_5	21.37	18.59	19.90	17.44	12.50
ambush_6	64.23	62.93	62.70	61.34	48.50
ambush_7	5.46	4.04	5.03	3.87	2.38
bamboo_1	4.46	3.56	4.42	3.54	2.02
bamboo_2	2.60	1.99	2.46	1.93	1.19
bandage_1	8.70	7.35	8.16	6.98	5.09
bandage_2	10.92	9.78	10.79	9.64	8.26
cave_2	15.74	15.22	15.30	14.84	8.71
cave_4	15.81	15.54	15.54	15.45	10.20
market_2	10.00	8.93	10.03	9.11	6.21
market_5	49.39	46.92	48.61	46.45	36.35
market_6	23.47	21.51	22.11	20.07	12.31
mountain_1	14.91	10.27	13.96	10.13	5.00
shaman_2	4.36	1.96	4.05	1.95	0.76
shaman_3	5.26	3.55	4.41	2.94	2.09
sleeping_1	5.50	4.16	5.33	4.12	0.01
sleeping_2	0.55	0.36	0.42	0.27	0.00
temple_2	22.59	19.67	22.90	20.91	9.00
temple_3	25.39	22.40	25.04	22.12	8.35

TABLE 5.4. Sintel optical flow 2px errors in percent. Similarly to the end point error, the DiscreteFlow method performs better than the proposed hierarchical approach. The 2px error reflects error in the image domain, whereas the end point error measures th error in the flow domain.

CHAPTER 6

Structure from Motion

Assuming that point correspondences are known, we want to reconstruct the world, which was recorded by the camera. Many methods select a set of keyframes and compute bundle adjustment on the frames between them, these are *batch approaches*. This separation into keyframes increases latency, since many frames need to be captured before bundle adjustment can be performed. An alternative approach is given by *filtering*, where the reconstruction of the current frame is computed directly using prior information from previous frames. The filter remains fully responsive, since the reconstruction is computed on each input frame immediately. In the case of car driving assistance, fast response and dense reconstruction are vital requirements.

There are different mathematical models for the filtering problem, of which we present the Gauss-Newton and the minimum energy filters in the next section. Afterwards, we present a piecewise planar dense scene reconstruction method for two frames, which can form the basis of a filter for long sequences.

6.1. Monocular Reconstruction based on Filtering Techniques

We briefly present the *Gauss-Newton filter*, which is the basis for the approach presented in the next section. The state at frame $k+1$ depends on the new observation and the states at frames 1 to k, thus achieving a smooth transition between frames. For example, we may have a quadratic prior x^k of our variables x^{k+1}, which is added to a quadratic data driven energy with observation z^{k+1},

$$\begin{aligned} E(x^{k+1}) =& \frac{1}{2}(x^{k+1} - x^k)^\top H_{\text{prior}}(x^{k+1} - x^k) \\ &+ \frac{1}{2}(z^{k+1} - z(x^{k+1})^\top H_{\text{data}}(z^{k+1} - z(x^{k+1}). \end{aligned}$$

An application to monocular SLAM is presented in [**81**], where each frame is mapped into the same reference view.

6.1.1. Variational Structure from Motion. In [**9**], camera movement and dense scene depth are propagated and updated at each frame. The approach formulates inference in a probabilistic framework, where state $X^k = (d^k, C^k)$ consisting of depth and camera parameters in

terms of observation Y^k is given by

$$p(X^k|Y^{1:k}) \propto \underbrace{p(Y^k|X^k)}_{\text{observation}} \underbrace{p(X^k|X^{k-1})}_{\text{prior}}.$$

The respective energy is given by a photometric unary energy, and a quadratic regularity term,

$$\begin{aligned} E(X^k) &= -\log p(X^k|Y^{1:k}) \\ &= E_u(d^k, C^k) + E_C(C^k) + E_d(d^k). \end{aligned}$$

The probabilistic formulation provides the method with uncertainties, which improve the prediction, and may also be of interest in the output. The uncertainty is defined as second derivative of the corresponding energy,

$$\sigma_x^{-2} = \frac{\partial^2}{\partial x^2} \log -p(x).$$

We explain the components very briefly here, more details are given in the paper. $E_u(d^k, C^k)$ penalizes the quadratic difference to observed optical flow,

$$E_u(d^k, C^k) = (\hat{u}^k - u(C^k, d^k)) \left(\Sigma_{\hat{u}}^k\right)^{-1} (\hat{u}^k - u(C^k, d^k)).$$

The variables $\hat{u}$ and $\Sigma_{\hat{u}}$ are determined by the Lucas and Kanade optical flow algorithm with Gaussian weight G_ρ,

$$\begin{aligned} \hat{u} &= \Sigma_{\hat{u}} \left((G_\rho(x) * \nabla I(x))^\top \partial_t I\right) \\ \Sigma_{\hat{u}} &= \left((G_\rho(x) * \nabla I(x))^\top \nabla I(x)\right)^{-1}. \end{aligned}$$

$E_C(C^k)$ is small, if the camera motion is close to its predicted value,

$$E_C(C^k) = \frac{1}{2} \operatorname{dist}^2_{\mathrm{SE}(3)}(C^k, C^{k-1}, \Sigma_C^k).$$

Σ_C^k denotes the current standard deviation on SE(3), which defines the distance of the extrinsic camera parameters on the special Euclidean manifold.

The depth energy $E_d(d^k)$ is given by

$$E_d(d^k) = \frac{1}{2} \sum_{x \in \Omega} \left(\frac{d^k(x) - \hat{d}^k(x))^2}{\sigma_d^k(x)} + \frac{1}{\sigma_s^2} \|\nabla d^k\|^2.$$

The predicted depth $\hat{d}^k$ is determined from the previous frame, together with depth uncertainty σ_d^k. σ_s is a constant regularity parameter.

Optimization is performed using a Newton method with backtracking line search. The method is performed on a multiscale pyramid, as this is required by the Lucas and Kanade approach.

6.1.2. A Novel Minimum Energy Filter for Visual Odometry. In contrast to the approach presented above, which implements temporal smoothing with local prediction, the minimum energy filter provides a global optimality criterion for the entire sequence. The minimum energy filter can be computed recursively, using the reformulation as optimal control problem presented by Mortensen [**64**].

In our approach [**11**], we describe a minimum energy filter on the special Euclidean group SE(3). The camera movement is modeled as continuous curve $E(t) \in \mathrm{SE}(3)$, assuming zero acceleration,

$$\begin{aligned} \dot{E}(t) &= V(t) \\ \dot{V}(t) &= 0 \end{aligned} \tag{6.1}$$

Our state variable $x(t)$ is given by the pair (E, V). Due to the Lie group structure of SE(3), the derivatives are subject to the respective tangent spaces, and we can rewrite the equation system (6.1),

$$\dot{x}(t) = f(x) = (E(t)v(t), 0).$$

More details are given in [**11**]. The zero acceleration assumption does not explain the camera poses perfectly, due to acceleration and noise in the process, hence we add noise variable $\delta(t)$ to our state equation. We cannot observe the state variable directly, but only through estimated optical flow $\hat{u}$, which is expected to be close to the parameterized optical flow field as given by eq. (6.3). Again, both flow fields will not match perfectly in general, and we require noise variable $\epsilon(t)$. The state and observation system is given by

$$\begin{aligned} \text{(state)} \quad & \dot{x}(t) = x(t)(f(x(t)) + \delta(t)), x(0) = x_0 \\ \text{(observation)} \quad & \hat{u}(t) = u(x(t), d(t)) + \epsilon(t). \end{aligned}$$

Note that the depth d(t) is not part of the variables, but assumed to be known or estimated independently.

We introduce energy function $\mathcal{J}(\delta, \epsilon, t_0, t)$, in order to minimize the noise variables $\delta(t)$ and $\epsilon(t)$ as well as deviation from the initial state $x_0 = (I_4, 0)$,

$$\begin{aligned} \mathcal{J}(\delta, \epsilon, t_0, t) &= \frac{1}{2} e^{-\alpha(t-t_0)} \left(\|x - (x_0, 0)\|^2 + \int_{t_0}^{t} c(\delta, \epsilon, \tau, t) d\tau \right) \\ c(\delta, \epsilon, \tau, t) &= \|\delta(\tau)\|_S^2 + \|\epsilon(\tau)\|_Q^2. \end{aligned}$$

$S \in \mathbb{R}^{12\times12}$ and $Q \in \mathbb{R}^{2\times2}$ are arbitrary positive definite weighting matrices. This objective forms the basis of the minimum energy filter, which is reformulated into the recursive framework in [**11**]. The approach also extends to constant acceleration rather than constant velocity and higher order models.

6.2. Depth and Normal Regularization

We model the scenes as piecewise planar, which is well suited especially for urban environments. A plane in 3D can be described by plane equation

$$n^\top X = q, \text{ with } n \in \mathbb{R}^3,\ q \in \mathbb{R}.$$

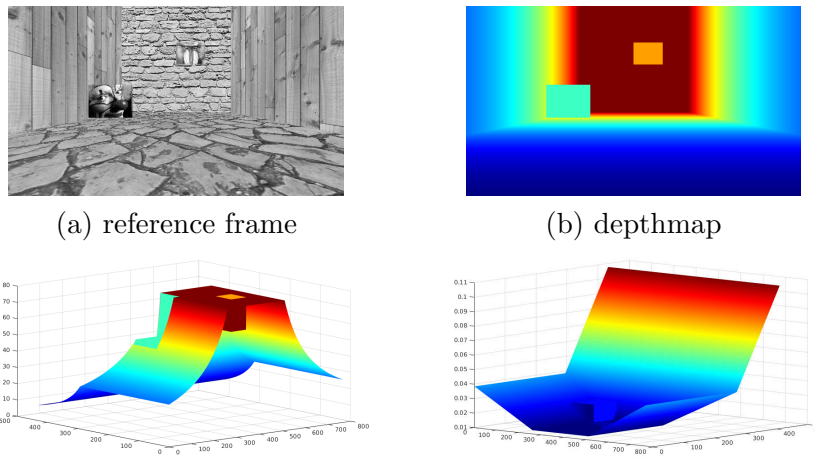
Capital letters represent 3D scene points, lowercase letters denote homogeneous points, that are represented on the image plane at $z = 1$. Assuming that $q \neq 0$, we can divide this equation by q, and writing $v = n/q$, we get

$$v^\top X = 1.$$

With $q = 0$, we would have a degenerate plane that intersects with the camera center, hence the assumption $q \neq 0$ does not restrict our model. There is an affine relation between (inverse) scene depth z and plane parameters,

$$X = \frac{x}{z} \ \Rightarrow\ 1 = \frac{v^\top x}{z} \ \Rightarrow\ z = v^\top x.$$

Also, we can see here that the first two components of v denote the affine parameters of inverse depth in the image plane, if viewing the same plane. Absolute depth of a plane is not affine, which makes it more natural to represent depth by its inverse, as shown in Figure 6.1.



(a) reference frame (b) depthmap

(c) depth $d(x)$ (d) $z(x) = 1/d(x)$

FIGURE 6.1. Simple rendered scene consisting of a few planes. The inverse depth parameterization z is affine on planar scene segments, while absolute depth d induces a curved relation.

6.2.1. Piecewise Planar Depth Map Smoothing. We assume that we are given an estimated depth map $\tilde{z}$, and compare different piecewise planar regularity energies, in order to recover the original scene. The data energy is given by the quadratic depth difference,

$$E_{\text{data}} = \frac{1}{2}\|z - \tilde{z}\|^2.$$

Total generalized variation (TGV) is known to preserve piecewise affine structures, which in our case represents a piecewise planar scene. The TGV model is given by

$$\text{TGV}(z, w) = \alpha_1\|\nabla z - w\|_1 + \alpha_0\|\nabla w\|_1.$$

We rewrite the energy as saddle point problem,

$$\begin{aligned} E(z, w, p, q) =& \frac{1}{2}\|z - \tilde{z}\|^2 + p^\top \begin{bmatrix} A & -I \end{bmatrix} \begin{pmatrix} z \\ w \end{pmatrix} + q^\top C w \\ & - \delta_{\alpha_1[-1\ 1]^{|p|}}(p) - \delta_{\alpha_0[-1\ 1]^{|q|}}(q). \end{aligned}$$

The matrix A denotes the gradient operator, where x and y gradients stacked. C is the second order gradient, therefore it contains mixed gradients as well. We can then apply the primal-dual algorithm to this energy, as shown in Algorithm 12.

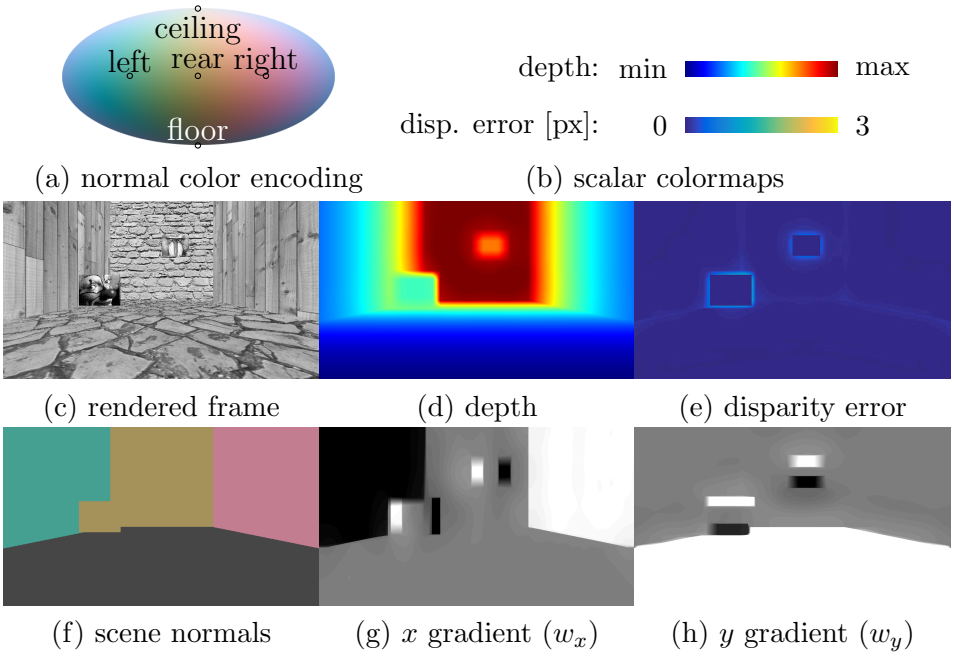
Input : $\tilde{z}$, parameters α_1, α_0, τ_1, τ_2, τ_3, τ_4
Output: (inverse) depth z
repeat
  $p^{k+1} = \Pi_{\alpha_1[-1\ 1]^{|p|}}\left(p^k + \tau_1 \begin{bmatrix} A & -I \end{bmatrix} \begin{pmatrix} \overline{z} \\ \overline{w} \end{pmatrix}\right)$;
  $q^{k+1} = \Pi_{\alpha_0[-1\ 1]^{|q|}}\left(q^k + \tau_2 C\overline{w}\right)$;
  $z^{k+1} = \text{Prox}_{\tau_3 E_{\text{data}}}(z^k - \tau_3 A^\top p^{k+1})$;
  $w^{k+1} = w^k - \tau_4\left(C^\top q^{k+1} - p^{k+1}\right)$;
  $\overline{z} = 2z^{k+1} - z^k$;
  $\overline{w} = 2w^{k+1} - w^k$;
until *convergence*;

Algorithm 12: Depth map smoothing with TGV regularization.

The projection onto the set $\alpha[-1\ 1]^m$ can be performed element wise by clipping to the interval $[-\alpha\ \alpha]$. The proximal operator of the quadratic data energy is a weighted average,

$$\text{Prox}_{\tau E_{\text{data}}}(y) = \frac{y + \tau\tilde{z}}{1 + \tau}.$$

Figure 6.2 shows an example output of the TGV regularized depth reconstruction with ground truth depth as reference. We can see that depth discontinuities are not preserved, but interpolated with planar surfaces, leading to blur in the depth map.



(a) normal color encoding (b) scalar colormaps

(c) rendered frame (d) depth (e) disparity error

(f) scene normals (g) x gradient (w_x) (h) y gradient (w_y)

FIGURE 6.2. TGV reconstruction with the corresponding color encoding. Hinges are reconstructed accurately, but depth discontinuities get interpolated in the reconstruction. The spherical normal color encoding is based on the LAB color space, floor and ceiling are shown in dark and light gray, on the equator we have strongest intensities of the colors. The gradients in x and y direction are defined on shifted domains. They would need to be mapped to the pixel grid, in order to generate a pixel wise planar representation.

In order to encourage sharp depth discontinuities, we introduce variables j_x and j_y, representing depth jumps along x and y components. We assume depth discontinuities to be sparse, thus we introduce an $L1$ penalty on $j = (j_x \; j_y)$. The total energy reads

$$E(z, w, j) = \frac{1}{2}\|z - \tilde{z}\|^2 + \alpha_1\|\nabla z - w - j\|_1 + \alpha_0\|\nabla w\|_1 + \beta\|j\|_1.$$

The algorithm is very similar to the original TGV formulation. We do not require additional dual variables, the proximal operator of the L1 norm is given by the shrinkage operator,

$$\begin{aligned}\text{Prox}_{\tau|\beta.|}(x) &= \arg\min_y \frac{1}{2\tau}(y - x)^2 + |\beta y| \\ &= \begin{cases} x - \tau\beta & \text{if } x > \tau\beta \\ x + \tau\beta & \text{if } x < -\tau\beta \\ 0 & \text{if } |x| \leq \tau\beta \end{cases}.\end{aligned}$$

For vector valued input, the shrinkage operation needs to be performed element wise, the full method is shown in Algorithm 13.

Input : $\tilde{z}$, parameters α_1, α_0, β, τ_1, τ_2, τ_3, τ_4, τ_5
Output: (inverse) depth z
repeat

$$p^{k+1} = \Pi_{\alpha_1[-1\ 1]^{|p|}}\left(p^k + \tau_1\begin{bmatrix}A & -I & -I\end{bmatrix}\begin{pmatrix}\overline{z}\\ \overline{w}\\ \overline{j}\end{pmatrix}\right);$$
$$q^{k+1} = \Pi_{\alpha_0[-1\ 1]^{|q|}}\left(q^k + \tau_2 C\overline{w}\right);$$
$$z^{k+1} = \mathrm{Prox}_{\tau_3 E_{\mathrm{data}}}(z^k - \tau_3 A^\top p^{k+1});$$
$$w^{k+1} = w^k - \tau_4\left(C^\top q^{k+1} - p^{k+1}\right);$$
$$j^{k+1} = \mathrm{Prox}_{\tau_5\|\beta.\|_1}\left(j^k + \tau_5 p\right);$$
$$\overline{z} = 2z^{k+1} - z^k;$$
$$\overline{w} = 2w^{k+1} - w^k;$$
$$\overline{j} = 2j^{k+1} - j^k;$$

until *convergence*;

Algorithm 13: Depth map smoothing with extended TGV regularization. The new variable j estimates depth discontinuities.

Figure 6.3 shows the output on the test frame. The depth is reconstructed with higher accuracy, we can observe sharp depth discontinuities. Also, artifacts in the gradients are not as strong as in the original TGV model. Depth discontinuities are represented well by the new variables j_x and j_y.

6.2.2. Scene Estimation from Dense Optical Flow. Given an optical flow field, we are interested in the corresponding camera motion and scene parameters.A similar method is presented in [**66**]. Synthesizing optical flow from egomotion and scene depth relies on a static scene, dynamic objects would need to be modeled explicitly. Translation vector t is restricted to the unit sphere, since scene scale cannot be estimated from a monocular video alone, hence normalization needs to be imposed.

The scene point X can be transformed into the coordinate system of the second camera,

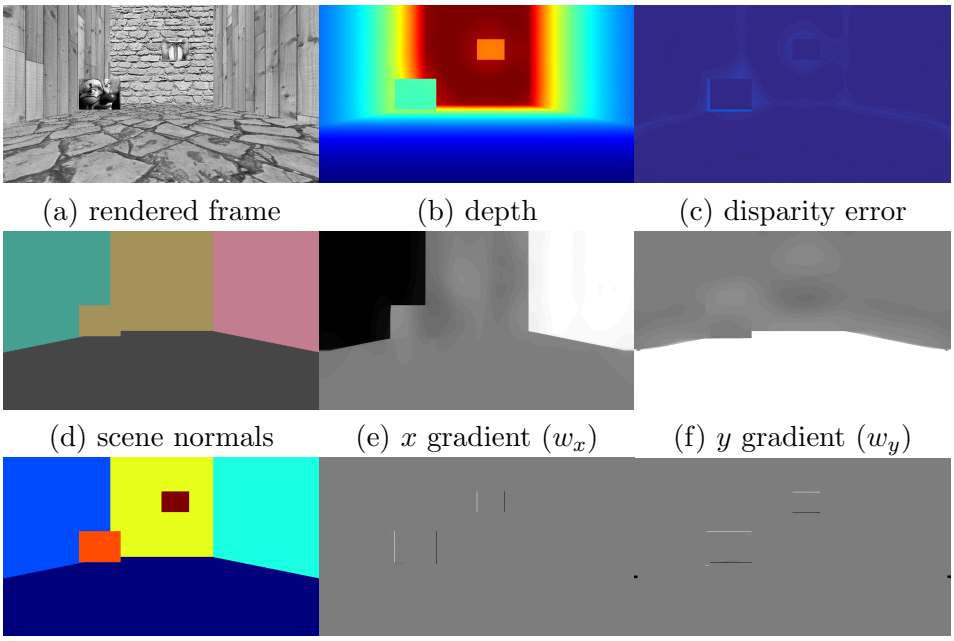
$$X' = R^\top(X - t).$$

An illustration of the camera setup is shown in Figure 6.4. We can determine the x' by projection onto the image plane,

$$x' = \pi(X'), \quad \pi(x_1, x_2, x_3) = \left(\frac{x_1}{x_3}, \frac{x_2}{x_3}, 1\right).$$

Since the projection π is invariant under scaling, we can express x' as a function of R, t and inverse depth z at x,

$$x' = \pi(R^\top(x - tz)).$$



(a) rendered frame (b) depth (c) disparity error

(d) scene normals (e) x gradient (w_x) (f) y gradient (w_y)

(g) scene objects (h) x jumps (j_x) (i) y jumps (j_y)

FIGURE 6.3. Extended TGV reconstruction. Depth discontinuities are reflected well by new variable j. The overall error in depth reconstruction and artifacts in w decreased as well.

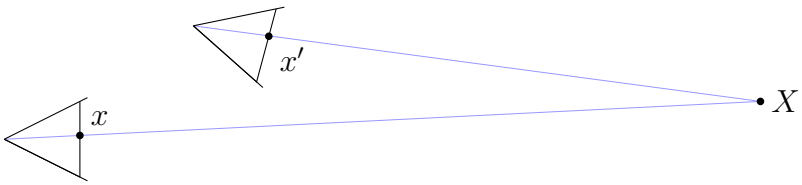


FIGURE 6.4. If relative camera poses and scene depth are known, the induced optical flow can be calculated. In three dimensions, viewing rays do not have to intersect, but can be skew.

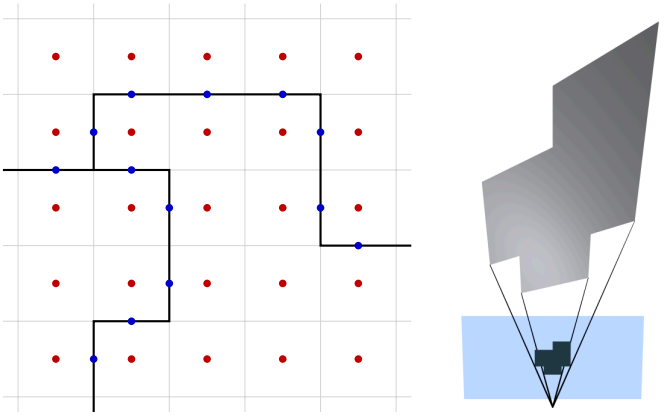
If we also have plane parameters v, z is given by $v^\top x$, and we can define a homography mapping $H(R,t,v)$,

$$x' = \pi(H(R,t,v)x), \quad H(R,t,v) = R^\top(I - tv^\top). \tag{6.2}$$

The induced optical flow is given by the difference between x' and x,

$$u(R,t,v) = \pi(H(R,t,v)x) - x. \tag{6.3}$$

6.2.3. Plane Estimation on Superpixels. We fit scene planes into superpixels rather than individual pixels, since a planar description in each pixel would yield an overparametrization. Merging pixels into



(a) superpixel discretization (b) plane segment

FIGURE 6.5. Superpixel discretization of our approach. Pixels are marked in red, and provide a flow vector, which is used by the data energy. Blue points on the superpixel boundary are used for regularization, we assume a small deviation in depth of both neighboring planes in these locations.

superpixels reduces the problem size.At the same time, superpixel segmentation provides scene regularization, since it defines the granularity of the planar reconstruction.

The superpixels are denoted with Ω_i, which form a partition of the image domain Ω,

$$\Omega_i \cap \Omega_j = \varnothing, \ \forall i, j \in \mathcal{I}_\Omega$$
$$\bigcup_i \Omega_i = \Omega.$$

$\overline{\Omega_i}$ denotes the boundary of superpixel Ω_i. The shared boundary of superpixels Ω_i and Ω_j is denoted ∂^{ij},

$$\partial^{ij} = \overline{\Omega_i} \cup \overline{\Omega_j}.$$

Figure 6.5 illustrates the locations of the boundary points on the pixel grid. The pixels are represented by red points, the blue boundary points are shifted by half a pixel.

We define and minimize an energy function on the scene and vehicle egomotion, which compares measured and parameterized optical flow and imposes scene regularity,

$$E(R, t, v) = E_u(R, t, v) + \lambda_z E_z(v) + \lambda_v E_v(v) + \lambda_p E_p(v). \tag{6.4}$$

The fidelity term $E_u(R, t, v)$ in our optimization problem is the deviation of an *observed* optical flow $\hat{u}(x)$ from our model (6.3) and is

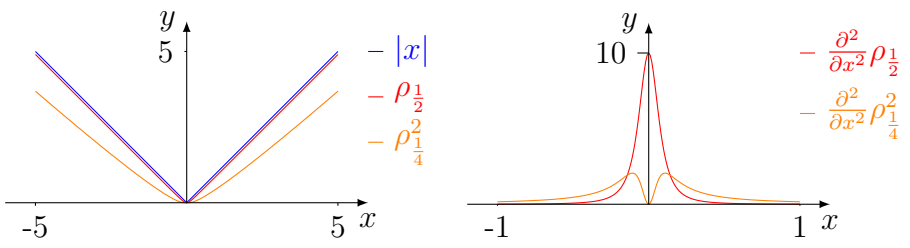


FIGURE 6.6. The L1 norm, Charbonnier norm with $\alpha = \frac{1}{2}$ and the squared Charbonnier with $\alpha = \frac{1}{4}$. In both cases, $\epsilon = 0.01$. Both versions of the Charbonnier function are smooth at $x = 0$, and approximate the L1 norm. The squared Charbonnier can be minimized with the Levenberg-Marquardt method, which requires a quadratic energy. The second derivative of the squared Charbonnier vanishes at zero, which improves numerical stability.

defined as

$$E_u(R,t,v) := \sum_{i=1}^{n} \sum_{x \in \Omega_i} w_{\hat{u}}(x) \|u(x;R,t,v_i) - \hat{u}(x)\|_2^2 \,.$$

Here, $w_{\hat{u}}(x) \geq 0$ denotes a spatially varying weighting of the data term which is provided by a confidence measure of the optical flow algorithm as detailed next.

The optical flow $\hat{u}$ between images I_1 and I_2 as required by the data term is computed in a pre-processing step using the algorithm *DeepFlow* [**93**], because the method is accurate and also very fast.

We complement the output obtained from DeepFlow with a confidence map $w_{\hat{u}}(x)$, which avoids the influence of flow vectors which are considered incorrect. To this end we also estimate the *backward* flow between I_2 and I_1, providing an estimate $\hat{u}^{-1}(x)$ of the inverse mapping of $\hat{u}(x)$. Only points that are consistently mapped forth and back are considered correct and we define the confidence map as

$$w_{\hat{u}}(x) := \exp\left(-\tfrac{1}{2}\|x - (\hat{u}^{-1} \circ \hat{u})(x)\|_2^2 / \sigma_{\hat{u}}^2\right)$$

with value $\sigma_{\hat{u}} > 0$. Experimentally, we found the value $\sigma_{\hat{u}} = \frac{1}{2\sqrt{2}}$ to be suitable.

In order to enforce that planes of neighboring superpixels form a seamlessly connected surface in most parts of the image, we introduce the prior $E_z(v)$ as follows. We consider points on the common boundary $x_\partial \in \partial^{ij}$ of superpixel i and j and penalize deviations of their inverse depth $z(x_\partial, v) = x_\partial^\top v$ according to the two plane models v_i and v_j, Figure 6.5 illustrates the boundary points on the pixel grid.

In order to encourage sharp depth edges we make use of the generalized Charbonnier functional

$$\rho_\alpha(x) := (x^2 + \epsilon)^\alpha - \epsilon^\alpha. \tag{6.5}$$

Figure 6.6 shows a comparison of the Charbonnier functional to the original $L1$ norm. We choose $\epsilon = 10^{-10}$ and $\alpha = 1/4$, so that $\rho_C^2(x)$ smoothly approximates the L1 norm. Then the energy function for one boundary ∂^{ij} reads

$$E_z^{ij}(v) := \sum_{x \in \partial^{ij}} \rho_\alpha^2(x^\top v_i - x^\top v_j).$$

The global smoothness term consists of a weighted sum of E_z^{ij} over all neighboring superpixels $(i, j) \in \mathcal{N}_\Omega$:

$$E_z(v) := \sum_{(i,j) \in \mathcal{N}_\Omega} E_z^{ij}(v)\,.$$

In addition to seamless surfaces on superpixel boundaries, we aim at plane parameters which up to a small set of discontinuities are constant over the image domain. This property encourages large connected planar structures.

For the plane smoothness prior we employ again the Charbonnier function ρ_α (see eq. (6.5), here applied component-wise),

$$E_v(v) = \sum_{(i,j) \in \mathcal{N}_\Omega} \|\rho_\alpha(v_i - v_j)\|_2^2$$

As a further constraint, we require all observed space points to be in front of the camera. Thus, we introduce an additional prior E_p, where we apply a soft hinge function

$$\rho_+(x) := \begin{cases} 1 - 2x & x \leq 0 \\ (1-x)^2 & 0 < x \leq 1 \\ 0 & 1 < x \end{cases},$$

to the inverse depth $z(x_c^i, v_i)$, evaluated at superpixel centers $x_c^i \in \Omega_i$.The center point is defined as mean value of all pixels in Ω_i,

$$x_c^i = \frac{1}{|\Omega_i|} \sum_{x \in \Omega_i} x$$

Summing over all superpixels, this leads to

$$E_p(v) = \sum_{i=1}^{n} \rho_+^2(z(x_c^i, v_i))\,.$$

6.2.4. Optimization Framework. The considered optimization task comprises a sum of a non convex data energy function and convex regularity terms (6.4), where manifold constraints $R \in \mathsf{SO}(3)$ and $t \in \mathsf{S}^2$ need to be respected. In order to find a local minimum of $E(R,t,v)$, we choose the Levenberg-Marquardt method [**63**], which has been adapted to Riemannian manifolds in [**1**].

The proposed energy function $E(R,t,v)$ can be decomposed into a sum of m squared functions $f_j(R,t,v)$,

$$E(R,t,v) = \sum_{j=1}^{m} (f_j(R,t,v))^2 = \|f(R,t,v)\|_2^2\,, \tag{6.6}$$

with $f(R,t,v) := (f_1(R,t,v), \ldots, f_m(R,t,v))^\top \in \mathbb{R}^m$ and $m = 2|\Omega| + \sum_{(i,j)\in\mathcal{N}_\Omega} |\partial_{ij}| + 3|\mathcal{N}_\Omega| + n$.

We combine the variables into a vector $Y := (R,t,v)$ and locally re-parametrize Y near (R^k, t^k, v^k) by parameters $\eta := (\omega, \delta t, \delta v)^\top \in \mathbb{R}^{3+3+3n}$ as

$$Y^k(\eta) := (R^k \operatorname{Exp}([\omega]_\times), \Pi_{\mathsf{S}^2}(t^k + \delta_t), v^k + \delta_v).$$

$\operatorname{Exp}(\cdot)$ denotes the matrix exponential function, which is applied to the skew-symmetric matrix

$$[\omega]_\times = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}.$$

A closed form expression for the matrix exponential is given by Rodrigues' rotation formula [**36**]. Furthermore,

$$\Pi_{\mathsf{S}^2}(t) := t/\|t\|_2$$

denotes the orthogonal projection of t to S^2.

Using first order Taylor expansion we obtain an approximation of $f(Y(\eta))$ in $Y = (R^k, t^k, v^k)$,

$$\tilde{f}^k(\eta) := f^k(0) + \mathrm{J}(f^k)\eta,$$

with Jacobian $\mathrm{J}(f^k)$ of f^k. The Jacobian is obtained for the rotation and translation by differentiating the function compositions $\frac{\partial}{\partial\omega}(f \circ \operatorname{Exp})(\omega)$ and $\frac{\partial}{\partial t}(f \circ \Pi_{\mathsf{S}^2})(t)$, respectively. Substituting this approximation in (6.6) yields a model of the original energy function $E_m^k(\eta)$,

$$E_m^k(\eta) = \min_\eta \|\tilde{f}^k(\eta)\|_2^2.$$

We add a soft trust region term $\frac{\mu^k}{2}\|\eta\|^2$, in order to control the step size of the update, the resulting objective is quadratic in η and thus can be solved efficiently. The update of state variables Y^{k+1} and trust region parameter μ^{k+1} is presented in algorithm 14. The update is based on the quotient ρ of the change in the original and model energies. If ρ is close to one, the model is accurate, and μ can be decreased. If ρ

is close to zero, the model is inaccurate, and the trust region needs to be reduced. For negative ρ the energy has decreased, and we omit the update.

Input : Current state Y^k, update vector η^k, parameter μ^k
Output: Y^{k+1}, μ^{k+1}
$\rho \leftarrow \frac{E(Y^k)-E(Y^k(\eta))}{E_m(0)-E_m(\eta)}$;
if $\rho < \frac{1}{4}$ **then**
  $\mu^{k+1} = 4\mu^k$;
else if $\rho > \frac{3}{4}$ **then**
  $\mu^{k+1} = \frac{1}{2}\mu^k$;
end
if $\rho > 0$ **then**
  $Y^{k+1} = Y^k(\eta)$;
else
  $Y^{k+1} = Y^k$;
end

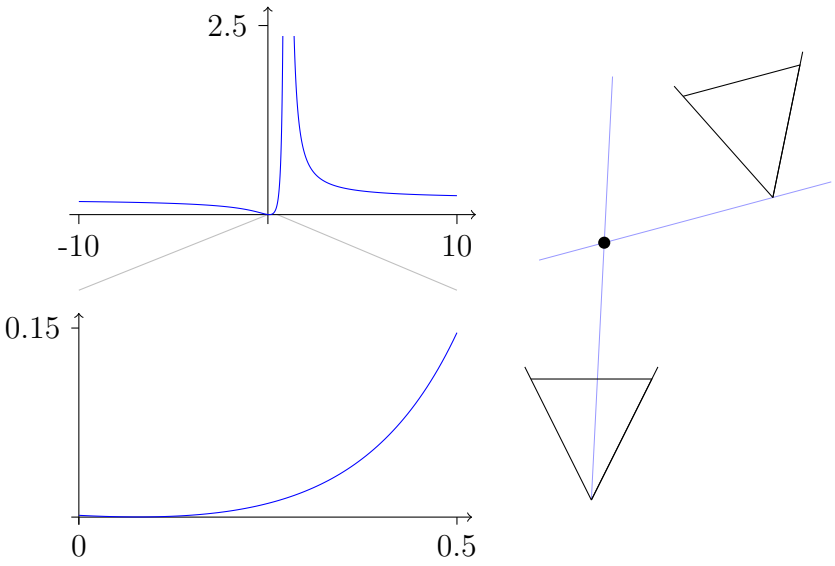
Algorithm 14: Update rule for variables Y^{k+1} and trust region μ^{k+1}.

A limit of 80 iterations was used as stopping criterion which was sufficient for most of the considered data. The complete method is summarized in algorithm 15. We describe the construction of the Jacobian in detail in the next section.

Input : Reference frame I_1, optical flow $\hat{u}$, reverse flow $\hat{u}^{-1}$
Output: Scene planes $\{v_i\}$, normalized egomotion
  $R \in \mathrm{SO}(3), t \in \mathsf{S}^2$
Compute superpixels $\{\Omega_i\}$ using SLIC and weights $w_{\hat{u}}$;
Initialize $R^0 \leftarrow I_3$, $t^0 \leftarrow (0\ 0\ 1)^\top$, $v_i^0 \leftarrow (0\ 0\ 0.001)^\top\ \forall i \in \mathcal{I}_\Omega$;
$\mu^0 \leftarrow 40$;
while $k \leq 80$ **do**
  Determine $F = f^k$ and $J = \mathrm{J}(f^k)$;
  $\eta \leftarrow (J^\top J + \mu^k I_m)^{-1}(-J^\top F)$;
  Determine Y^{k+1} and μ^{k+1} using algorithm 14;
  $k \leftarrow k + 1$;
end

Algorithm 15: Our method for monocular two frame piecewise planar scene reconstruction.

Calculating the derivative of the pixel correspondence in eq. (6.2) is challenging in two ways. Firstly, the projection π is not convex, not differentiable everywhere and not bounded. And secondly, the variables $R \in \mathrm{SO}(3)$ and $t \in \mathsf{S}^2$ are subject to manifold constraints, so that the respective differential geometry needs to be considered.



(a) optical flow energy (b) pole location

FIGURE 6.7. Optical flow energy for fixed R and t. While the function is not well behaved on large scale, it is convex in the region of interest. The pole occurs at infinite optical flow, when the viewing ray of the second camera is parallel to its image plane.

Figure 6.7 shows the flow energy $\frac{1}{2}\|u(R,t,z)-\hat{u}\|^2$ for a single pixel as a function of z. There is a unique minimum at the true inverse depth, and a pole at which $u(R,t,z)=\infty$. Since we look at the inverse depth here, at zero the actual depth changes continuously from $+\infty$ to $-\infty$, and at $z=\pm\infty$, we approach the camera center.

Inserting the definition of $u(R,t,v)$ into the data energy, we get

$$E_{\text{flow}} = \sum_{i\in\mathcal{I}_\Omega}\sum_{x\in\Omega_i} \frac{w_x}{2}\left\|\pi(H(R,t,v_i)x)-\hat{x}'\right\|^2, \tag{6.7}$$

where $\hat{x}' = x+\hat{u}(x)$ denotes the observed point correspondence. By applying the rule $\text{vec}(ABC) = (C^\top\otimes A)\vec{B}$, we can vectorize the homography matrix,

$$E_{\text{flow}} = \sum_{i\in\mathcal{I}_\Omega}\sum_{x\in\Omega_i} \frac{w_x}{2}\left\|\pi\left((x^\top\otimes I_3)\vec{H}_i\right)-\hat{x}'\right\|^2. \tag{6.8}$$

We can construct the derivative in terms of the vectorized homography,

$$\frac{\partial E_{\text{flow}}}{\partial \vec{H}_i} = \sum_{i\in\mathcal{I}_\Omega}\sum_{x\in\Omega_i} w_x(\pi(H_ix)-\hat{x}')^\top\frac{\partial\pi}{\partial p}(x^\top\otimes I_3) \tag{6.9}$$

The derivative of the projection π at $p = (p_x\ p_y\ p_z)^\top = H_i x$ is given by

$$\frac{\partial \pi}{\partial p} = \begin{bmatrix} \frac{1}{p_z} & 0 & -\frac{p_x}{p_z^2} \\ 0 & \frac{1}{p_z} & -\frac{p_y}{p_z^2} \\ 0 & 0 & 0 \end{bmatrix} \tag{6.10}$$

For the inner derivative of the homography matrix, we can vectorize the homography matrix in different ways, writing R' for $R^\top$,

$$\begin{aligned} H_i = R'(I_3 - t v_i^\top) = R' - R' t v_i^\top \\ H_i = I_3 R'(I_3 - t v_i^\top) \ \Rightarrow\ \frac{\partial \vec{H}_i}{\partial R'} = (I_3 - v_i t^\top \otimes I_3) \\ H_i = R' - R' t v_i^\top \ \Rightarrow\ \frac{\partial \vec{H}_i}{\partial t} = -(v_i \otimes R') \\ H_i = R' - (R't) v_i^\top I_3 \ \Rightarrow\ \frac{\partial \vec{H}_i}{\partial v_i} = -(I_3 \otimes R't) \end{aligned}$$

We need to extend the derivative further, in order to respect the manifold constraints on R' and t. The tangent space of R' is given by $R'[\omega]_\times$, thus we get the derivative

$$\frac{\partial \vec{R}}{\partial \omega} = \mathrm{vec}(R'[\omega]_\times).$$

We project the gradient into the sphere's tangent space,

$$\frac{\partial t}{\partial t^0} = I_3 - t t^\top.$$

The total derivative is given by applying the respective inner derivative to eq. (6.9).

The regularity terms do not depend on the egomotion parameters, but only on plane parameters v_i. Therefore, we do not need to consider manifold constraints, and can compute the Jacobian using the Euclidean derivatives. The derivative of the depth smoothness term is given by

$$\begin{aligned} \frac{\partial}{\partial v_i} E_z^{ij} &= \sum_{x \in \partial^{ij}} \rho_{0.25}^2(x^\top (v_i - v_j)) \\ &= \left((x^2 + \epsilon)^{0.25} - \epsilon^{0.25} \right) (x^2 + \epsilon)^{0.75} x \\ \frac{\partial}{\partial v_j} E_z^{ij} &= -\frac{\partial}{\partial v_i} E_z^{ij}. \end{aligned}$$

The derivative of the plane smoothness term is very similar, since $v_i - v_j = I(v_i - v_j)$, which corresponds to the depth smoothness case with $x \in \{e^1, e^2, e^3\}$. The positive depth energy consists of linear and quadratic functions, which can be differentiated directly.

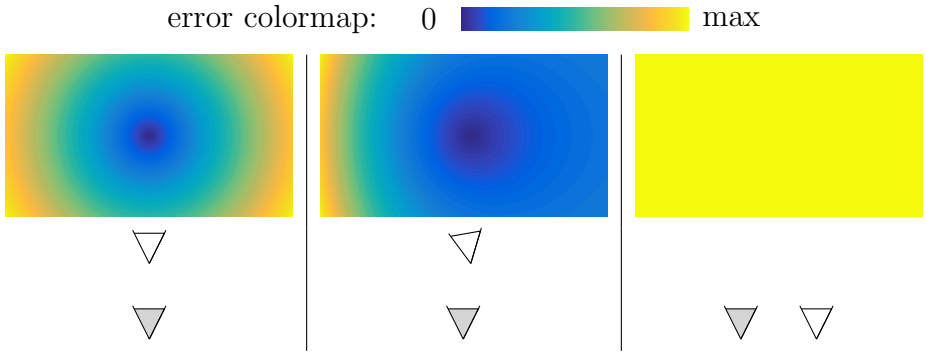


FIGURE 6.8. Reprojective error between $z \equiv 0.01$ (depth 100) and $z_{gt} \equiv 0$ (infinite depth), the reference camera is shaded in gray. In case of stereo, the optical flow (or disparity) difference is constant, but in the monocular case, we get a non linear optical flow deviation, with zero error at the epipole. The epipole has zero error independent of the estimated depth. In case of stereo estimation, the epipole lies at infinity, and does not influence the error measure.

6.2.5. Evaluation on the KITTI Dataset. Before we evaluate our approach, we discuss our evaluation strategy. In case of stereo evaluation, the error is defined as the difference between ground truth and estimated disparities. This error measure respects the stereo camera setup well, since the disparity is directly observed by the camera system.

Similar to the disparity comparison in stereo evaluation, we compare the difference in optical flow as given by ground truth and estimated inverse depth. In contrast to stereo, the camera setup is unknown, and hence the estimated camera displacement needs to be taken for evaluation. Also, scene scale is unknown, and needs to be estimated, which we implement by computing the median scale between estimated and ground truth inverse depth maps.

Figure 6.8 shows the reprojection error between two constant depth maps for different egomotion parameters. While the error is constant for the stereo case, the epipole lies within the image for a forward camera motion, and is clearly visible. The induced flow at the epipole is independent of the depth value, hence the error is zero between any depth maps. The authors of [**9**] propose a similar depth evaluation scheme, by defining an uncertainty measure in the estimated depth which relates to the induced optical flow field.

The KITTI stereo/flow dataset [**32**] contains traffic scenes with ground truth depth.The ground truth depth is sparse and has been acquired with a Velodyne laser scanner. Due to the rotation frequency of the laser scanner, the camera frame rate is rather low at 10 frames per second, which implies large displacements between successive frames.

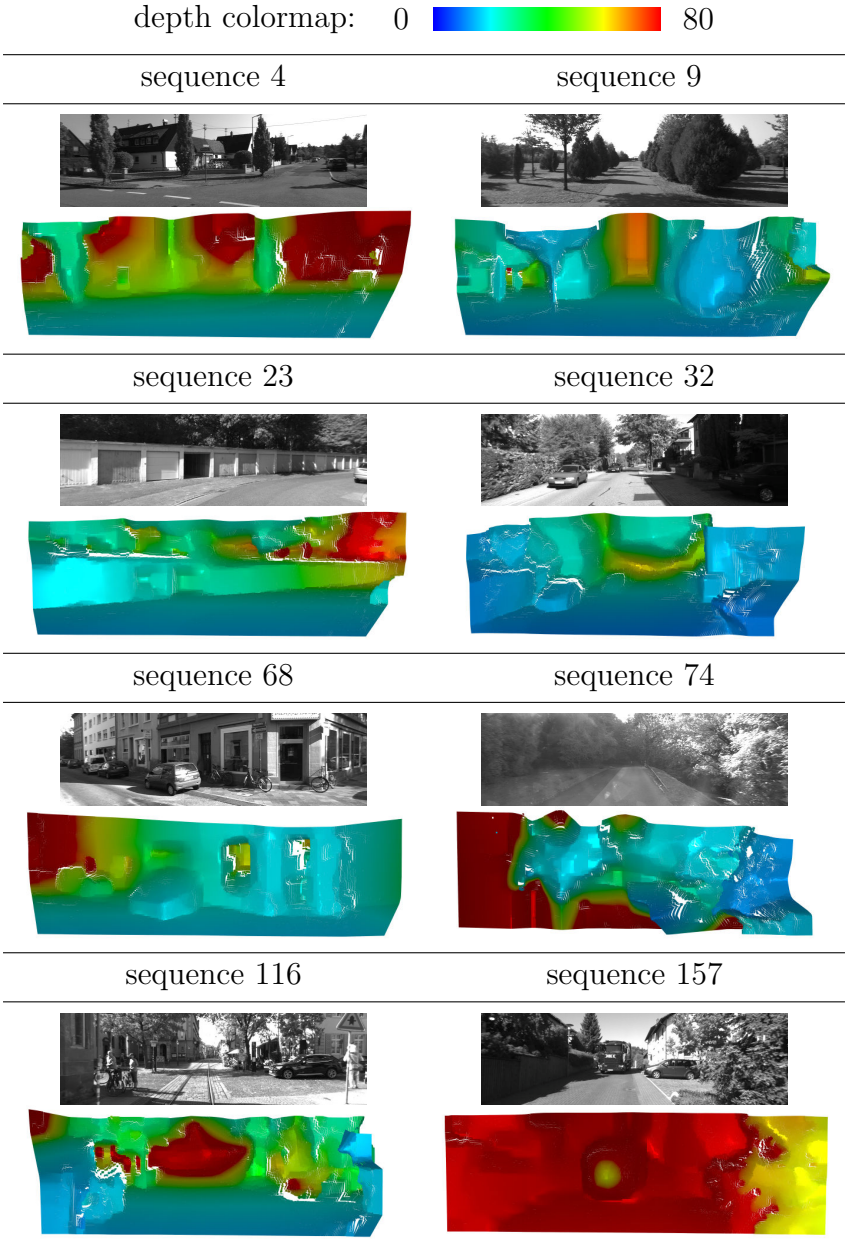


FIGURE 6.9. Rendered KITTI reconstructions found by our monocular approach. The scene is not normalized to metric units, but scaled according to $\|t\| = 1$. The reconstructed depth depends on the vehicle speed, the scene will appear closer with higher speed, the color encoding is shown on top. In the last two cases (116 and 157), the static scene assumption is violated. Therefore, the car coming from the right in seq. 116, and the truck approaching the camera in seq. 157 are not reconstructed correctly.

As we can see if figure 6.9, our reconstruction algorithm captures the main layout of the respective scenes. Also, the ground surface is reconstructed well in most cases, and appears as a closed surface. However, our regularity term also introduces artifacts, objects tend to be merged with the background at the edge, while sharp depth discontinuities would be preferable.

In sequence 74, the estimated optical flow field is incorrect in large regions due to illumination changes and reflections. Our method can compensate small errors in the optical flow field, but it would require separate input data in order to compensate large errors.

Sequences 116 and 157 show dynamic scenes, the car to the right in sequence 116, and the truck in sequence 157 are moving forward. In case of the truck in sequence 157, its motion happens to align with the camera egomotion, thus our method is tricked by an optical illusion. The truck appears to the observer as a very close and very small object. The large scene scale in sequence 157 is due to the low vehicle speed in this example.

Figures 6.10, 6.11, 6.12 and 6.13 show our test frames in more detail with a direct comparison to the sps-stereo method. Both methods reconstruct most test images well, especially the ground surface is oriented correctly in most locations.

We can observe that our approach lacks reconstruction accuracy in the image center, which is approximately the location of the epipole. Especially in sequences 4 and 32, we see that the trees are blurred into the background, while other trees at the sides are reconstructed with better detail. The stereo method does not have this disadvantage in camera setup, and provides constant reconstruction accuracy. In addition to the better suited camera setup, the stereo method has calibrated cameras, and does not need to estimate camera poses.

Our approach favors smooth transitions between different regions, while the stereo approach has more sharp discontinuities. In some cases, we oversmooth the scene and merge foreground and background, but sps-stereo detects too many discontinuities is some cases, and produces noisy estimates of homogeneous regions.

The flow weights mostly vanish in occluded regions, as we would expect, since these cannot me estimated accurately from the images alone. The weights of sequence 74 show that the flow field is very inaccurate, which implies an inaccurate scene reconstruction as well. Homogeneous regions, as for example the sky, usually have a small weight as well, since these cannot be matched reliably.

Table 6.1 shows depth reconstruction errors on the KITTI dataset, in comparison with sps-stereo. Our approach copes well with occluded parts of the image, but sps-stereo shows higher accuracy in depth reconstruction. The calibrated side by side camera setup is better suited for depth estimation.

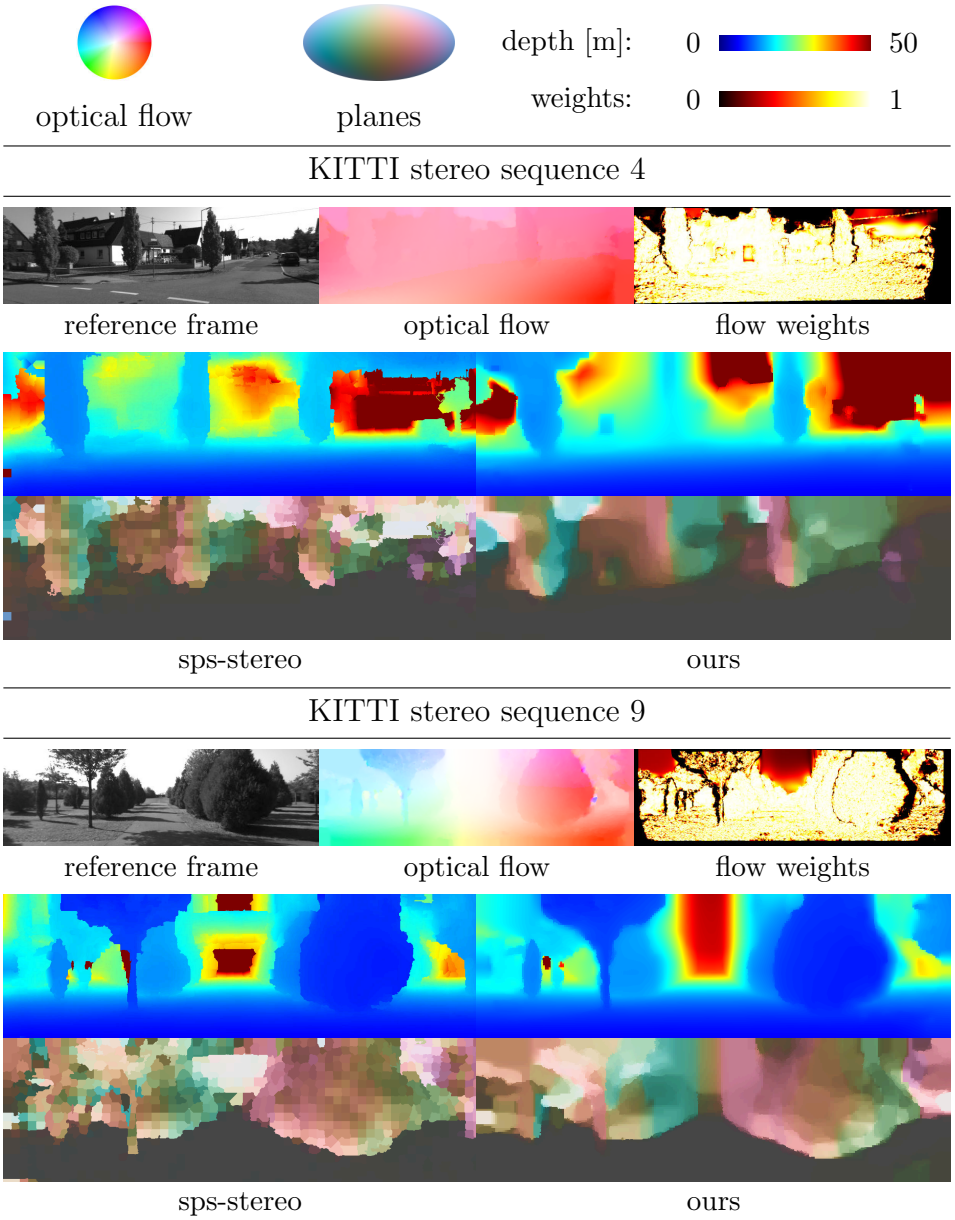


FIGURE 6.10. Results on example frames from KITTI sequences 4 and 9. In case of sequence 4, the tree in the image center is almost completely blurred, since it lies in the region of the epipole. In the other example, the region around the epipole does not contain any particular object, but shows the horizon, and can be interpolated from the neighborhood without much loss in accuracy. The flow weights represent the occlusion pattern well in both cases. Our approach favours objects, especially trees, to stand out of the scene as sharp objects, while sps-stereo favors noisier estimates in the plane reconstructions with sharper discontinuities in depth.

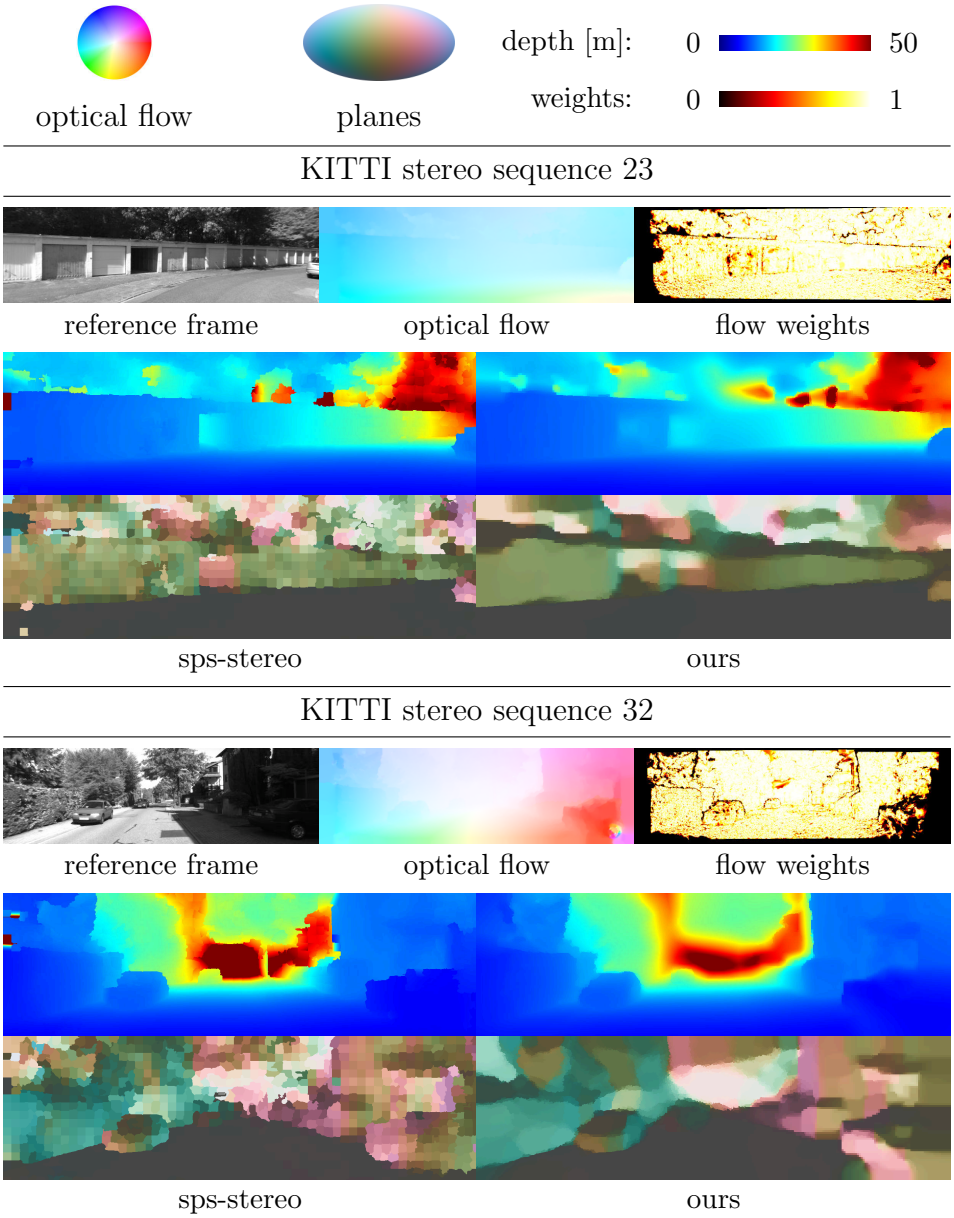


FIGURE 6.11. Results on example frames from KITTI sequences 23 and 32. Sequence 23 shows that our method merges the garages into a uniform planar surface, while the sps-stereo reconstruction is more noisy. In sequence 32, we can again observe blurriness at the epipole. The tree in front of the car is unclear, whereas the parked car on the left is reconstructed in detail. Especially the normals provide a good representation of the car, with sharp edges inside the object. The opitical flow field is not accurate in the underexposed right part of the image, therefore our reconstruction is lacking details in the bottom right corner. The ground surface is reconstructed well by both methods in both examples.

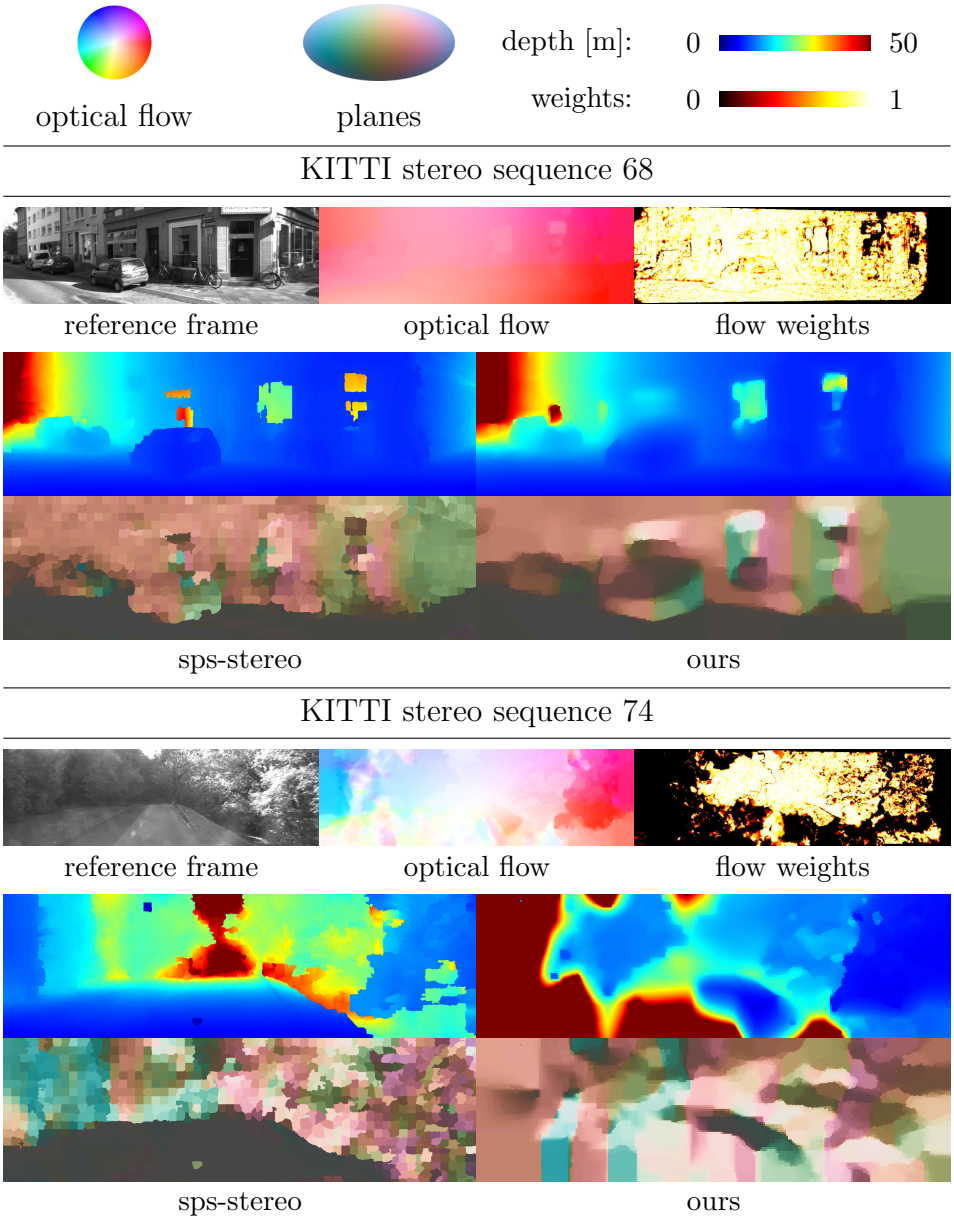


FIGURE 6.12. Results on example frames from KITTI sequences 68 and 74. In the first example, the large building is reconstructed well, but we have errors in depth reconstruction due to reflections in the window in both methods. The example frame from sequence 74 has strong changes in lighting conditions and optical lens effects in the monocular, but not in the stereo case. The computed optical flow field has large errors in large parts of the image, as shown by the weight mask. Our method detects a few parts of the scene in the, but does not reflect the surrounding environment well. In contrast to this, the stereo approach shows similar accuracy as on the previous examples.

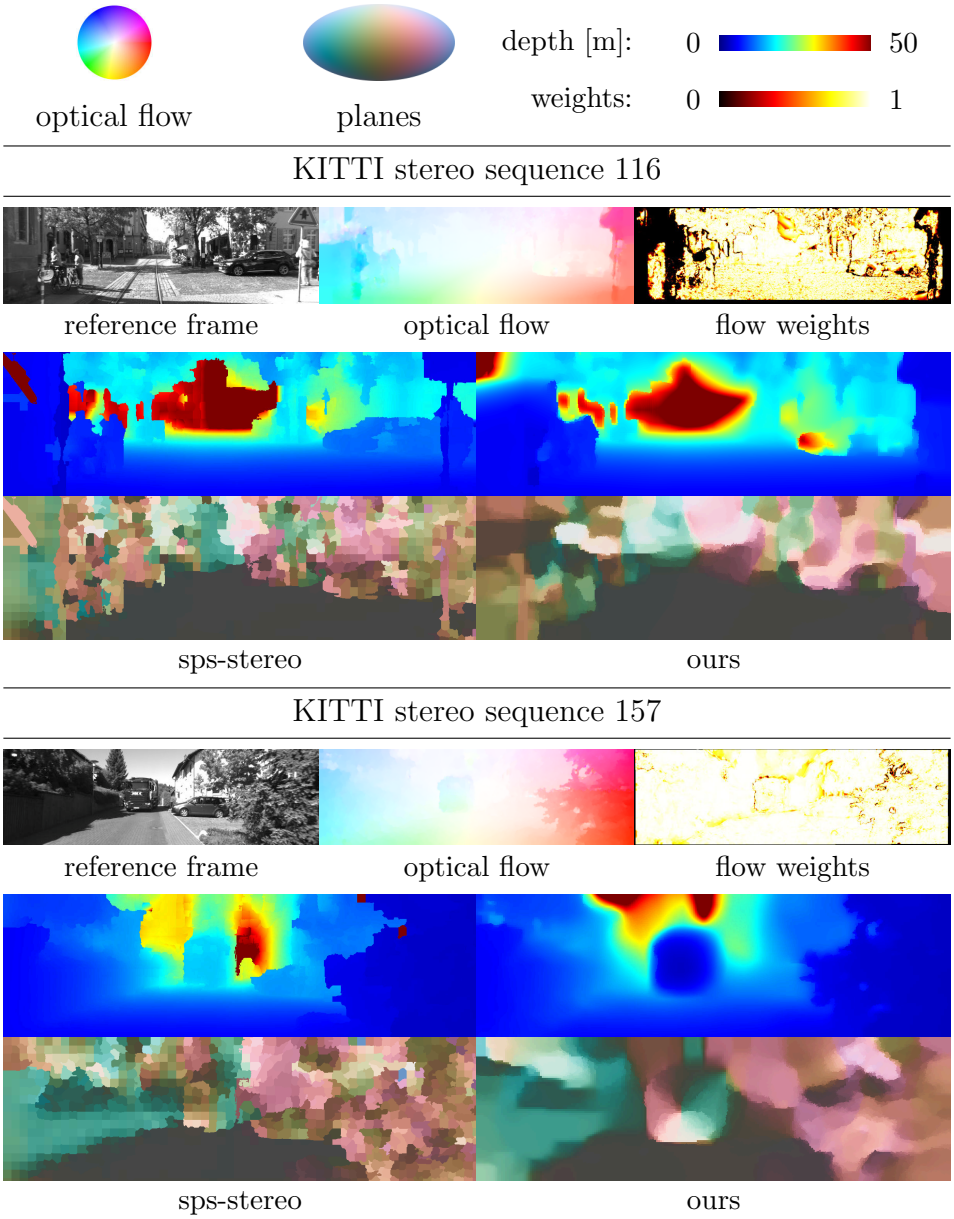


FIGURE 6.13. Results on example frames from KITTI sequences 116 and 157. In case of sequence 116, the motion of the dynamic car does not align with the epipolar lines, and we get a wrong reconstruction. However, sequence 157 shows an inherent limitation of the monocular camera setup. The movement with respect to the truck and the environment only differ in scale, hence the motion aligns with the epipolar lines given by the egomotion. The truck appears as a much smaller, and much closer object, than it actually is. The front of the truck is reconstructed to a similar distance as the tree to the right of the camera, and as a sharp pointed object.

		occ			noc	
	seq.	sps-st.	flow	ours	flow	ours
end point error [px]	4	0.22	1.06	0.63	0.36	0.62
	9	0.40	1.51	0.59	0.92	0.56
	19	0.80	2.40	1.41	1.63	1.05
	23	0.26	1.00	0.35	0.39	0.33
	32	0.88	3.62	1.38	1.08	1.34
	68	0.17	1.24	0.73	0.67	0.72
	74	0.78	21.21	13.96	15.14	13.64
	116	0.67	1.77	1.34	1.08	1.08
	157	0.07	0.19	0.16	0.19	0.16
	all	**0.97**	**6.98**	**3.93**	**1.97**	**3.50**
2px [%]	4	0.93	6.05	2.73	1.31	2.68
	9	2.68	14.05	4.90	5.34	4.35
	19	4.80	23.05	16.00	15.33	13.62
	23	0.95	8.55	0.28	1.38	0.28
	32	8.65	21.87	14.29	9.30	13.15
	68	0.31	7.21	4.27	2.51	4.17
	74	5.83	62.10	84.01	56.36	83.60
	116	4.89	16.33	11.97	8.75	8.19
	157	0.36	0.14	0.09	0.14	0.09
	all	**6.47**	**21.11**	**17.59**	**10.67**	**16.41**

TABLE 6.1. KITTI monocular reprojective error and input optical flow error for all pixels (occ) and non occluded ones (noc). We do not include the sps-stereo method in the non occluded case, since the occlusion pattern is different in the stereo camera setup. Our approach shows similar accuracy on the visible and on all locations, which suggests that occluded regions are detected correctly, and filled with accurate data. However, the parameterized flow has larger error than the input flow, we lose some information by imposing regularity terms. The stereo approach performs better, as we would expect from a stereo system.

6.2.6. Evaluation on Rendered Data. Plane normals are not trivial to measure directly, and most structure estimation methods do not evaluate or even estimate plane normals. We did not find any dataset providing the data we need, and we decided to generate our own dataset with ground truth egomotion, optical flow, depth and normals. We decided to render artificial scenes, where ground truth normals are known from the scene setup.

In addition to estimated optical flow, we evaluate our method on ground truth input, in order to remove influence of the optical flow model and estimation errors from our reconstruction. We do not need to estimate scene scale with the median, since the exact scale is known.

		stereo	DeepFlow		GT
	seq.	sps-st.	flow	ours	ours
end point error [px]	1	0.12	0.54	0.15	0.02
	2	0.17	1.75	0.32	0.01
	3	0.11	2.14	0.43	0.02
	4	0.09	1.34	0.28	0.01
	5	0.57	5.34	5.25	0.04
	6	0.16	0.38	0.18	0.02
	7	0.59	6.67	27.69	0.01
	8	0.32	27.13	42.29	0.01
	9	0.10	0.35	0.25	0.04
	10	0.11	4.48	6.65	2.01
2px [%]	1	0.43	6.44	1.11	0.15
	2	1.86	10.08	4.45	0.09
	3	0.20	11.14	5.87	0.08
	4	0.14	9.51	3.36	0.08
	5	1.32	12.78	24.96	0.35
	6	0.40	3.86	1.47	0.15
	7	3.30	12.12	13.26	0.08
	8	0.74	11.79	31.30	0.01
	9	0.28	2.87	1.37	0.23
	10	0.23	12.92	41.24	14.18

TABLE 6.2. Rendered scenes depth evaluation. We observe that our method improves the input optical flow in all sequences except 7, 8 and 10, which are sequences with large errors in estimated optical flow. The rendered scenes consist of only very few planar objects, which fits our piecewise planar model well. The stereo approach is outperforming our monocular method, as expected. On ground truth input flow, we have very low error except for sequence 10, which has large gaps in scene depth.

The dataset consists of four scene environments, using multiple camera tracks for each environment, each containing 25 frames, which yields 24 optical flow fields.

Figures 6.14 to 6.18 show the results of our reconstruction for frame 10 of each sequence in comparison to sps-stereo and ground truth. We note a very insignificant fuzziness at the epipole, which is again approximately in the image center. The effect is weaker than on the KITTI benchmark, since the rendered scenes consist of less objects, and piecewise planar interpolation describes the scene better.

Sequences 7 and 8, depicted Figure 6.17, contain very fine texture, which is difficult for optical flow estimation. We get strong artifacts in the reconstruction from estimated flow, while the reconstruction from

	estimated flow			ground truth		
seq.	e_R [deg.]	e_t [met.]	e_α [deg.]	e_R [deg.]	e_t [met.]	e_α [deg.]
1	0.00	0.00	0.03	0.00	0.00	0.01
2	0.00	0.00	0.02	0.00	0.00	0.00
3	0.00	0.00	0.00	0.00	0.00	0.01
4	0.00	0.00	0.03	0.00	0.00	0.01
5	0.24	0.20	5.38	0.00	0.00	0.02
6	0.00	0.00	0.08	0.00	0.00	0.06
7	1.70	0.52	8.40	0.00	0.00	0.01
8	2.60	0.93	29.04	0.00	0.00	0.03
9	0.00	0.00	0.14	0.00	0.00	0.16
10	0.82	0.61	25.52	0.25	0.17	7.05

TABLE 6.3. Rendered scenes camera motion evaluation. The table show rotation error in degrees, and translation error both in metric units and as angle in degrees. We evaluate translation error as angle, because translation norm cannot be estimated from a single optical flow field without further assumptions. Most sequences are reconstructed accurately from ground truth input. However, the depth discontinuities of sequence 10 are not represented well by our regularity term, which results in errors of egomotion estimation.

ground truth optical flow is accurate. The gap in depth reconstruction accuracy between estimated and ground truth flow is shown in Table 6.2. In most cases, the estimated depth has lower error than the input optical flow, our piecewise planar scene assumption represents the rendered sequences well. Similarly, we can observe a discrepancy between estimated and ground truth optical flow for the estimated camera motion in Table 6.3.

As we have seen on the KITTI dataset, foreground objects are merged with the background. This results in reconstruction errors, especially in sequence 10, where depth discontinuities are particularly large, as we can see in Figure 6.18. Tables 6.2 and 6.3 show large reconstruction errors for sequence 10 from ground truth input.

Table 6.4 show normal reconstruction errors of our approach and sps-stereo. The average error is similar to the stereo reconstruction, but the distributions vary for both cases.

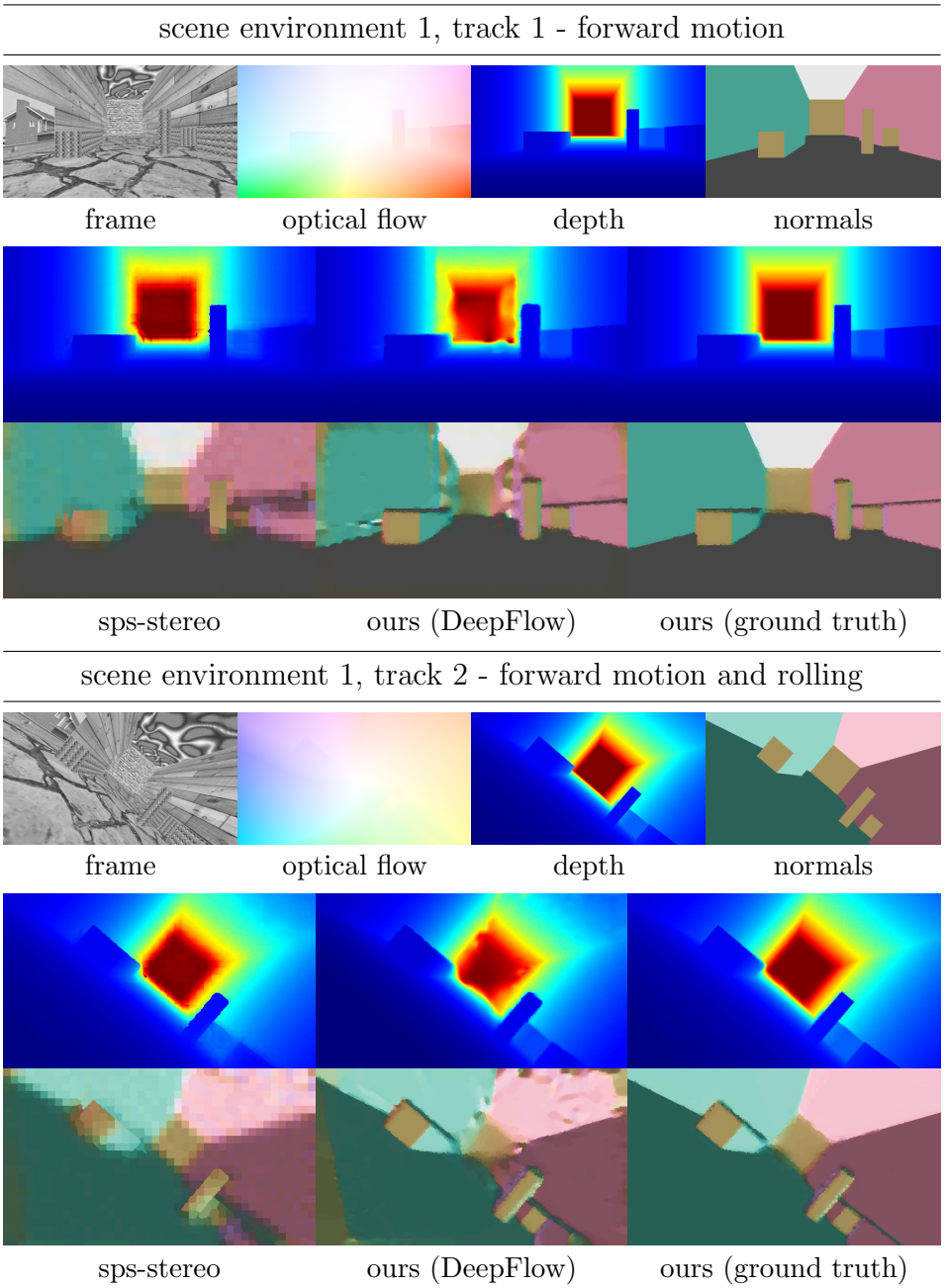


FIGURE 6.14. Rendered sequences 1 and 2, showing environment 1 with straight forward and forward rolling egomotion. The sps-stereo method runs on stereo images independently of the egomotion. Our method reconstructs both scenes accurately, with small artifacts on the estimated flow. Also, the optical flow estimation does not work reliably on occluded areas. As in the KITTI benchmark, we can observe that our method merges objects with the background.

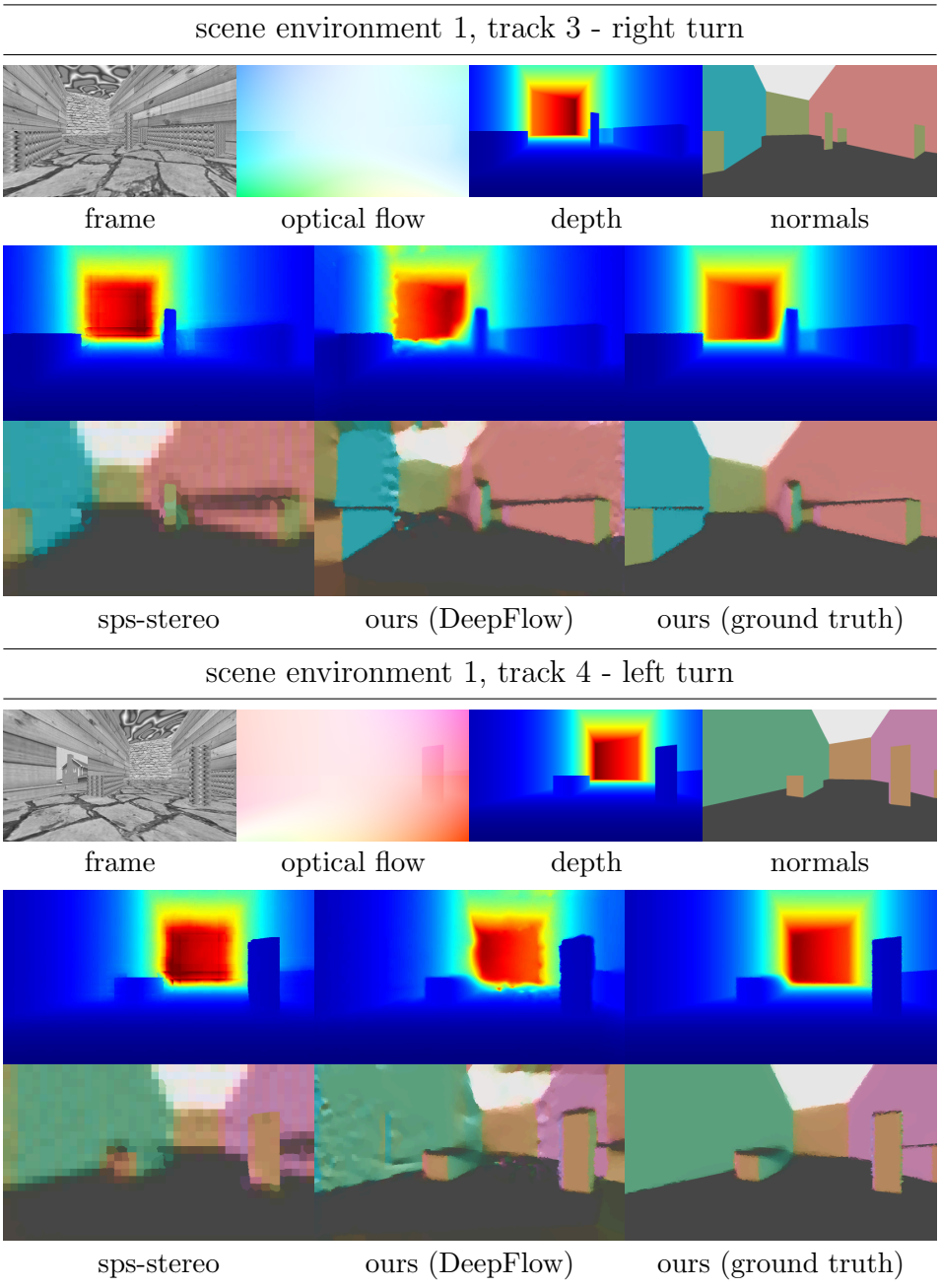


FIGURE 6.15. Rendered sequences 3 and 4, showing as same environment as in Fig. 6.14 with left and right turn motion. We see a similar behavior as in the previous case, the occlusion pattern differs due to the new camera motion, which explains reconstruction errors on the left edge in the right motion case, and vice versa.

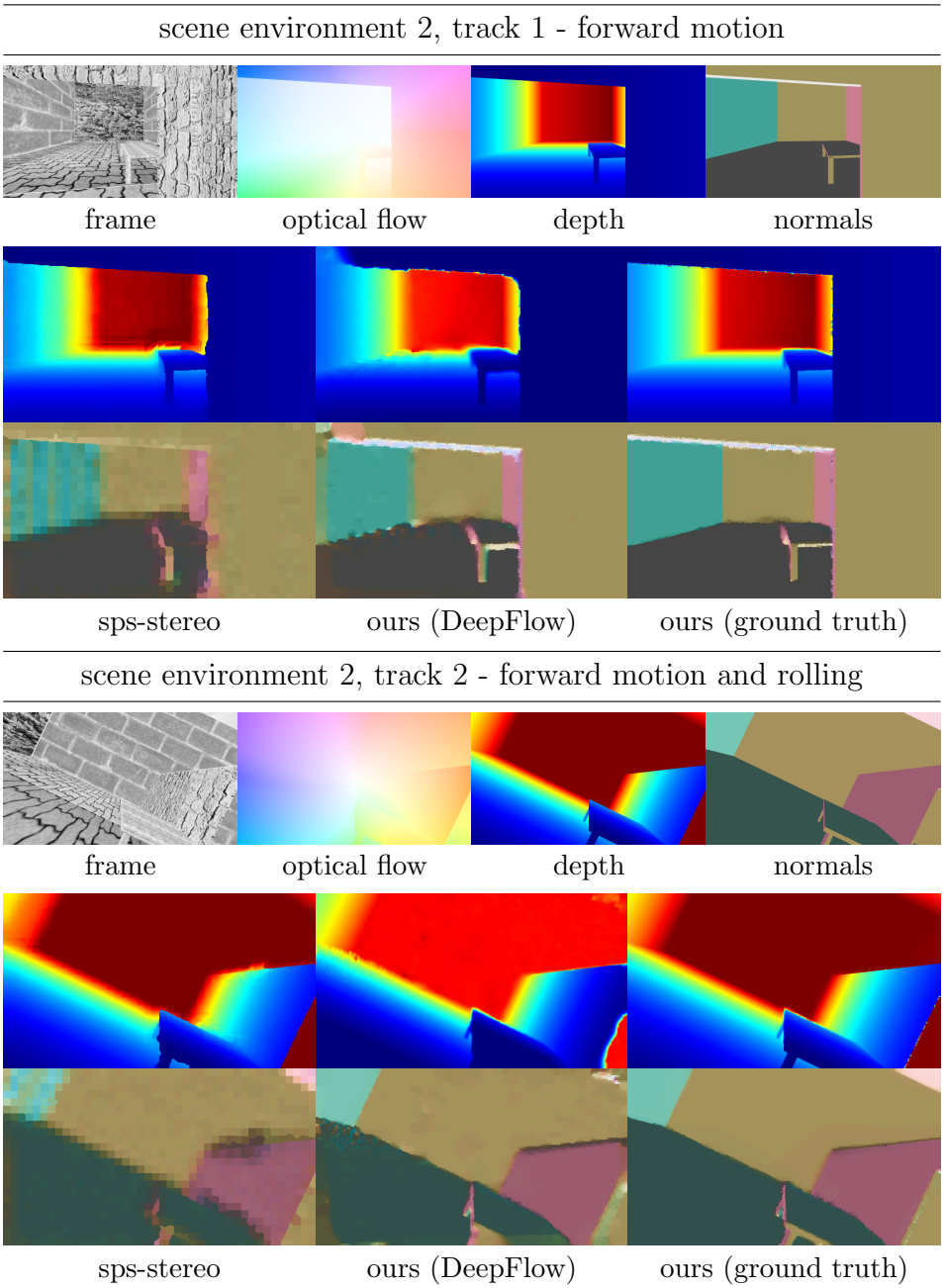


FIGURE 6.16. Rendered sequences 5 and 6. The environment is more complex than the previous one, it has more and smaller objects. Our method reconstructs details such as the table and and top facing wall and the entrance, while the large superpixels of sps-stereo do not support reconstruction on the same level of detail. In the second case using DeepFlow, the maximum depth is estimated too small, which suggest an error in the estimated camera translation.

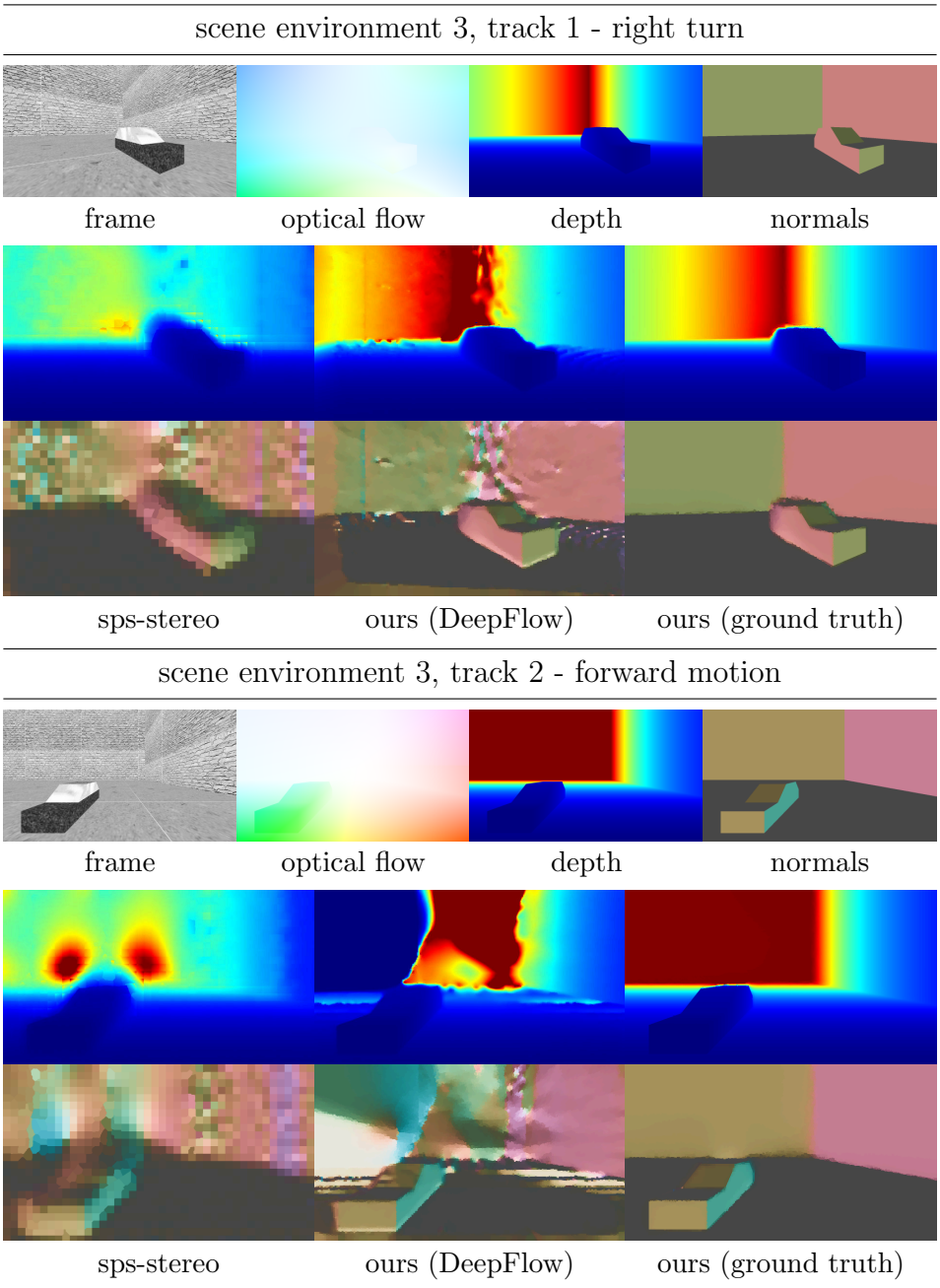


FIGURE 6.17. Rendered sequences 7 and 8. The environment is very large, compared to the camera motion, and the wall has very small texture. These are difficult conditions for optical flow estimation, since the wall texture is lost on a small picture scale. As consequence, our reconstruction is less accurate on estimated flow. On ground truth flow we get high quality depth and normal reconstruction, but objects tend to be blended with the background.

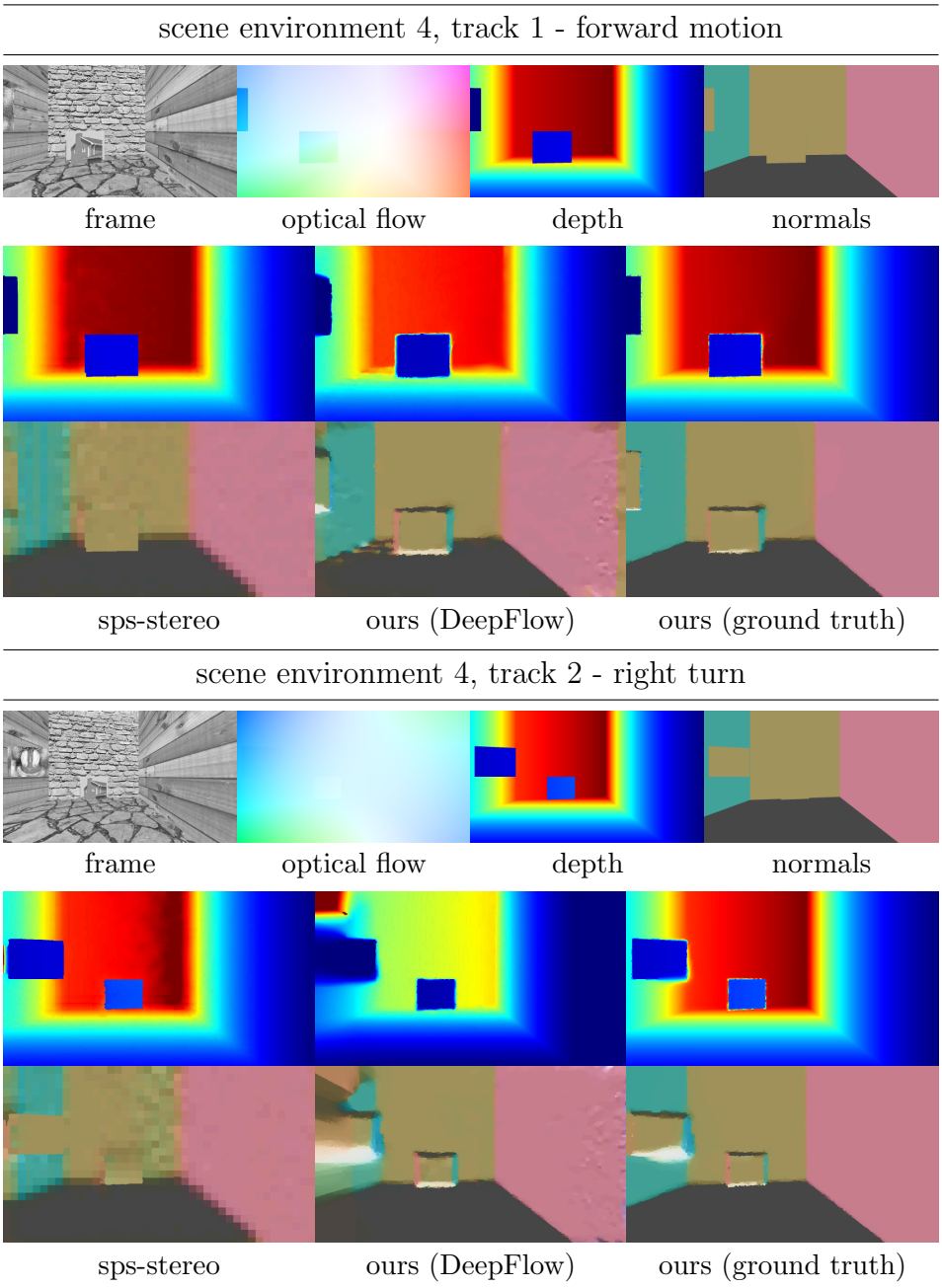


FIGURE 6.18. Rendered sequences 9 and 10, the environment contains large depth discontinuities and close objects. In this case, we have low discrepancy between estimated and ground truth flow, our regularity term oversmoothes the scene, which leads to strong artifacts at depth discontinuities. In case of estimated flow, we have an error in depth scale.

	seq.	mean [°]	2° [%]	5° [%]	10° [%]	20° [%]
ours (ground truth)	1	2.99	12.66	8.36	5.93	4.02
	2	2.93	12.85	8.98	6.04	4.00
	3	3.64	15.01	9.07	6.64	4.86
	4	2.61	12.62	7.34	4.95	3.34
	5	2.66	12.18	7.20	4.86	3.33
	6	2.61	14.58	7.27	4.49	3.04
	7	2.11	11.02	5.30	3.60	2.42
	8	2.48	18.52	8.21	4.58	2.81
	9	2.30	11.50	5.30	3.74	2.53
	10	9.24	23.61	21.04	19.71	18.11
ours	1	10.27	46.74	33.00	22.39	14.31
	2	10.00	48.77	33.27	23.17	15.43
	3	13.46	46.35	34.62	26.47	18.38
	4	11.55	52.39	35.48	24.50	15.83
	5	20.15	54.93	46.23	39.00	29.63
	6	7.14	33.56	18.68	13.38	9.90
	7	18.48	67.77	52.00	40.00	29.06
	8	38.25	84.67	75.53	65.27	51.23
	9	8.83	35.68	22.64	16.24	11.86
	10	29.78	63.99	57.55	52.93	48.10
sps-stereo	1	11.49	60.85	38.91	25.62	17.47
	2	11.23	61.50	42.86	28.48	16.48
	3	12.85	64.30	46.92	30.55	19.24
	4	10.84	60.60	37.27	24.73	16.84
	5	13.84	69.19	50.20	33.90	19.42
	6	10.44	60.15	34.44	23.59	15.91
	7	27.43	81.24	68.95	60.93	46.93
	8	34.99	89.77	75.97	67.81	56.67
	9	7.18	53.79	28.19	16.09	8.88
	10	9.03	68.06	44.27	25.72	11.24

TABLE 6.4. Rendered scenes normal evaluation. Sequences 7 and 8 show large error due to erroneous optical flow estimates, and sequence 10 has larger error on ground truth input data as well. In the other cases, our method shows similar average error as sps-stereo. When comparing the fraction of pixels that lie under a certain threshold, our approach shows larger variance, while the sps-stereo plane error distribution is more uniform.

CHAPTER 7

Conclusion

We presented novel approaches to discrete optical flow in chapter 5 and monocular structure estimation in chapter 6. The discrete optical flow problem is addressed in two ways, with sparse optical flow for outdoor traffic scenes, and a dense approach for general optical flow. Monocular structure estimation is approached with two methods as well. We formulate a minimum energy filter, which computes the optimal camera trajectory in a long monocular video sequence, and present a structure estimation approach is taking two frames only as input for estimating camera motion and a scene representation. The scene is assumed to be piecewise planar, and estimated with depth and normal information.

The sparse optical flow approach performs dense matching in a search window around distinct locations in the reference image. We compare different data energies, which are robust against changes in illumination, that occur frequently in outdoor traffic scenes. A limited number of matches is preserved, and further filtered according to distances from epipolar lines and parameterized depth. A discrete graphical model based on matching scores and a regularity energy on the optical flow field find the globally optimal displacement vectors. The approach achieves accurate but very sparse results.

In order to overcome the sparsity of the resulting optical flow fields, we designed a dense estimation method based on exhaustive matching and inference on discrete graphical models. Direct inference in the large graphical model is not feasible. We cluster the optical flow labels into a tree structure, and navigate through the tree by means of discrete optimization, where different strategies for label reduction are compared. Our method can track multiple branches in the label tree. However, following the single best cluster and applying the most greedy reduction method show best performance.

The recursive minimum energy filter computes the optimal camera trajectory in terms of a frame wise objective function depending on optical flow and depth observations. The filter is applied frame wise, and does not require caching of multiple frames, thus it remains most responsive.

Our scene estimation method fits egomotion parameters, depth and plane normals to estimated optical flow in a continuous framework. The scene is assumed to be smooth in most parts and piecewise planar.

Future Work

The approaches we proposed in this thesis have limitations. The matching energies we applied do not model changes in scale and orientation, which appear often in outdoor sequences. Furthermore, the exhaustive matching is very time consuming. An efficient hierarchical matching over different image resolutions may improve on runtime and performance of the approach, while maintaining the advantages of discrete optimization. A discrete optical flow method can incorporate further related tasks more easily than a continuous approach, such as occlusion detection or segmentation of optical flow into independent layers.

The recursive minimum energy filter for localization in a monocular sequence has the drawback, that optical flow and depth are assumed to be known. Especially depth is not directly available in a monocular camera configuration. The two frame scene reconstruction approach estimates camera motion, depth and plane normals from a single optical flow field. Two frames only provide very limited information, the approach can be extended to multiple frames by maintaining and updating a global model of the scene as future work. Explicit modeling of depth discontinuities could increase reconstruction accuracy of both depth and camera motion. A photometric error term could improve image alignment, and is another aspect for future research. The restriction to static scenes is unrealistic, traffic scenes are not static, and the dynamic objects are of particular interest to the (autonomous) driver. Incorporating a segmentation of dynamic objects, potentially with monocular scene flow estimation, is a very interesting direction for further work.

Bibliography

[1] P.-A. Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[2] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building rome in a day. *Commun. ACM*, 54(10):105–112, October 2011.

[3] Luis Alvarez, Rachid Deriche, Théo Papadopoulo, and Javier Sánchez. Symmetrical Dense Optical Flow Estimation with Occlusions Detection. *Int. J. Comput. Vision*, 2007.

[4] Michael Athans and Peter L. Falb. *Optimal Control. An Introduction to the Theory and Its Applications*. McGraw-Hill, 1966.

[5] A. Ayvaci, M. Raptis, and S. Soatto. Sparse Occlusion Detection with Optical Flows. *Int. J. Comp. Vision*, 2013.

[6] Christian Bailer, Bertram Taetz, and Didier Stricker. Flow Fields: Dense Correspondence Fields for Highly Accurate Large Displacement Optical Flow Estimation. In *Proc. Int. Conf. Comput. Vision*, 2015.

[7] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *ACM Trans. Graph.*, 28(3):24:1–24:11, July 2009.

[8] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

[9] Florian Becker, Frank Lenzen, Jörg H. Kappes, and Christoph Schnörr. Variational Recursive Joint Estimation of Dense Scene Structure and Camera Motion from Monocular High Speed Traffic Sequences. *Int. J. Comput. Vision*, 105:269–297, 2013.

[10] Florian Becker, Stefania Petra, and Christoph Schnörr. Optical Flow. In O. Scherzer, editor, *Handbook of Mathematical Methods in Imaging*, pages 1945–2004. Springer, 2nd edition, 2015.

[11] Johannes Berger, Frank Lenzen, Florian Becker, Andreas Neufeld, and Christoph Schnörr. Second-Order Recursive Filtering on the Rigid-Motion Lie Group SE(3) Based on Nonlinear Observations. *J. Math. Imag. Vision*, 58(1):102–129, 2017.

[12] Michael Bleyer, Christoph Rhemann, and Carsten Rother. PatchMatch Stereo - Stereo Matching with Slanted Support Windows. In *Proc. Brit. Mach. Vision Conf.*, 2011.

[13] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.*, 2010.

[14] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, November 2001.

[15] Matthew Brown and David G. Lowe. Automatic Panoramic Image Stitching Using Invariant Features. *Int. J. Comput. Vision*, 74(1):59–73, August 2007.

[16] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(3):500–513, 2011.

[17] Andres Bruhn, Joachim Weickert, and Christoph Schnörr. Lucas/Kanade meets Horn/Schunck: Combining Local and Global Optic Flow Methods. *Int. J. Comp. Vision*, 61(3):211–231, 2005.

[18] Florian Bugarin, Adrien Bartoli, Didier Henrion, Jean-Bernard Bernard Lasserre, Jean-José Orteu, and Thierry Sentenac. Rank-Constrained Fundamental Matrix Estimation by Polynomial Global Optimization Versus the Eight-Point Algorithm. *J. Math. Imag. Vision*, 2015.

[19] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary Robust Independent Elementary Features. In *Proc. Europ. Conf. Comp. Vision*, 2010.

[20] Antonin Chambolle and Thomas Pock. A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging. *J. Math. Imag. Vision*, 2010.

[21] Qifeng Chen and Vladlen Koltun. Fast MRF Optimization with Application to Depth Reconstruction. In *Proc. Conf. Comp. Vision Patt. Rec.*, June 2014.

[22] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *Proc. Conf. Comp. Vision Patt. Rec.*, 2005.

[23] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Patt. Anal. Mach. Intell.*, 29(6):1052–1067, June 2007.

[24] Amaël Delaunoy and Marc Pollefeys. Photometric Bundle Adjustment for Dense Multi-View 3D Modeling. In *Proc. Conf. Comp. Vision Patt. Rec.*, pages 1486–1493, 2014.

[25] Oliver Demetz, David Hafner, and Joachim Weickert. The Complete Rank Transform: A Tool for Accurate and Morphologically Invariant Matching of Structure. In *Proc. Brit. Mach. Vision Conf.*, 2013.

[26] Alan Edelman, Tomás A. Arias, and Steven T. Smith. The Geometry of Algorithms with Orthogonality Constraints. *SIAM J. Matrix Anal. Appl.*, 20(2):303–353, 1998.

[27] David Eigen and Rob Fergus. Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture. In *Proc. Int. Conf. Comput. Vision*, 2015.

[28] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-Scale Direct Monocular SLAM. In *Proc. Europ. Conf. Comput. Vision*, 2014.

[29] Anders Eriksson and Anton van den Hengel. Optimization on the Manifold of Multiple Homographies. In *Proc. Works. Int. Conf. Comp. Vision*, 2009.

[30] Anders Eriksson and Anton van den Hengel. Efficient Computation of Robust Weighted Low-Rank Matrix Approximations Using the L_1 Norm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(9):1681–1690, 2012.

[31] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient Belief Propagation for Early Vision. *Int. J. Comp. Vision*, 70(1):41–54, 2006.

[32] Andreas Geiger. Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. Conf. Comput. Vision Patt. Rec.*, 2012.

[33] Tom Goldstein, Xavier Bresson, and Stan Osher. Global Minimization of Markov Random Fields with Applications to Optical Flow. *J. Inv. Probl. Imag.*, 6(4):623–644, 2012.

[34] David Hafner, Oliver Demetz, and Joachim Weickert. Why Is the Census Transform Good for Robust Optic Flow Computation? In *Proc. Scale Space Var. Methods*, 2013.

[35] P. L. Hammer, P. Hansen, and B. Simeone. Roof Duality, Complementation and Persistency in Quadratic 0–1 Optimization. *Math. Progr.*, 28(2):121–155, 1984.

[36] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[37] Uwe Helmke, Knut Hüper, Pei Y. Lee, and John Moore. Essential Matrix Estimation Using Gauss-Newton Iterations on a Manifold. *Int. J. Comp. Vision*, 74(2):117–136, 2007.

[38] Uwe Helmke and John B. Moore. *Optimization and Dynamical Systems*. 1996.

[39] Heiko Hirschmüller. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Trans. Patt. Anal. Mach. Intell.*, 2008.

[40] Berthold K.P. Horn and Brian G. Schunck. Determining Optical Flow. *Art. Intell.*, 17:185–203, 1981.

[41] Hiroshi Ishikawa. Exact Optimization for Markov Random Fields with Convex Priors. *IEEE Trans. Patt. Anal. Mach. Intell.*, 25(10):1333–1336, October 2003.

[42] Jörg Hendrik Kappes, Bogdan Savchynskyy, and Christoph Schnörr. A Bundle Approach To Efficient MAP-Inference by Lagrangian Relaxation. In *Int. Conf. Comp. Vision*, 2012. in press.

[43] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Proc. Int. Symp. Mix. Augm. Real.*, 2007.

[44] Vladimir Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1568–1583, October 2006.

[45] Solomon Kullback and Richard A. Leibler. On Information and Sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951.

[46] Abhijit Kundu, Yin Li, Frank Dellaert, Fuxin Li, and James M. Rehg. Joint Semantic Segmentation and 3D Reconstruction from Monocular Video. In *Proc. Europ. Conf. Comput. Vision*, 2014.

[47] Lubor Ladicky, Paul Sturgess, Christopher Russell, Sunando Sengupta, Yalin Bastanlar, William F. Clocksin, and Philip H. S. Torr. Joint Optimization for Object Class Segmentation and Dense Stereo Reconstruction. *Int. J. Comp. Vision*, 100(2):122–133, 2012.

[48] J.M. Lee. *Introduction to Smooth Manifolds*. Graduate Texts in Mathematics. Springer, 2003.

[49] Jan Lellmann and Christoph Schnörr. Continuous Multiclass Labeling Approaches and Algorithms. *J. Imag. Sci.*, 4(4):1049–1096, 2011.

[50] Victor Lempitsky, Stefan Roth, and Carsten Rother. FusionFlow: Discrete-Continuous Optimization for Optical Flow Estimation. In *Proc. Conf. Conput. Vision Patt. Rec.*, 2008.

[51] Victor Lempitsky, Carsten Rother, and Andrew Blake. LogCut - Efficient Graph Cut Optimization for Markov Random Fields. In *Proc. Int. Conf. Comput. Vision*, 2007.

[52] Ce Liu, Jenny Yuen, and Antonio Torralba. SIFT Flow: Dense Correspondence Across Scenes and Its Applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):978–994, May 2011.

[53] Ming-Yu Liu, Shuoxin Lin, Srikumar Ramalingam, and Oncel Tuzel. Layered Interpretation of Street View Images. *Proc. Robot. Sci. Sys.*, 2015.

[54] Stephan Liwicki, Christopher Zach, Ondrej Miksik, and Philip H. S. Torr. Coarse-to-fine Planar Regularization for Dense Monocular Depth Estimation. In *Proc. Europ. Conf. Comput. Vision*, 2016.

[55] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[56] Bruce D. Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proc. Int. Conf. Art. Intell.*, pages 674–679, 1981.

[57] Ezio Malis and Manuel Vargas. Deeper Understanding of the Homography Decomposition for Vision-Based Control. Research Report RR-6303, INRIA, 2007.

[58] Ran Margolin, Ayellet Tal, and Lihi Zelnik-Manor. What Makes a Patch Distinct? In *Proc. Conf. Comput. Vision Patt. Rec.*, 2013.

[59] Moritz Menze and Andreas Geiger. Object Scene Flow for Autonomous Vehicles. In *Proc. Conf. Comput. Vision Patt. Rec.*, 2015.

[60] Moritz Menze, Christian Heipke, and Andreas Geiger. Discrete Optimization for Optical Flow. In *Proc. German Conf. Patt. Rec.*, 2015.

[61] C. Michelot. A Finite Algorithm for Finding the Projection of a Point onto the Canonical Simplex of $\mathbb{R}^n$. *J. Opt. Theory Applic.*, 1986.

[62] Thomas Minka. Divergence Measures and Message Passing. Technical report, Microsoft Research Ltd., Cambridge, UK, 2005.

[63] Jorge J Moré. The Levenberg-Marquardt algorithm: Implementation and Theory. In *Numerical analysis*, pages 105–116. Springer, 1978.

[64] R. E. Mortensen. Maximum-Likelihood Recursive Nonlinear Filtering. *J. Opt. Theory Appl.*, 2(6):386–394, 1968.

[65] Mikhail G. Mozerov. Constrained Optical Flow Estimation as a Matching Problem. *IEEE Trans. Imag. Process.*, 2013.

[66] Andreas Neufeld, Johannes Berger, Florian Becker, Frank Lenzen, and Christoph Schnörr. Estimating Vehicle Ego-Motion and Piecewise Planar Scene Structure from Optical Flow in a Continuous Framework. In *Proc. German Conf. Patt. Rec.*, 2015.

[67] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. DTAM: Dense Tracking and Mapping in Real-Time. In *Proc. Int. Conf. Comput. Vision*, 2011.

[68] David Pfeiffer and Uwe Franke. Efficient Representation of Traffic Scenes by Means of Dynamic Stixels. In *Proc. Intell. Vehic. Symp.*, 2010.

[69] Thomas Pock, Thomas Schoenemann, Gottfried Graber, Horst Bischof, and Daniel Cremers. A Convex Formulation of Continuous Multi-label Problems. In *Proc. Europ. Conf. Comput. Vision*, 2008.

[70] Huga Raguet, Jalal Dadili, and Gabriel Peyré. A Generalized Forward-Backward Splitting. *SIAM J. Imaging Sci.*, 2012.

[71] René Ranftl, Kristian Bredies, and Thomas Pock. Non-Local Total Generalized Variation for Optical Flow Estimation. In *Proc. Europ. Conf. Comput. Vision*, 2014.

[72] Rene Ranftl, Stefan Gehrig, Thomas Pock, and Horst Bischof. Pushing the Limits of Stereo using Variational Stereo Estimation. In *Proc. Intell. Vehic. Symp.*, 2012.

[73] Hatem A. Rashwan, Mahmoud A. Mohamed, Miguel Angel Garcia, Bärbel Mertsching, and Domenec Puig. Illumination Robust Optical Flow Model Based on Histogram of Oriented Gradients. In *Proc. German Conf. Patt. Rec.*, 2013.

[74] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. In *Proc. Comp. Vision Patt. Rec.*, 2015.

[75] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *Proc. Int. Conf. Comput. Vision*, 2011.

[76] Neus Sabater, Andrés Almansa, and Jean-Michel Morel. Meaningful Matches in Stereovision. *IEEE Trans. Patt. Anal. Mach. !ntell.*, 34(5):930–942, 2012.

[77] Dmitrij Schlesinger. Exact Solution of Permuted Submodular MinSum Problems. In *Proc. Energy Min. Methods Comput. Vision Patt. Rec.*, 2007.

[78] Laura Sevilla-Lara, Deqing Sun, Varun Jampani, and Michael J. Black. Optical Flow with Semantic Segmentation and Localized Layers. In *Proc. Conf. Comput. Vision Patt. Rec.*, 2016.

[79] Paul Smith, Tom Drummond, and Roberto Cipolla. Layered Motion Segmentation and Depth Ordering by Tracking Edges. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(4):479–494, 2004.

[80] Frank Steinbruecker, Thomas Pock, and Daniel Cremers. Large Displacement Optical Flow Computation without Warping. In *Proc. Int. Conf. Comput. Vision*, Kyoto, Japan, 2009.

[81] Hauke Strasdat, J. M. M. Montiel, and Andrew J. Davison. Visual SLAM: Why Filter? *Image Vision Comput.*, 2012.

[82] Jan Stühmer, Stefan Gumhold, and Daniel Cremers. Real-Time Dense Geometry from a Handheld Camera. In *Patt. Rec. DAGM Symp.*, 2010.

[83] Deqing Sun, Stefan Roth, and Michael J. Black. Secrets of Optical Flow Estimation and Their Principles. In *Proc. Conf. Comput. Vision Patt. Rec.*, 2010.

[84] Deqing Sun, Erik B. SuSudderth, and Michael J. Black. Layered Image Motion with Explicit Occlusions, Temporal Consistency, and Depth Ordering. In *Proc. Advan. Neural Infor. Process. Sys.*, 2010.

[85] Deqing Sun, Jonas Wulff, Erik Sudderth, Hanspeter Pfister, and Michael Black. A Fully-Connected Layered Model of Foreground and Background Flow. In *Proc. Conf. Comput. Vision Patt. Rec.*, 2013.

[86] Jian Sun, Nan-Ning Zheng, and Heung-Yeung Shum. Stereo Matching using Belief Propagation. *IEEE Trans. Patt. Anal. Mach. Intell.*, 2003.

[87] Keisuke Tateno, Federico Tombardi, Iro Laina, and Nassir Navab. CNN-SLAM: Real-Time Dense Monocular SLAM with Learned Depth Prediction. In *Proc. Comput. Vision Patt. Rec.*, 2017.

[88] Levi Valgaerts, Andrés Bruhn, Markus Mainberger, and Joachim Weickert. Dense versus Sparse Approaches for Estimating the Fundamental Matrix. *Int. J. Comput. Vision*, 96(2):212–234, 2012.

[89] Bart Vandereycken. Low-Rank Matrix Completion by Riemannian Optimization. *SIAM J. Optim.*, 23(2):1214—1236, 2013.

[90] Christoph Vogel, Stefan Roth, and Konrad Schindler. An evaluation of data costs for optical flow. In *GCPR*, 2013.

[91] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3D Scene Flow Estimation with a Piecewise Rigid Scene Model. *Int. J. Comput. Vision*, 115(1):1–28, October 2015.

[92] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP Estimation Via Agreement on Trees: Message-Passing and Linear Programming. *IEEE Trans. Inf. Theory*, 2005.

[93] Philippe Weinzaepfel, Jérome Revaud, Zaïd Harchaoui, and Cordelia Schmid. DeepFlow: Large Displacement Optical Flow with Deep Matching. In *Proc. Int. Conf. Comput. Vision*, 2013.

[94] Andreas Wendel, Michael Maurer, Gottfried Graber, Thomas Pock, and Horst Bischof. Dense Reconstruction On-the-Fly. In *Conf. Comput. Vision Patt. Rec.*, 2012.

[95] Koichiro Yamaguchi, Tamir Hazan, David McAllester, and Raquel Urtasun. Continuous Markov Random Fields for Robust Stereo Estimation. In *Proc. Europ. Conf. Comput. Vision*, 2012.

[96] Koichiro Yamaguchi, David A. McAllester, and Raquel Urtasun. Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation. In *Proc. Europ. Conf. Comput. Vision*, 2014.

[97] Ramin Zabih and John W. Ll. Non-parametric Local Transforms for Computing Visual Correspondence. In *Proc. Europ. Conf. Comput. Vision*, 1994.

[98] Christopher Zach. Fast and High Quality Fusion of Depth Maps. In *Proc. 3D Data Process. Visual. Transm.*, 2008.

[99] Christopher Zach, Thomas Pock, and Horst Bischof. A Globally Optimal Algorithm for Robust TV-L1 Range Image Integration. In *Proc. Int. Conf. Comput. Vision*, 2007.