MONOCULAR SCENE FLOW ESTIMATION FOR DYNAMIC TRAFFIC SCENES

Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften

vorgelegt beim Fachbereich Informatik und Mathematik
der Johann Wolfgang Goethe-Universität
in Frankfurt am Main

von

FABIAN BRICKWEDDE

aus Düsseldorf

Frankfurt am Main 2021

vom Fachbereich Informatik und Mathematik der
Johann Wolfgang Goethe-Universität als Dissertation angenommen.

Dekan:	Prof. Dr. Martin Möller
1. Gutachter:	Prof. Dr. Rudolf Mester
2. Gutachter:	Prof. Dr. Kostas Alexis
Datum der Disputation:	09.12.2021

ABSTRACT

The main topic of the present thesis is *scene flow* estimation in a *monocular camera* system. Scene flow describes the joint representation of 3D positions and motions of the scene. A special focus is placed on approaches that combine two kinds of information, deep-learning-based *single-view depth* estimation and model-based *multi-view geometry*.

The first part addresses single-view depth estimation focussing on a method that provides single-view depth information in an advantageous form for monocular scene flow estimation methods. A convolutional neural network, called *ProbDepthNet*, is proposed, which provides pixel-wise *well-calibrated depth distributions*. The experiments show that different strategies for quantifying the measurement uncertainty provide overconfident estimates due to overfitting effects. Therefore, a novel *recalibration technique* is integrated as part of the ProbDepthNet, which is validated to improve the calibration of the uncertainty measures. The monocular scene flow methods presented in the subsequent parts confirm that the integration of single-view depth information results in the best performance if the neural network provides depth distributions instead of single depth values and contains a recalibration.

Three methods for monocular scene flow estimation are presented, each one designed to combine multi-view geometry-based optimization with deep learning-based single-view depth estimation such as ProbDepthNet. While the first method, *SVD-MSfM*, performs the motion and depth estimation as two subsequent steps, the second method, *Mono-SF*, jointly optimizes the motion estimates and the depth structure. Both methods are tailored to address scenes, where the objects and motions can be represented by a set of rigid bodies. *Dynamic traffic scenes* are one kind of scenes that essentially fulfill this characteristic. The method, *Mono-Stixel*, uses an even more specialized scene model for traffic scenes, called *stixel world*, as underlying scene representation.

The proposed methods provide *new state of the art* for monocular scene flow estimation with Mono-SF being the first and leading monocular method on the KITTI scene flow benchmark at the time of submission of the present thesis. The experiments validate that both kind of information, the multi-view geometric optimization and the single-view depth estimates, contribute to the monocular scene flow estimates and are necessary to achieve the new state of the art accuracy.

ZUSAMMENFASSUNG

Motivation

Die menschliche Fähigkeit zu Sehen wird durch zwei Teile des Gehirns ermöglicht, dem ventralen und dorsalen Pfad. Der ventrale Pfad umfasst die Kategorisierung von Objekten sowie die Wahrnehmung von Formen. Diese Fähigkeit ist wesentlich durch die menschliche Erfahrung geprägt. Der dorsale Pfad hingegen reagiert auf schnelle zeitliche Veränderungen und Bewegungen zur Wahrnehmung von Entfernungen und zur Interaktion mit Objekten. Eine vergleichbare Unterteilung lässt sich auch im Bereich der *Computer Vision* feststellen. Die Detektion von Objekten, die semantische Klassifizierung und die Entfernungsschätzung basierend auf nur einem Bild ist wesentlich geprägt durch Machine bzw. Deep Learning Verfahren. Diese Verfahren basieren auf der erlernten Erscheinung von Objekten und Szenen in einem Bild. Im Gegensatz dazu zeigen im Bereich der Multi-View geometriebasierten Entfernungs- und Bewegungsschätzung (z.B. Simultaneous Localization and Mapping) weiterhin traditionelle Optimierungsverfahren die höchste Genauigkeit. Diese Verfahren basieren auf den geometrischen Zusammenhängen zwischen Bildern aus verschiedenen Perspektiven.

Im Bereich der Computer Vision werden die geometriebasierten Verfahren und Machine Learning Verfahren zum Großteil individuell betrachtet. Diese Arbeit adressiert die Kombination der traditionellen geometriebasierten Ansätze mit Deep Learning Verfahren zur Scene Flow Schätzung im Monokamerasetup. Insbesondere wird die Kombination mit Deep Learning Verfahren zur Entfernungsschätzung basierend auf einem Bild untersucht. Scene Flow beschreibt die Repräsentation der 3D Position der Bildpunkte sowie deren Bewegung in der Szene. Der Fokus dieser Arbeit liegt auf multirigiden Szenen, welche sich durch einen Satz von rigiden Objekten darstellen lassen. Diese Eigenschaft wird von Straßenszenen weitestgehend erfüllt.

Die Repräsentation des Scene Flows findet insbesondere Anwendung im automotive Bereich und in der Robotik. Basierend auf der Entfernung und Bewegung werden Kollisionen vermieden und Trajektorien können sicher abgefahren werden. Es sind allerdings auch weitere Anwendungen denkbar, beispielsweise im Bereich der Augmented Reality.

Probabilistische Single-View Depth Schätzung

Single-View Depth Schätzung beschreibt Verfahren zur Rekonstruktion der Tiefenstruktur einer Szene basierend auf nur einem Bild. Der wesentliche Durchbruch wurde durch Deep Learning Methoden ermöglicht, welche die Tiefenstruktur anhand der typischen Erscheinung von Szenen erlernen. So liefert beispielsweise die Größe und Textur von Objekten, der typische Verlauf von Straßenoberflächen oder auch die Relationen zwischen Objekten einen Hinweis auf die Tiefenstruktur der Szene.

Während die meisten Verfahren nur einen einzigen Schätzwert für die Entfernung pro Pixel ausgeben, liefern nur wenige Verfahren ein zusätzliches Maß für die Unsicherheit. Die Unsicherheiten eines neuronalen Netzes lassen sich in 3 Kategorien unterteilen, (1) der Unsicherheit basierend auf den Messungen und der Mehrdeutigkeit des Problems, (2) der Unsicherheit der Modellparameter und (3) die Unsicherheit aufgrund großer Abweichungen der Testdaten von den Trainingsdaten. Moderne neuronale Netze geben typischerweise eine zu hohe Konfidenz aus und zeigen damit eine schlechte Kalibrierung. Dadurch lassen sich die Unsicherheitsmaße nicht direkt als Wahrscheinlichkeiten interpretieren oder Konfidenzintervalle ableiten.

Der erste Teil dieser Arbeit adressiert probabilistische Single-View Depth Schätzung mit dem Fokus auf der Unsicherheitsschätzung und Kalibrierung. Die Analyse der empirischen Verteilung einer Single-View Depth Schätzung [Godard et al., 2017] zeigt folgende Charakteristiken auf. Die Verteilung über die inverse Tiefe lässt sich durch ein Mixture Model, bestehend aus einer Laplace und einer Gaußverteilung, approximieren. Während sich der Fehler der inversen Tiefe nahezu konstant über die Entfernungen verhält, zeigt der Fehler eine hohe Abhängigkeit zur semantischen Klasse. Insbesondere Fahrzeuge und die Straße zeigen eine höhere Genauigkeit verglichen mit beispielsweise der Vegetation.

In dieser Arbeit wird das *ProbDepthNet* präsentiert. ProbDepthNet ist ein Convolutional Neural Network, welches pixelweise die Parameter einer Tiefenverteilung schätzt. Das Netz wird trainiert, den negativen Log-Likelihood der Tiefenverteilung auf den Trainingsdaten zu minimieren. Die Experimente zeigen, dass die daraus resultierenden Verteilungen ein Maß für die Unsicherheit darstellen, allerdings schlecht kalibriert sind. ProbDepthNet integriert zusätzlich eine neuartige Rekalibrierungstechnik. Wenige zusätzliche Layer, das *CalibNet*, werden integriert, welche die Parameter der Verteilung auf rekalibrierte Parameter abbildet. Diese Layer werden trainiert, den negative Log-Likelihood auf einem separaten Teil der Trainingsdaten zu minimieren. Die Experimente zeigen, dass dadurch mit dem CalibNet eine wesentlich bessere Kalibrierung erzielt wird, wobei die Rekalibrierungstechnik auch auf andere Verfahren zur Unsicherheitsschätzung anwendbar ist. ProbDepthNet adressiert die Unsicherheiten basierend auf den Messungen und der Mehrdeutigkeit des Problems.

Die darauffolgenden Verfahren zur monokamerabasierten Scene Flow Schätzung zeigen eine höhere Genauigkeit, wenn die Single-View Depth Schätzung als probabilistische Verteilungen integriert werden, wobei auch die Kalibrierung der Verteilung entscheidend ist.

Monokamerabasierte Scene Flow Schätzung

Die darauffolgenden Teile der Arbeit befassen sich mit der Scene Flow Schätzung. Scene Flow Schätzung wird typischerweise in einem Stereokamerasetup adressiert. Zur Schätzung der Bewegung und Entfernung werden korrespondierende Bildpunkte sowohl in den statischen Stereobildpaaren als auch in aufeinanderfolgenden Stereobildern einander zugeordnet. Eine Monokamera ist oft bevorzugt aufgrund geringerer Kosten, kleinerer Verbaugröße und der nicht notwendigen Kalibrierung der Stereokameras zueinander. Allerdings ist die Scene Flow Schätzung ein mehrdeutiges Problem aus einer geometrischen Perspektive im Monokamerasystem. Es besteht eine Skalierungsmehrdeutigkeit zwischen der translatorischen Bewegung einer Kamera und der Skalierung der Umgebungskarte.

Diese Arbeit präsentiert drei Verfahren, welche die geometrischen Ansätze mit Deep Learning-basierten Verfahren kombinieren, wodurch die beschriebenen Limitierungen adressiert und eine höhere Genauigkeit erreicht wird.

SVD-MSfM Methode: Multi-Body Structure from Motion

Die erste Methode zur Scene Flow Schätzung in dieser Arbeit wird mit *SVD-MSfM* (Abkürzung für Single-View Depth meets Multi-body Structure from Motion) bezeichnet. SVD-MSfM integriert die probabilistische Single-View Depth Schätzungen des ProbDepthNets in einem *Multi-body Structure from Motion*-basierten Ansatz (wie z.B. [Ranftl et al., 2016]). Das Verfahren lässt sich in zwei wesentliche Schritte unterteilen.

Der erste Schritt entspricht der Bewegungsschätzung für jedes Objekt, welches mittels einer Instanzensegmentierung detektiert wurde, sowie der Kamerabewegung zwischen zwei aufeinanderfolgenden Bildern. Die Schätzung der 6D Bewegung ist formuliert als ein nichtlineares Minimierungsproblem, welches die Bewegungsparameter als auch einen Satz von 3D Szenenpunkten gemeinsam optimiert. Der erste Anteil des Optimierungsproblems minimiert den Reprojektionsfehler basierend auf optischen Flussvektoren. Der zweite Anteil führt zu 3D Szenenpunkten, dessen Entfernungen plausibel zu den Tiefenverteilungen des ProbDepthNets sind. Durch die Supervision des ProbDepthNets sind die Schätzungen metrisch skaliert und liefern damit auch implizit die korrekte Skalierung der Bewegungsschätzung. Während die Skalierung sonst oftmals über die bekannte Kamerahöhe über den Boden bestimmt wird, ist diese Methode auch auf bewegte Objekte anwendbar. Die Levenberg-Marquardt Methode wird zur Optimierung des Minimierungsproblems angewandt. SVD-MSfM stellt damit ein Verfahren dar, welches eine 6D Bewegung inklusive Skalierung für bewegte Objekte liefert basierend auf der Kombination von Single-View Depth Schätzung und Multi-View geometriebasierter Optimierung. Die Experimente zeigen, dass SVD-MSfM eine Genauigkeit der Bewegungsschätzung im Bereich von State-of-the-Art Methoden erreicht und robustere Schätzungen liefert, wenn das Verfahren mit Initialisierungsproblemen konfrontiert wird. Diese Eigenschaft ist insbesondere für bewegte Objekte wichtig, welche regelmäßig im Sichtfeld erscheinen und verschwinden.

Während der erste Schritt einzelne 3D Szenenpunkte optimiert, liefert der zweite Schritt eine dichte Tiefenschätzung für alle Pixel. Basierend auf der Assoziation von Pixeln zu Objekten anhand der Instanzensegmentierung sowie der zugehörigen Bewegungsschätzung, ist für jede Tiefe eines Pixels die zugehörige Bildposition im darauffolgenden Bild eindeutig bestimmt. Dieses ermöglicht es, die photometrische Distanz für jeden Eintrag in einem Kostenvolumen über die Tiefe zu bestimmen. Zusätzlich wird die Wahrscheinlichkeit jedes Tiefenwertes basierend auf den vom ProbDepthNet geschätzten Tiefenverteilungen bewertet. Die Tiefenschätzungen werden anschließend basierend auf dem Tiefenkostenvolumen durch Anwendung der Semi Global Matching [Hirschmuller, 2005] und Slanted Plane Smoothing [Yamaguchi et al., 2014] Methoden bestimmt.

Die Experimente vergleichen erstmalig verschiedene Methode zur monokamerabasierten Rekonstruktion basierend auf einer Scene Flow Metrik, wobei vier Kategorien an Methoden analysiert werden: (1) Multi-Task Convolutional Neural Networks zur optischen Fluss- und Tiefenschätzung, (2) die Kombination individueller Methoden zur optischen Fluss- und Single-View Depth Schätzung, (3) Multi-body Structure from Motion und (4) die Kombination geometrischer Optimierung und Single-View Depth Schätzung. Die Experimente zeigen, dass SVD-MSfM höhere und neue State-of-the-Art Genauigkeit für monokamerabasierte Scene Flow Schätzung liefert. Weitere Experimente analysieren die wesentlichen Komponenten von SVD-MSfM. Die Deaktivierung der Multi-View geometriebasierten Information oder der Single-View Depth Schätzung führen jeweils zu einer Verschlechterung, welches die Aussage stützt, dass beide Informationen einen Beitrag zur Scene Flow Schätzung liefern. Des Weiteren zeigen Experimente die Wichtigkeit der probabilistischen Formulierung und Rekalibrierungsmethode des ProbDepthNets zur Integration von Single-View Depth Schätzungen.

Mono-SF Methode: Objekt Scene Flow

SVD-MSfM folgt dem Multi-body Structure from Motion-Konzept und formuliert die Bewegungs- und Tiefenschätzung als zwei aufeinanderfolgende Schritte. Die zweite Methode zur Scene Flow Schätzung in dieser Arbeit wird als *Mono-SF* bezeichnet und zeichnet sich durch die gemeinsame Bewegungs- und Tiefenschätzung aus.

Die Szene wird durch einen Satz von planaren Oberflächenelementen, einem Satz von rigiden Objekten sowie der Assoziation von Oberflächenelementen zu Objekten repräsentiert. Jedes Oberflächenelement ist einem Superpixel im Bild zugeordnet und ein skalierter Normalenvektor definiert die Fläche in der Szene. Die rigiden Objekte folgenden der Definition von SVD-MSfM, wobei diese durch eine Instanzensegmentierung detektiert und durch eine 6D Bewegung beschrieben werden. Die statische Umgebung bildet einen weiteren rigiden Körper. Aufgrund der objektorientierten Repräsentation der Bewegung wird Mono-SF der Kategorie der *Objekt Scene Flow*-Verfahren [Menze and Geiger, 2015] zugeordnet.

Die Scene Flow Optimierung ist als Energieminimierungsproblem formuliert. Ein Glattheitsterm modelliert, dass benachbarte Oberflächenelemente typischerweise koplanar sind und geringe Unterschiede in der Tiefe aufweisen. Die Datenterme folgen den wesentlichen Erkenntnissen von SVD-MSfM und integrieren probabilistische Single-View Depth Schätzungen des ProbDepthNets sowie eine photometrische Distanz basierend auf der Multi-View Geometrie. Zur Optimierung wird Sequential Tree-reweighted Message Passing verwendet.

Die Experimente zeigen, dass die gemeinsame Optimierung gegenüber SVD-MSfM, welches als Initialisierung verwendet wird, in eine weitere Verbesserung der Genauigkeit resultiert. Wesentliche Erkenntnisse von SVD-MSfM werden durch weitere Experimente gestützt. Beide Informationen, die Single-View Depth und photometrische Distanz, tragen zur Erhöhung der Genauigkeit bei und sind notwendig zur Erreichung der neuen State-of-the-Art Genauigkeit für eine monokamerabasierte Scene Flow Schätzung. Außerdem wird auch für Mono-SF die Wichtigkeit der probabilistischen Repräsentation und Kalibrierung des ProbDepthNets unterstrichen.

Mono-Stixel Methode: Stixel Scene Flow

Die Ergebnisse von SVD-MSfM und Mono-SF motivieren einen praktischen Einsatz der Verfahren zur Scene Flow Schätzung in einem Monokamerasystem. Allerdings ist die reine Genauigkeit nicht der alleinige Aspekt, um für einen praktischen Einsatz geeignet zu sein. Typischerweise ist die Scene Flow Repräsentation nicht das endgültige Ziel, sondern viel mehr eine Zwischenrepräsentation, auf der Anwendungen wie Notbremsfunktionalitäten aufbauen. Die Notwendigkeit, die Repräsentation weiterzuverarbeiten, zeigt die zusätzlichen Anforderungen auf, dass die Repräsentation möglichst kompakt sein sollte.

Eine kompakte und trotzdem detaillierte Repräsentation speziell für Straßenszenen ist die sogenannte *Stixel World* [Badino et al., 2009, Pfeiffer and Franke, 2011b]. In dieser Arbeit wird die *Mono-Stixel* Methode präsentiert, welche eine monokamerabasierte Scene Flow Schätzung umsetzt mit einer Stixel World als zugrunde liegendes Szenenmodell.

Die Stixel World entspricht der spaltenweisen Unterteilung des Bildes in planare Segmente, den Stixeln. Durch die eingeschränkte Form lässt sich die Segmentierung eines Stixel durch die obere und untere Begrenzung in der Spalte darstellen. Jeder Stixel wird zusätzlich durch die folgenden Labels beschrieben, welche eine detaillierte Szenenrepräsentation ermöglichen: (1) Der Stixel Typ unterscheidet die vier Typen Boden, statisches Objekt, dynamisches Objekt und Himmel, (2) ein Label definiert die semantische Klasse wie Straße, Fahrzeug oder Gebäude, (3) weitere Label definieren die Geometrie und Bewegung, (4) jedes Segment wird einem rigiden Objekt oder der statischen Umgebung zugeordnet und (5) ein weiterer Wert repräsentiert die Independent Moving Object Wahrscheinlichkeit des Stixels in Bewegung zu sein.

Die vier Stixel Typen definieren spezifische Modellannahmen. Während ein Bodenstixel eine liegende Orientierung hat, stehen Objektstixel senkrecht auf dem Boden und der Himmel ist unendlich weit entfernt. Nur Stixel des dynamischen Objekttyps besitzen eine Eigenbewegung, welche entweder durch eine 2D Bewegung über den Boden oder durch die Zuordnung zu einem rigiden Objekt wie bei SVD-MSfM und Mono-SF definiert ist. Die anderen Typen sind dem rigiden Körper der statischen Umgebung zugeordnet und die relative Bewegung ist entsprechend durch die Kamerabewegung definiert. Außerdem ist eine eindeutige Zuordnung von semantischen Klassen zu Stixel Typen gegeben. Die Modellannahmen definieren somit Zusammenhänge zwischen den Labels und schränken die Menge an konsistenten Lösungen ein. Dadurch stützt zum Beispiel die Integration einer semantischen Segmentierung auch die Schätzung der richtigen Stixeltypen und damit die Anwendung des richtigen Modells bezüglich der Orientierung und Bewegung.

Aufgrund der Unterteilung in Spalten lässt sich die Mono-Stixel Segmentierung als ein 1D Segmentierungsproblem ausdrücken. Die Schätzung basiert auf einem optischen Fluss, probabilistischen Single-View Depth Schätzungen des ProbDepthNets, semantischen Segmentierung und einer Instanzensegmentierung. Die Definition eines Energieterms integriert die verschiedenen Eingänge basierend auf deren Messmodellen. Zur Bestimmung des erwarteten optischen Flusses eines Stixels definiert jeder Stixeltyp ein spezifisches Homographiemodell. Die Abweichung zum erwarteten optische Fluss wird als Reprojektionsfehler bewertet. Zusätzlich repräsentieren paarweise Terme das Priorwissen über die typische Struktur von Straßenszenen. Objekte stehen typischerweise auf dem Boden, vordere Objekte verdecken dahinterliegende und die Bodenoberfläche ist weitestgehend eben ohne größere Sprünge in der Höhe.

Das 1D Energieminimierungsproblem lässt sich global optimal mithilfe des Viterbi Algorithmus lösen. Um allerdings den Rechenaufwand zu reduzieren, wird nur die Segmentierung und der Stixel Typ global optimiert und die anderen Label eines Stixels lokal geschätzt. Die lokale Schätzung basiert unter anderem auf den Homographiemodellen der Stixel Typen, wobei eine Direct Linear Transform die Freiheitsgrade der Homographie unter Berücksichtigung der Stixel Modellannahmen basierend auf den optischen Flussmessungen liefert.

Die beschriebene Mono-Stixel Segmentierung führt die Schätzung für jede Spalte individuell durch. Basierend auf der spaltenweisen Lösung wird der Mono-SF Algorithmus zur globalen Optimierung angewandt mit der Stixel World als zugrunde liegenden Repräsentation.

Zusätzlich wird für jedem Stixel eine Independent Moving Object Wahrscheinlichkeit basierend auf einem Hypothesentest bestimmt. Der Hypothesentest vergleicht die Wahrscheinlichkeit eines Stixels des dynamischen Objekttyps zu der Wahrscheinlichkeit der statischen Stixeltypen.

Zusätzlich zur kompakteren Darstellung und weiteren Informationen wie die Independent Moving Object Detektion, zeigt die Mono-Stixel Methode eine leicht höhere Genauigkeit als Mono-SF. Die qualitativen Ergebnisse zeigen außerdem bessere Charakteristiken bei der Rekonstruktion von dünnen Objekte (z.B. Pfosten) und bei geringer translatorischer Bewegung.

Zusammenfassung

Die vorliegende Arbeit adressiert erstmalig explizit die Scene Flow Schätzung in einem Monokamerasetup. Ein spezieller Fokus ist auf die Kombination der Multi-View Geometrie mit Deep Learning Methoden insbesondere zur Single-View Depth Schätzung gelegt. Es werden neuartige Methoden zur monokamerabasierten Scene

Flow Schätzung vorgestellt, wobei Mono-SF die erste und aktuell führende[1] Methode auf dem KITTI Scene Flow Benchmark [Menze and Geiger, 2015] darstellt. Beide Informationen, die Multi-View Geometrie und Single-View Depth, liefern einen wesentlichen Beitrag zur Genauigkeit der Scene Flow Schätzungen in den vorgestellten Methoden. Ein neuronales Netz zur Ausgabe einer probabilistischen und kalibrierten Darstellung von Single-View Depth Schätzungen wird vorgestellt und es wird gezeigt, dass diese Darstellung vorteilhaft zur Integration in die Scene Flow Methoden ist.

[1] bezogen auf veröffentlichte Methoden des KITTI Scene Flow Benchmark am 02. Januar 2021

CONTENTS

PUBLICATIONS

Brickwedde, F., Abraham, S., and Mester, R. (2018a). Exploiting Single Image Depth Prediction for Mono-Stixel Estimation. In *Proc. of European Conference on Computer Vision Workshops (ECCVW)*. IEEE.

Brickwedde, F., Abraham, S., and Mester, R. (2018b). Mono-Stixels: Monocular Depth Reconstruction of Dynamic Street Scenes. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Brickwedde, F., Abraham, S., and Mester, R. (2019). Mono-SF: Multi-View Geometry Meets Single-View Depth for Monocular Scene Flow Estimation of Dynamic Traffic Scenes. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*. IEEE.

ACRONYMS

ADAS	advanced driver assistance system
ADF	assumed density filtering
AUSE	area under sparsification error
AP	average precision
BA	bundle adjustment
CNN	convolutional neural network
CRF	conditional random field
ECE	expected calibration error
FN	false negative
FP	false positive
GAN	generative adversial network
HMM	hidden Markov model
IoU	intersection over union
IMO	independent moving object
LC	loop closing
LiDAR	light detection and ranging
MoG	mixture of Gaussians
MSfM	multi-body structure from motion
NLL	negative log-likelihood
OOD	out of distribution
P	precision
R	recall
ROC	receiver operating characteristic
RMSE	root mean squared error
SfM	structure from motion
SLAM	simultaneous localization and mapping
SotA	state of the art
SSIM	structural similarity
TN	true negative
TP	true positive
VAE	variational auto-encoder

NOTATION

General Notation

$\mathbf{M}$	matrix
$\mathbf{v}$	vector
s	scalar value
$\mathcal{S}$	set of elements
$\mathbf{M}_{i,j}, \mathbf{v}_i$	indexed element at i or i, j
$[\mathbf{x}]_\times$	cross product matrix
$\mathbf{I}_{n\times m}$	identity matrix of size $n \times m$
$\mathbf{0}_{n\times m}$	zero matrix of size $n \times m$
$diag(\mathbf{M})$	diagonal matrix with the elements on the diagonal of $\mathbf{M}$

Scene Geometry

$\mathbf{X} = (X, Y, Z, 1)^T$	3D scene point coordinates (typically in homogenous coordinates)
$\mathbf{p} = (u, v, 1)^T$	2D image coordinates (typically in homogenous coordinates)
d	distance or disparity
ρ	inverse depth
$\mathbf{n}$	normal vector of plane
$\mathbf{T} \in SE(3)$	transformation matrix
$\mathbf{R} \in SO(3)$	rotation matrix
$\mathbf{t}$	translation vector
$\xi \in se(3)$	Lie-algebra elements of transformation

Camera Parameter and Projective Geometry

$I_{(h\times w)}$	image of height h and width w
$\mathcal{W}_{\mathbf{p}}^{m\times n}$	window of size $m \times n$ centered at $\mathbf{p}$
$\mathcal{G}$	gradient image
$\mathcal{B}$	census transform
$\mathbf{R}^{v2c}, \mathbf{R}^{c2v}$	extrinsic rotation between camera and vehicle coordinates

$\mathbf{C}$	camera center
$\mathbf{P}$	projection matrix
$\mathbf{K}$	intrinsic camera matrix
$\pi(\mathbf{X})$	projection of 3D point into image coordinates
m_x, m_y	pixel size
f_x, f_y	camera constant normalized by pixel size
c_x, c_y	principle point normalized by pixel size
s	skew or scaling factor
$\mathbf{F}$	fundamental matrix
$\mathbf{E}$	essential matrix
$\mathbf{e}$	epipole
$\mathbf{l}$	epipolar line
$\mathbf{H}$	homography

Statistic and Optimization Problems

$p(A\|B)$	probability of A conditioned on B
$\mathcal{N}(\mu, \sigma^2)$	Gaussian distribution with variance σ^2 and mean μ
Σ	covariance matrix
$\mathcal{L}(b, \mu)$	Laplace distribution with scale b and mean μ
$\hat{\mathbf{x}}$	estimated value of system variables $\mathbf{x}$
r	residual
$\mathcal{L}$	loss function
E	energy term
Φ	unary term (e.g. data likelihood)
Ψ	pairwise term (e.g. regularization)
Θ	model-/ hyperparameters
τ	truncation value
λ	weighting factor
$\mathbf{J}$	Jacobi matrix
δ	increment
γ	threshold
$\mathcal{H}$	hypothesis
$\mathcal{O}(\cdot)$	O-notation for complexity

1 INTRODUCTION

CONTENTS

Human vision is enabled by two essential parts of the brain, the *ventral* and the *dorsal stream* [Mishkin and Ungerleider, 1982, Goodale and Milner, 1992]. The ventral stream is also known as the 'what pathway' and refers to the object recognition and representation of object forms. The dorsal stream represents the 'where and how pathway', which allows locating objects especially important to interact with them. The ventral stream is characterized by the fact to be sensitive to high spatial frequencies and details. The ability to recognize objects is essentially based on the learned appearance. In contrast to that, the dorsal stream perceives high temporal frequencies to recognize the location and motion of objects. An object shows a visually greater movement if it is closer to the moving observer. This is known as motion parallax and an important visual cue exploited by the dorsal stream.

The methods developed in the field of computer vision are often motivated by the human way of visual perception. The tasks of *semantic* and *instance segmentation* are focused on recognizing object classes and which parts of an image correspond to the same object instance. These approaches refer to the ventral stream and are today dominated by *deep learning*-based methods for learning the appearance of object categories. The ability of the ventral stream to recognize object shapes and forms based on the learned experience is emulated by convolutional neural networks (CNNs) for *single-view depth estimation*. In connection with the dorsal stream, methods such as *structure from motion* (SfM) should be mentioned, which exploit the principles of *multi-view geometry* to perceive the environment structure in terms of depth and motion. These methods are based on temporal information and are still dominated by traditional optimization and model-based approaches.

Distinguishing the human visual perception into two separate parts was challenged by several research works (e.g. [Milner, 2017]), which show that there are interactions between both streams. While these tasks are mostly addressed individually in the computer vision domain, the combination of the principles of multi-view geometry with deep learning-based perception is investigated in the context of a *scene flow estimation* in the present thesis. A special emphasis is placed on the combination with deep learning-based single-view depth estimation.

Scene flow is defined as the 3D motion field and 3D structure observed by at least two cameras at two consecutive time steps ([Vogel, 2015, p. 2]). Scene flow methods typically estimate the motion and structure of the scene *jointly* and provide *dense* results, which means providing a scene flow estimate for each pixel of at least one reference image. Four degrees of freedom per pixel are needed to define a scene flow estimate. The 3D translational motion of each pixel between the time points at which the images are captured defines the 3D motion field. The depth of each pixel defines the 3D structure for a calibrated camera. In the present thesis, the focus is placed on the scene flow estimation with a single monocular camera instead of the typically used stereo cameras.

Due to the explicit estimation of motion, scene flow is designed for *dynamic scenes*. Dynamic scenes are scenes that include objects with an individual motion. The relative motion of the camera to the scene is consequently not only describable by the camera motion itself. The present thesis is tailored to dynamic scenes, where the motion inherent in the scene can be described by the motion of a set of rigid moving bodies. Each rigid body motion is defined by six degrees of freedom, composed of the three translational and three rotational degrees of freedom. Dynamic scenes restricted to rigid moving objects are denoted as *multi-rigid-object dynamic scenes* or briefly as *multi-object dynamic scenes* in the present thesis. One kind of scene, which can be almost completely described by a set of rigid bodies, is a *dynamic traffic scene*. The domain of dynamic traffic scenes serves as a basis for evaluation and domain for future applications.

In summary, the core research question addressed in this thesis can be formulated as follows:

How to combine the principles of multi-view geometry with deep learning-based perception for scene flow estimation in a monocular camera setup focusing on multi-rigid-object dynamic scenes?

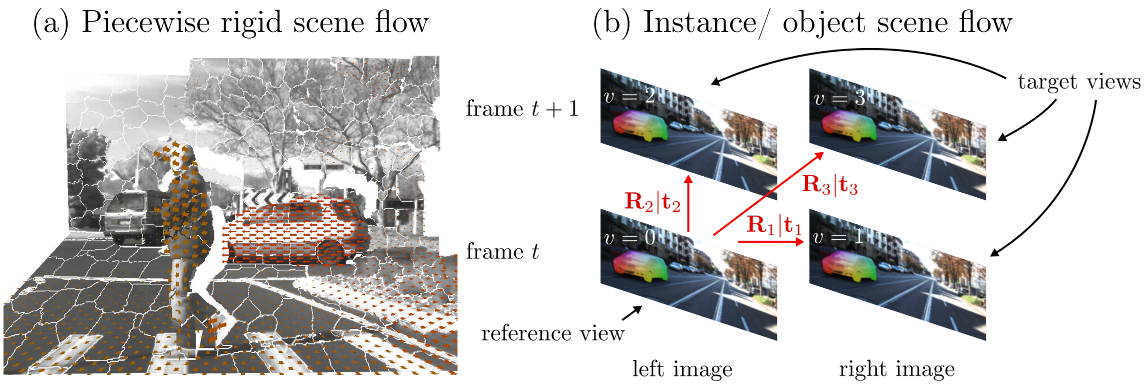

(a) Piecewise rigid scene flow
(b) Instance/ object scene flow
frame $t+1$
frame t
reference view
target views
$v=2$
$v=3$
$v=0$
$v=1$
$\mathbf{R}_2|\mathbf{t}_2$
$\mathbf{R}_3|\mathbf{t}_3$
$\mathbf{R}_1|\mathbf{t}_1$
left image
right image

Figure 1.1: Stereo-based scene flow estimation methods represent the 3D position and 3D motion of pixels in the image. (a) Piecewise rigid scene flow [Vogel et al., 2013] divides the scene into a set of 3D rigid planar surface elements including their 3D position and 6D motion. The relative motion of each surface element is indicated by the arrows. (b) Instance or object scene flow [Behl et al., 2017] divides the scene into a set of planar surface elements and assigns each surface element to a rigid body (e.g. a vehicle) represented by its 6D motion. The figures are taken from the corresponding papers [Vogel et al., 2013] (©2013 IEEE) and [Behl et al., 2017] (©2017 IEEE).

1.1 Motivation

To motivate the thesis, the present section gives answers to two important questions: First, why focus on monocular scene flow estimation in general? Second, why study in particular the combination of multi-view geometry-based principles with deep learning-based approaches?

1.1.1 Scene Flow Estimation

Methods fo 3D scene flow estimation provide the 3D structure and 3D motion of a scene. They gained a lot of interest, for example in the context of advanced driver assistance systems (ADASs) or autonomous driving. For example, such methods can be used to determine a collision risk with a pedestrian or vehicle or to navigate safely through the scene.

Stereo-based scene flow estimation: Scene flow estimation is traditionally based on a temporal series of stereo images. Two popular *stereo-based scene flow* methods are illustrated in figure 1.1. *Piecewise rigid scene flow* [Vogel et al., 2013] divides the scene into a set of planar surface elements oriented in 3D space. Each plane is defined by its 3D geometry in the scene and is considered as a rigid moving element with its corresponding 6D motion (three translational and three rotational degrees of freedom). Based on two stereo image pairs, the basic principle is to op-

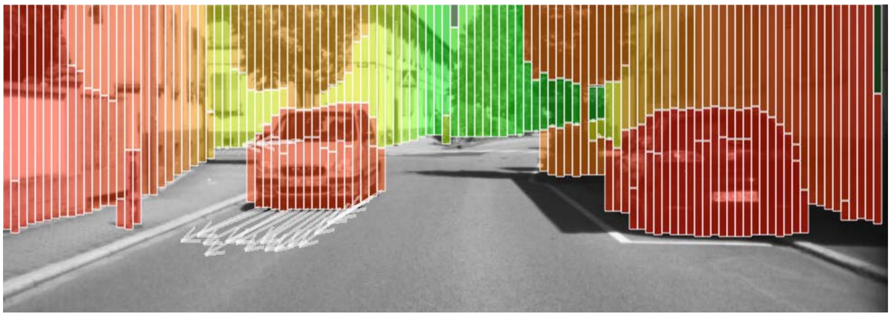


Figure 1.2: Stereo-based stixel world representation [Franke et al., 2013]. The image is divided into column-wise segments, each one corresponding to a stick-like element (the stixel) in the scene. Using a Kalman-Filter-based tracking, the motion state of each stixel is estimated (visualized by the arrows). The figure is taken from the corresponding paper [Franke et al., 2013] (©2013 IEEE).

timize the geometry and motion of the planes such that a photometric distance is minimized by warping each plane in all stereo images. The result in figure 1.1 (a) indicates that this approach is able to represent the depth structure as well as the motion of the scene.

To exploit the special conditions of *dynamic traffic scenes*, Menze and Geiger [2015], Behl et al. [2017] proposed the *object* or *instance scene flow* approach (see figure 1.1 (b)). A traffic scene, in particular, typically consists of a few independent motions by vehicles and other objects. Therefore, the object scene flow model introduces a set of rigid body motions and assigns each plane to one rigid body. The scene flow estimation is formulated as a joint optimization of the 6D object motion parameters, 3D plane parameters and the association of planes to objects.

Stixel world: While the presented stereo scene flow methods provide an accurate representation of the depth and motion of a scene, Badino et al. [2009], Pfeiffer and Franke [2011b,a] additionally addressed the challenge of providing a compact, but detailed representation and proposed the so-called *stixel world* (see figure 1.2). Such compact representation is beneficial to reduce the amount of data that needs to be stored, transferred to, or processed by a subsequent application. The stixel world representation encodes the scene as a column-wise segmentation in ground and object stixels defined by its geometry. The estimation is typically based on a dense disparity map. Initially, the stixel world approaches were designed to represent the first row of closest objects in each column [Badino et al., 2009]. However, the stixel world representation was later extended to provide the depth structure of the whole image [Pfeiffer and Franke, 2011b]. By integrating optical flow estimates in addition to the dense disparity map, the stixels are tracked over time and the motion of each stixel is estimated using a Kalman Filter [Pfeiffer and Franke, 2011a] (see figure 1.2). Thereby, a complete scene flow is defined by the stixel world representation in a compact way. The stixel world representation is also used as the medium-level environment abstraction for the autonomous driving research project presented in [Franke et al., 2013].

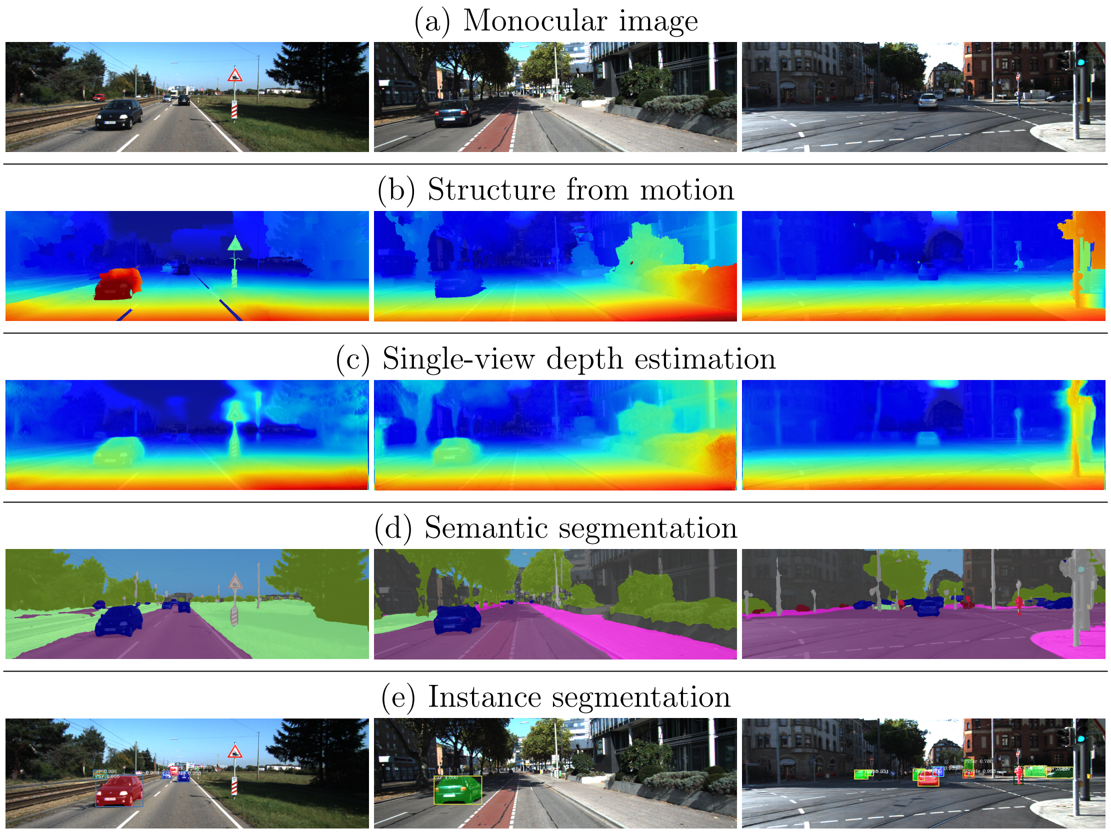
Stereo-based scene flow methods are suitable for an accurate representation of the depth and motion of a dynamic traffic scene and specialized representations for dynamic traffic scenes are introduced. However, monocular cameras are often preferred due to be less expensive and to avoid the effort of calibrating the stereo rig. This strongly motivates to address the scene flow estimation problem in a monocular camera setup.

1.1.2 Multi-View Geometry Meets Deep Learning

Scene flow estimation and scene reconstruction are traditionally based on the principles of *multi-view geometry* in a monocular camera setup. The displacement of a scene point between two or more images (*optical flow*) depends on the relative motion of the scene point and its distance to the camera.

Multi-view geometry-based approaches for static environments: Methods have been proposed that address the estimation of the depth and camera motion assuming a static environment. Structure from motion (SfM) (see figure 1.3 (b)) describes the principle to triangulate the depth of each scene point based on its optical flow given the camera motion or the simultaneous estimation of depth and camera motion. *Simultaneous localization and mapping* (SLAM) methods (e.g. [Engel et al., 2017, Mur-Artal and Tardós, 2017]) are applied on image sequences and jointly estimate the camera poses and depth structure of the scene including a mapping stage. Even given perfect optical flow estimates, there exists a *scale-ambiguity* between the depth estimates and translational camera motion. It is not possible to distinguish a small camera translation in a miniature world from a large camera translation in the real world based on a monocular image sequence and the multi-view geometric principles. To overcome this ambiguity, additional constraints are introduced such as a known camera height above the ground plane.

Multi-view geometry-based approaches for dynamic environments: The concept of *multi-body structure from motion* (MSfM) is a generalization of single-body SfM to *dynamic scenes* including objects with individual motion. The scene is segmented into parts that undergo the same individual motion (e.g. each segment corresponds to a different vehicle) and each part is reconstructed based on the SfM principle separately. However, the scale ambiguity is present for every reconstruction and even more, the relative scales between the reconstructions are unknown as



(a) Monocular image

(b) Structure from motion

(c) Single-view depth estimation

(d) Semantic segmentation

(e) Instance segmentation

Figure 1.3: Overview of multi-view geometric (structure from motion) and deep learning methods (single-view depth estimation, semantic segmentation, and instance segmentation). The figure shows results using [Godard et al., 2017] for single-view depth estimation, [Long et al., 2015] for semantic segmentation, and [He et al., 2017] for instance segmentation. The SfM-based depth estimation builds on the MirrorFlow [Hur and Roth, 2017] for optical flow estimation. The results are overlaid with the input images and colored in the following scheme: (b,c) the estimated depth is colored from close (red) to far (dark blue), (d) each color corresponds to a different class (e.g. blue for vehicles), and (e) each color corresponds to a different object instance.

well. To solve the ambiguity, previous monocular methods assumed that the moving objects are in contact with the ground, on which they move [Ranftl et al., 2016, Bullinger et al., 2018], or that the scene follows a smoothness prior regarding surface and motion [Mitiche et al., 2015, Xiao et al., 2017, Kumar et al., 2017, Di et al., 2019]. These assumptions might be violated and require a highly accurate detection of the ground contact point and reconstruction of the ground plane. Additionally, there exist degenerate situations where moving objects are not detectable based on multi-view geometry alone. Moving objects with collinear translational motion are one example, which is common in traffic scenes for approaching or preceding traffic participants on the same or adjacent lane.

Figure 1.3 (b) shows for an example that SfM is able to provide a reasonable reconstruction of the depth structure for most parts of the static environment. This reconstruction assumes highly accurate optical flow estimates, which is especially challenging for low textured image regions or repetitive structures. The *aperture problem* is known as the ambiguity of the motion direction of an optical flow vector in the direction of a line [Jähne, 2005, p. 401]. Erroneous optical flow estimates (e.g. on the lane markers in figure 1.3 (b)) directly result in incorrect depth estimates. Furthermore, moving objects are systematically reconstructed at the wrong position, e.g. an oncoming object appears closer to the camera than it actually is and a preceding object appears at a far distance. The assumption that all objects are static is violated in this case. However, the same errors occur in a multi-body structure from motion (MSfM) method if the moving object is not detected due to a degenerated situation. In summary, multi-view geometry provides powerful information for depth and motion estimation, but also with some challenges or limitations: (1) there exists a scale-ambiguity, (2) optical flow estimation is prone to low-textured or repetitive parts of the scene, (3) there exists degenerated situations in terms of moving object detection, and (4) a translational camera motion is required.

In contrast to the multi-view geometry-based approaches, many types of information are inherent in the appearance of the scene in a single image. Figure 1.3 (c-e) provides qualitative results for three different tasks. Today, leading solutions for these tasks are dominated by *deep learning* methods.

Single-view depth estimation: A human is able to recognize the depth structure of a scene by looking at a single image. This ability is especially based on the experience of how a known scene typically looks like, e.g. that objects at a far distance appear smaller in the image and mostly close to the horizon. The breakthrough of *single-view depth estimation* was achieved by applying deep learning methods (e.g. [Eigen et al., 2014, Godard et al., 2017]), which are able to exploit these depth cues in a single image to provide plausible depth estimates (see figure 1.3 (c)). Comparing the results to SfM, this depth estimation shows different characteristics. The depth estimates are less sharp and also less accurate in some parts. However, these methods are less prone to errors for low-textured or repetitive structures and moving objects. The depth estimates are also provided in a metric scale if supervision of a stereo camera or ground truth data is used during training. A translational motion of the camera relative to the scene is not required anymore. While SfM is based on the motion of pixels between different views, only a single image is given to these approaches. Therefore, single-view and multi-view approaches base their estimates on different kinds of information, which results in different advantages and disadvantages. This makes it attractive to combine both for monocular scene flow estimation.

Semantic and instance segmentation: Figure 1.3 (d-e) shows results for *semantic segmentation* and *instance segmentation*. While semantic segmentation (e.g. [Long et al., 2015]) assigns a class label to each pixel, instance segmentation (e.g. [He et al., 2017]) also classifies which pixels belong to the same object. Instance segmentation is applicable to countable objects such as pedestrians or vehicles. However, there is also an attempt to combine both tasks as panoptic segmentation (e.g. [Kirillov et al., 2019, Xiong et al., 2019]). Such information can also be beneficial for scene flow estimation, e.g. to define a more distinctive scene model distinguishing the different classes, or to identify which parts of the image undergo the same rigid body motion.

All these methods provide different powerful information for the task of monocular scene reconstruction. Even more, deep learning methods provide information complementary to the challenges and limitations of multi-view geometry-based approaches. However, they are typically still addressed as individual tasks or combined in a way that is limited to static scenes (e.g. [Tateno et al., 2017, Fácil et al., 2017, Yin et al., 2017]). This strongly motivates to focus the present thesis on the topic of combining multi-view geometry with deep learning approaches to overcome previous limitations and to provide accurate monocular scene flow estimates.

1.2 Organization and Contributions

The present thesis addresses the task of scene flow estimation for multi-object dynamic scenes in a monocular camera setup. The core research question addresses the combination of multi-view geometry with deep learning-based methods for monocular scene flow estimation. A special focus is placed on the combination of multi-view geometry and single-view depth estimation. The present section gives an overview of the organization of the thesis and highlights the contributions of each chapter in general and in particular regarding the research question.

Chapters 1 and 2 set the motivation and provide an introduction to the relevant technical background including an overview of multi-view geometry and optimization methods. Chapter 3 presents *ProbDepthNet*, a CNN for single-view depth estimation including uncertainty quantification of the provided depth estimates. The subsequent chapters 4 to 6 present different approaches for monocular scene flow estimation – each one designed to combine multi-view geometry with the probabilistic single-view depth estimates provided by ProbDepthNet. These methods present also ways to exploit the results of a semantic or instance segmentation. While *SVD-MSfM* (chapter 4) basically follows a MSfM-based formulation, *Mono-SF* (chapter 5) is formulated as a scene flow optimization problem. Finally, the *Mono-Stixel* method (chapter 6) is designed to exploit a scene model specialized for dynamic traffic scenes as an underlying representation.

Chapter 3 - Probabilistic Single-View Depth Estimation: Even though the idea of *single-view depth estimation* is by far not new, the real breakthrough was achieved by using deep learning methods. However, most of the deep learning methods for single-view depth estimation lack *uncertainty quantification* of their estimates – an important characteristic, which is needed to combine multi-view geometry with single-view depth estimates in a statistical manner.

Chapter 3 provides an analysis regarding the error distribution and presents *ProbDepthNet*, a CNN that estimates pixel-wise depth distributions from a single image rather than single depth values that just encode the most likely depth estimate. Different strategies for uncertainty estimation are analyzed and the experiments reveal that these strategies suffer from overfitting effects and provide overconfident uncertainty measures. A novel *recalibration* technique by adding a few subsequent layers, called *CalibNet*, is proposed, which compensate for such overfitting effects resulting in well-calibrated distributions. This recalibration technique is applicable to different probabilistic approaches. The subsequent chapters give evidence that the probabilistic design and recalibration technique of ProbDepthNet are important to provide high accurate monocular scene flow estimates by combining multi-view geometry with single-view depth estimates.

Chapter 4 - Single-View Depth Meets Multi-Body Structure from Motion: In a monocular camera setup, the estimation of the depth and motion of a dynamic scene is traditionally addressed by MSfM-based approaches. In contrast to these multi-view geometry-based approaches, previous works and the ProbDepthNet method presented here provide depth estimates from a single image. However, single-view depth estimation and multi-view geometry are mostly tackled as two individual tasks or fused in a way that is only applicable to static scenes.

Chapter 4 presents a novel method, denoted as *SVD-MSfM*, which combines *probabilistic single-view depth estimation* with *multi-view geometry* in a MSfM-based approach. More precisely, in the first step, scale-aware motion estimation is performed for the camera and potentially moving objects detected by an instance segmentation. While previous methods merely exploited single-view depth estimates for camera pose estimation, SVD-MSfM presents a novel scale-aware motion estimation for moving objects. As a second step, a depth probability volume serves as the basis to estimate the depth structure by combining the probabilistic single-view depth estimates with the multi-view geometry-based photometric consistency. The monocular scene flow estimates provided by SVD-MSfM are confirmed by the experiments to be superior to previous methods. Furthermore, the method generalizes to standstill scenarios and several components and design choices are validated by additional ablation studies. The experiments give evidence for the claimed combination of multi-view geometric optimization with single-view depth estimates. They show that both parts contribute to the final accuracy. This combination benefits from single-view depth estimates provided in a probabilistic and well-calibrated form – the two characteristics for which the ProbDepthNet model is designed.

Chapter 5 - Monocular Instance Scene Flow: SVD-MSfM consists basically of the two separate steps of motion and depth estimation. However, both tasks highly depend on each other, which motivates a joint optimization of both.

Chapter 5 presents *Mono-SF* for joint optimization of the motion and depth structure of the scene. The scene is decomposed into 3D planar surface elements, each one assigned to an object or the background. The objects and background are assumed to be rigid and form a set of rigid bodies. Following this model, Mono-SF jointly estimates the 3D geometry of each plane and 6D motion of each rigid body considering (1) the multi-view geometry by warping the reference image into the subsequent image, (2) probabilistic single-view depth estimates, and (3) scene model smoothness priors. SVD-MSfM is used for the initialization of the non-linear optimization problem. Mono-SF provides a further improvement in terms of monocular scene flow accuracy compared to SVD-MSfM and previous methods. Furthermore, Mono-SF has been the first monocular method published in the KITTI scene flow benchmark. Several components and design choices are analyzed by the experiments. The two essential claims in terms of combining multi-view geometry with single-view depth estimates and the ProbDepthNet design are further confirmed by the experiments.

Chapter 6 - Monocular Stixel Scene Flow: Mono-SF has provided new state of the art (SotA) monocular scene flow estimates by combining multi-view geometry with probabilistic single-view depth estimates. A special representation for traffic scenes, the stixel world representation, was introduced in section 1.1.1. Chapter 6 presents the *Mono-Stixel* method, which addresses monocular scene flow estimation with the stixel world representation as underlying scene model. While SVD-MSfM and Mono-SF distinguish static and potentially moving objects based on an instance segmentation, the differentiation between static and moving objects is part of the Mono-Stixel optimization and an independent moving object (IMO) detection identifies which objects are really in motion.

The experiments confirm that the Mono-Stixel method provides SotA monocular scene flow estimates and IMO detections using a stixel world representation. Due to the higher flexibility of the Mono-Stixel method that the deep-learning inputs are optional, the experiments provide further insights into the different benefits of each deep-learning component for a monocular scene flow estimation.

2

TECHNICAL BACKGROUND

CONTENTS

The present chapter provides a summary of the technical background and used notation and serves to support the understanding of the thesis. The first section 2.1 introduces the *projective geometry* of a single image and the geometric principles of two views. The second section 2.2 presents methods for minimizing *continuous non-linear energy terms* and for inferring *discrete problems in graphical models*. Both sections are tailored to the concepts and methods used in the present thesis and described in a compact introductory manner.

2.1 Single and Two-View Geometry

The purpose of the present section is to explain the concept of projective geometry for single cameras and pairs of cameras and covers (1) the *projection* of a 3D point into the image, (2) the *epipolar geometry* that constraints the image position of a 3D point in multiple views, and (3) the *plane-induced homography* that defines the mapping of a 3D point lying on a scene plane. For a more comprehensive and in-

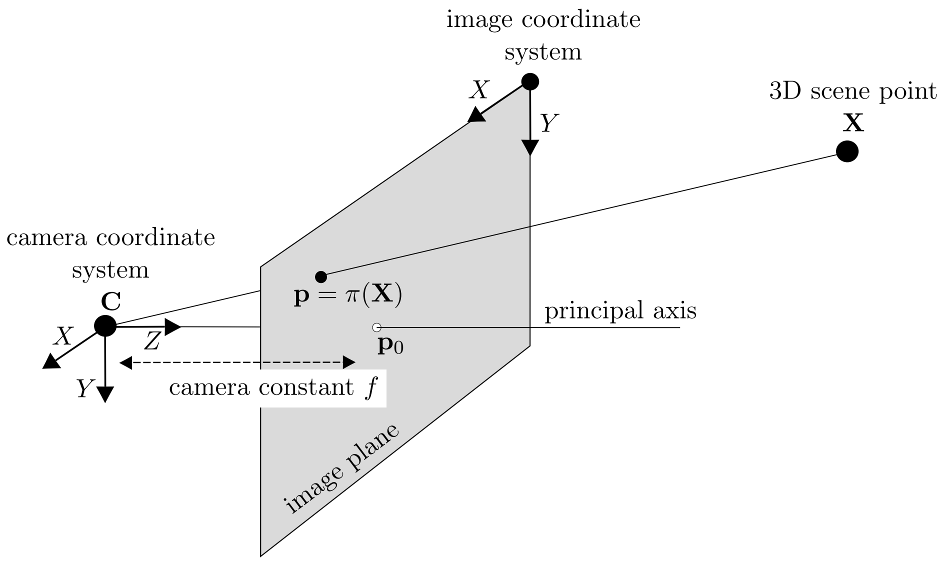


Figure 2.1: Illustration of the pinhole camera model (figure following [Hartley and Zisserman, 2003, p. 154]). The pinhole camera model defines the projection π of a 3D scene point $\mathbf{X}$ into the image $\mathbf{p} = \pi(\mathbf{X})$. The origin of the camera coordinate system is denoted by $\mathbf{C}$. The principle point $\mathbf{p}_0$ is defined by the intersection of the principal axis with the image plane. The distance of the image plane to the camera center is known as the camera constant f.

depth presentation, the reader is referred to relevant literature such as [Hartley and Zisserman, 2003, Part 1, 2].

2.1.1 Pinhole Camera Model

To define the projection of a 3D scene point $\mathbf{X}$ into the 2D image coordinates $\mathbf{p} = \pi(\mathbf{X})$, the *pinhole camera model* is assumed. For an ideal pinhole model distortions due to an optical lens are neglected. For practical use, the camera needs to be properly calibrated and the distortions should be compensated in advance or incorporated into the projection model.

Figure 2.1 illustrates the geometric principles of the pinhole model to derive $\mathbf{p} = \pi(\mathbf{X})$. The projection is expressed in homogenous coordinates. This means that the image coordinates are defined as $\mathbf{p} = (u \;\; v \;\; 1)^T$ in projective space. All points that are equal up to an arbitrary scale $k \in \mathbb{R}, k \neq 0$ form an equivalence class $\mathbf{p} = (ku \;\; kv \;\; k)^T$ and refer to the same non-homogenous point u, v. Most equations in the present thesis employ $\mathbf{p}$ and $\mathbf{X}$ in projective space.

The camera parameters that define the projection are illustrated in figure 2.1. The *camera center* $\mathbf{C}$ is the origin of the camera coordinate system. The z-axis is perpendicular to the image plane, the x-axis points to the right in a horizontal direction, and the y-axis is used as the vertical direction pointing downwards. The distance of the image plane to the origin of the camera coordinate system is denoted as the *camera constant* f. Furthermore, the intersection of the z-axis of the camera coordinates with the image plane defines the *principal point* $\mathbf{p}_0$. The image coordinate system is defined to be at the upper left corner of the image as shown in figure 2.1. For mapping the image coordinates to pixels, the parameters are additionally normalized by the pixel dimension m_x, m_y, which defines $f_x = f/m_x$, $f_y = f/m_y$, $c_x = \mathbf{p}_{0,x}/m_x$, and $c_y = \mathbf{p}_{0,y}/m_y$.

Based on these camera parameters, the projection $\pi(\mathbf{X})$ of a 3D point $\mathbf{X}$ into image coordinates $\mathbf{p}$ is defined by the following equation:

$$\mathbf{p} = \pi(\mathbf{X}) = \underbrace{\begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \mathbf{X} \tag{2.1}$$

The matrix $\mathbf{K}$ is the so-called *camera matrix*. The *skew* parameter s is zero for most cameras [Hartley and Zisserman, 2003, p. 157].

The camera parameters are derived by camera calibration, which is not covered here. The reader is referred to literature such as [Hartley and Zisserman, 2003, pp. 178–193].

2.1.2 Epipolar Geometry

While the previous section 2.1.1 explained a model to describe the projective geometry of a single camera image, multi-view geometry is introduced in the present subsection. As shown in figure 2.2, two images are given with the camera centers $\mathbf{C}_0$ and $\mathbf{C}_1$. The images are taken by two different cameras with individual camera matrices $\mathbf{K}_0$ and $\mathbf{K}_1$ or by a single moving camera $\mathbf{K}_0 = \mathbf{K}_1 = \mathbf{K}$.

Representation of transformations: First, the used representation of transformations is introduced. The transformation between the camera coordinate systems is defined by $\mathbf{T} \in SE(3)$, which transforms a 3D point $\mathbf{X}_0$ from the coordinate system $\mathbf{C}_0$ to $\mathbf{C}_1$:

$$\mathbf{X}_1 = \underbrace{\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{3\times 1} & 1 \end{bmatrix}}_{\mathbf{T}} \mathbf{X}_0 \tag{2.2}$$

The points $\mathbf{X}_0, \mathbf{X}_1$ are given in homogenous coordinates in this equation with the homogenous coordinate normalized to one. The transformation matrix $\mathbf{T}$ comprises

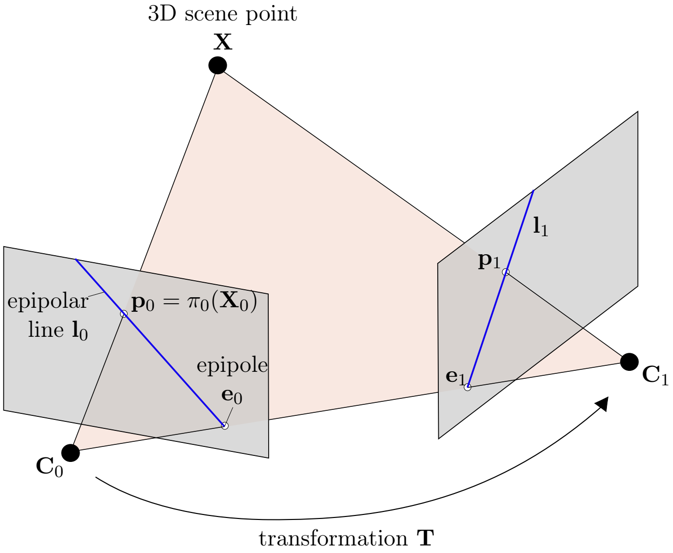


Figure 2.2: Illustration of epipolar geometry (figure following [Hartley and Zisserman, 2003, p. 240]). The epipolar plane (red plane) intersects the image planes in the epipolar lines. Two matching image points $\mathbf{p}_0 = \pi_0(\mathbf{X}_0)$ and $\mathbf{p}_1 = \pi_1(\mathbf{X}_1)$ are constraint to lie on the respective epipolar line. The camera centers $\mathbf{C}_0$ and $\mathbf{C}_1$ projected into the other image defines the epipoles $\mathbf{e}_0$ and $\mathbf{e}_1$. All epipolar lines intersect the epipole.

the rotation matrix $\mathbf{R} \in SO(3)$ and translation vector $\mathbf{t} \in \mathbb{R}^3$ as shown in equation (2.2).

Transformations are mostly expressed by the introduced matrix multiplication in the present thesis. Alternative representations of the rotation are quaternions, Euler angles, or Lie algebra elements. The optimization presented in section 4.2.1 is based on the Lie algebra elements because this is a minimal representation of the 6 degrees of freedom. Since a deeper explanation of Lie algebra is not necessary to understand the present thesis in general, the reader is referred to the literature. For example, [Engel, 2017, pp. 25–30] provides a compact overview of different parameterizations for representing and optimizing transformations.

Essential and fundamental matrix: Considering a 3D point $\mathbf{X}$ in the scene, it can be transformed into the corresponding camera coordinates $\mathbf{X}_0$ or $\mathbf{X}_1$ and projected into the image coordinates by $\mathbf{p}_0 = \pi_0(\mathbf{X}_0)$ or $\mathbf{p}_1 = \pi_1(\mathbf{X}_1)$. Even if the exact 3D position of the point is not given, the corresponding image positions $\mathbf{p}_0$ and $\mathbf{p}_1$ are constrained by the *epipolar geometry*. The projections of the camera centers $\mathbf{C}_0$ and $\mathbf{C}_1$ in the other image are known as the *epipoles* $\mathbf{e}_0, \mathbf{e}_1$. The plane that is spanned by the rays through the image position $\mathbf{p}_i$ and the epipole $\mathbf{e}_i$ is the *epipolar plane* (see plane in figure 2.2). The intersection of this plane with the images defines the *epipolar lines* $\mathbf{l}_i$ and illustrates the fact that both corresponding image positions $\mathbf{p}_0$ and $\mathbf{p}_1$ need to lie on its corresponding epipolar line (see epipolar lines in figure 2.2). The method presented in section 4.2.1 exploits the epipolar geomerty to find corresponding image features in the subsequent image. The epipolar constraint is used to reduce the search space of corresponding features and to prefer matches close to the epipolar line.

Formally, the epipolar constraint is expressed by the following equation, which depends on the transformation and camera parameters.

$$\begin{aligned} \mathbf{p}_1^T \mathbf{K}_1^{-T} \underbrace{[\mathbf{t}]_\times \mathbf{R}}_{\mathbf{E}} \mathbf{K}_0^{-1} \mathbf{p}_0 &= 0 \\ \mathbf{p}_1^T \underbrace{\mathbf{K}_1^{-T} \mathbf{E} \mathbf{K}_0^{-1}}_{\mathbf{F}} \mathbf{p}_0 &= 0 \\ \mathbf{p}_1^T \mathbf{F} \mathbf{p}_0 &= 0 \end{aligned} \tag{2.3}$$

The matrix $\mathbf{E}$ is the so-called *essential matrix* defined as the cross product matrix $[\mathbf{t}]_\times$ multiplied with $\mathbf{R}$. The *fundamental matrix* $\mathbf{F}$ integrates additionally the camera matrices $\mathbf{K}_0$ and $\mathbf{K}_1$.

2.1.3 Plane-Induced Homography

The epipolar geometry defines the relation of two image positions for one 3D scene point. The present subsection covers the projection of a scene plane between two images, which is defined by a *plane-induced homography* (see figure 2.3).

Plane geometry: A scene plane is defined by its normal vector $\mathbf{n}$ and its distance d to the camera such that each 3D point $\mathbf{X}$ on the plane fulfills the equation $\mathbf{n}^T\mathbf{X} + d = 0$. An equivalent representation $\bar{\mathbf{n}}^T\mathbf{X} = 1$ is based on the scaled normal vector $\bar{\mathbf{n}}$, which is defined as the normal vector $\mathbf{n}$ divided by $-d$. The three degrees of freedom of the scaled normal vector are a minimal representation to define a scene plane. The 3D scene point $\mathbf{X}$ corresponding to an image point $\mathbf{p}$ lying on the scene plane $(\mathbf{n}, d)$ is defined by the intersection of the ray of $\mathbf{p}$ with the scene plane, which is expressed by the following equation:

$$\mathbf{X} = -d \frac{\mathbf{K}^{-1}\mathbf{p}}{\mathbf{n}^T\mathbf{K}^{-1}\mathbf{p}} \tag{2.4}$$

Homography: In the context of the two-view geometry, a homography defines the projection of a scene plane (or a point on the plane) from one image to another image (see figure 2.3). Formally, the relation of two corresponding image points $\mathbf{p}_0$ and $\mathbf{p}_1$ lying on the scene plane defined by $(\mathbf{n}, d)$ is expressed by the homography $\mathbf{H}$:

$$\mathbf{p}_1 = \underbrace{\mathbf{K}_1 \left(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^T}{d} \right) \mathbf{K}_0^{-1}}_{\mathbf{H}} \mathbf{p}_0 \tag{2.5}$$

The homography $\mathbf{H}$ is a 3×3 matrix that depends on the transformation between the camera coordinates systems $(\mathbf{R}, \mathbf{t})$, the scene plane $(\mathbf{n}, d)$, and the camera matrices $(\mathbf{K}_0, \mathbf{K}_1)$.

Due to the arbitrary scale using homogenous coordinates, the homography has 8 degrees of freedom. A short proof for deriving the homography equation in equation (2.5) is given as follows with the arbitrary scale s made explicit:

$$\begin{aligned} && s\mathbf{p}_1 &= \mathbf{K}_1 \left(\mathbf{R}\mathbf{X}_0 + \mathbf{t} \right) \\ \Longleftrightarrow && s\mathbf{p}_1 &= \mathbf{K}_1 \left(\mathbf{R} \frac{-d\mathbf{K}_0^{-1}\mathbf{p}_0}{\mathbf{n}^T\mathbf{K}_0^{-1}\mathbf{p}_0} + \mathbf{t} \right) \\ \Longleftrightarrow && \underbrace{\frac{s\mathbf{n}^T\mathbf{K}_0^{-1}\mathbf{p}_0}{-d}}_{s'} \mathbf{p}_1 &= \mathbf{K}_1 \left(\mathbf{R}\mathbf{K}_0^{-1}\mathbf{p}_0 - \frac{\mathbf{t}\mathbf{n}^T\mathbf{K}_0^{-1}\mathbf{p}_0}{d} \right) \\ \Longleftrightarrow && s'\mathbf{p}_1 &= \underbrace{\mathbf{K}_1 \left(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^T}{d} \right) \mathbf{K}_0^{-1}}_{\mathbf{H}} \mathbf{p}_0 \end{aligned} \tag{2.6}$$

The second step substitutes $\mathbf{X}_0$ with equation (2.4) and the third step exploits that a homography is only defined up to an unknown scale.

The described plane-induced homography is a core element to represent the optical flow of planar surface elements in chapters 5 and 6. Even though the epipolar geometry and plane-induced homography are illustrated for a moving camera and static scene point or plane, these principles are not limited to static scenes. For moving points or planes, the transformation $\mathbf{T}$ needs to be replaced by the relative motion of the camera to the scene element, which makes both applicable also to dynamic scenes.

2.2 Optimization Methods

The previous section 2.1 described the projective geometry for single and multiple views. Methods are proposed, also in the present thesis, to estimate e.g. the scene structure, motion, or scene flow based on the geometric relations. Such methods typically are formulated as *minimization problems* of a *loss* function, also denoted

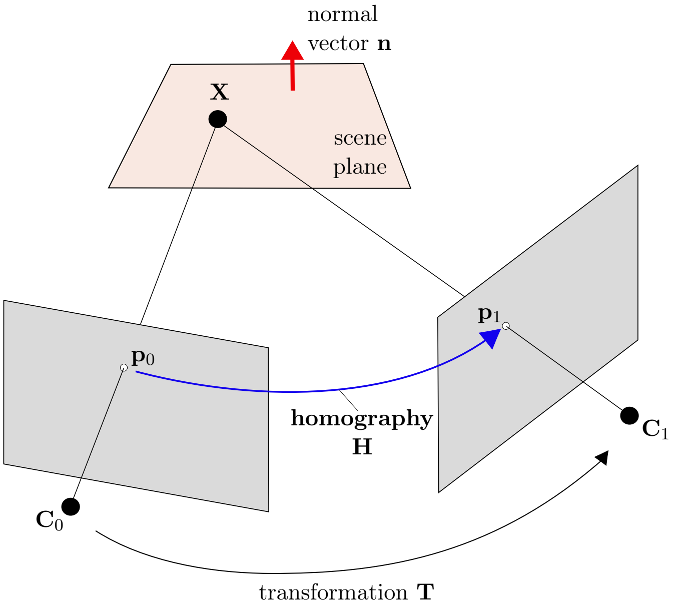


Figure 2.3: Illustration of plane-induced homography (figure following [Hartley and Zisserman, 2003, p. 243]). The projection of scene points lying on a scene plane between two images is defined by a plane-induced homography $\mathbf{H}$. The scene plane is defined by its normal vector $\mathbf{n}$ and its distance to the camera d. The homography additionally depends on the transformation $\mathbf{T}$ between the camera coordinate systems $\mathbf{C}_0$, $\mathbf{C}_1$ and the camera matrices $\mathbf{K}_0$, $\mathbf{K}_1$.

as *energy minimization*. Even though the optimization of these energy terms itself is not the main focus of the thesis, the optimization is an essential component to derive accurate estimates. The present section is focused on the optimization methods applied in the thesis. The first section 2.2.1 is related to the optimization of non-linear continuous problems minimizing the squared residuals. The second section 2.2.2 covers discrete problems, which are formulated in a graph-based representation. A broader and more detailed representation can be found in the relevant literature such as [Bishop, 2006, pp. 140–146, 207–210, 383–418, 610–631].

2.2.1 Non-linear Least-Squares Optimization

The first kind of problem considered here is optimizing the variable $\mathbf{x}$ such that it minimizes a non-linear continuous energy function $E(\mathbf{x})$, for the special case that $E(\mathbf{x})$ can be expressed by a *sum of squared residuals* $r_i^2(\mathbf{x})$:

$$E(\mathbf{x}) = \sum_{i=1}^{n} r_i^2(\mathbf{x}) \tag{2.7}$$

An equivalent representation of the energy term $E(\mathbf{x})$ is defined by introducing the vector $\mathbf{r}(\mathbf{x}) = [r_1(\mathbf{x}), r_2(\mathbf{x}), ..., r_n(\mathbf{x})]^T$:

$$E(\mathbf{x}) = \mathbf{r}(\mathbf{x})^T \cdot \mathbf{r}(\mathbf{x}) \tag{2.8}$$

For example, the negative log-likelihood of statistical independent Gaussian measurements result in the given form neglecting some constant factors. The following two approaches cover methods to optimize $\mathbf{x}$ such that the energy term E is minimized.

Gradient descent: The first approach presented here is the *gradient descent* method. This method is an iterative optimization in the direction of the strongest descent defined by the gradient of the energy term. Each update step is defined as applying an increment δ to the current solution $\mathbf{x}_0$:

$$\mathbf{x} \leftarrow \mathbf{x}_0 + \delta \tag{2.9}$$

The gradient of the energy term depends on the partial derivation of the residuals $\mathbf{r}(\mathbf{x})$ by $\mathbf{x}$ evaluated at $\mathbf{x}_0$, which is defined as the Jacobi-matrix $\mathbf{J}_r(\mathbf{x}_0)$:

$$\mathbf{J}_r(\mathbf{x}_0) = \left.\frac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}_0} \tag{2.10}$$

The increment δ is defined as the strongest descent at $\mathbf{x}_0$, which is defined by the negative gradient of the energy term. An additional *dampening parameter* λ controls the gradient descent stepwidth.

$$\begin{aligned} \left.\frac{\partial E(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}_0} &= 2\mathbf{J}_r^T(\mathbf{x}_0) \cdot \mathbf{r}(\mathbf{x}_0) \\ \iff \delta &= -\lambda \mathbf{J}_r^T(\mathbf{x}_0) \cdot \mathbf{r}(\mathbf{x}_0) \end{aligned} \tag{2.11}$$

Gauss-Newton optimization: In contrast to the gradient descent method, the *Gauss-Newton* iteratively applies increments, which are derived by finding the minimum of the energy term for linearized residuals. The first-order Tailor approximation of the residuals $\mathbf{r}_i(\mathbf{x}_0 + \delta)$ at $\mathbf{x}_0$ is introduced and defined by

$$\mathbf{r}_i^T(\mathbf{x}_0 + \delta) \approx \mathbf{r}(\mathbf{x}_0) + \mathbf{J}_r \delta. \tag{2.12}$$

Replacing the residual with its first-order Tailor approximation, results in an approximated energy term $E(\mathbf{x}_0 + \delta)$ of the following form:

$$\begin{aligned} E(\mathbf{x}_0 + \delta) &= \mathbf{r}_i^T(\mathbf{x}_0 + \delta) \cdot \mathbf{r}_i(\mathbf{x}_0 + \delta) \\ &\approx \left(\mathbf{r}(\mathbf{x}_0) + \mathbf{J}_r\delta\right)^T \left(\mathbf{r}(\mathbf{x}_0) + \mathbf{J}_r\delta\right) \\ &= \mathbf{r}(\mathbf{x}_0)^T \cdot \mathbf{r}(\mathbf{x}_0) + 2\delta^T\mathbf{J}^T \cdot \mathbf{r}(\mathbf{x}_0) + \delta^T\mathbf{J}^T\mathbf{J}\delta \end{aligned} \tag{2.13}$$

The minimum of the approximated energy function is derived by setting the first derivative of the energy function by δ to zero, which defines the increment δ and update rule:

$$\begin{aligned} \frac{\partial E(\mathbf{x}_0 + \delta)}{\partial \delta} &= 2\mathbf{J}_\mathbf{r}^T\mathbf{J}_\mathbf{r}\delta + 2\mathbf{J}_\mathbf{r}^T \cdot \mathbf{r}(\mathbf{x}_0) = 0 \\ \iff \delta &= -\left(\mathbf{J}_\mathbf{r}^T\mathbf{J}_\mathbf{r}\right)^{-1}\mathbf{J}_\mathbf{r}^T \cdot \mathbf{r}(\mathbf{x}_0) \end{aligned} \tag{2.14}$$

Levenberg-Marquardt optimization: *Levenberg-Marquardt* is a combination of the *Gauss-Newton* algorithm with *gradient descent*. The addition of Gauss-Newton (equation (2.14)) and gradient descent increment (equation (2.11)) defines the combined increment δ of Levenberg-Marquardt as:

$$\delta = -(\mathbf{J}_\mathbf{r}^T\mathbf{J}_\mathbf{r} + \lambda \cdot \mathbf{I})^{-1}\mathbf{J}_\mathbf{r}^T \cdot \mathbf{r}(\mathbf{x}_0) \tag{2.15}$$

The dimension of the identity matrix $\mathbf{I}$ corresponds to the dimension of $\mathbf{x}_0$.

An alternative common choice instead of the identity matrix is $diag(\mathbf{J}_\mathbf{r}^T\mathbf{J}_\mathbf{r})$ to adjust each component of the gradient descent to the gurvature and increase stepwidth for smaller gradients. Introducing the variables $\mathbf{H} = \mathbf{J}_\mathbf{r}^T\mathbf{J}_\mathbf{r}$ and $\mathbf{b} = \mathbf{J}_\mathbf{r}^T \cdot \mathbf{r}(\mathbf{x}_0)$, the alternative increment δ is defined as follows:

$$\delta = -(\mathbf{H} + \lambda \cdot diag(\mathbf{H}))^{-1}\mathbf{b} \tag{2.16}$$

The factor λ can be defined adaptively to change the weighting of the Gauss-Newton part compared to the gradient descent. Typically, λ is decreased if the energy term decreases $E(\mathbf{x}_0 + \delta) < E(\mathbf{x}_0)$ and increased otherwise.

The Levenberg-Marquardt algorithm is also applicable to *weighted least-squares problems* of the following form:

$$E(\mathbf{x}) = \mathbf{r}(\mathbf{x})^T \cdot \mathbf{W} \cdot \mathbf{r}(\mathbf{x}) \tag{2.17}$$

The update rule follows equation (2.16) by defining $\mathbf{H} = \mathbf{J}_\mathbf{r}^T\mathbf{W}\mathbf{J}_\mathbf{r}$ and $\mathbf{b} = \mathbf{J}_\mathbf{r}^T\mathbf{W}\cdot\mathbf{r}(\mathbf{x}_0)$.

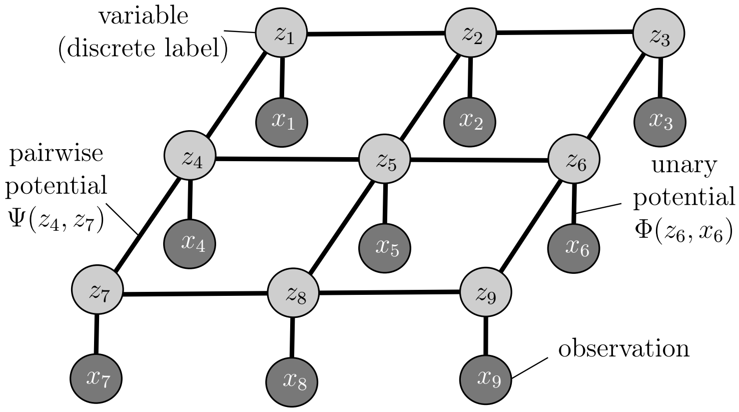


Figure 2.4: Example of a graph representation of a conditional random field (CRF). Each node represents a variable z_i (light grey) or an observation x_i (dark grey). The edges model the unary $\Psi(z_i, x_i)$ and pairwise potentials $\Phi(z_i, z_j)$.

The Levenberg-Marquardt algorithm is applied in section 4.2.1 to estimate the transformation of rigid bodies between two images using the implementation of Kümmerle et al. [2011].

2.2.2 Discrete Inference on Graphical Models

While the previous section 2.2.1 deals with the optimization of non-linear continuous functions, the present subsection is placed on discrete energy minimization and labeling problems. Two kinds of problems are presented here, both are essential for the methods proposed in the present thesis. First, the optimization of a discrete labeling problem is described with a *conditional random field* (CRF) as the underlying representation of the energy minimization problem. Second, inferring a sequence of hidden states of a *hidden Markov model* (HMM) via the *Viterbi algorithm* is presented.

Conditional random field: The first kind of problem considered here is the optimization of discrete variables $\mathbf{Z}$ based on a set of observations $\mathbf{X}$. The most likely solution of $\mathbf{Z}$ is defined by the maximum of the posterior conditional probability $p(\mathbf{Z} \mid \mathbf{X})$. A CRF corresponds to the category of discriminative random fields and serves as a graphical representation to model the posterior probability $p(\mathbf{Z} \mid \mathbf{X})$. Figure 2.4 illustrates the graphical representation as a CRF. The vertices $\mathcal{V}$ correspond to the discrete variables $z_i \in \mathbf{Z}$ and observations $x_i \in \mathbf{X}$. The shown graph is a *pairwise Markov random field* for the case that the probability fulfills the Markov property, which says that each node is independent of any other node given all its neighbors.

Switching to the logarithmic space, the optimization is converted to an energy minimization E. Each data term is represented by an edge in the graphical representation. One part, the unary potentials $\Phi_i(z_i, x_i)$, captures the likelihood of a certain label z_i based on the corresponding observations x_i. A second part, the pairwise potential $\Psi_{i,j}(z_i, z_j)$, is related to the joint probability of neighboring variables $(i, j) \in \mathcal{N}$, e.g. a regularization or smoothness term:

$$E(\mathbf{Z}, \mathbf{X}) = \sum_{i \in \mathcal{V} \setminus \mathbf{X}} \Phi_i(z_i, x_i) + \sum_{(i,j) \in \mathcal{N}} \Psi_{i,j}(z_i, z_j) \tag{2.18}$$

In computer vision tasks, the nodes of the variables often correspond to a pixel or superpixel, while the label represents a certain attribute such as the semantic class. The observations might be directly the image intensities or intermediate representations such as classifier responses or optical flow estimates.

Belief propagation: Based on the graphical representation of the energy term using a CRF, *belief propagation* or *message passing* are optimization methods to find a solution of $\mathbf{Z}$ with a high probability. The main idea is that the nodes iteratively exchange the probability or energy terms as messages. For the max-product algorithm, the belief of a certain label for a given node is defined as the product of the incoming messages from its neighboring nodes. The messages are defined as probabilities and the belief needs to be maximized. Alternatively, the message could be defined as cost or energy terms. Thereby, the belief is analogously defined as the sum of the incoming messages, which needs to be minimized. This version is known as the min-sum variant, which is the basis for the following description. Formally, the belief b for a certain label z_i of node i is defined by the incoming messages $m_{j \to i}$ of all neighboring nodes $j \in \mathcal{N}_i$ and the unary potential $\Phi(z_i, x_i)$:

$$b_i(z_i) = \Phi(z_i, x_i) + \sum_{j \in \mathcal{N}_i} m_{j \to i}(z_i) \tag{2.19}$$

The solution is defined as the label, which minimizes the belief function:

$$\hat{z}_i = \arg\min_{z_i} b_i(z_i) \tag{2.20}$$

The messages are recursively defined and iteratively updated:

$$\begin{aligned} m^t_{j \to i}(z_i) &= \min_{z_j} \left(\Psi(z_i, z_j) + \Phi(z_j, x_j) + \sum_{(j,k) \in \mathcal{N} \setminus i} m^{t-1}_{k \to j}(z_j) \right) \\ &= \min_{z_j} \left(\Psi(z_i, z_j) + b^{t-1}_j(z_j) - m^{t-1}_{i \to j} \right) \end{aligned} \tag{2.21}$$

The equation shows that the messages integrate the pairwise potentials $\Psi(z_i, z_j)$ taking into account the belief that the neighboring node is of label z_j. The messages need to be evaluated for all combinations of labels of the neighboring nodes, which results in a quadratic complexity in the number of labels. It has been shown that belief propagation provides the optimal solution for *tree-structured graphs* [Bishop, 2006], but is not guaranteed to converge for graphs including cycles.

Tree-reweighted message passing: An extension based on belief propagation designed for graphs including cycles has been proposed by Wainwright et al. [2005] and is called *tree-reweighted message passing*. The main idea is to divide the graph including cycles in a set of trees to exploit the characteristic of belief propagation to find the optimal solution for each of the trees. Each tree is defined as a spanning tree, which means that it covers all nodes. Furthermore, each edge is part of at least one tree. The optimal solution of each tree is derived by applying belief propagation. If the solutions of all nodes are consistent for all trees, the optimization has converged. Until this is the case, the belief of each node is averaged over the trees and the solution of each individual tree is recalculated with the updated belief values. While the original algorithm updates all messages after one iteration in parallel, Kolmogorov [2006] proposed to update the messages in a stepwise manner, which is denoted as *sequential tree-reweighted message passing*.

The scene flow optimization presented in chapter 5 applies sequential tree-reweighted message passing to optimize the depth of planar surface elements and the motion of rigid-bodies using the implementation of Kolmogorov [2006]. Even though the depth and motion are continuous variables, the discrete optimization could be applied by creating discrete samples as described in section 5.2.3.

Hidden Markov model: A hidden Markov model (HMM) represents the probability of a sequence of variables. In the definition of an HMM, the variables are not directly observable and defined as a sequence of hidden states $\mathbf{Z} = [z_1, z_2, ..., z_T]$. Additionally, observable events are introduced and defined as a sequence of observations $\mathbf{X} = [x_1, x_2, ..., x_T]$. To give an illustrative example, HMMs are often used in the domain of speech recognition. While the sequence of audio data samples represents the observations, the hidden states are for example the spoken words. The probability of a sequence of spoken words for the observed audio stream is modeled by the HMM.

How to model the probabilistic relations is described in more detail in the following paragraph. The HMM embodies the Markov assumption that defines that the transition probability only depends on the previous state:

$$p(z_t \mid z_1, z_2, ..., z_{t-1}) = p(z_t \mid z_{t-1}) \tag{2.22}$$

The emission probability only depends on the state at the same position:

$$p(x_t \mid z_1, z_2, ..., z_{t-1}) = p(x_t \mid z_t) \tag{2.23}$$

The initial probability for the first state is defined by $p(z_1)$. Based on these introduced variables and assumptions, the probabilities of the sequence are defined as follows:

$$\begin{aligned} p(\mathbf{Z}) &= \prod_{t=1}^{T} p(z_t \mid z_{t-1}) \text{ with } p(z_1 \mid z_0) = p(z_1) \\ p(\mathbf{X}, \mathbf{Z}) &= \prod_{t=1}^{T} p(z_t \mid z_{t-1}) \cdot \prod_{t=1}^{T} p(x_t \mid z_t) \\ p(\mathbf{X} \mid \mathbf{Z}) &= \prod_{t=1}^{T} p(x_t \mid z_t) \\ p(\mathbf{X}) &= \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}) \end{aligned} \tag{2.24}$$

An analogous definition could be defined in the logarithmic space with the negative log-likelihoods $\Psi(z_t, z_{t-1}) = -\log p(z_t \mid z_{t-1})$, $\Psi(z_1) = -\log p(z_1)$, and $\Phi(x_t, z_t) = -\log p(x_t \mid z_t)$.

Viterbi algorithm: Decoding of an HMM is known as the task of finding the optimal sequence of hidden states $\hat{\mathbf{Z}}$ given a sequence of observations $\mathbf{X}$. The optimal sequence of hidden states is defined by maximizing the posterior probability:

$$\begin{aligned} p(\mathbf{Z} \mid \mathbf{X}) &= \frac{p(\mathbf{X} \mid \mathbf{Z}) \cdot p(\mathbf{Z})}{p(\mathbf{X})} \\ &= \frac{p(\mathbf{X}, \mathbf{Z})}{p(\mathbf{X})} \end{aligned} \tag{2.25}$$

The probability of the observations $p(\mathbf{X})$ remains constant during optimization, which leads to the following optimization objective expressed with negative log-likelihoods:

$$\hat{\mathbf{Z}} = \arg\min_{\mathbf{Z}} \sum_{t=1}^{T} \left(\Phi(x_t, z_t) + \Psi(z_t, z_{t-1})\right) \tag{2.26}$$

The *Viterbi algorithm* is an approach to find the optimal solution w.r.t the defined HMM. The trellis diagram in figure 2.5 represents all possible sequences as paths from the source to the sink. The sum over the weights, which are defined as $\Phi(x_t, z_t) + \Psi(z_t, z_{t-1})$, represents the joint likelihood of the respective path. Therefore, finding the shortest path is an equivalent representation of finding the optimal sequence of states. The forward-path probabilities $v(z_t = i)$ represent the probability

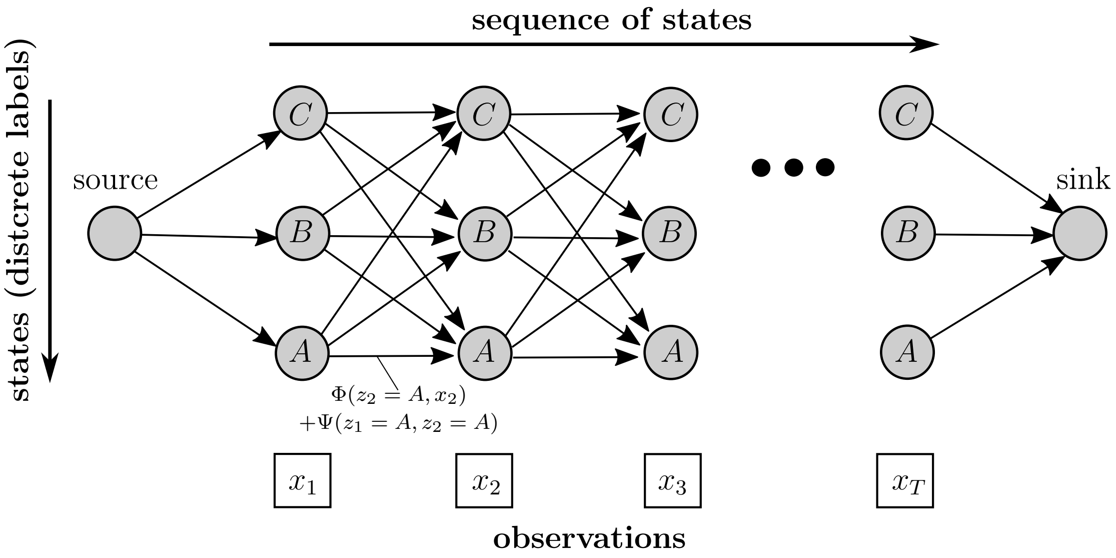


Figure 2.5: Trellis graph representing a sequence of states of a hidden Markov model (HMM). Each node represents a certain state at a certain position in the sequence. The horizontal position represents the position in the sequence and the vertical position the discrete label. The directed edges correspond to a transition between two adjacent states. The assigned weight rates the likelihood in terms of the transition $\Psi(z_{i-1}, z_i)$ and emission likelihood $\Phi(z_i, x_i)$. A path from the source to the sink models one possible sequence of hidden states. The minimum path related to the sum over the weights encodes the most likely sequence of hidden states.

of being in state $z_t = i$ at position t based on previous observations. Exploiting the structure of the trellis diagram, $v(z_t = i)$ is defined in a recursive form:

$$v_t(z_t = i) = \Phi(x_t, z_t = i) + \min_j \left(v_{t-1}(z_{t-1} = j) + \Psi(z_t = i, z_{t-1} = j)\right) \tag{2.27}$$

Referring to the trellis diagram in figure 2.5, the Viterbi algorithm determines the values $v_t(z_t = i)$ for each node from the left to the right beginning at the source node. Instead of evaluating all possible paths, only the likelihoods to reach the previous nodes needs to be taken into account. Consequently, the Viterbi algorithm is a kind of dynamic programming [Bishop, 2006, pp. 411–415]. The minimal cost of a node at the end of the sequence $v_T(z_T)$ defines the cost of the optimal sequence and the final state z_T. Additionally, for each node, its predecessor is stored, which allows backtracing the path to get the optimal sequence of states $\hat{\mathbf{Z}}$. The complexity of this algorithm is $\mathcal{O}\left(N^2 T\right)$ with N being the number of hidden states and T the length of the sequence.

An extension of the Viterbi algorithm to *hidden semi-Markov models* models explicitly to stay in a state for a certain duration. This introduces additional edges in the trellis diagram, which skip nodes while remaining in the same state. Due to the additional edges, the computational effort increases to $\mathcal{O}\left(N^2T^2\right)$.

The Viterbi algorithm is applied in chapter 6 for a column-wise segmentation of the image. The column is considered as a sequence, whereby the discrete labels of the hidden states correspond to the type of the underlying segment. The optimal segmentation and object types are inferred based on observations such as optical flow or single-view depth estimates.

3 PROBABILISTIC SINGLE-VIEW DEPTH ESTIMATION

CONTENTS

This chapter extends parts of the works that have been published previously in [Brickwedde et al., 2018a, 2019].

To derive the depth structure of a scene from a moving monocular camera, methods are traditionally based on the structure from motion (SfM) estimation. The depth of a pixel is estimated based on its optical flow by triangulation given a known camera motion. This concept was also extended to multi-body structure from motion (MSfM) to handle dynamic scenes.

In contrast to the multi-view geometry-based approaches, deep learning-based methods have been proposed (e.g. [Eigen et al., 2014, Godard et al., 2017, Fu et al., 2018]) that provide depth estimates from a single image at a reasonable level of quality. These methods do not require a relative translational motion of the camera

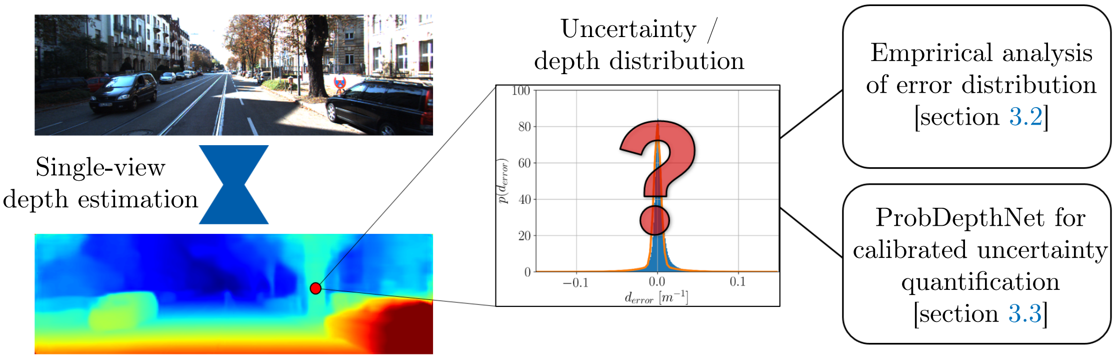


Figure 3.1: Overview of the present chapter regarding probabilistic single-view depth estimation. While most of the previous methods provide the most likelihood depth estimates, this chapter addresses to analyze and quantify the uncertainty or depth distribution of the single-view depth estimates. The first section provides an empirical analysis, which also reveals some dependencies of the error. The second section presents the ProbDepthNet model, a convolutional neural network (CNN) that provides well-calibrated depth distributions.

to the scene and do not suffer from a scale ambiguity using stereo or ground truth supervision.

However, only a few methods represent the *uncertainty* in addition to the raw depth estimates (e.g. [Kendall and Gal, 2017, Xia et al., 2020, Liu et al., 2019]). An uncertainty measure is important for safety-critical applications and beneficial for a probabilistic fusion with other modalities such as the SfM-based depth estimates. Therefore, the main focus of the present chapter is to analyze and quantify the uncertainty or *depth distributions* of the *single-view depth estimates* (see figure 3.1).

In the first section 3.2, the empirical error distribution of single-view depth estimates is analyzed to get a deeper understanding of the uncertainties. The experiments show that the error distribution of the method by Godard et al. [2017] mainly follows a mixture of a Gaussian and Laplace distribution. Additionally, high dependence of the error on the semantic classes such as road, vehicle or building is identified.

Even though the empirical error distribution could be used as a measurement model of the single-view depth estimates as shown in [Brickwedde et al., 2018a], a more distinctive pixel-wise uncertainty measure is beneficial. Therefore, I propose a convolutional neural network (CNN) that estimates pixel-wise depth distributions instead of single depth values that only encode the most likely depth estimate. The network is called *ProbDepthNet* and presented in section 3.3. While the problem of overconfident uncertainty measures is a well-known problem in classification [Guo et al., 2017], it is widely ignored in probabilistic approaches for regression such as [Ilg et al., 2018, Gast and Roth, 2018, Kendall and Gal, 2017, Klodt and Vedaldi, 2018]. Therefore, I propose a novel *recalibration technique*: The final layers of the network, which are denoted as *CalibNet*, are trained on a separate split of the training data to compensate for overfitting effects and to provide well-calibrated distributions. The experiments show that several previous probabilistic approaches suffer from overconfident estimates – an effect that can be compensated by adding the proposed CalibNet for recalibration. The suitability of ProbDepthNet for combining single-view depth information with multi-view geometry for monocular scene flow estimation is confirmed in the following chapters 4 to 6 – especially due to the characteristic of providing single-view depth information in a probabilistic and well-calibrated form.

3.1 Related Work

The works related to probabilistic single-view depth estimation are divided into two categories. First, related works in terms of *single-view depth estimation* are summarized. It includes an overview of the depth cues inherent in a single image, a short discussion of conventional approaches and a more detailed presentation of deep learning-based methods. Second, methods for *probabilistic deep learning* are explained, which are related to the probabilistic design of ProbDepthNet.

3.1.1 Single-View Depth Estimation

In general, depth estimation from a single view is an ill-posed problem. However, humans are still able to perceive the distances using only one eye or by looking at a photo. This ability is based on the experience of how objects and scenes typically look like. The first section 3.1.1.1 gives an overview of these single-view depth cues. The second section 3.1.1.2 and third section 3.1.1.3 cover conventional and deep learning-based approaches for single-view depth estimation.

3.1.1.1 *Single-View Depth Cues*

I propose a categorization of the *single-view depth cues* into three types:

Local and object-wise information: The following *local and object-wise depth cues* are considered as the first group: The concept of *shape from shading* [Horn and Brooks, 1986, Ruo Zhang et al., 1999] takes into account that the image intensity level depends on the light source direction and surface normal. *Shape from texture* [Malik and Rosenholtz, 1997] exploits the fact that the same texture in the scene has a different appearance in the image depending on its distance and orientation.

Ooi et al. [2001] analyzed that humans judge the distance of an object based on the *angular declination* to the eye level. This is highly related to the characteristic in computer vision that objects at far distances are typically closer to the horizon [Konrad et al., 2013, Dijk and Croon, 2019]. Furthermore, if the object dimensions in 3D and the perspective view are known, the distance can be directly estimated based on the size of the object in the 2D image [Cutting and Vishton, 1995, Dijk and Croon, 2019]. Saxena et al. [2009] stated the assumption that long straight 2D lines (e.g. wall or street boundaries) in the image typically correspond to straight 3D lines in the scene. Additionally, the semantic class of an image patch also provides a single-view depth cue, e.g. the sky is at infinite distance and the ground plane is typically horizontal.

Contextual information: The second group comprises *contextual information* between different objects or parts of the scene. First, objects that are closer occlude those that are behind them [Cutting and Vishton, 1995]. Second, the ground contact point of objects defines the distance assuming a known ground surface [Zhongfei Zhang et al., 1997, Hoiem et al., 2005]. Third, small local surface elements in the image (e.g. represented by superpixels) are mostly connected and coplanar in the 3D scene [Saxena et al., 2009].

Global scene model assumptions: Assumptions regarding the *global scene model* are considered as a third group: The Manhattan world assumption [Coughlan and Yuille, 1999] defines that all planes in the image are aligned with the x, y, or z-axis of a global world coordinate system. This means that all planes are coplanar or orthogonal to each other. Regarding the ground plane, the flat world assumption [Zhongfei Zhang et al., 1997] defines that there exists one flat ground plane, which could be either estimated or given if the relative camera height and orientation are known.

3.1.1.2 *Conventional Approaches for Single-View Depth Estimation*

Based on the described single-view depth cues, approaches have been presented to reconstruct the depth structure of a scene from a single image. The present subsection is devoted to approaches without deep learning techniques. For indoor scene reconstruction, a Manhatten world is assumed [Gupta et al., 2010, Hedau et al., 2010, Schwing and Urtasun, 2012]. The walls of a room are detected based on the lines in the image and objects are typically represented as boxes inside the room.

Karsch et al. [2012], Karsch et al. [2014], Konrad et al. [2013] directly formulated the single-view depth estimation as a template matching problem exploiting a database of images with known depth structure. The first machine learning-based approaches are formulated as depth classifiers of local image patches [Ladicky et al., 2014, Saxena et al., 2009, Liu et al., 2014, Zhuo et al., 2015, Liu et al., 2015]. These classifiers are combined based on a conditional random field (CRF), where the classifier responses serve as the unary data terms. The pairwise terms additionally prefer that superpixels are connected and coplanar [Liu et al., 2014, Zhuo et al., 2015, Liu et al., 2015, Saxena et al., 2009], model occlusions [Liu et al., 2014, Zhuo et al., 2015], or favor collinearity of lines in the image [Saxena et al., 2009].

3.1.1.3 *Deep learning-based Approaches for Single-View Depth Estimation*

Even though the conventional approaches provide first reasonable results, the real breakthrough of single-view depth estimation was achieved by the usage of deep learning-based methods and CNNs.

Supervised learning: In his pioneering work, Eigen et al. [2014] proposed a CNN that is trained in a *supervised manner* and estimates the depth as a *regression problem* in a coarse-to-fine scheme. Different training losses were proposed for supervised learning such as the scale-invariant training loss [Eigen et al., 2014], berHu norm [Kuznietsov et al., 2017], or L2-norm [Qi et al., 2018]. Alternatively, Fu et al. [2018] formulated the estimation as an *ordinal classification problem* by discretizing the depth values and training the CNN with an ordinal regression loss. The currently leading approach[1] in the KITTI depth benchmark [Uhrig et al., 2017] has been proposed by Lee et al. [2019]. The single-view depth estimation is formulated as a regression problem trained with the scale-invariant loss and integrates a novel *local planar guidance*. At each scale of the decoder, features are learned to recover the depth estimation back to full resolution with local planarity assumptions. This improves, on the one hand, the upsampling step in contrast to the nearest neighbor approach. On the other hand, for some parts of the image, it might be easier to estimate the local structure instead of the absolute depth value. This characteristic could be exploited explicitly in the proposed architecture including local planar guidance. Ground truth data is typically acquired by RGB-D cameras for indoor scenes or light detection and ranging (LiDAR) sensors for outdoor scenes.

Self-supervised learning: To reduce the effort of ground truth data collection, several *self-supervised learning* approaches using a stereo camera setup have been proposed [Garg et al., 2016, Godard et al., 2017, Kumar et al., 2018, Guo et al., 2018, Godard et al., 2019, Luo et al., 2018, Tosi et al., 2019, Kuznietsov et al., 2017]. The main idea is that based on the estimated depth, it should be possible to reconstruct the right image by warping the left image. Consequently, the training loss is

[1] Benchmark on October 02, 2020. Methods with a publication are considered.

formulated as an *image reconstruction loss*, which is defined as a photometric distance between the reconstructed and original right image. The photometric distance is defined as an L2-norm [Garg et al., 2016] or as a combination of the L1-norm and structural similarity (SSIM) of the pixel intensities [Godard et al., 2017]. In addition to this concept, several adaptions have been proposed to improve the accuracy. Godard et al. [2017] proposed to integrate a loss-term that prefers consistency of the depth estimates of the left and right image. Aleotti et al. [2018], Kumar et al. [2018] proposed to train a discriminator that rates the similarity of the reconstructed and real image following the concept of generative adversial networks (GANs) [Goodfellow et al., 2014a]. The discriminator response is used for fine-tuning or as an additional part of the loss for training the depth estimation network, which is the generator in terms of the GAN concept. Alternatively, instead of taking the depth estimates of the network directly, a subsequent stereo matching network is fed with the original left and reconstructed right image [Luo et al., 2018, Tosi et al., 2019]. The stereo setup could also be used to generate pseudo ground truth using stereo algorithms [Guo et al., 2018, Gan et al., 2018].

Unsupervised learning: The concept of the image reconstruction loss is transferred to an *unsupervised learning* approach in a monocular image sequence [Zhou et al., 2017, Yin and Shi, 2018, Yang et al., 2018b, Teng et al., 2018, Zhan et al., 2018, Wang et al., 2018, Mahjourian et al., 2017, Casser et al., 2019, Almalioglu et al., 2019, Chen et al., 2019]. Based on the estimated depth, the image is warped to reconstruct the subsequent image and the network is trained by minimizing the photometric distance between the original and reconstructed images. However, in such a setup also the *camera motion* needs to be known for the warping step and is typically estimated by an additional CNN. Ideas similar to the self-supervised approaches are proposed to improve accuracy: Almalioglu et al. [2019], Wu et al. [2019] proposed an additional discriminator for training and Mahjourian et al. [2017] presented a loss that prefers consistency between the point clouds of subsequent images. The generation of pseudo ground truth was also adapted by using SfM-based approaches [Klodt and Vedaldi, 2018]. Furthermore, *multi-task networks* were proposed that additionally estimate the optical flow [Yang et al., 2018b, Teng et al., 2018, Zou et al., 2018, Chen et al., 2019]. A consistency loss between optical flow, pose and depth results in a further improvement of accuracy. The unsupervised methods are trained based on the multi-view consistency in a monocular setup assuming a static environment. Consequently, these methods suffer from a *scale ambiguity* and are not able to *handle moving objects*. For better handling of moving objects, Mou et al. [2019] proposed to predict moving object masks and Casser et al. [2019] proposed a relative pose estimation for moving objects. Chen et al. [2019] proposed an adaptive photometric loss, which allows some parts of the image to differ from the global rigid displacement.

These different training strategies can also be combined. e.g. by combining supervised and self-supervised learning [Kuznietsov et al., 2017, Jiang et al., 2018] or combining self-supervised and unsupervised learning [Godard et al., 2019, Jiang et al., 2018, Yang et al., 2018b, Teng et al., 2018, Zhan et al., 2018]. The ProbDepthNet proposed in section 3.3 is trained in a supervised manner. However, the ground truth is collected by combining a LiDAR sensor and stereo setup. The stereo setup is exploited to overcome the limitations of the LiDAR sensor in terms of range and field of view.

To get a deeper understanding of the used *single-view depth cues*, Dijk and Croon [2019] performed several experiments based on the CNN for single-view depth estimation by Godard et al. [2017]. These experiments revealed that the depth estimation of an object highly depends on the image position of the ground contact point. The object size in the image is less important. Furthermore, some parts of the object have nearly no impact on the depth estimates. Even if the center of a vehicle is removed, the entire vehicle is reconstructed. The CNN is able to recognize situations where the ground surface differs from the flat world assumption. However, the difference is typically underestimated, which reveals that the extrinsic camera position and flat world assumption are taken into account as prior knowledge. The experiments also show that the accuracy significantly drops by removing the texture, while removing the color has only a minor impact.

3.1.2 Probabilistic Deep Learning

In contrast to the methods presented in the previous section 3.1.1.3, ProbDepthNet is designed to estimate pixel-wise depth distributions. Therefore, the present subsection presents works related to *probabilistic deep learning* in general. There exist slightly different definitions and namings regarding the types of uncertainty in the literature. Following these definitions, I propose that the types of uncertainties are mainly distinguishable into three groups.

Types of uncertainty: First, the *measurement uncertainty* (also called *aleatoric* or *intrinsic* uncertainty) refers to the uncertainty inherent in the measurement data and the nature of the problem. On the one hand, it comprises the uncertainty due to some measurement noise. On the other hand, the measurements could be ambiguous and not enough to determine a unique solution. For example, consider a neural network that defines a function that maps a 2D input image to a depth map. A toy vehicle close and a real vehicle further away from the camera can result in the same image including the same size and appearance of both vehicle in the image, even though the vehicles are at different distances. This uncertainty can not be removed even with infinite training data [Kendall and Gal, 2017] – but it is observable during the training process.

Second, the *model uncertainty* (also called *epistemic* uncertainty) comprises the uncertainty of the model parameters. There might exist more than one set of model parameters that can explain the given training data equally well. This ambiguity in selecting the model parameters is expressed by the model uncertainty. In contrast to the measurement uncertainty, this uncertainty could decrease given enough training data [Kendall and Gal, 2017]. The influence of the model uncertainty could also be propagated to the output uncertainty.

Third, the *out of distribution* (OOD) *uncertainty* (also called *distributional* uncertainty) defines the uncertainty due to a mismatch of training and test data. There is a high probability that a test sample that is highly different from the training data results in erroneous estimates. For example, a model trained on outdoor scenes will typically fail in indoor scenes. In contrast to the previous uncertainties, this uncertainty is not observable during the training process due to the fact that OOD data samples are not given during training by definition.

These uncertainties and how to estimate them are explained in more detail in the following subsections. Methods that applied the corresponding strategy to single-view depth estimation are highlighted.

3.1.2.1 *Measurement Uncertainty*

As described above, the *measurement uncertainty* refers to the uncertainty inherent in the measurement data and the nature of the problem.

Measurement uncertainty for classification: For a discrete classification problem, a probability score p_y of the estimated class y is typically derived by the *softmax function* σ_{SM}. The softmax function is applied on the *logits* $\mathbf{z}_i$, which is the network output that represents an unnormalized score for each class:

$$\begin{gathered}\sigma_{SM}(\mathbf{z}_y) = e^{z_y} / \sum_j e^{z_j} \\ p = \max_y \sigma_{SM}(\mathbf{z}_y)\end{gathered} \tag{3.1}$$

However, due to the known issues of overfitting effects, these scores should not directly be interpreted as probabilities. These scores are, in particular for modern networks, overconfident and not well-calibrated [Guo et al., 2017]. Therefore, different *recalibration* methods were proposed to derive well-calibrated probabilities. In general, recalibration is formulated as a post-processing step, which produces well-calibrated probabilities $\hat{p}$ based on the probability scores p or the logits $\mathbf{z}$. The transformation from uncalibrated to well-calibrated probabilities is determined on a hold-out split of training samples. The hold-out split is a part of the training data, which is not used to train the weights of the network.

For histogram binning [Zadrozny and Elkan, 2002], a set of bins $B_1, ..., B_N$ is defined. Each bin corresponds to a certain interval of uncalibrated probability scores $B_j = (p_j, p_j + 1]$, which are mapped to a calibrated probability $\hat{p}_j$. Alternatively, isotonic regression [Zadrozny and Elkan, 2002, Niculescu-Mizil and Caruana, 2005] is used as a recalibration technique. *Isotonic regression* is defined as a piecewise constant function, which transforms the probabilities by $\hat{p} = f(p)$. It generalizes the discretized form of histogram binning to a continuous function. Bayesian binning into quantiles [Naeini et al., 2015] considers the whole space $\mathcal{S}$ of possible binning schemes. Based on the probability of each binning scheme $s \in \mathcal{S}$ determined on a dataset $\mathcal{D}$, the recalibrated probabilities are defined by $p(\hat{p} \mid p, D) = \sum_{s \in \mathcal{S}} p(\hat{p} \mid p, s) \cdot p(s \mid \mathcal{D})$. While these calibration methods are directly based on the uncalibrated probabilities p, the following methods manipulate the logits $\mathbf{z}_i$ to derive calibrated probabilities $\hat{p}$. Platt scaling or matrix and vector scaling [Platt, 1999] transforms the logits by $\mathbf{Wz} + \mathbf{b}$ so that the probabilities after applying the softmax function $\hat{p}_y = \sigma_{SM}(\mathbf{Wz}_y + \mathbf{b})$ are well-calibrated. Alternatively, the logits are scaled by a single scalar parameter $\mathbf{z}/T$. Thereby, the maximum of the softmax remains the same and the recalibration does not affect the model accuracy. This approach is known as temperature scaling [Guo et al., 2017].

Liu et al. [2019] proposed to formulate single-view depth estimation as a classification problem to derive depth distributions. The inverse depth is uniformly discretized and a depth probability volume is estimated using a softmax function. A recalibration technique is not applied.

Measurement uncertainty for regression: In contrast to classification problems, a regression network typically provides only a single maximum likelihood estimate (e.g. a single depth value d) without any probability score or uncertainty measure.

The first strategy to quantify the measurement uncertainty is to estimate *distribution parameters* instead of a single estimate, e.g. $p(y_i) \approx \mathcal{N}(\mu_i, \sigma_i^2)$. The parameters defining the distributions are trained by minimizing the negative log-likelihood (NLL)-loss [Kendall and Gal, 2017]. For example, assuming a Gaussian distribution of the network output, the network estimates the mean value $\mu(\mathbf{x}_i)$ and variance $\sigma^2(\mathbf{x}_i)$ for each data sample $\mathbf{x}_i$. Based on the training data $\mathcal{D}$ with the corresponding ground truth values y_i, the network weights are trained by minimizing

$$\mathcal{L} = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \frac{1}{2 \cdot \sigma(\mathbf{x}_i)^2} \, ||y_i - \mu(\mathbf{x}_i)||^2 + \log \sigma(\mathbf{x}_i)^2. \tag{3.2}$$

This strategy is also apllied to probabilistic single-view depth estimation [Kendall and Gal, 2017, Laidlow et al., 2019, Ma et al., 2018, Klodt and Vedaldi, 2018] .

The second type of approaches proposed by Ilg et al. [2018] is to estimate a *set of hypotheses* instead of a single estimate. Thereby, the distribution is defined by the set of hypotheses $\mathcal{H}$, e.g. by computing the empirical mean and variance. Ilg et al. [2018] proposed a loss, which rates the distance to the best hypotheses. This encourges the network to make diverse hypotheses, especially for uncertain estimates, to have at least one hypothesis close to the ground truth.

$$\mathcal{L}_{hypo} = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \left(\min_{j \in \mathcal{H}} \mathcal{L}_i + \sum_{k \in \mathcal{H}} \mathcal{L}_{k,reg} \right) \tag{3.3}$$

The regression $\mathcal{L}_{k,reg}$ prefers that similar solutions are from the same hypothesis. The first two approaches could be combined by predicting a set of distributions instead of a set of single estimates [Ilg et al., 2018]. In this case, $\mathcal{L}_i$ corresponds to the NLL-loss as defined in equation (3.2). Additionally, Ilg et al. [2018] proposed a subsequent network that directly provides an uncertainty measure based on the set of hypothesis. Xia et al. [2020] used the GAN concept to estimate a set of hypotheses. The decoder is interpreted as a generator that estimates different hypotheses depending on an additional noise source. Each hypothesis represents a plausible depth estimate for an image patch. The set of hypotheses can also covers the correlation inside the image patch. While the method by Ilg et al. [2018] is applied to the task of optical flow estimation, the approach by Xia et al. [2020] is used for probabilistic single-view depth estimation.

Gast and Roth [2018] proposed a new type of network, which they called *lightweight probabilistic deep networks*. This approach is considered as a third type of strategy. While the approaches described above represent the output as a distribution, these networks also replace the output of intermediate activations by distributions. These distributions are derived by propagating input and activation uncertainties through the network using *assumed density filtering* (ADF). The network is still trained by minimizing the NLL on the training data. The number of parameters and network architecture remains the same. However, replacing a network with its ADF-counterpart roughly doubles the inference time.

The ProbDepthNet proposed in section 3.3 is designed to estimate the measurement uncertainty in a regression problem. While the calibration is well-studied for classification tasks, it is ignored for the presented approaches that quantify the measurement uncertainty in a regression problem. The experiments in section 3.4.2 evaluate the calibration of these strategies and show that these strategies suffer from overconfident estimates. Moreover, a novel recalibration method is proposed to achieve well-calibrated distributions.

3.1.2.2 *Model Uncertainty*

Model uncertainty refers to the fact that more than one set of model parameter values might explain the given training data equally well. Therefore, in contrast to traditional deep learning, *Bayesian deep learning* additionally models the uncertainty of model parameters $p(\mathbf{W} \mid \mathbf{X}, \mathbf{Y})$ based on the training data $\mathbf{X}, \mathbf{Y}$. The probability of the output $p(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y})$ is defined as

$$p(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y}) = \int p(y \mid \mathbf{x}, \mathbf{W}) \cdot p(\mathbf{W} \mid \mathbf{X}, \mathbf{Y}) d\mathbf{W}. \tag{3.4}$$

In many cases, the closed-form solution of the integral does not exist or is computationally infeasible. Therefore, the integral is often approximated based on N sampled model parameters $\mathbf{W}_i$ using Monte Carlo integration:

$$p(y \mid \mathbf{x}, \mathbf{X}, \mathbf{Y}) \approx \frac{1}{N} \sum_{i=1}^{N} p(y \mid \mathbf{x}, \mathbf{W}_i) \cdot p(\mathbf{W}_i \mid \mathbf{X}, \mathbf{Y}) \tag{3.5}$$

As evaluating the true probability might be computationally infeasible or intractable, variational distributions represented by the parameters Θ are introduced to approximate $p(\mathbf{W} \mid \mathbf{X}, \mathbf{Y}) \approx q(\mathbf{W} \mid \Theta)$. The variational distributions are typically restricted to the tractable family of distributions. For example, it is assumed that the distribution factorizes with respect to the model parameters $q(\mathbf{W} \mid \Theta) = \prod_i q(\mathbf{W}_i \mid \Theta_i)$ and that the distribution of each weight could be represented by a Gaussian distribution [Blundell et al., 2015] with the mean and variance as parameters. A popular and efficient approximation was proposed by [Gal and Ghahramani, 2016] and is known as *Monte Carlo Dropout*. A unit j of the input $\mathbf{M}_i$ for layer i is dropped out with some probability p_i. Formally, the weights of the Monte Carlo Dropout approach are defined based on a Bernoulli distribution:

$$\begin{gathered} \mathbf{W}_i = \mathbf{M}_i \cdot diag(z_{i,j}|_{j=1}^{K}) \\ \text{with } z_{i,j} \approx \text{Bernoulli}(p_i) \end{gathered} \tag{3.6}$$

Consequently, the output with dropout during inference can be seen as the result of one sampled network drawn from the distribution of weight parameters. The probability $p(y|\mathbf{x}, \mathbf{X}, \mathbf{Y})$ is derived by Monte Carlo integration. In order to avoid the disadvantage of performing the inference multiple times for the same image, Huang et al. [2018] proposed a Monte Carlo sampling over time by warping previous estimates based on an optical flow field. Monte Carlo BatchNorm [Azizpour et al., 2018] is comparable to Monte Carlo Dropout but uses BatchNorm as stochastic regularization instead. Using a stochastic regularization method can be interpreted as injecting noise at certain layers during training and inference. Postels et al. [2019] proposed to explicitly formulate the injected noise at certain layers as an addition or an element-wise multiplication of the activation values with a random vector. Based on this formulation, the output uncertainty is derived by error propagation based on an assumed noise covariance matrix. This approach avoids sampling of the network multiple times and reduces the computational effort.

Krueger et al. [2018], Pawlowski et al. [2017] proposed a *hypernetwork* that estimates the weight distributions. The model weights of the hypernetwork serve as the variational parameters that approximate the distribution of model parameters.

Furthermore, an *ensemble of networks* could be used to perform Monte Carlo integration over the network outputs [Lakshminarayanan et al., 2017]. Even though it does not directly define the distributions of model parameters, the ensemble could be interpreted as N networks drawn from the distribution of model parameters.

Kuleshov et al. [2018] analyzed the *calibration* of Monte Carlo Dropout [Gal and Ghahramani, 2016] and the ensemble strategy [Lakshminarayanan et al., 2017] for regression problems such as single-view depth estimation. The experiments show that both methods benefit from a recalibration method that manipulates the cumulative distribution function using isotonic regression. Even though the recalibration is applied to methods that quantify the model uncertainty, this recalibration is most related to our proposed recalibration because both are focused on regression problems. In contrast to this recalibration method, the proposed recalibration in the present thesis maintains the type of distribution and provides continuously differentiable distributions. These characteristics are beneficial for many approaches to work with the probabilistic estimates. The estimation of the model uncertainty is not addressed as part of ProbDepthNet.

The approaches for quantification of measurement and model uncertainties can be combined. For example, Kendall and Gal [2017] combined Monte Carlo Dropout with an output distribution that is trained by minimizing the NLL and Segù et al. [2020] proposed to use ADF in combination with Monte Carlo Dropout. The approach by Kendall and Gal [2017] is also applied to probabilistic single-view depth estimation.

3.1.2.3 *Out-of-Distribution Uncertainty*

The measurement and model uncertainty show a dependency on the training data. The measurement uncertainty is quantified by observing the output distribution on the training data and the model uncertainty represents the uncertainty of the parameters based on the training data. However, there exists a third uncertainty that is related to a mismatch of a test sample to the training data. Those test samples are called OOD data. Malinin and Gales [2018] defined the uncertainty based on the mismatch as *distributional uncertainty*, while other approaches are designed to explicitly detect OOD samples.

Some works reveal that standard approaches are not directly applicable for OOD detection. Hendrycks and Gimpel [2017] analyzed the softmax output of a standard classifier for OOD detection. The probability score can be a slight indicator, but in many cases, the classifier also provides high confidences for OOD data [Nguyen et al., 2015, Hendrycks and Gimpel, 2017]. Some works consider the model or epistimic uncertainty as all the uncertainties that can be reduced by further training data including the OOD uncertainty. This is also supported by the fact that approaches addressing the model uncertainty such as Monte Carlo Dropout [Gal and Ghahramani, 2016] or an ensemble strategy [Lakshminarayanan et al., 2017] show better characteristics for detecting OOD samples. However, Lis et al. [2019], Mundt et al. [2019] showed that also the model uncertainty does not provide the intended detection accuracy. Furthermore, variational auto-encoders (VAEs) [Kingma and Welling, 2014] are able to assign a probability to an image to be generated. Intuitively, one might expect that a VAE outputs a low probability for OOD samples. However, this is not the case in general as shown by Nalisnick et al. [2019].

In summary, several works show that standard approaches are not directly suitable for OOD detection. Consequently, special strategies and approaches have been designed that explicitly address OOD detection. I propose a categorization of these methods into four groups.

OOD data generation: Approaches that *generate a representation* of OOD data for training are considered as the first category of methods. Given such representation, the network is, for example, trained to predict a uniform distribution for OOD samples [Lee et al., 2018a, Malinin et al., 2017, Vyas et al., 2018, Malinin and Gales, 2018]. However, OOD data is not given during training by definition and the challenge is how to find a suitable representation for such data. Different datasets are used to represent OOD data [Malinin and Gales, 2018, Masana et al., 2018]. For training an ensemble of networks, even different splits of the in-distribution data are used to represent the OOD data for training [Vyas et al., 2018]. Alternatively, synthetic generations of OOD data samples are proposed using factor analysis in spoken language assessment [Malinin et al., 2017] or using an additional GAN [Lee et al., 2018a]. The GAN is trained to generate samples at the boundary of the distribution of the original training data. These approaches are sensitive to the generated data samples that represents OOD samples. High accuracy is achieved if the OOD samples are from the dataset that is used to test the OOD detection [Yu and Aizawa, 2019]. However, this is typically not the case and OOD samples are actually not given during training by definition.

Controlled input perturbations: The second category represents methods that perform *controlled perturbations* of the input for OOD detection. A small but worst-case perturbation of an input that results in an incorrect classification is called an adversarial example [Goodfellow et al., 2014b]. This example can be generated based on the gradient of the loss function. Liang et al. [2018], DeVries and Taylor [2018], Lee et al. [2018b] proposed a similar strategy to detect OOD. In contrast to adversarial examples, small perturbations are applied to the input such that the confidence of the detected class increases. This confidence increases more rapidly for in-distribution data than for OOD data. Thus, applying such input processing could be used to separate in-distribution from OOD data based on its confidence. After applying the controlled input pertubation, Liang et al. [2018] proposed a threshold based on the softmax output to detect OOD data samples.

Neural activation pattern: While the first both categories are designed to separate the confidences of OOD data from in-distribution data, the third category analyzes intermediate activations and feature vectors. Even though the final confidence of OOD data often indicates high confidence, usually a different set of neurons shows high activations compared to in-distribution data. The proposed approaches basically differ in terms of how to characterize the typical *activation pattern* for each class. The detection could be based on (1) the distance to the mean activations of the penultimate layer (the layer before the final softmax output) [Bendale and Boult, 2016], (2) the Mahalanobis distance to the activations of the penultimate layer [Lee et al., 2018b], or (3) the distance to a signature representing the K-highest activations [Schultheiss et al., 2017]. Mundt et al. [2019] applied this strategy also to the latent space of a generative model.

Image resynthesis: The fourth category is related to OOD detection based on *image resynthesis* as proposed by Lis et al. [2019]. A GAN is trained to resynthesize the image based on its semantics. Furthermore, a discrepancy network is trained to identify unexpected objects of unknown classes by comparing the features of the original and the resynthesized image. To train the discrepancy network, the semantic label is swapped for some objects, which are then considered as unexpected objects

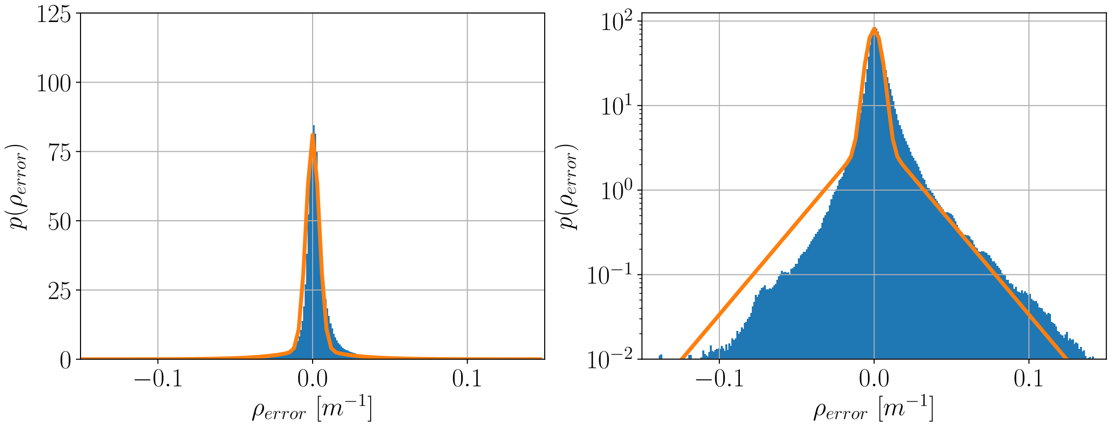


Figure 3.2: Empirical distribution of the inverse depth error ρ_{error} of the single-view depth estimates by Godard et al. [2017]. The blue histograms show the empirical distribution of the error on the KITTI scene flow training set [Menze and Geiger, 2015]. The orange curves show the approximated distribution as a mixture model of a Laplacian and a Gaussian distribution. The distribution is shown with a logarithmic scale of the frequency in the right diagram.

during training. This approach is able to predict a pixel-wise score to belong to an unexpected object.

An OOD sample is related to a mismatch of a test sample to the training data. However, it remains unclear to what extent a network generalizes and which scenes should be regarded as OOD samples. Therefore, a generalization experiment is performed in section 3.4.3 to show examples of scenes that can be considered as OOD for the ProbDepthNet model. The detection of those samples is out of the scope.

3.2 Empirical Analysis of Error Distribution

The present section provides some analysis regarding the *empirical error distribution* of single-view depth estimates. The main motivation of the analysis is to give an impression of how the error behaves and for which parts of the scenes the estimates can be expected to be more accurate. A previous work [Brickwedde et al., 2018a] additionally has shown that a measurement model for single-view depth estimations can be derived from the results of the analysis.

Type of error distribution: The error distribution of single-view depth estimates is shown in figure 3.2. I chose the approach by Godard et al. [2017] for single-view depth estimation to analyze its accuracy on the KITTI scene flow dataset [Menze and Geiger, 2015]. The error is defined in terms of the inverse depth $\rho = Z^{-1}$, where Z corresponds to the distance of the point to the camera in the z-direction. The error distribution mainly consists of two parts. First, there is one part with

Distance $[m]$	**0-10**	**10-20**	**20-30**	**30-40**	**40-50**	**50-60**	**60-70**	**70-80**
RMSE $[m]$ (depth)	1.852	3.284	6.389	9.735	12.81	18.45	22.3	28.43
RMSE $[m^{-1}]$ (inverse depth)	0.01404	0.01225	0.01266	0.0123	0.01074	0.0128	0.01183	0.0126

Table 3.1: Root mean squared error (RMSE) in terms of depth and inverse depth of the single-view depth estimation by Godard et al. [2017] dependent on the distance of the scene point to the camera.

slowly decreasing tails, which mainly models the distribution of large errors and follows a linear decreasing shape on a logarithmic scale. Second, there is another part that corresponds to a peak and high probabilities for small errors. This characteristic is approximated by a mixture model consisting of a Laplacian and a Gaussian distribution:

$$p(\rho_{error}) = \frac{1-\lambda}{\sqrt{2\pi}\sigma} e^{-\rho_{error}^2/(2\sigma^2)} + \frac{\lambda}{2b} e^{-|\rho_{error}|/b} \tag{3.7}$$

Figure 3.2 shows the approximated density function for $\sigma = 0.0042$, $b = 0.02$ and $\lambda = 0.2$.

The figure shows the error distribution over all estimates. However, the error distribution is likely not the same for all parts of the scene. Some parts of the scene probably have stronger single-view depth cues than others.

Dependence on distance: To analyze the dependence on the distance to the camera, the root mean squared error (RMSE) in terms of depth and inverse depth is evaluated for certain depth intervals $[d_{min}, d_{max}]$ separately:

$$\begin{aligned} \text{RMSE(depth)} &= \sqrt{\frac{1}{|\Omega_{GT}(\rho_{min}, \rho_{max})|} \sum_{\mathbf{p} \in \Omega_{GT}(\rho_{min}, \rho_{max})} (\hat{\rho}(\mathbf{p})^{-1} - \rho_{GT}(\mathbf{p})^{-1})^2} \\ \text{RMSE(inverse depth)} &= \sqrt{\frac{1}{|\Omega_{GT}(\rho_{min}, \rho_{max})|} \sum_{\mathbf{p} \in \Omega_{GT}(\rho_{min}, \rho_{max})} (\hat{\rho}(\mathbf{p}) - \rho_{GT}(\mathbf{p}))^2} \end{aligned} \tag{3.8}$$

All pixels $\mathbf{p} \in \Omega_{GT}(\rho_{min}, \rho_{max})$ with valid ground truth $\rho_{GT}(\mathbf{p}) \in [\rho_{min}, \rho_{max}]$ are considered to calculate the metric.

The results in table 3.1 show that the error is nearly uniformly distributed over the distance in terms of the inverse depth. This motivates to approximate the distribution of the inverse depth instead of the depth directly.

Dependence on semantic class: The error distribution is evaluated separately for different semantic classes c_i as shown in figure 3.3. This experiment is designed to reveal the dependence on the semantic class. The figures show that the error distribution depends on the semantic class. The estimated parameters σ_{c_i}, b_{c_i} and λ_{c_i}

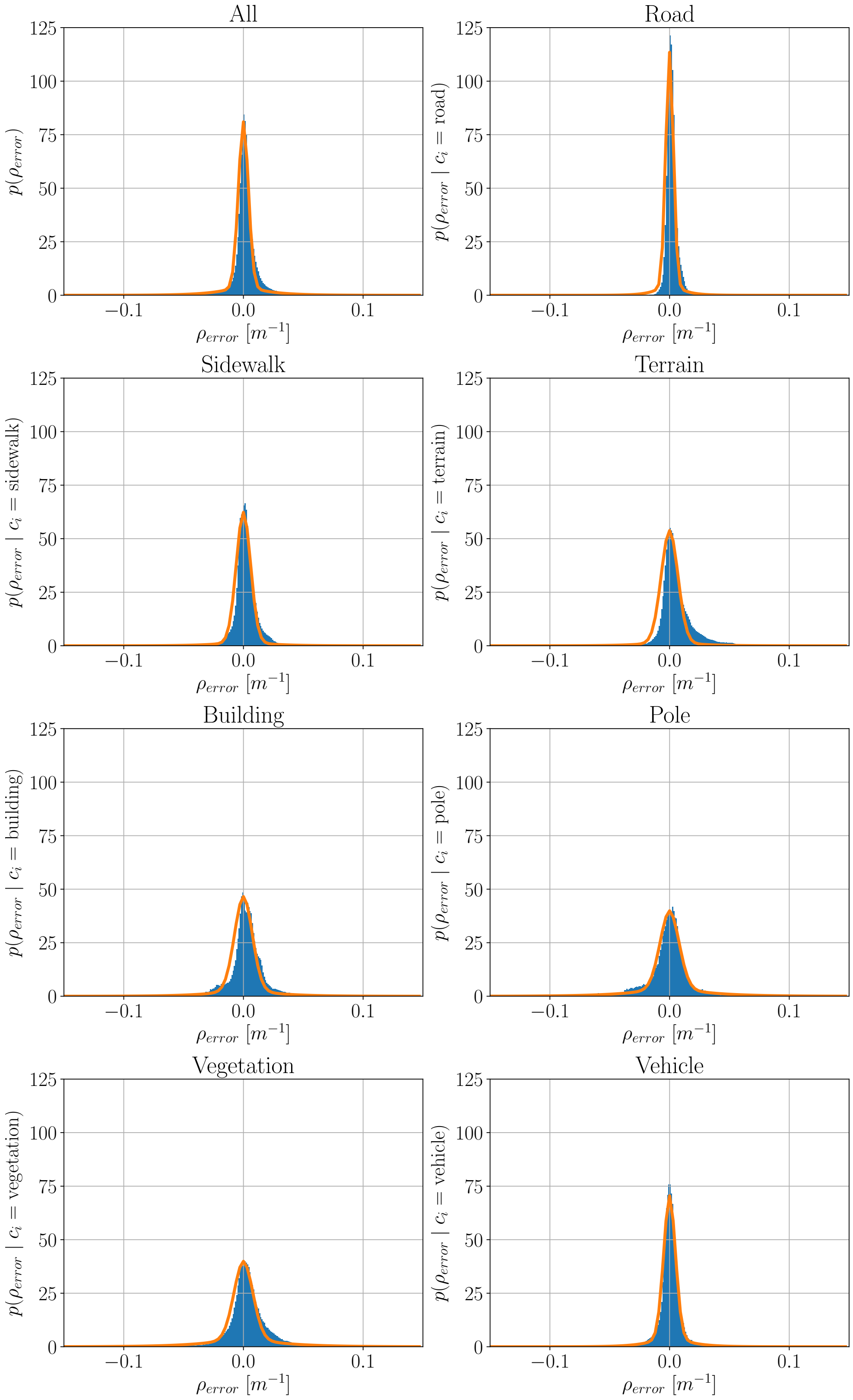


Figure 3.3: The empirical distribution of the inverse depth error ρ_{error} of the single-view depth estimation by Godard et al. [2017] dependent on the semantic class. The blue histograms show the empirical distribution of the error on the KITTI scene flow training set [Menze and Geiger, 2015]. The orange curves show the approximated distributions as a mixture model of a Laplacian and a Gaussian distribution.

Class c_i	Road	Sidewalk	Terrain	Building	Pole	Vegetation	Vehicle
σ_{c_i}	0.0032	0.006	0.007	0.0075	0.008	0.008	0.005
b_{c_i}	0.01	0.02	0.02	0.025	0.03	0.03	0.015
λ_{c_i}	0.15	0.1	0.1	0.2	0.3	0.3	0.2

Table 3.2: Semantic class-dependent approximation of the error distribution for the single-view depth estimation by Godard et al. [2017].

of the distribution specified in equation (3.7) are shown in table 3.2 for each class. The corresponding density functions are shown in figure 3.3.

The classes *road* and *vehicle* show the highest accuracy with a variance σ_{c_i} approximately half of the variance of classes such as building, pole, or vegetation. The classes pole and vegetation additionally show the highest weighting factor λ_{c_i} for the Laplace distribution, which models outliers and high errors. The other ground classes, sidewalk and terrain, show a medium accuracy with higher variances σ_{c_i} than the road and vehicle classes, but also with a low weighting factor λ_{c_i} for the Laplace distribution. The results indicate a higher accuracy for classes that follow strict model assumptions, for example, regarding surface, shape, or size and which are frequently represented in the training dataset.

This observation shows a dependence on the semantic class and motivates to approximate the distribution separately for each semantic class. The previously published work [Brickwedde et al., 2018a] gives evidence that such semantic class-dependent measurement models are suitable to describe the error distribution – especially for integrating single-view depth estimates in a multi-view geometry-based scene flow estimation method.

Note that this analysis is only valid for the method by Godard et al. [2017]. For example, while the distribution in figure 3.2 shows some asymmetry with a longer tail for positive errors, the method by Fu et al. [2018] shows such asymmetry in the inverse form with a long tail for negative errors. However, the dependence on the semantic classes is nearly the same.

3.3 ProbDepthNet Model

Even though previous work [Brickwedde et al., 2018a] gives evidence that the analyzed distributions shown in the previous section 3.2 could serve as a measurement model, it assumes the same distribution for all pixels belonging to the same semantic class. To provide a more distinctive *uncertainty quantification*, I propose a CNN, called *ProbDepthNet*, that is designed to provide the pixel-wise uncertainty of each estimate. Thus, the main objective of ProbDepthNet is not to provide a single esti-

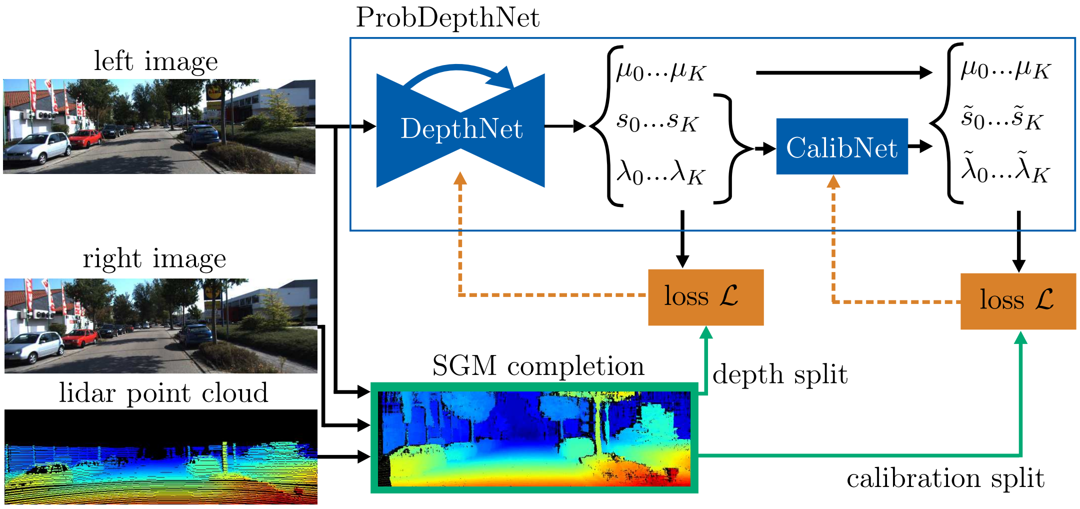


Figure 3.4: Overview of ProbDepthNet architecture and training process. The architecture consists of two parts: DepthNet and CalibNet for recalibration (blue). Both parts provide a parameterized form (μ_i, s_i / $\tilde{s}_i$ and λ_i / $\tilde{\lambda}_i$) of a MoG. Each part is trained on a different split of the training data using a NLL-loss (orange). The ground truth data is provided by a stereo SGM [Hirschmuller, 2005]-based completion of a LiDAR point cloud (green).

mate for the most likely depth value, but to provide a probability density function of the depth for each pixel $\mathbf{p}$ given an input image I. Motivated by the analysis in section 3.2 (table 3.1), depth is encoded by its inverse form $\rho = Z^{-1}$, where Z is the z-coordinate of the 3D position of the regarded point in camera coordinates. ProbDepthNet predicts a pixel-wise probability density function $p_{\mathbf{p}}(\rho \mid I)$ parameterized as a mixture of Gaussians (MoG):

$$\begin{aligned} p_{\mathbf{p}}(\rho \mid I) &= \sum_{i=1}^{K} \lambda_i(\mathbf{p}) \cdot \mathcal{N}\left(\rho - \mu_i(\mathbf{p}), \sigma_i^2(\mathbf{p})\right) \\ \text{with } & \sum_{i=1}^{K} \lambda_i(\mathbf{p}) = 1 \end{aligned} \tag{3.9}$$

The distribution is parameterized by the number of components K, the weights λ_i, the mean values μ_i, and the standard deviations σ_i^2 of each i-th component. Compared to a single Gaussian distribution, a mixture model is able to capture more general distributions, e.g. a multimodal distribution, but other parameterizations of a probability distribution can be used as well. Figure 3.4 gives an overview of the architecture, training process, and ground truth generation.

Architecture: *ProbDepthNet* consists of two parts: *DepthNet* and *CalibNet*. The outputs of DepthNet are the parameters of the MoG, whereby the variance is provided in the log-space $s_i = \log \sigma_i^2$. While DepthNet already outputs a depth distribution, the CalibNet serves as a recalibration technique and reshapes the dis-

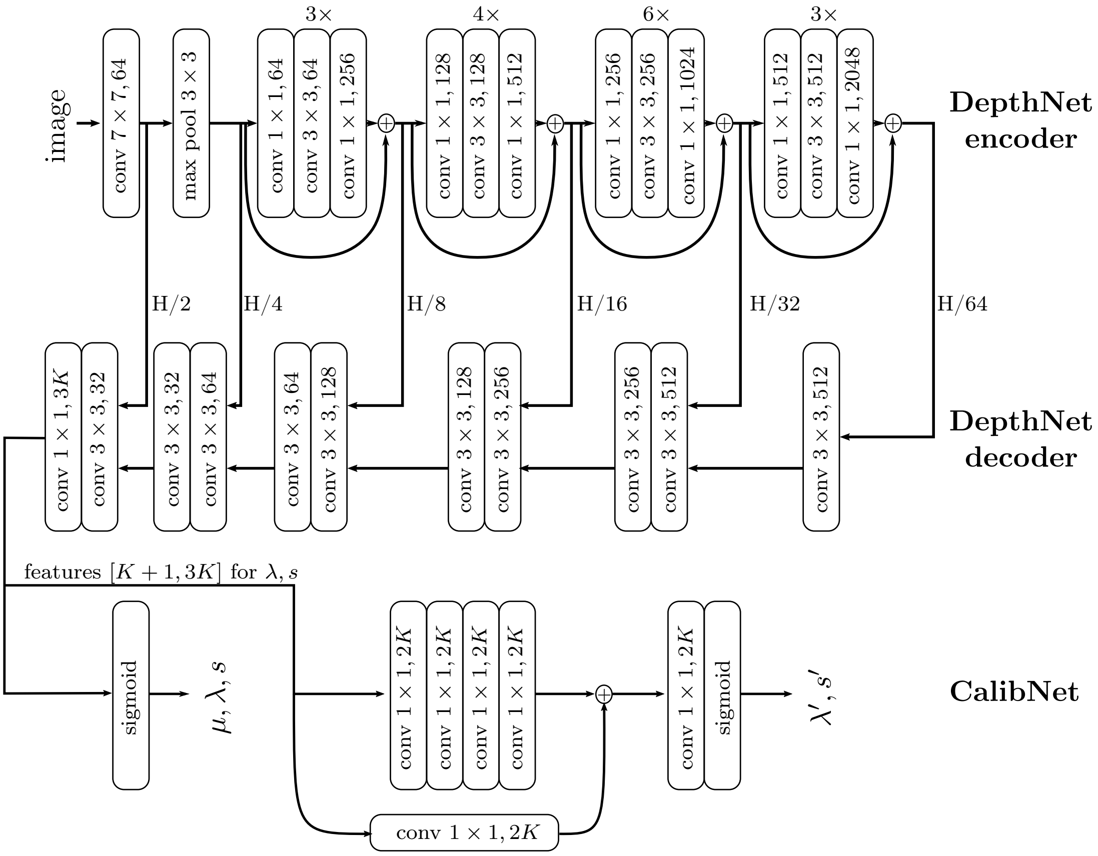


Figure 3.5: Architecture of ProbDepthNet model. The DepthNet encoder corresponds to a ResNet-50 [He et al., 2016] architecture plus a concatenated residual block denoted as CalibNet.

tribution of DepthNet such that it is better calibrated. A detailed illustration of the architecture is shown in figure 3.5. DepthNet is a fully convolutional ResNet-50 [He et al., 2016] with skip connections between corresponding encoder and decoder layers. The same residual blocks are connected several times in a row. The number of repetitions is given above each block in figure 3.5. Each block consists of convolutional layers, whereby the first two numbers in figure 3.5 represent the kernel size and the third number represents the feature dimension. Following the residual network concept, there is an additional shortcut connection. If the dimensions of the shortcut and output of the convolutional blocks do not match, an additional 1 × 1 convolutional layer is placed into the shortcut with a stride and feature dimension aligned to the other path. The output size after each block in relation to the input image size H is specified at the skip connections in figure 3.5. The reduction of the size is achieved by using a stride of 2 at the last layer of each block. The corresponding DepthNet decoder consists of convolutional layers, whereby the output size is aligned by upsampling the feature in advance to the last layer of each clock.

The logarithmic variances s_i and weights λ_i of DepthNet are recalibrated by CalibNet, which outputs the corresponding recalibrated values $\tilde{s}_i$ and $\tilde{\lambda}_i$. The CalibNet architecture follows a concatenated residual block and is shown in the bottom part of figure 3.5. The inputs are the features of the DepthNet decoder that represent the weights λ and logarithmic variances s of the MoG with K components. The CalibNet model is similar to a single residual block, whereby the shortcut path additionally includes a convolutional layer to represent a scaled version of the input features.

An activation function is used after each convolutional layer or summation. The elu function is used as activation function expect for the final outputs, which are the result of a sigmoid activation function. The sigmoid activation function are explicitly shown in figure 3.5. For the CalibNet, there is no activation after the summation to ensure that the shortcut path can directly represent a scaled version of the input.

CalibNet directly transforms the parameters of the depth distribution. Thus, in contrast to [Kuleshov et al., 2018], the type of recalibrated distribution remains the same and is continuously differentiable.

Training: The network parameters of DepthNet and CalibNet are trained on different, non-overlapping splits of the training data. The split, on which the CalibNet layers are trained on, is denoted as calibration split and deliberately taken out of the training data of the DepthNet to avoid overfitting of DepthNet on the calibration split. The NLL-loss $\mathcal{L}$ is minimized during training similar to [Kendall and Gal, 2017, Klodt and Vedaldi, 2018]:

$$\mathcal{L} = \sum_{\mathbf{p} \in \Omega_{GT}} \left[-\log \left(\sum_{i=1}^{K} \lambda_i(\mathbf{p}) \cdot \mathcal{N}\left(\rho_{GT}(\mathbf{p}) - \mu_i(\mathbf{p}), \sigma_i^2(\mathbf{p})\right) \right) \right] \tag{3.10}$$

All pixels $\mathbf{p} \in \Omega_{GT}$ with valid ground truth inverse depth values ρ_{GT} are used for training. The outputs of the respective trained network, either DepthNet or CalibNet, are μ_i, λ_i, and $s_i = \log \sigma_i^2$.

The architecture and the separate training process are the essential elements of the recalibration technique. The DepthNet potentially overfits on its training split, which results in higher accurate estimates on the training data than on general data samples. The loss forces the network to output depth distributions that represent the error statistic on the training data – which is not representative and typically overconfident for data samples that are not part of the training data. In contrast to that, the CalibNet is trained to provide a distribution that represents the statistic of the single-view depth estimates on the hold-out calibration split, which is not affected by overfitting effects because it is not used for training the DepthNet. In principle, the CalibNet part could overfit on the hold-out calibration split. However, the mean depth values are not adjusted by the CalibNet and keep the same accuracy. The overfitting effects of the CalibNet part are also substantially lower due to the small amount of input features $2K$ and the number of model parameters $6 \cdot (4K^2 + 2K)$ (e.g. 1632 for $K = 8$).

Ground truth generation: To overcome the limitations of LiDAR data in terms of density, range, and field of view, an intermediate fusion based on stereo images is used for ground truth depth generation. First, the LiDAR point cloud is projected to the image. If there are depth measurements in a local window whose depth difference exceeds a certain threshold, these measurements are marked as inconsistent and removed. This step is included to handle occlusion problems due to the different mounting position and perspective of the LiDAR sensor. Second, these sparse depth maps are completed considering a photometric distance between the two stereo images. Therefore, a depth cost volume $E(\mathbf{p}, d)$ is created spanned over all pixels $\mathbf{p}$ and depth values represented as discretized disparities $d \in [0, 255]$. Each depth value is rated by its photometric distance based on the stereo image pair $E_{stereo}(I_l, I_r, \mathbf{p}, d)$ for each pixel $\mathbf{p}$. For pixels $\mathbf{p}$ with a valid LiDAR measurement, an additional penalty cost $E_{lidar,err}$ is added for depth values different to the LiDAR measurement $d_{lidar}(\mathbf{p})$.

$$E(\mathbf{p}, d) = \begin{cases} E_{stereo}(I_l, I_r, \mathbf{p}, d) + E_{lidar,err} & \text{, if } d_{lidar}(\mathbf{p}) \neq d \\ E_{stereo}(I_l, I_r, \mathbf{p}, d) & \text{, else} \end{cases} \tag{3.11}$$

$E_{lidar,err}$ could represent the probability that the LiDAR measurement is incorrect and overruled by the stereo part. However, because the LiDAR sensor has high robustness and accuracy, the value $E_{lidar,err}$ is chosen very high. Consequently, in practice pixels with valid LiDAR measurements take over the LiDAR depth value and the stereo part serves as a depth completion for pixels without valid LiDAR measurements. The photometric distance is defined in the same way as in [Hirschmuller, 2005] and the same SGM-based approach is applied to compute the depth estimates based on the depth cost volume. For further details, the reader is referred to the corresponding paper [Hirschmuller, 2005]. The presented stereo-based completion of the LiDAR point cloud serves as the ground truth for training. The conversion from ground truth disparities d_{GT} to ground truth inverse depth ρ_{GT} is defined for a calibrated stereo setup with camera constant f and baseline b as $\rho_{GT} = d_{GT}/(f \cdot b)$. The example in figure 3.4 shows that this densifies the sparse point cloud. Even more important, this approach provides depth estimates also for the upper part of the image and at far distances.

ProbDepthNet learns to estimate a pixel-wise depth distribution by observing the depth distribution during the training process. This way, the depth distribution captures the measurement uncertainty. The main contribution of ProbDepthNet is the novel recalibration technique to provide well-calibrated distributions. The experiments in section 3.4.2 validate the recalibration technique and show that it is also applicable to different probabilistic approaches similar to [Ilg et al., 2018, Gast and Roth, 2018].

3.4 Experimental Evaluation of ProbDepthNet Model

In the present section ProbDepthNet is analyzed: (1) Qualitative results of ProbDepthNet are shown, (2) the uncertainty quantification and calibration of different ProbDepthNet variants are analyzed, (3) the generalization capabilities to different outdoor scenes are shown, and (4) the accuracy of the underlying depth estimates is evaluated.

The experiments are conducted on a ProbDepthNet model trained for the KITTI scene flow training set [Menze et al., 2018]. The model is trained on 33 sequences of the KITTI raw dataset [Geiger et al., 2013], which are not part of the scene flow set. Around 75% of the sequences are used for training DepthNet and 25% of the sequences serve as the calibration split for training the CalibNet layers. It is trained for 15 epochs using Adam optimizer [Kingma and Ba, 2014] with a learning rate of 10^{-4} halved every 5 epochs and a small batch size of 4. The input images are scaled to a size of 512 × 256 and a MoG with 8 components is used.

3.4.1 Qualitative Results

Figure 3.6 shows the output of ProbDepthNet for some images of the KITTI scene flow dataset [Menze and Geiger, 2015] in terms of the mean depth value, variance, and recalibrated variance of the first component. The visualizations show that the component is visually representative also for the other components and the distribution in general. The estimated recalibrated variances $\tilde{s}_0$ provided by CalibNet are significantly higher than the variances s_0. This shows that the CalibNet layers reshape the distribution toward less confident distributions. A quantitative evaluation of the resulting recalibration is presented in section 3.4.2. The variances correlate with the depth errors and high variances are typically estimated in the following situations. First, ProbDepthNet estimates high variances correctly for challenging parts such as thin objects (e.g. the poles in figure 3.6 (a,c)) or object boundaries (e.g. the object boundaries of the vehicles in figure 3.6 (a,b,d,e)). The variances are lower for the object boundaries at the bottom than those at the top of the vehicles. This is due to the fact that the difference in depth is larger between the vehicle and the

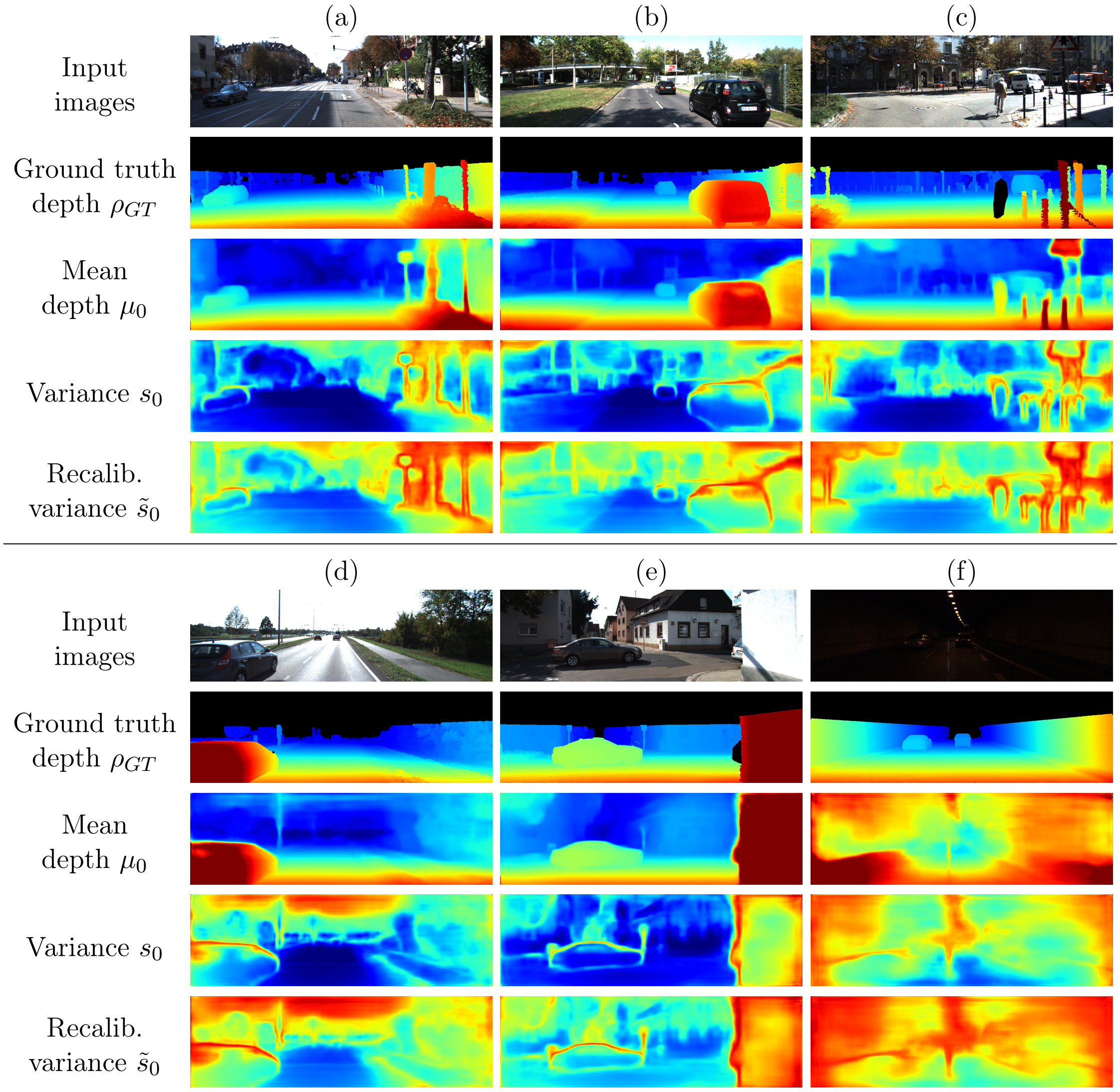

(a)
(b)
(c)
Input images
Ground truth depth ρ_{GT}
Mean depth μ_0
Variance s_0
Recalib. variance $\tilde{s}_0$
(d)
(e)
(f)
Input images
Ground truth depth ρ_{GT}
Mean depth μ_0
Variance s_0
Recalib. variance $\tilde{s}_0$

Figure 3.6: Qualitative results of ProbDepthNet on the KITTI scene flow dataset [Menze and Geiger, 2015] in the form of the mean depth values μ_0, log-variances s_0 and recalibrated log-variances $\tilde{s}_0$ of the first component of the MoG. The color encodes the inverse depth from close (red) to far (blue) or the variance from high (red) to low (blue).

background than between the vehicle and the ground plane. Second, ProbDepthNet is able to estimate high variances for parts that lack valuable ground truth data for training. For example, the stereo-based completion of the LiDAR data does not provide valuable ground truth data for the low-textured sky (see figure 3.6 (d)). Third, ProbDepthNet is able to identify scenarios that result in an erroneously estimated depth structure such as the dark tunnel in figure 3.6 (f). In this scenario, ProbDepthNet estimates high variances correctly for almost the whole image.

3.4.2 Evaluation of Uncertainty Quantification and Calibration

The following experiments analyze the uncertainty quantified by the distributions and the proposed recalibration by adding the CalibNet trained on a hold-out split. Three ProbDepthNet variants are analyzed, each one representing one category of the presented strategies in terms of quantifying the measurement uncertainty (section 3.1.2.1). The proposed training by minimizing the NLL is related to the first category such as in [Kendall and Gal, 2017]. To represent the second category of multi-hypothesis approaches, the DepthNet part is adapted to estimate a set of 8 hypotheses, each one represented by a single Gaussian. Following the proposed loss by Ilg et al. [2018], only the best hypothesis is penalized during training. This variant is denoted as 'Hypo'. [Ilg et al., 2018] also proposed to use a subsequent network to estimate a distribution based on the set of hypotheses. In contrast to the subsequent network proposed by Ilg et al. [2018], CalibNet only consists of a few layers and is additionally designed to provide well-calibrated distributions. The third ProbDepthNet variant corresponds to its 'ADF'-counterpart [Gast and Roth, 2018] representing the third category. Therefore, each layer is replaced by its probabilistic version to propagate an input uncertainty through the whole network as proposed by [Gast and Roth, 2018].

Three metrics are analyzed to assess the quality of the distributions in terms of quantifying the uncertainty and calibration – the sparsification error, the expected calibration error (ECE) and mean NLL.

Sparsification error: The sparsification error reveals how much the estimated uncertainty or variance is related to the error of the estimates. If the estimated variance represents the uncertainty well, the error should monotonically decrease by removing gradually the pixels with the highest uncertainty. In the best case, the pixels with the highest errors are removed, which is denoted as *oracle* sparsification. Formally, the distributions are approximated by its total mean $\hat{\rho}$ and total variance $\hat{\sigma}^2$ to define a single value for the depth estimate and uncertainty. The error is defined as the root mean squared error (RMSE) in terms of inverse depth for a subset $\mathbf{p} \in \Omega_k$ of pixels with the corresponding ground truth values $\rho_{GT}(\mathbf{p})$:

$$RMSE_k = \sqrt{\frac{1}{|\Omega_k|} \sum_{\mathbf{p} \in \Omega_k} \left(\hat{\rho}(\mathbf{p}) - \rho_{GT}(\mathbf{p})\right)^2} \tag{3.12}$$

The subset excludes the estimates with the k-percentage of highest variances $\hat{\sigma}^2$. To measure the difference of the sparsification defined by the estimated variances and

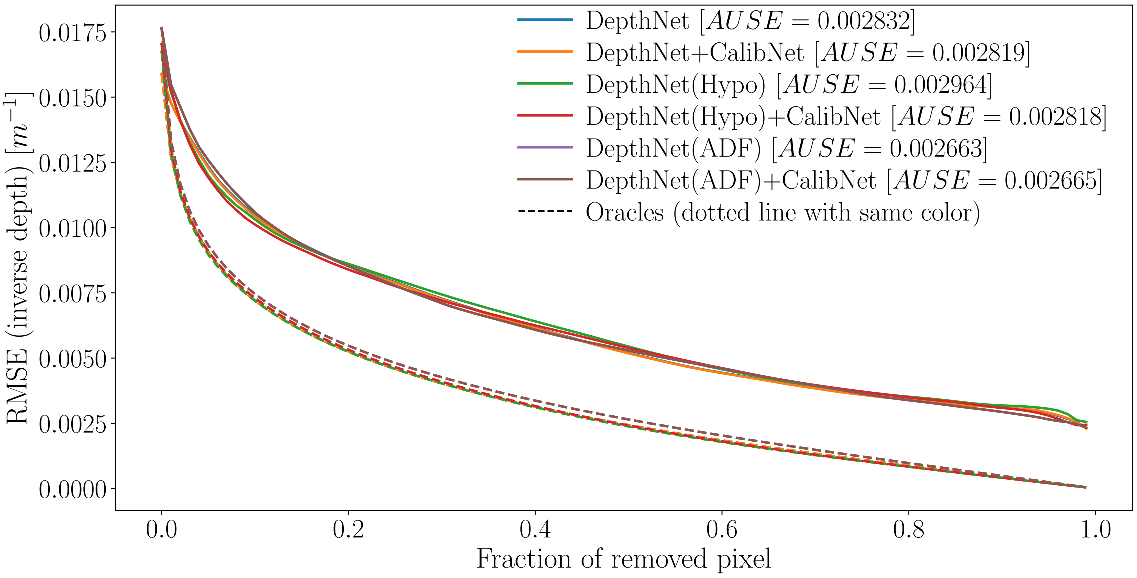


Figure 3.7: Sparsification plots of ProbDepthNet variants. The plot shows the root mean squared error (RMSE) in terms of inverse depth for gradually removing the fraction of pixels with the highest uncertainties. All curves are monotonically decreasing, which shows that the uncertainty of all methods coincides with the errors. The dotted curves correspond to an 'oracle', which removes the best fraction of pixels with the highest errors. The distance to the oracle is expressed as the area under sparsification error (AUSE) error stated in the legend. The different variants show comparable accuracy, which is basically not affected by adding the CalibNet for recalibration.

the oracle sparsification, the area under sparsification error (AUSE) is introduced as $AUSE = \int_{k=0}^{1}(RMSE_k - RMSE_{k-oracle})$ [Ilg et al., 2018].

The sparsification plots and the AUSE errors are shown in figure 3.7. All methods show a monotonically decreasing curve by removing the fraction of pixels with the highest uncertainty. This validates that the variances coincide with the error and uncertainty of the estimates. The different ProbDepthNet variants show similar performance. Adding the CalibNet for recalibration has no significant impact, which shows that CalibNet does not change the order of uncertainties.

Expected calibration error (ECE): The sparsification plot is able to show if higher uncertainties also correspond to higher errors, but it says nothing about the calibration of the uncertainty. For a well-calibrated distribution, the variance or uncertainty also needs to define a reasonable confidence interval. The calibration of the final models is shown in figure 3.8. The frequency of ground truth depth values inside a given interval should be the same as the cumulative probability of the estimated distribution. Analogously to the sparsification metric, the ECE [Guo et al., 2017] is defined as the area between the calibration curve and the ideal calibration. The extend of being overconfident varies among the different approaches – but all approaches suffer from such an effect and provide overconfident estimates.

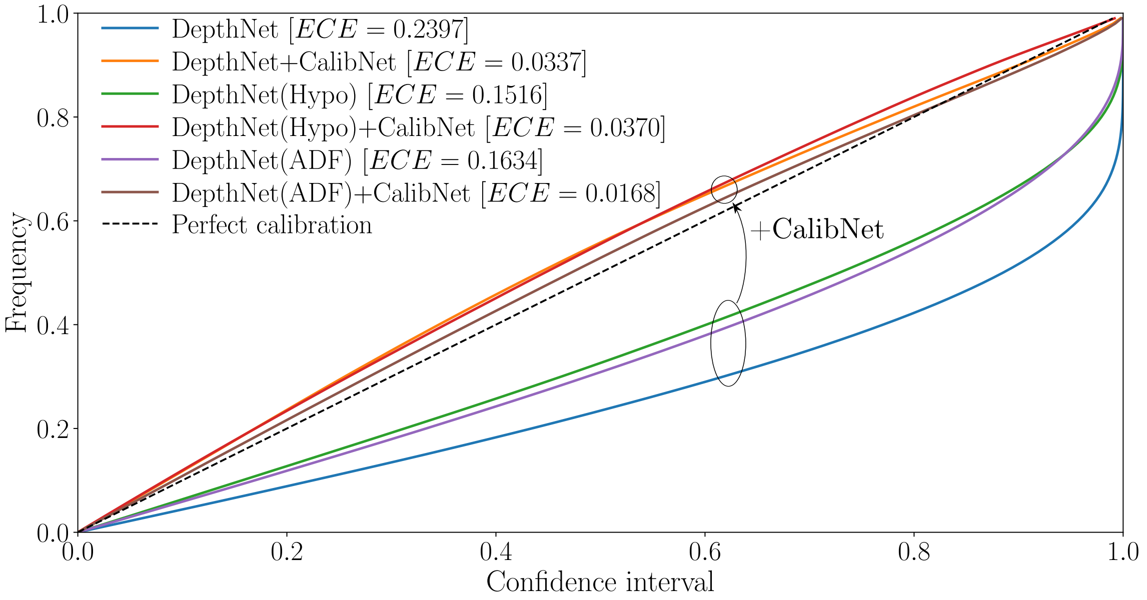


Figure 3.8: Calibration plots of ProbDepthNet variants. The calibration plot analyzes the frequency of ground truth depth values lying in a given confidence interval. This frequency is equal to the confidence interval for a perfect calibrated model (dotted line). The distance to a perfect calibrated model is defined by the ECE metric (stated in the legend). By including CalibNet for recalibration, a much better calibrated model is achieved for all ProbDepthNet variants.

The proposed recalibration by adding CalibNet is able to improve the calibration of the distributions significantly resulting in a decreased ECE of a factor of 5 to 10 depending on the variant as shown in the legend of figure 3.8.

Mean NLL over training process: Evaluating the mean NLL of the models is a metric that directly assesses the accuracy of the distribution. This covers the accuracy of the underlying depth estimates as well as the calibration of the distribution. Figure 3.9 shows the mean NLL on the KITTI scene flow set (which is not part of the training data) every 1000 training steps. This experiment reveals the reason for the overconfident distributions without the CalibNet. As soon as a network starts overfitting, the accuracy of the network gets better on the training data than on the test set. All strategies to cover the measurement uncertainty derive their estimated distributions from the observed accuracy during training. Intuitively, these strategies can not prevent the network to provide overconfident distributions. In contrast to that, the distributions provided by CalibNet are based on the accuracy on a hold-out split. The results show that the accuracy on the hold-out split is much more representative and consequently CalibNet is able to compensate for the overfitting effects to provide well-calibrated distributions. This supports the claim that CalibNet is a useful recalibration technique applicable to different probabilistic approaches.

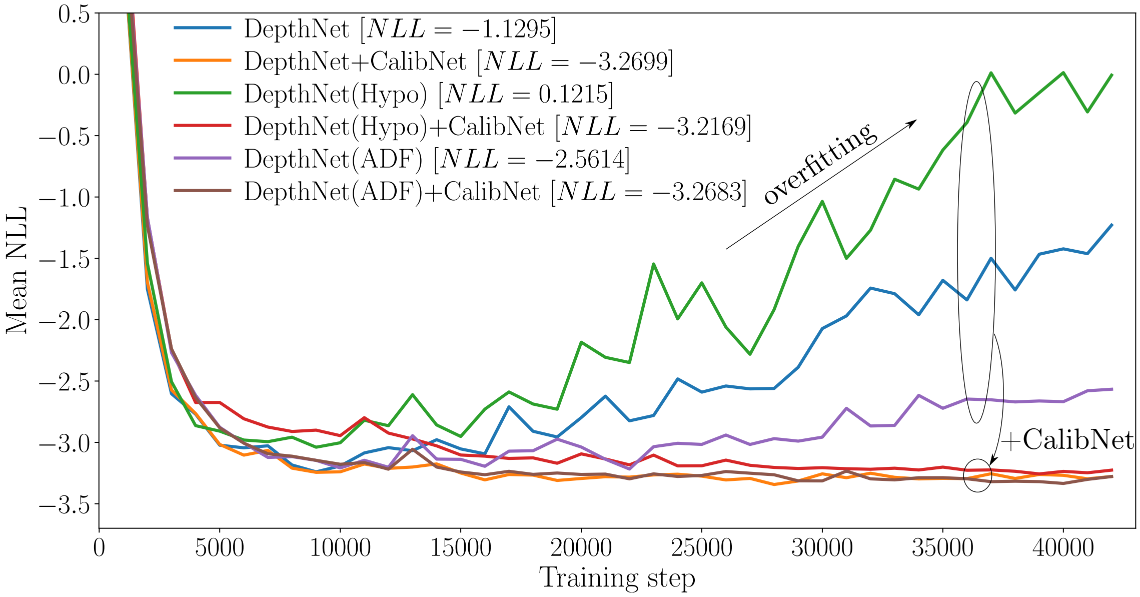


Figure 3.9: Mean NLL of ProbDepthNet variants over the training process. The mean NLL (lower is better) is evaluated on the KITTI scene flow set [Menze et al., 2018], which is not part of the training data. The accuracy decreases at some point for all variants without CalibNet due to an overfitting effect, which results in overconfident estimates. CalibNet is validated as a reasonable recalibration technique to compensate for the overfitting effect.

Comparing the different strategies (minimizing NLL, hypothesis strategy 'Hypo', and assumed density filtering 'ADF'), the experiments show that the main difference is in terms of calibration and being sensitive to overfitting effects. However, CalibNet addresses exactly these points and by including CalibNet for recalibration a similar performance is achieved for the different strategies.

3.4.3 Generalization Capabilities

To analyze the generalization capabilities, ProbDepthNet results are provided for images of different scenes. Figure 3.10 shows the results of a ProbDepthNet model, which is trained on KITTI (with pretraining on Cityscapes [Cordts et al., 2016]) on images of the Make3D dataset [Saxena et al., 2009]. The Make3D dataset comprises images of a wide range of outdoor scenes captured by a hand-held camera. The results are based on a central crop of the Make3D image to provide images with a similar aspect ratio as the training data. The estimates of ProbDepthNet are not only reasonable for street scenes (see figure 3.10 (a,b)), but also for some different scenes (see figure 3.10 (c-f)) such as parks. However, there are also limitations in terms of generalization capabilities. Images that are too different from the training data such as the close-up views of buildings (see figure 3.10 (g,h)) can result in erroneous estimates of both, mean depth values and variances. Note that ProbDepthNet is

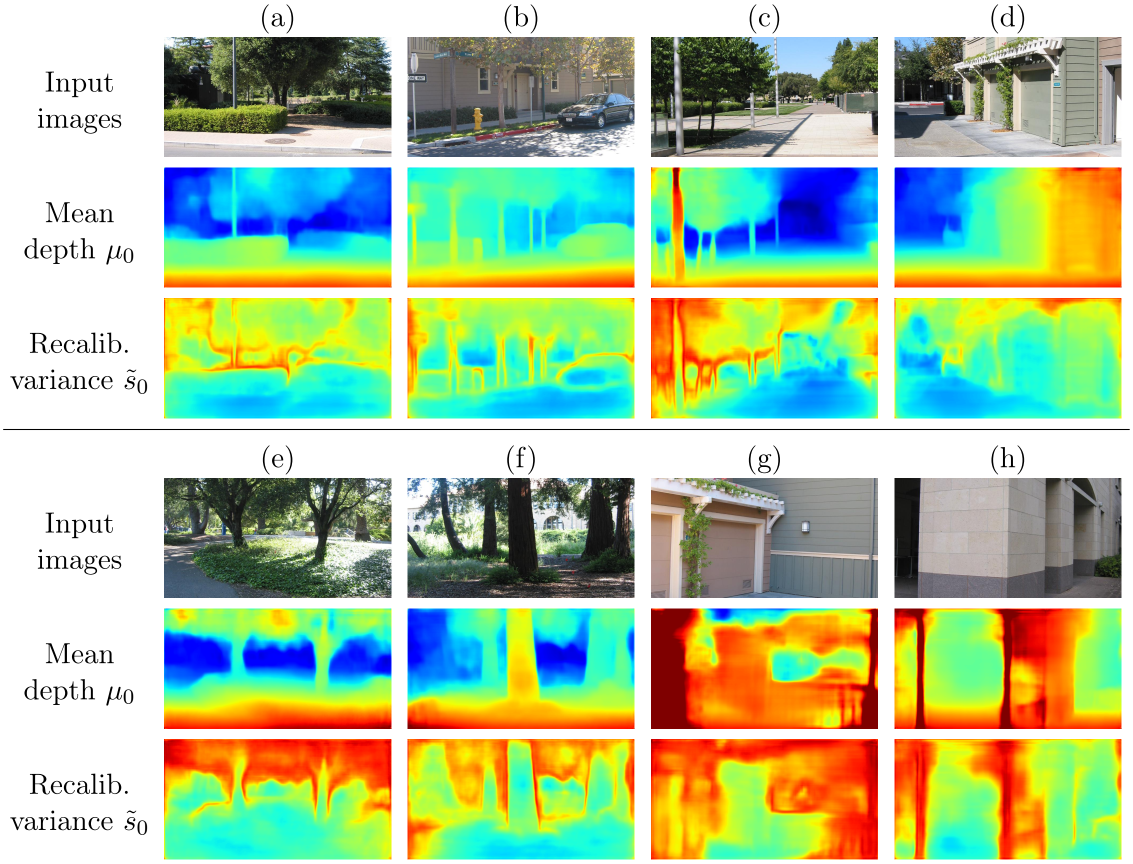


Figure 3.10: Generalization of ProbDepthNet (trained on Cityscapes [Cordts et al., 2016] and KITTI [Menze and Geiger, 2015]) on the central crop of Make3D [Saxena et al., 2009]. The figure shows the estimates based on the input image (top) in the form of the mean depth values μ_0 (middle) and recalibrated log-variances $\tilde{s}_0$ (bottom) of the first component of the MoG. The color encodes the inverse depth from close (red) to far (blue) or the variance from high (red) to low (blue).

designed to capture the measurement uncertainty. Such images as in figure 3.10 (g,h) need to be considered as OOD data and an additional OOD data detection would be needed to estimate correctly the uncertainty in such situations.

3.4.4 Evaluation of Maximum Likelihood Depth Estimation

Previous methods for single-view depth estimation such as [Eigen et al., 2014, Liu et al., 2016, Garg et al., 2016, Godard et al., 2017, Kuznietsov et al., 2017, Fu et al., 2018, Lee et al., 2019] are typically designed to provide a single depth value per pixel that represents the maximum likelihood estimate. In contrast to these methods, the advantage of ProbDepthNet is to provide well-calibrated pixel-wise depth distributions instead of maximum likelihood depth estimates. This probabilistic design is beneficial for combining single-view depth information with multi-view geometry as shown in the following chapters 4 to 6.

However, to get an impression of the underlying accuracy of the depth estimates provided by ProbDepthNet with respect to previous methods, the total means or first moments of the pixel-wise depth distributions are interpreted as estimates of single depth values $\hat{\rho} = \sum_{i=1}^{K} \lambda_i \mu_i$. Table 3.3 shows the quantitative evaluation of these values with respect to several methods for single-view depth estimation following the evaluation metric and KITTI test split proposed by Eigen et al. [2014]:

$$\begin{aligned}
\textbf{Abs Rel} &= \frac{1}{|\Omega_{GT}|} \sum_{\mathbf{p}\in\Omega_{GT}} \frac{|\hat{\rho}(\mathbf{p})^{-1} - \rho_{GT}(\mathbf{p})^{-1}|}{\rho_{GT}(\mathbf{p})^{-1}} \\
\textbf{Sq Rel} &= \frac{1}{|\Omega_{GT}|} \sum_{\mathbf{p}\in\Omega_{GT}} \frac{\left(\hat{\rho}(\mathbf{p})^{-1} - \rho_{GT}(\mathbf{p})^{-1}\right)^2}{\rho_{GT}(\mathbf{p})^{-1}} \\
\textbf{RMSE} &= \sqrt{\frac{1}{|\Omega_{GT}|} \sum_{\mathbf{p}\in\Omega_{GT}} \left(\hat{\rho}(\mathbf{p})^{-1} - \rho_{GT}(\mathbf{p})^{-1}\right)^2} \qquad (3.13) \\
\textbf{RMSE}_{log} &= \sqrt{\frac{1}{|\Omega_{GT}|} \sum_{\mathbf{p}\in\Omega_{GT}} \left(\log \hat{\rho}(\mathbf{p})^{-1} - \log \rho_{GT}(\mathbf{p})^{-1}\right)^2} \\
\boldsymbol{\gamma < 1.25^k} &: \ \% \text{ of } \mathbf{p} \text{ with } \max\left(\frac{\hat{\rho}(\mathbf{p})}{\rho_{GT}(\mathbf{p})}, \frac{\rho_{GT}(\mathbf{p})}{\hat{\rho}(\mathbf{p})}\right) = \gamma < 1.25^k
\end{aligned}$$

$\mathbf{p} \in \Omega_{GT}$ are all pixels with valid ground truth $\rho_{GT}(\mathbf{p})$. While the estimated depth by ProbDepthNet is represented as inverse depths, the proposed metric by Eigen et al. [2014] evaluates the depth in the linear or logarithmic space. Furthermore, the depth values are capped at 50 or 80 meters.

The results are based on a ProbDepthNet model pretrained on Cityscapes [Cordts et al., 2016] and fine-tuned on the KITTI training split specified by Eigen et al. [2014]. Additionally, the results of the following baseline methods are provided: [Eigen et al., 2014, Liu et al., 2016, Garg et al., 2016, Godard et al., 2017, Kuznietsov et al., 2017, Fu et al., 2018, Lee et al., 2019]. A more detailed description of these methods is given in section 3.1.1.3. The results stated in Table 3.3 are taken from the corresponding papers except for the DORN [Fu et al., 2018] method. While the other methods evaluate their depth estimates against the raw LiDAR point cloud, the DORN method was evaluated against the ground truth data provided by the KITTI depth prediction benchmark [Uhrig et al., 2017]. To provide a fair comparison, the published estimates of the DORN method are evaluated against the raw LiDAR point cloud as ground truth for this evaluation.

The accuracy of the estimates of the DORN [Fu et al., 2018] and BTS [Lee et al., 2019] methods are superior to ProbDepthNet. However, even though ProbDepthNet is focused on providing depth distributions, it is also comparative to methods

Method	Cap	lower is better				higher is better		
		Abs Rel	**Sq Rel**	**RMSE**	**RMSE**$_{log}$	$\gamma < \mathbf{1.25}$	$\gamma < \mathbf{1.25^2}$	$\gamma < \mathbf{1.25^3}$
Eigen [Eigen et al., 2014]	80m	0.190	1.515	7.156	0.270	69.2	89.9	96.7
Liu et al. [Liu et al., 2016]	80m	0.217	1.841	6.986	0.289	64.7	88.2	96.1
LRC [Godard et al., 2017]	80m	0.114	0.898	4.935	0.206	86.1	94.9	97.6
Kuznietsov [Kuznietsov et al., 2017]	80m	0.113	0.741	4.621	0.189	86.2	96.0	**98.6**
DORN [Fu et al., 2018]	80m	0.111	$\underline{0.618}$	**3.659**	**0.168**	$\underline{89.4}$	$\underline{96.4}$	$\underline{98.4}$
BTS [Lee et al., 2019]	80m	**0.091**	**0.555**	$\underline{4.033}$	$\underline{0.174}$	**90.4**	**96.7**	$\underline{98.4}$
ProbDepthNet	80m	$\underline{0.103}$	0.762	4.680	0.195	87.1	95.3	97.9
Garg [Garg et al., 2016]	50m	0.169	1.080	5.104	0.273	74.0	90.4	96.2
LRC [Godard et al., 2017]	50m	0.108	0.657	3.729	0.194	87.3	95.4	97.9
Kuznietsov [Kuznietsov et al., 2017]	50m	0.108	0.595	3.518	0.179	87.5	96.4	**98.8**
DORN [Fu et al., 2018]	50m	0.108	$\underline{0.535}$	**2.884**	**0.162**	$\underline{90.2}$	$\underline{96.6}$	98.5
BTS [Lee et al., 2019]	50m	**0.088**	**0.437**	$\underline{3.127}$	$\underline{0.165}$	**91.4**	**97.0**	$\underline{98.6}$
ProbDepthNet	50m	$\underline{0.098}$	0.567	3.530	0.183	88.3	95.9	98.1

Abs Rel [$\frac{m}{m}$], ***Sq Rel***[$\frac{m^2}{m}$]: *absolute and squared relative depth error;* ***RMSE*** [m]: *RMSE of depth*
RMSE$_{log}$: *RMSE of logarithmic depth;* $\gamma < \mathbf{1.25^k}$[%]: *percentage fulfilling a quality threshold*

Table 3.3: Quantitative evaluation of methods for single-view depth estimation on the KITTI dataset [Geiger et al., 2013] using the test split by Eigen et al. [2014].

such as LRC [Godard et al., 2017], which uses a similar network architecture. The experiments in sections 4.3.2 and 5.3.2 show a significantly better accuracy for integrating single-view depth estimates in a monocular scene flow method using the probabilistic ProbDepthNet instead of the LRC method. In terms of estimating a maximum likelihood depth value, the quality of ProbDepthNet is just similar or slightly superior to the LRC method. This additionally supports the claim that the main benefits of ProbDepthNet are due to the probabilistic design and due to providing well-calibrated distributions.

3.5 Conclusion

The present chapter addressed the topic to analyze and estimate the uncertainties or distributions in the context of single-view depth estimation. The first section analyzed the type of distribution as well as the dependence on semantic classes. The second section presented ProbDepthNet, a CNN for probabilistic single-view depth estimation. ProbDepthNet provides depth distributions instead of merely maximum likelihood depth estimates. The experiments reveal that previous methods for estimating the measurement uncertainty in a regression problem suffer from overconfident estimates – an effect that is compensated by a novel recalibration technique. The experiments also confirm a reasonable accuracy of the underlying depth estimates. Even though the ProbDepthNet provides well-calibrated distributions covering the measurement uncertainty, the experiments also illustrated that a quantification of the measurement uncertainty reaches its limits for images that are too different from the training data (e.g. close-up views of buildings). In such cases, neither the depth nor the uncertainty is guaranteed to be reasonable. Such images need to be considered as OOD data, which need special treatment to be detected. This detection was out of the scope for ProbDepthNet, but is clearly important to increase the robustness of probabilistic single-view depth estimates against unknown scenarios.

4 SINGLE-VIEW DEPTH MEETS MULTI-BODY STRUCTURE FROM MOTION

CONTENTS

This chapter extends parts of the work that has been published previously in [Brickwedde et al., 2019].

In applications such as mobile robots or autonomous vehicles, a dynamic representation of the surrounding environment is needed. In addition to depth estimates, the motion of the camera and other traffic participants is important. For example, dynamic parts of the scene need to be detected and tracked to navigate safely through a scene and to avoid collisions.

From a computer vision point of view, the 3D position and motion of a pixel in the image is denoted as *3D scene flow* [Vedula et al., 1999, 2005]. As distances are a crucial element of scene flow, it is estimated based on a temporal series of stereo images [Vogel et al., 2013, Menze and Geiger, 2015, Behl et al., 2017]. However, from an economic point of view, monocular camera systems are often preferred over stereo cameras due to being more cost-efficient and to avoid the effort of calibrating the stereo rig.

3D scene flow estimation is an ill-posed problem in terms of multi-view geometry in a monocular camera setup. To solve the scale-ambiguity, previous monocular methods assumed that the moving objects are in contact with the surrounding environment [Ranftl et al., 2016, Song and Chandraker, 2015, 2014, Yuan and Medioni, 2006] or that the scene follows a smoothness prior regarding surface and motion [Mitiche et al., 2015, Xiao et al., 2017, Kumar et al., 2017, Di et al., 2019]. These assumptions might be violated, e.g. if the ground contact point of a moving vehicle is occluded. Furthermore, these methods still require a relative translational motion of the camera to the scene and they need an accurate estimate of the road ground plane.

The previous chapter 3 showed that convolutional neural networks (CNNs) are able to provide depth estimates from a single image at a reasonable level of quality. However, single-view depth estimation and multi-view geometry are mostly tackled as two individual tasks or fused in a way that is only applicable to static scenes [Tateno et al., 2017, Fácil et al., 2017, Yin et al., 2017].

Therefore, I propose in the present chapter an approach that combines *multi-view geometry* with *single-view depth information* to exploit both kinds of information for a monocular scene flow estimation problem, which results in new state of the art (SotA) accuracy. Even more, the single-view depth serves to solve the multi-view geometry-based ambiguity and the method generalizes to standstill scenarios. The proposed approach is denoted as *SVD-MSfM* because it is an approach that integrates probabilistic single-view depth estimates (SVD) in a multi-body structure from motion (MSfM)-based approach.

An overview of SVD-MSfM and the corresponding sections is shown in figure 4.1. In the first step (section 4.2.1), the camera motion and the motion of objects detected by an instance segmentation are estimated. The motion estimation is based on a sparse optical flow field and integrates single-view depth distributions provided by ProbDepthNet. The single-view depth distributions can be considered as prior depth information that supports the matching process of sparse optical flow estimation and optimization of motion estimation. The motions are expressed by all 6 degrees of freedom. Due to the integration of single-view depth information, the motion estimates are also provided in the correct metric scale. Traditionally, the known

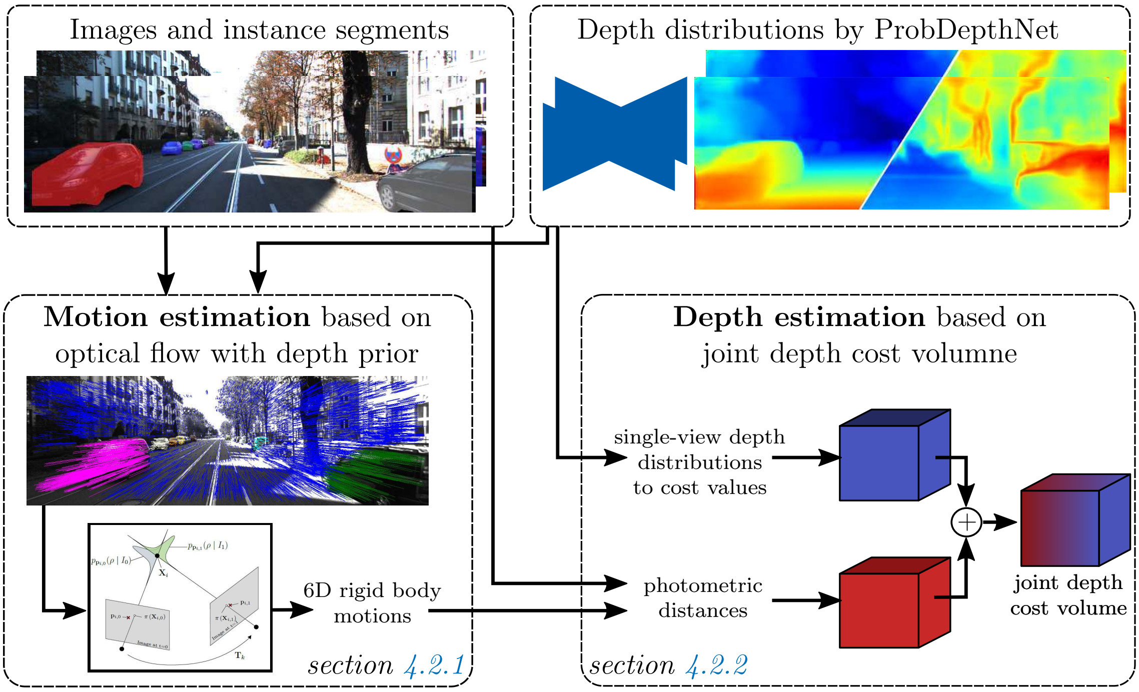

Images and instance segments
Depth distributions by ProbDepthNet
Motion estimation based on optical flow with depth prior
6D rigid body motions
section 4.2.1
Depth estimation based on joint depth cost volumne
single-view depth distributions to cost values
photometric distances
joint depth cost volume
section 4.2.2

Figure 4.1: Overview of SVD-MSfM. The proposed approach is divided into two steps. The motion of objects detected by an instance segmentation as well as the camera motion are estimated based on a sparse optical flow field [section 4.2.1]. The integration of the single-view depth distribution can be considered as a depth prior for matching and motion optimization. A dense depth map is estimated based on a depth cost volume, which comprises the single-view depth information and a multi-view photometric distance [section 4.2.2].

camera height or an additional inertial measurement unit is used for scale-aware monocular visual odometry in the automotive domain. However, this only provides scale information for the camera motion. In contrast to [Barnes et al., 2018, Yin et al., 2017, Yang et al., 2018a] that integrate single-view depth estimates for scale-aware camera motion estimation, this idea is applied here additionally for scale-aware motion estimation of moving objects. Most methods exploit the possibility to track the camera motion over a long time and evaluate the results on long sequences. To be closer to the characteristic of moving objects that frequently appear and disappear, the evaluation in the present thesis is based on short sequences. While the evaluation shows that the accuracy of most methods significantly drops on short sequences, SVD-MSfM has high robustness even on short sequence snippets.

In the second step (section 4.2.2), the depth is estimated based on a depth cost volume spanned over each pixel and discretized depth values. The depth cost volume combines two kinds of information. Cost values derived from the depth distributions provided by ProbDepthNet serve as the first part and integrate the single-view depth information. Based on the scale-aware motion estimates and pair of monocular images, the photometric distances rate each discretized depth value for each pixel and contribute the multi-view geometric information. Standard techniques, SGM [Hirschmuller, 2005] and SPS [Yamaguchi et al., 2014], are applied to derive final depth estimates based on the joint depth cost volume.

The experiments evaluate different categories of monocular baseline methods for scene flow estimation and show that SVD-MSfM outperforms previous methods significantly on a scene flow metric. Furthermore, ablation studies confirm the suitability of subcomponents and design choices such as (1) the proposed motion estimation, (2) the combination of multi-view and single-view information, and (3) ProbDepthNet for integrating single-view depth information. The advantage of ProbDepthNet can be attributed to the importance of providing single-view depth information in a probabilistic and well-calibrated form.

4.1 Related Work

Previous work related to SVD-MSfM is categorized into three groups. The first category comprises simultaneous localization and mapping (SLAM) methods that estimate the camera poses and a map of the static environment. These works are mostly related to the formulation of SVD-MSfM motion estimation. The second category comprises the methods based on the multi-body structure from motion (MSfM) principle. While the methods of the first category provide a reconstruction of the static environment, the second category consists of methods that provide motion and depth estimates for dynamic scenes. These methods are related to the general concept of SVD-MSfM and serve as one category of baseline methods. Multi-task networks that provide a monocular scene flow representation, e.g. in terms of depth and optical flow or depth and motion estimates, are considered as the third category and an additional group of baselines.

4.1.1 Simultaneous Localization and Mapping

SLAM methods jointly estimate the camera poses and a set of 3D map points of the static environment. The approaches differ regarding certain aspects: (1) the type of the estimation ('filtering vs. optimization'), (2) the definition of the optimized data terms ('direct vs. indirect'), (3) the density of the map points ('dense vs. sparse'), and (4) the applied constraints to derive metric scale-aware estimates ('scale estimation'). A similar categorization is proposed in [Engel, 2017]. Table 4.1 provides an overview of monocular SLAM methods and their classifications. D3VO [Yang et al., 2020]

Method	Filtering vs. Optimization	Direct vs. Indirect	Sparse vs. Dense	Scale estimation
PTAM [Klein and Murray, 2007]	BA	indirect	sparse	No
DTAM [Newcombe et al., 2011]	Filtering	direct	dense	No
SVO [Forster et al., 2014]	Filtering	hybrid	sparse	No
VISO2-M [Geiger et al., 2011]	Filtering	indirect	sparse	Ground Plane
LSD-SLAM [Engel et al., 2014]	BA+LC	direct	semi-dense	No
ORB-SLAM [Mur-Artal et al., 2015]	BA+LC	indirect	sparse	No
DSO [Engel et al., 2017]	BA	direct	sparse	No
LDSO [Gao et al., 2018]	BA+LC	direct	sparse	No
PMO [Fanani et al., 2017]	Filtering	direct	sparse	Ground Plane
CNN-SLAM [Tateno et al., 2017]	BA	direct	dense	Single-View Depth
Yin et al. [Yin et al., 2017]	BA	indirect	sparse	Single-View Depth
Barnes et al. [Barnes et al., 2018]	Filtering	indirect	dense	Single-View Depth
DVSO [Yang et al., 2018a]	BA	direct	sparse	Single-View Depth
D3VO [Yang et al., 2020]	BA	direct	sparse	Single-View Depth

***BA**: bundle adjustment ; **LC**: loop closing*

Table 4.1: Overview and classification of SLAM methods. The methods are categorized according to four aspects: (1) filtering vs. optimization, (2) direct vs. indirect, (3) sparse vs. dense, and (4) scale estimation.

is the currently leading approach[1] as reported by the KITTI odometry benchmark [Geiger et al., 2012]. The general concepts that are the basis for the classification are explained in the following paragraphs.

Filtering vs. optimization: Formally, SLAM methods estimate a set of camera poses $\mathcal{T} = \{\mathbf{T}_0, ...\mathbf{T}_N\}$ with $\mathbf{T}_j \in SE(3)$ and 3D scene points $\mathcal{X} = \{\mathbf{X}_0, ..., \mathbf{X}_M\}$ with $\mathbf{X}_i \in \mathbb{R}^3$. The estimation problem is typically considered as a maximum likelihood estimation based on observations $\mathbf{Y}$:

$$\hat{\mathcal{T}}, \hat{\mathcal{X}} = \arg\max_{\mathcal{T},\mathcal{X}} p\left(\mathbf{Y} \mid \mathcal{T}, \mathcal{X}\right) \tag{4.1}$$

The observations $\mathbf{Y}$ are, for example, tracked image positions of the 3D scene points.

[1] Referring to monocular methods with a publication submitted to the benchmark until Januar 02, 2021.

Filtering-based methods, on the one hand, continuously predict and update the probability density of the current camera pose and observed scene points over time, e.g. [Geiger et al., 2011, Fanani et al., 2017]. *Optimization*-based methods, on the other hand, formulate the joint estimation of camera poses and the 3D scene points as an energy minimization problem based on a factor graph, e.g. [Dellaert et al., 2017, Mur-Artal et al., 2015, Engel et al., 2017]. Each vertex corresponds to either a camera pose $\mathbf{T}_j$ or a 3D scene point $\mathbf{X}_i$. Each edge describes the observation of a 3D scene point at a certain camera pose. The optimization is formulated as an energy minimization problem, where each observation $\mathbf{Y}_{i,j}$ contributes to the energy term E by $\Phi(\mathbf{Y}_{i,j}, \mathbf{X}_i, \mathbf{T}_j)$:

$$E = \sum_j \sum_i \Phi(\mathbf{Y}_{i,j}, \mathbf{X}_i, \mathbf{T}_j) \tag{4.2}$$

Based on this formulation, a set of camera poses is jointly optimized, which is denoted as *bundle adjustment* (BA). To reduce the computational effort, typically only a subset of camera poses is optimized instead of the whole camera trajectory. The subset is either defined by a sliding window approach [Engel et al., 2017] or by systematically chosen keyframes [Mur-Artal et al., 2015]. Furthermore, the camera might return to a previously visited location, which corresponds to a *loop* in the pose graph. Approaches have been proposed [Mur-Artal et al., 2015, Gao et al., 2018] to detect these loop closings (LCs) and optimize the camera trajectory to be consistent with the detected loop.

Direct vs. indirect: This classification is based on the definition of the observations $\mathbf{Y}$ and the optimized energy term $\Phi(\mathbf{Y}_{i,j}, \mathbf{X}_i, \mathbf{T}_j)$. Two types of methods are distinguished: *direct* and *indirect* methods. While the optimization of indirect methods is based on detected and tracked scene point image coordinates, direct methods perform the optimization on image intensities. For indirect methods, the detected scene point image coordinates $\mathbf{Y}$ serve as the measurements for the error term $\Phi(\mathbf{Y}_{i,j}, \mathbf{X}_i, \mathbf{T}_j)$. The set of 3D scene points corresponds to extracted feature keypoints, which are matched and tracked over the image sequence. For each 3D scene point $\mathbf{X}_i$, a set of measured image positions $\mathbf{Y}_{i,j} = \mathbf{p}_{i,j}$ corresponding to a certain pose $\mathbf{T}_j$ is derived. These measured image positions are considered as the observations. The error term $\Phi^{proj}(\mathbf{p}_{i,j}, \mathbf{X}_i, \mathbf{T}_j)$ quantifies the distance between the measured $\mathbf{p}_{i,j}$ and reprojected 2D keypoint image position $\pi(\mathbf{X}_i, \mathbf{T}_j)$ [Klein and Murray, 2007, Mur-Artal et al., 2015]:

$$\begin{aligned} E &= \sum_j \sum_i \Phi^{proj}(\mathbf{p}_{i,j}, \mathbf{X}_i, \mathbf{T}_j) \\ &= \sum_j \sum_i ||\mathbf{p}_{i,j} - \pi(\mathbf{X}_i, \mathbf{T}_j)|| \end{aligned} \tag{4.3}$$

For the norm $||\cdot||$ usually the Euclidian- or more rarely the Mahalanobis-distance is used.

While indirect methods minimize a geometric reprojection error, direct methods skip the intermediate feature matching and minimize a photometric distance directly on the image intensities. Each 3D scene point $\mathbf{X}_i$ refers to a reference pixel $\mathbf{p}_i$ in a reference image I_{ref}. The camera poses $\mathbf{T}_j$ and 3D scene points $\mathbf{X}_i$ are optimized to align the reference image intensities $I_{ref}(\mathbf{p}_i)$ with the image intensities $I_j(\pi(\mathbf{X}_i, \mathbf{T}_j))$:

$$\begin{aligned} E &= \sum_j \sum_i \Phi^{photo}(I_{ref}, I_j, \mathbf{p}_i, \mathbf{X}_i, \mathbf{T}_j) \\ &= \sum_j \sum_i ||I_{ref}(\mathbf{p}_i) - I_j(\pi(\mathbf{X}_i, \mathbf{T}_j))|| \end{aligned} \tag{4.4}$$

Consequently, the images are directly considered as the observations $\mathbf{Y}$. The corresponding image patches could be rated as a weighted sum of squared distances [Fanani et al., 2017, Engel et al., 2017]. Furthermore, the energy term can be extended to consider different exposure times [Engel et al., 2017, Bergmann et al., 2018] and rolling shutter effects [Schubert et al., 2018].

Dense vs. sparse: The third criterion distinguishes the methods based on the density of the reconstructed map. It basically refers to the definition of the set of 3D scene points $\mathcal{X}$. The reprojection (equation (4.3)) and photometric distance (equation (4.4)) are based on local feature matching or local image intensities. Thus, only a *sparse* subset of local image patches with sufficient structure is suitable for optimization. Engel et al. [2014] proposed to use all parts of the image with a minimum length of the image gradient vector, which is denoted as *semi-dense*. However, for a *dense* reconstruction, the energy term needs to be extended by a regularization term [Newcombe et al., 2011] or by exploiting single-view depth estimates [Barnes et al., 2018, Tateno et al., 2017].

Scale estimation: In a monocular setup, the translational motion of the camera and the scene points are only known up to an unknown scale. An additional constraint needs to be applied to solve this *scale ambiguity*. For a vehicle-mounted camera, the *known camera height* above the ground is exploited as metric scale information [Fanani et al., 2017, Geiger et al., 2011]. More recently, single-view depth estimates are used to provide the scale information [Yin et al., 2017, Tateno et al., 2017, Yang et al., 2018a, Barnes et al., 2018, Yang et al., 2020], which is learned based on a stereo or ground truth supervision. Yin et al. [2017] proposed a scale estimation and correction based on *single-view depth estimates* as a subsequent postprocessing step to a not scale-aware monocular SLAM method based on ORB-SLAM [Mur-Artal et al., 2015]. Alternatively, Barnes et al. [2018] proposed to directly integrate the single-view depth estimates d_i as additional measurements in the energy minimization problem:

$$E = \sum_j \sum_i \Phi^{proj}(\mathbf{X}_i, \mathbf{T}_j) + ||d_i - d(\mathbf{X}_i, \mathbf{T}_j)|| \tag{4.5}$$

These methods are mostly related to the proposed motion estimation of SVD-MSfM, which would be categorized as follows: (1) SVD-MSfM is formulated as an energy minimization problem based on a factor graph ('optimization'), (2) a general sparse optical flow field is considered as the measurements ('indirect'), (3) the estimation is based on a sparse set of scene points ('sparse'), and (4) single-view depth estimates are exploited for scale estimation. The main contribution of SVD-MSfM's motion estimation is its applicability to moving objects. The experiments show that SVD-MSfM provides robust estimates on short image sequences up to only two consecutive images. This is an important characteristic as moving objects frequently appear and disappear in the scene and require a fast recognition and motion estimation without having a long history of the trajectory.

4.1.2 Multi-Body Structure from Motion

The methods from the literature presented in section 4.1.1 are focused on static environments and merely estimate the camera motion. Some methods address the robustness against moving objects so that they do not have a negative effect on the camera motion estimation. However, estimating the motion of the moving objects itself is out of the scope of these works.

The present subsection describes works that extended the general concepts of SLAM or structure from motion (SfM) to *dynamic scenes*. Therefore, parts of the scene that undergo a different motion need to be reconstructed individually. Such methods are denoted as multi-body structure from motion (MSfM) and can typically be divided into three steps: (1) *independent moving object* (IMO) detection and segmentation, (2) multi-body motion estimation and reconstruction, and (3) scale estimation.

Independent moving object detection (IMO) and segmentation: The objective of the independent moving object (IMO) detection and segmentation is to identify parts of the image that belong to the same rigid-body motion.

Assuming that the camera motion is given (e.g. by a traditional SLAM method), several *geometric cues* based on *optical flow* estimates have been proposed to identify IMOs. Each optical flow vector belonging to the static environment needs to fulfill the epipolar geometry (section 2.1). If the distance of the optical flow vector to the epipolar line exceeds a certain threshold, it is considered as belonging to an IMO [Klappstein, 2008, Kundu et al., 2010, 2011]. The trifocal constraint [Hartley, 1997, Klappstein, 2008] is defined based on the geometric relation in three views. It states that the triangulated 3D point based on two views needs to be consistent with the feature position in the third view. Fanani et al. [2018] proposed a circular check to detect violations of the trifocal constraint. These constraints detect IMOs with non-colinear translational motion. Additionally, the positive depth constraint [Klappstein, 2008] defines that a triangulated 3D scene point needs to be in front of the camera. This is equivalent to the constraint that the optical flow vector needs to point away from the focus of expansion. These constraints are based on the principles of multi-view geometry without loss of generality. Additionally, constraints have been proposed that are based on scene model assumptions. The positive height constraint [Klappstein, 2008] requires that a 3D scene point lies on or above the ground plane. Kundu et al. [2010, 2011] proposed to define a range of plausible values for the depth or the length of optical flow vectors.

The previous constraints for IMO detection are based on a given camera motion. Alternatively, IMO segmentation is formulated as a *model selection* problem. The scene is divided into a set of motions and each optical flow vector is assigned to one motion model. The fundamental matrix [Ranftl et al., 2016], the trifocal tensor [Vidal and Hartley, 2008] or a 2-DoF planar motion model [Sabzevari and Scaramuzza, 2016] is used to represent the motion. For the segmentation, which means finding the number of motion segments, the motion parameters per segment, and associate each optical flow vectors to one motion segment, Ranftl et al. [2016], Sabzevari and Scaramuzza [2016] proposed iterative methods. The motion parameters for one motion segement are estimated using a robust estimator such as RANSAC [Sabzevari and Scaramuzza, 2016] or median-least-squares [Ranftl et al., 2016]. The optical flow vectors are classified as inlier and outlier for the estimated motion parameters. Iteratively, new motion segments are created and estimated for the remaining optical flow outliers.

Limitations of the geometric constraints exist for *epipolar-conformant* motion. A common example for traffic scenes is an oncoming object with translational motion *collinear* to the camera motion. The optical flow of these objects is the same as for a static object closer to the camera and is therefore not distinguishable from a geometric point of view.

In addition to the geometric considerations, deep learning-based inputs are exploited for IMO detection and segmentation. First, an *instance segmentation* is used to identify the set of potentially moving objects, e.g. vehicles or pedestrians [Bai et al., 2016, Fanani et al., 2018, Fanani et al., 2018, Bullinger et al., 2017]. Second, Fanani et al. [2018] proposed the additional constraint that a triangulated 3D scene point needs to be consistent with a corresponding *single-view depth estimate*.

Multi-body motion estimation and reconstruction: Given the IMO detection and segmentation, the traditional concepts of SLAM or SfM are applied to reconstruct each rigid body individually. The relative motion of the camera to the IMO is interpreted as a *virtual camera motion*. For example, BA-based optimization [Yuan and Medioni, 2006, Kundu et al., 2010, Bullinger et al., 2018, Chhaya et al., 2016] or a particle filter-based approach [Kundu et al., 2011] is used for reconstructing the IMO trajectory and position. Additionally, Yuan and Medioni [2006], Sabzevari and Scaramuzza [2016] proposed to exploit specific object motion models, in particular, that vehicle motion is perpendicular to the normal vector of the ground plane.

Scale estimation: Analogously to SLAM methods for the static environment, the *absolute* and even more the *relative scales* of the reconstructions are *ambiguous* in a MSfM-based approach. While the scale of the camera ego-motion might be derived by the known camera height or an additional inertial measurement unit, these approaches are not applicable for IMOs. To derive the absolute scale of an IMO, previous methods integrate assumptions regarding the objects' depth, velocity, and size [Kundu et al., 2011] or fit object shapes in the 3D point cloud [Chhaya et al., 2016].

However, most approaches tackle the scale ambiguity by estimating the relative scale to the static environment. IMOs typically stand on the *ground plane*, which provides the IMO position relative to the static environment. Song and Chandraker [2015, 2014] proposed to triangulate the bottom of a 2D bounding box with the ground plane. Yuan and Medioni [2006] proposed to scale the 3D points of the IMO such that one point lies on and the rest above the ground plane. Ranftl et al. [2016] proposed to scale the different reconstructions considering an ordering and smoothness prior. The ordering prior refers to the assumption that an IMO typically occludes the static environment, while the smoothness prior prefers that the dynamic objects are in contact with the surrounding environment. These approaches require to observe and accurately detect the ground contact point and assume a highly accurate reconstruction of the ground surface.

Other approaches consider the whole *object trajectory* for scale estimation. The trajectory of the relative object motion $\tilde{\mathbf{t}}_{oc}$ can be estimated up to the unknown scale s. A one-parameter family of possible object trajectories $\mathbf{t}_o(s)$ is defined by applying different scales s [Ozden et al., 2004]:

$$\mathbf{t}_o(s) = s \cdot \tilde{\mathbf{t}}_{oc} + \mathbf{t}_c \tag{4.6}$$

The camera trajectory $\mathbf{t}_c$ is assumed to be known. Assuming that the true scale is $s = 1$, the relation of the reconstructed trajectory to the true object and camera trajectories is defined as follows:

$$\mathbf{t}_o(s) = s \cdot \mathbf{t}_o + (1 - s) \cdot \mathbf{t}_c \tag{4.7}$$

This equation reveals an increased coupling of the camera and object motion at false scales. Therefore, Ozden et al. [2004] proposed to optimize the scale s such that it minimizes the linear dependence of camera and object trajectory. Furthermore, the *non-accidental* criterion [Ozden et al., 2004] has been proposed. If for a certain scale the object undergoes a special motion, there is a high probability that this is not by sheer accident. For example, the scale s is optimized such that the IMO trajectory corresponds to a planar motion [Ozden et al., 2004], a piecewise circular motion [Namdev et al., 2013] or such that the IMO moves on a ground plane [Bullinger et al., 2018]. However, a dependency between camera and object motion could result in *degenerated situations*. For example, (1) if both motions are coplanar, all IMO trajectories of the one-parameter family will be planar or (2) if the true camera and IMO trajectories are collinear, all possible IMO trajectories will be collinear to these trajectories as well.

3D jigsaw puzzle: Kumar et al. [2017] proposed a slightly different approach, which they denoted as a *3D jigsaw puzzle* problem. In contrast to the other methods that utilize object-level motion segmentation, a *superpixel segmentation* is performed and it is assumed that each superpixel corresponds to a *rigid planar element* of the scene. The joint optimization of the 3D geometry and motion of each superpixel plane is formulated as an energy minimization problem. The energy terms are defined (1) to favor a smooth motion and structure of neighboring superpixels and (2) to minimize a reprojection error based on a dense optical flow field. Since only small superpixels are assumed to be rigid, this method is also considered as a non-rigid SfM method. This concept is extended in [Kumar et al., 2019] in such a way that an explicit representation of the motion parameters is not necessary and single-view depth estimates are exploited for depth initialization. Di et al. [2019] proposed a scene model for the optimization, which also handles the correlation between the spatial relation and the motion relation of the superpixel planes. The approach by Di et al. [2019] is the currently leading method[2] of MSfM-based approaches evaluated on the KITTI dataset [Geiger et al., 2013].

SVD-MSfM follows the approaches that utilize an instance segmentation for IMO detection and segmentation. The main motivation behind this is that this strategy does not suffer from degenerated situations such as a collinear translation of the camera and IMO. Furthermore, SVD-MSfM provides a novel strategy for scale-aware motion estimation of IMOs by integrating single-view depth information.

4.1.3 Multi-Task Networks for Monocular Scene Flow Estimation

While the first and second category comprises multi-view geometry-based scene reconstructions for static and dynamic environments, the third category covers deep learning-based approaches that provide a scene flow representation. For example, a CNN that outputs transformation matrices in addition to the depth estimates define a scene flow. But also, providing the optical flow and depth of two subsequent images can be considered as a scene flow representation.

In section 3.1.1.3, multi-task networks have been presented and their ability to learn single-view depth estimation in an unsupervised manner based on the geometric constraints in a monocular image sequence has been highlighted. The training loss in such methods is closely related to the optimization objectives in a SLAM or MSfM-based approach. However, these networks provide actually more than single-view depth estimates. These CNNs also provide the camera motion [Mou et al., 2019, Mahjourian et al., 2017, Zhou et al., 2017, Godard et al., 2019, Shen et al., 2019], the camera motion and optical flow estimates [Zhang et al., 2019, Chen et al., 2019, Yin and Shi, 2018, Ranjan et al., 2019, Yang et al., 2018b, Zou et al., 2018, Teng et al., 2018, Luo et al., 2019], or the motion of the camera and all objects [Casser et al., 2019]. The object motion is separately inferred for each object by an additional network in [Casser et al., 2019]. The inputs of the object motion network are the images, in which everything is masked out except the corresponding object. The object masks are assumed to be given by an instance segmentation including consistent object identifiers over time.

Even though the mentioned CNNs use short sequences of at least two images as inputs to exploit multiple views for optical flow and motion estimation, the depth estimation is still based on a single image during inference for most approaches. Casser et al. [2019], Chen et al. [2019] proposed an online refinement, which fine-

[2] Referring to published works until October 02, 2020 to the best of my knowledge

tunes the network for a few iterations on the current image sequence during test time. This deliberately forces the network to overfit on this sequence and implicitly exploits multi-view information for inference. Wang et al. [2019] proposed to insert LSTMs in the encoder to exploit multi-view information. Alternatively, Wu et al. [2019] integrated features that describe the spatial correspondence of pixels between two images. Thus, the features cover also some information regarding the optical flow. Even though these methods do not directly correspond to a multi-view depth estimation such as triangulation, the networks could at least exploit some multi-view information. A more explicit way of integrating the SfM principle is proposed by Ummenhofer et al. [2017]. The proposed method iteratively alternates optical flow estimation with the estimation of depth and camera motion in a CNN-based architecture. For each iteration, the initial depth is computed based on the previous optical flow and camera motion using triangulation. However, this method is only applied and focused on static scenes due to the fact that the network only estimates the camera motion.

The method proposed in [Yang and Ramanan, 2020] is not directly formulated as a multi-task network, but still related to this category. While a first CNN estimates the optical flow between two consecutive images, a second CNN performs a single-view depth estimation for the first image. Based on the optical flow and image appearance features, a third CNN is trained to estimate the motion-in-depth. The motion-in-depth parameterization relates to the ratio between the depth of corresponding points over two images. Using the optical flow estimates as input is motivated by the fact that the optical expansion of an object relates to the motion-in-depth for an orthographic projection onto a fronto-parrallel object plane. The accuracy of depth estimates in the reference image is still limited by the single-view depth estimation.

The most representative method of this category is called Self-Mono-SF [Hur and Roth, 2020], which is explicitly formulated and trained as a scene flow network. Self-Mono-SF outputs a scene flow representation in form of pixel-wise depths and 3D motion vectors using a joint decoder. The decoder is based on a classical optical flow cost volume following [Sun et al., 2018], which captures features of a single reference image and multi-view information by correlating image features of two subsequent images. The network is trained on sequences of stereo image pairs to minimize a photometric distance between a reference image and reconstructed images based on the scene flow outputs. The training loss is additionally designed to consider occlusions and prefer consistency between forward and backward scene flow estimates. The CNN provides scale-aware scene flow estimates due to the stereo supervision.

Referring to the categorization of monocular scene reconstruction methods, SVD-MSfM corresponds to a traditional optimization problem, which explicitly reasons about the multi-view geometry principles – but also integrates single-view depth information using a joint depth cost volume.

4.2 SVD-MSfM Method

The approach presented here, SVD-MSfM, combines probabilistic single-view depth estimates with multi-view geometry following a MSfM-based concept. The present section describes the proposed approach and is divided into two parts. First, the scale-aware rigid-body *motion estimation* is described including the detection of potentially moving objects. Second, the *depth estimation* based on a depth probability volume is explained.

4.2.1 Rigid-Body Motion Estimation

The present subsection describes the scale-aware estimation of the camera and object motions. Two consecutive monocular images I_0 and I_1 and their pixel-wise probabilistic single-view depth distributions $p(\rho \mid I_0)$ and $p(\rho \mid I_1)$ are given as inputs. Formally, the relative transformation $\mathbf{T}_k$ from $t = 0$ to $t = 1$ is estimated for all rigid-bodies $\mathbf{o}_k \in \mathcal{O}$. Each rigid body represents either the static environment or an independent moving object (IMO). The motion estimation can be divided into three main tasks: (1) the detection of parts, which undergoes the same rigid body transformation, (2) a sparse flow estimation to derive optical flow estimates, which are considered as the measurements for the motion optimization, (3) the motion optimization based on the sparse flow estimates and single-view depth distributions provided by ProbDepthNet.

4.2.1.1 *Object Detection and Segmentation*

In the first step, the set of rigid bodies is initialized. As discussed in section 4.1, several geometric cues based on an optical flow have been proposed. However, these constraints are not able to detect all moving objects. Common cases in traffic scenes are oncoming vehicles driving on the adjacent lane. The translational motion of these objects is typically collinear to the camera motion. These objects do not violate the epipolar constraint and appear as static objects closer to the camera. Assuming these objects to be static, they are potentially reconstructed in front of the ego-vehicle, even though they are driving on the adjacent lane. Therefore, SVD-MSfM identifies potentially moving objects based on an instance segmentation. For example, all

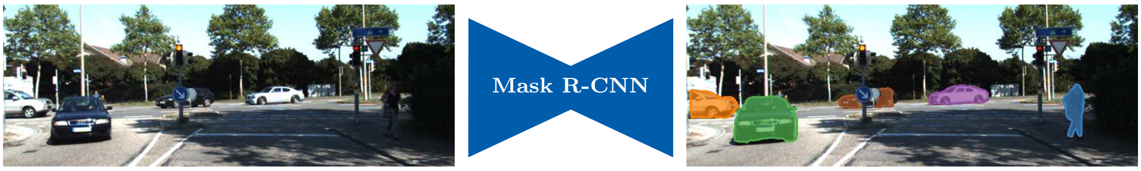


Figure 4.2: The set of rigid bodies $\mathcal{O}$ is defined as the set of objects detected by an instance segmentation (colored in the right image) and the background.

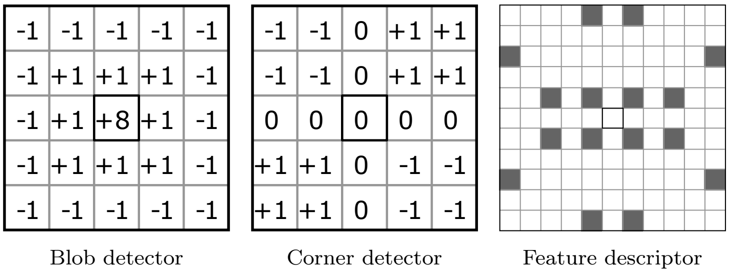
vehicles and pedestrians are taken as moving objects including those standing or parking.

The Mask R-CNN approach [He et al., 2017] (implementation of [Abdulla, 2017]) is used to provide instance labels $l_{\mathbf{p}}$ for each pixel $\mathbf{p}$ and both images I_0 and I_1. The set of detected instances in the first image I_0 as shown in figure 4.2 plus the background directly define the set of rigid bodies $\mathcal{O} = \{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_N\}$. Each object $\mathbf{o}_k$ is defined by its corresponding region of pixels $\mathcal{R}_k = \{\mathbf{p} \mid l_{\mathbf{p}} = k\}$. The estimation of its motion represented by a transformation matrix $\mathbf{T}_k$ is described in the following paragraphs.

4.2.1.2 *Sparse Flow Estimation*

The linearization of the image using a direct method is only valid in a small radius, e.g. Engel et al. [2017] mentioned a radius of 1-2 pixels. This requires a highly accurate initial motion to perform well. While this might be a valid assumption for the camera motion after a short period of initialization, moving objects frequently appear and disappear. Thus, motion estimation is often confronted with the situation that no accurate initial motion is given. Therefore, the motion estimation of SVD-MSfM corresponds to an indirect method based on a sparse optical flow that does not require an accurate initial motion. An analysis in [Engel et al., 2016] and the experiment in section 4.3.2.2 confirm that these indirect methods are more robust for initialization. The sparse flow estimation mainly follows a real-time capable approach proposed by Geiger et al. [2011] with some extensions.

Feature candidates: The feature candidates are a subset of pixels, whose local image structure has good characteristics to match them uniquely between both images. Therefore, blobs and corners are detected by convolving the image with the 5×5 blob and corner detector masks shown in figure 4.3. Applying non-maximum- and non-minimum-suppression [Neubeck and Van Gool, 2006] on the blob and corner filter responses defines the set of feature candidates for both images $\mathcal{F}_0 = \{\mathbf{f}_{1,0}, ..., \mathbf{f}_{N_0,0}\}$ and $\mathcal{F}_1 = \{\mathbf{f}_{1,1}..., \mathbf{f}_{N_1,1}\}$. In addition to the image position $\mathbf{p}_{i,t}$, a feature class $m_{i,t}$ is assigned to each feature candidate. The feature class is one of four classes, which are defined by the detector response: A feature that was



Blob detector

-1	-1	-1	-1	-1
-1	+1	+1	+1	-1
-1	+1	+8	+1	-1
-1	+1	+1	+1	-1
-1	-1	-1	-1	-1

Corner detector

-1	-1	0	+1	+1
-1	-1	0	+1	+1
0	0	0	0	0
+1	+1	0	-1	-1
+1	+1	0	-1	-1

Feature descriptor

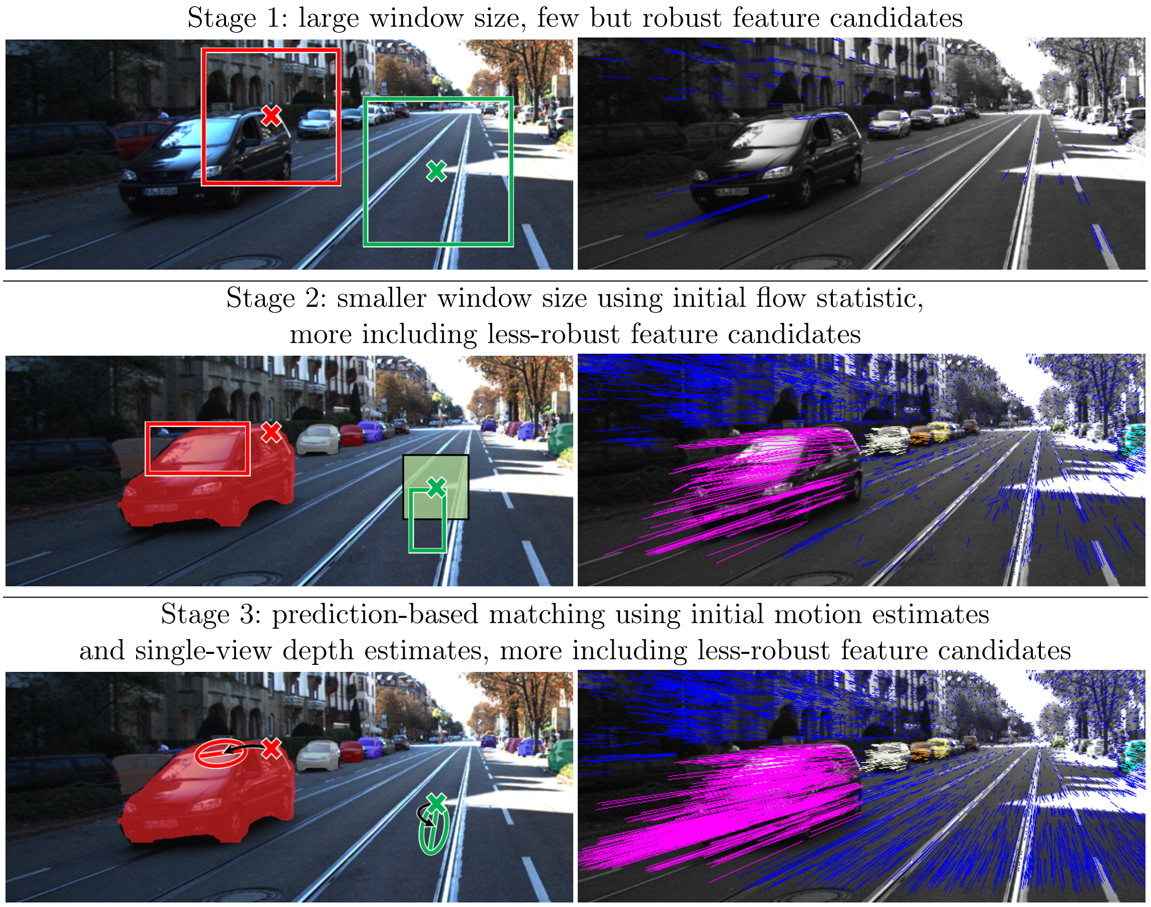
Figure 4.3: Feature candidates and feature vectors of libviso2 [Geiger et al., 2011]. Features candidates are the minima and maxima responses of the blob and corner detector. The feature descriptor is defined as the concatenated Sobel responses using the pattern in the right image. The figure is taken from [Geiger et al., 2011](©2011 IEEE)).

detected as a maximum of the blob detector responses is classified as 'blob max', a feature that was detected as a minimum of the blob detector responses is classified as 'blob min', and analogously features are classified as 'corner max' or 'corner min' if they were detected as the maximum or minimum of the corner detector responses.

Feature descriptor: The feature descriptor $\mathbf{d}_{i,t}$ of each feature candidates represents the local image structure and serves as a basis to define the distance and similarity between different feature candidates. It is defined as the concatenated vector of horizontal and vertical Sobel responses using the pattern in figure 4.3. The distance between two features $dist(\mathbf{f}_{i,0}, \mathbf{f}_{j,1})$ is defined as the sum of absolute differences of the feature descriptors, but only features inside the same class $m_{i,0} = m_{j,1}$ are considered as potential matches:

$$dist(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) = \begin{cases} \sum_k \left|\mathbf{d}_{i,0}^k - \mathbf{d}_{j,1}^k\right| & \text{, if } m_{i,0} = m_{j,1} \\ \infty & \text{, else} \end{cases} \tag{4.8}$$

The objective of the sparse flow estimation is to find a set of flow correspondences $\mathcal{F}$, which consist of paired image feature candidates $(\mathbf{f}_{i,0}, \mathbf{f}_{j,1})$ with $\mathbf{f}_{i,0} \in \mathcal{F}_0$ and $\mathbf{f}_{j,1} \in \mathcal{F}_1$. To find these corresponding feature candidates, an approach divided into three stages is proposed. The general idea of the three stages is that (1) the first stage estimates few but very robust feature matches, (2) the second stage increases the density of feature matches based on the flow statistics in the first stage, and (3) the third stage further increases the density of feature matches based on the predicted feature position exploiting initial motion and single-view depth estimates. Figure 4.4 shows an illustration for each stage, which are described in more detail in the following paragraphs.



Stage 1: large window size, few but robust feature candidates

Stage 2: smaller window size using initial flow statistic, more including less-robust feature candidates

Stage 3: prediction-based matching using initial motion estimates and single-view depth estimates, more including less-robust feature candidates

Figure 4.4: Illustration (left) and results (right) of different stages of optical flow estimation for one image pair of the KITTI dataset [Geiger et al., 2013]. The images are cropped for visualization purposes. The illustration is shown for a feature candidate of a moving object (red) and of the static environment (green). Stage 1: Feature matching within a fixed, large window centered at the feature candidate position. Stage 2: Feature matching within a window defined by the flow statistic of the first stage inside a cell (green) or object (red). Stage 3: Prediction of feature candidate based on initial motion estimates and single-view depth. Predicted feature position and epipolar geometry are exploited for feature matching.

Stage 1: In the first stage, the considered feature candidates, which can likely be matched robustly, are defined by the blob and corner detector described above using a non-maximum- and non-minimum-suppression with a window size of 9×9. The best match of a feature $\mathbf{f}_{i,0}$ is estimated by the lowest distance to a feature $\mathbf{f}_{j,1}$ within a local window of size 200×200 centered at the feature candidate position $\mathcal{W}_{\mathbf{p}_{i,0}}^{(200\times200)}$.

$$(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) = \left(\mathbf{f}_{i,0}, \underset{\mathbf{f}_{j,1}}{\arg\min} \left(\left\{ dist(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) \mid \mathbf{f}_{j,1} \in \mathcal{W}_{\mathbf{p}_{i,0}}^{(200\times200)} \right\} \right) \right) \tag{4.9}$$

Stage 2: The first stage results in relatively few matches (see figure 4.4). To increase the density of flow vectors, the second stage increase the number of feature candidates by using the blob and corner detector with a non-maximum- and non-minimum-suppression with a smaller window size of 3×3. Additionally, the statistic of the flow matches of the first stage is exploited to decrease the window size of the matching process to reduce the number of ambiguous matches. For this purpose, the image is divided into bins of 50×50 pixels. The flow statistics for each bin are defined by the minimum $(\Delta u_{min}, \Delta v_{min})$ and maximum displacements $(\Delta u_{max}, \Delta v_{max})$ of the flow vectors inside the bin. Based on these statistics, the matching is performed in a window defined by the corresponding flow statistic $\mathcal{W}_{(u_{mean}, v_{mean})}^{(\Delta u_{max} - \Delta u_{min} \times \Delta v_{max} - \Delta v_{min})}$.

$$(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) = \left(\mathbf{f}_{i,0}, \underset{\mathbf{f}_{j,1}}{\arg\min} \left(\left\{ dist(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) \mid \mathbf{f}_{j,1} \in \mathcal{W}_{(u_{mean}, v_{mean})}^{(\Delta u_{max} - \Delta u_{min} \times \Delta v_{max} - \Delta v_{min})} \right\} \right) \right) \tag{4.10}$$

The window is centered at $(u_{mean}, v_{mean}) = \mathbf{p}_{i,0} + (\Delta u_{max} + \Delta u_{min}, \Delta v_{max} + \Delta v_{min}))/2$. These windows are typically much smaller than the window in stage 1. The smaller windows reduce the computational effort and number of ambiguous matches inside a window, which enables to match more feature candidates including the less discriminative ones. In contrast to [Geiger et al., 2011], the instance segmentation is exploited in this stage. Due to the motion of IMOs, the optical flow often differs highly from the surrounding environment. Therefore, each instance segment is considered as an individual bin to determine an object-wise flow statistic.

Stage 3: While Geiger et al. [2011] originally used two stages of flow estimation, a third stage is proposed here. The third stage is applied after the first scale-aware motion estimations $\mathbf{T}_k$ described in the next section 4.2.1.3 and designed to exploit the initial motion estimate in combination with single-view depth estimates. Formally, for each feature candidate, the corresponding rigid body $\mathbf{o}_k$ is defined by the image position $\mathbf{p}_{i,0}$ and a single-view depth distribution $p_{\mathbf{p}_{i,0}}(\rho \mid I_0)$ is provided by ProbDepthNet. The total mean of the depth distribution is considered as the maximum likelihood depth estimate $\rho_i = Z^{-1}$. Based on the transformation of the corresponding rigid body $\mathbf{T}_k$ and the depth value ρ_i, the expected image position $\hat{\mathbf{p}}_{i,1}$ in image I_1 is defined as follows:

$$\hat{\mathbf{p}}_{i,1} = \mathbf{K} \left(\mathbf{R}_k \rho_i^{-1} \mathbf{K}^{-1} \mathbf{p}_{i,0} + \mathbf{t}_k \right) \tag{4.11}$$

The matrices $\mathbf{R}_k$ and $\mathbf{t}_k$ refer to the decomposition of $\mathbf{T}_k$ into a rotation matrix and translation vector. The best match is searched in a local window $\mathcal{W}_{\hat{\mathbf{p}}_{i,1}}^{(50 \times 50)}$ centered at the expected position $\hat{\mathbf{p}}_{i,1}$ in the next image:

$$(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) = \left(\mathbf{f}_{i,0}, \underset{\mathbf{f}_{j,1}}{\arg\min} \left(\left\{ dist'(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) \mid \mathbf{f}_{j,1} \in \mathcal{W}_{\hat{\mathbf{p}}_{i,1}}^{(50 \times 50)} \right\} \right) \right) \tag{4.12}$$

The distance is additionally rated by the distance to the epipolar line (2.1) and the distance to the expected position:

$$dist'(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) = dist(\mathbf{f}_{i,0}, \mathbf{f}_{j,1}) + \lambda_1 \left|\left|\mathbf{p}_{j,1} - \hat{\mathbf{p}}_{i,1}\right|\right|_2 + \lambda_2 \mathbf{e}^T \mathbf{p}_{j,1} \tag{4.13}$$

The vector $\mathbf{e}$ defines the epipolar line of the feature candidate $\mathbf{f}_{i,0}$. The additional terms are weighted by λ_1 or λ_2.

As shown in figure 4.4 this results in a denser flow field and the experiment in section 4.3 confirms the claimed extension of the flow estimation. The higher density is especially important to have a sufficient number of flow vectors on each rigid body, in the near field, and on areas with high confident single-view depth estimates.

In all stages, matches are only accepted if they pass the forward-backward consistency check defined as follows:

$$\left(\mathbf{f}_{i,0}, \underset{\mathbf{f}_{j,1}}{\arg\min} \left(dist(\mathbf{f}_{i,0}, \mathbf{f}_{j,1})\right)\right) \overset{!}{=} \left(\underset{\mathbf{f}_{i,0}}{\arg\min} \left(dist(\mathbf{f}_{i,0}, \mathbf{f}_{j,1})\right), \mathbf{f}_{j,1}\right) \tag{4.14}$$

Furthermore, matches are considered as outliers and removed if the match is not supported by at least two neighboring matches. The neighboring matches are defined by a 2D Delaunay triangulation [Delaunay et al., 1934]. A neighboring match is considered as a supporter if its flow difference is within some threshold τ_1.

4.2.1.3 *Motion Estimation*

Based on the set of matched features $\mathcal{F}$, the transformation of each rigid body (IMOs and camera motion) is estimated individually. This estimation is performed subsequently to stage 2 of the sparse flow estimation to derive initial transformations and subsequently to stage 3 to estimate the final transformations. The individual motion estimation of each rigid body is formulated as an energy minimization problem based on a factor graph of map points and poses. The energy term comprises a multi-view geometry term in the form of a reprojection error and prefers 3D map points consistent with the probabilistic single-view depth distribution provided by ProbDepthNet.

Even though it would be straightforward to utilize the energy term for bundle adjustment (BA), the optimization is only performed on two consecutive frames. Because objects frequently appear and disappear, it is important to provide high accuracy even for short image sequences.

One important aspect of integrating single-view depth information is to provide metric scale information. In contrast to [Barnes et al., 2018, Yin et al., 2017, Yang et al., 2018a], the approach presented here applies this idea additionally for scale-

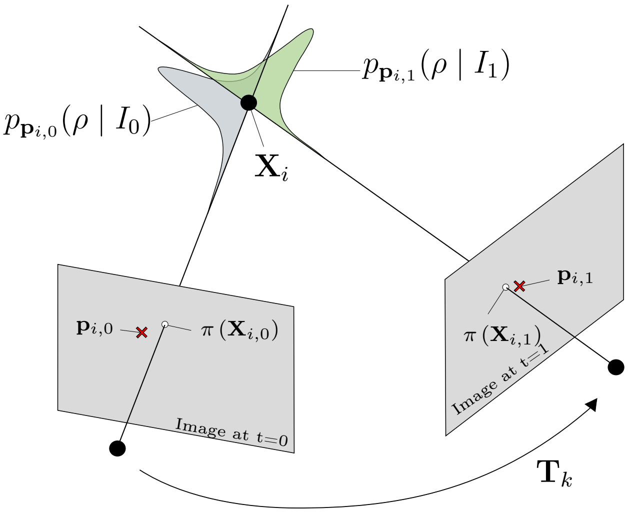

$p_{\mathbf{p}_{i,1}}(\rho \mid I_1)$
$p_{\mathbf{p}_{i,0}}(\rho \mid I_0)$
$\mathbf{X}_i$
$\mathbf{p}_{i,1}$
$\pi\left(\mathbf{X}_{i,1}\right)$
Image at t=1
$\mathbf{p}_{i,0}$
$\pi\left(\mathbf{X}_{i,0}\right)$
Image at t=0
$\mathbf{T}_k$

Figure 4.5: Joint optimization of transformation $\mathbf{T}_k$ and set of 3D points $\mathcal{X}_k$ (shown for one example of a 3D point $\mathbf{X}_i$). The optimization minimizes the distance between the flow-based image position $\mathbf{p}_{i,t}$ (red cross) and projected image position $\pi\left(\mathbf{X}_{i,t}\right)$ (white dot). Furthermore, the distance of the 3D point should be consistent to the single-view depth distributions $p_{\mathbf{p}_{i,t}}(\rho \mid I_t)$ (green/blue distributions).

aware motion estimation of moving objects. Compared to the approaches described in section 4.1.2 that optimize the scale based on the non-accidental criterion or by projecting the IMO on the ground plane, integrating single-view depth provides the following advantages: (1) a long history of the object trajectory is not required, (2) there is no limitation to certain motions and it is not prone to degenerate cases, and (3) the ground contact point of the IMO does not necessarily have to be observed.

The proposed motion optimization is described in more detail in the following paragraphs. The set of rigid bodies $\mathcal{O}$ is defined by the instance segmentation in the first image. In the first step, each rigid body $\mathbf{o}_k$ is assigned to an instance segment of the second image using a simple voting scheme. Each flow correspondence $(\mathbf{f}_{i,0}, \mathbf{f}_{j,1})$ that lies inside the object region $\mathbf{p}_{i,0} \in \mathcal{R}_k$ votes for the corresponding instance segment $l_{\mathbf{p}_{j,1}}$. The instance segment in the second image with the most votes is considered to be the same object. The subset of flow vectors $\mathcal{F}_k \subset \mathcal{F}$ that are inside the respective instance masks in both images are the basis for the motion estimation.

Energy minimization problem: The following steps are performed separately for each rigid body $\mathbf{o}_k$. For each flow correspondence in $\mathcal{F}_k$, the variables $\mathbf{X}_i \in \mathbb{R}^3$ are introduced to represent the 3D position of the scene point. The relative motion of the camera to the object is considered as a virtual camera motion without loss of generality. The basic geometric principles to jointly optimize the set of 3D points $\mathcal{X}_k$ and the transformation of the rigid body $\mathbf{T}_k$ are shown in figure 4.5.

Formally, the 6D motion $\mathbf{T}_k \in SE(3)$ of the rigid body is optimized jointly with the set of 3D points $\mathbf{X}_i \in \mathcal{X}_k$ by minimizing an energy term E. Without loss of generality, the poses are defined as $\mathbf{T}_{k,0} = \mathbf{I}_{4\times4}$ and $\mathbf{T}_{k,1} = \mathbf{T}_k$. The energy term consist of two parts, a reprojection error $\Phi_t^{proj}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_{k,t})$ weighted by λ_3 and a term that rates the likelihood of a 3D scene points based on the single-view depth distributions $\Phi_t^{svd}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_{k,t})$.

$$E = \sum_{\mathbf{X}_i \in \mathcal{X}_k} \sum_{t \in \{0,1\}} \lambda_3 \cdot \Phi_t^{proj}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_{k,t}) + \Phi_t^{svd}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_{k,t}). \tag{4.15}$$

Both terms of the energy term are explained in more detail in the following paragraphs.

First, the projected positions of each 3D point ($\pi(\mathbf{X}_{i,0})$ and $\pi(\mathbf{X}_{i,1})$) should be close to the image positions of the corresponding features ($\mathbf{p}_{i,0}$ and $\mathbf{p}_{i,1}$). While the position of a 3D point in the first camera $\mathbf{X}_{i,0} = \mathbf{X}_i$ is identical to the introduced variable $\mathbf{X}_i$, the transformation $\mathbf{T}_k$ needs to be considered to derive the position in the second image $\mathbf{X}_{i,1} = \mathbf{R}_k\mathbf{X}_i + \mathbf{t}_k$. The reprojection error $\Phi_t^{proj}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_k)$ is defined as follows:

$$\Phi_t^{proj}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_k) = \ell_H\left(||\mathbf{p}_{i,t} - \pi\left(\mathbf{X}_{i,t}\right)||_\Sigma^2\right) \tag{4.16}$$

The function ℓ_H is the robust Huber norm [Huber, 1992] and $||.||_\Sigma$ the Mahalanobis distance. A constant diagonal matrix is assumed to represent the covariance of all flow vectors $\Sigma = diag(\sigma_{flow}^2)$. However, it could be also replaced with an estimated covariance matrix.

Second, the single-view depth distributions define the probabilities $p_{\mathbf{p}_{i,0}}(\rho_{i,0} \mid I_0)$ and $p_{\mathbf{p}_{i,1}}(\rho_{i,1} \mid I_1)$ of the 3D point $\mathbf{X}_i$ along the rays (see figure 4.5), which should be maximized as well. The depths of the 3D point are defined as $\rho_{i,0} = Z_{i,0}^{-1}$ and $\rho_{i,1} = Z_{i,1}^{-1}$. Formally, the data term $\Phi_t^{svd}(\mathbf{p}_i, \mathbf{X}_i, \mathbf{T}_k)$ rates the consistency of the 3D points $\mathbf{X}_i$ in terms of single-view depth probabilities by

$$\Phi_t^{svd}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_j) = -\log\left(p_{\mathbf{p}_{i,t}}(Z_{i,t}^{-1} \mid I_t)\right). \tag{4.17}$$

This is essentially the part that provides the metric scale information and results in scale-aware estimates of the transformation $\mathbf{T}_k$ and 3D points $\mathcal{X}_k$.

Iterative optimization: The energy term of equation (4.15) is optimized iteratively using the Levenberg-Marquardt method (section 2.2.1). The optimization is performed for the 3D scene points $\mathbf{X}$ and for the transformation $\mathbf{T}$ of each rigid body individually. Referring to section 2.2.1, the Levenberg-Marquardt method for weighted least squares minimization requires to express the energy term as a weighted sum of residuals $E = \sum \mathbf{r}^T \mathbf{W} \mathbf{r}$. Both parts of the energy term Φ_t^{proj} and Φ^{svd} can be brought into the desired form by rearranging equations (4.16) and (4.17). First, the residual $\mathbf{r}^{proj}$ and weight matrix $\mathbf{W}^{proj}$ for the reprojection error Φ_t^{proj} are defined as follows:

$$\begin{aligned} \lambda_3 \Phi_t^{proj} =& \lambda_3 \cdot \ell_H \left(||\mathbf{p}_{i,t} - \pi\left(\mathbf{X}_{i,t}\right)||^2_{\Sigma} \right) \\ =& \ell_H \left(\underbrace{\left(\mathbf{p}_{i,t} - \pi\left(\mathbf{X}_{i,t}\right)\right)}_{\mathbf{r}^{proj}}{}^T \underbrace{\sqrt{\lambda_3}\Sigma^{-1}}_{\mathbf{W}^{proj}} \left(\mathbf{p}_{i,t} - \pi\left(\mathbf{X}_{i,t}\right)\right) \right) \end{aligned} \tag{4.18}$$

Second, assuming single-view depth distributions given as Gaussian distributions $\mathcal{N}(\mu_i, \sigma_i)$, the required form and definition of residual $\mathbf{r}^{svd}$ and weight matrix $\mathbf{W}^{svd}$ are given by the following equation:

$$\begin{aligned} \Phi_t^{svd}(\mathbf{p}_{i,t}, \mathbf{X}_i, \mathbf{T}_j) &= -\log\left(p_{\mathbf{p}_{i,t}}(Z_{i,t}^{-1} \mid I_t) \right) \\ &= \underbrace{\frac{1}{2\sigma_i^2}}_{\mathbf{W}^{svd}} \underbrace{\left(Z_{i,t}^{-1} - \mu_i\right)}_{\mathbf{r}^{svd}}{}^2 + \cancel{\log \sigma_i^2} \end{aligned} \tag{4.19}$$

The linearization of the Levenberg-Marquardt needs to derive the Jacobi matrices of the variables w.r.t. the residuals to define the update rule (equation (2.16)). To define an update step for the transformations $\mathbf{T}_t$, a minimal representation by its corresponding Lie-algebra elements $\xi \in se(3)$ represented as a 6D vector $\xi = [vw]^T \in \mathbb{R}^6$ is used. Before each iteration, the 3D position of all points is transformed to the current solution of the pose $\mathbf{T}_t$. Thereby, the Jacobi matrix could be evaluated at $\xi = 0$ and $\mathbf{X} = \mathbf{X}_t$ to express a differential motion at the current pose, which defines the update step as $\mathbf{T}_t \leftarrow exp^{\xi} \mathbf{T}_t$. The exponential map of Lie algebra elements to a transformation matrix is defined by exp^{ξ} (e.g. described in [Murray et al., 1994, p. 42]).

The derivations of the Jacobi matrices are provided in table 4.2. The implementation of the robust Huber-Kernel and Levenberg-Marquardt solver in [Kümmerle et al., 2011] is applied for each iteration. After each iteration, the variables of the transformation matrices and 3D positions are updated and the results after 50 iterations are considered as the final estimates.

4.2.2 Dense Depth Estimation

While the first step assigns depth values to a small subset of pixels, the second step is designed to provide a dense depth map for all pixels. Furthermore, it directly combines a photometric distance with the single-view depth distributions. The rigid

	Reprojection error (residual $\mathbf{r}^{proj}$)
3D scene point $\mathbf{X}$	$\begin{aligned} \mathbf{J}^{proj}_{\mathbf{X}} &= \frac{\partial \mathbf{r}^{proj}(\mathbf{X}, \mathbf{T})}{\partial \Delta \mathbf{X}} \\ &= \left.\frac{\partial \mathbf{r}^{proj}(\mathbf{X}_t, \mathbf{T})}{\partial \mathbf{X}_t}\right\vert_{\mathbf{X}=\mathbf{X}_t} \cdot \left.\frac{\partial \mathbf{X}_t}{\partial \Delta \mathbf{X}}\right\vert_{\mathbf{T}=\mathbf{T}_t} \\ &= \begin{bmatrix} -f_x Z_t^{-1} & 0 & f_x X_t Z_t^{-2} \\ 0 & -f_y Z_t^{-1} & f_y Y_t Z_t^{-2} \end{bmatrix} \mathbf{R}_t \end{aligned}$
Transformation ξ	$\begin{aligned} \mathbf{J}^{proj}_{\xi} &= \frac{\partial \mathbf{r}^{proj}(\mathbf{X}_t, \xi)}{\partial \Delta \xi} \\ &= \left.\frac{\partial \mathbf{r}^{proj}(\mathbf{X}_t, \xi)}{\partial \mathbf{X}_t}\right\vert_{\mathbf{X}=\mathbf{X}_t} \cdot \left.\frac{\partial \mathbf{X}_t}{\partial \Delta \xi}\right\vert_{\mathbf{X}=\mathbf{X}_t, \xi=0} \\ &= \begin{bmatrix} -f_x Z_t^{-1} & 0 & f_x X_t Z_t^{-2} \\ 0 & -f_y Z_t^{-1} & f_y Y_t Z_t^{-2} \end{bmatrix} \left[\mathbf{I}_{3x3} \mid -\left[\mathbf{X}_t\right]_{\times} \right] \end{aligned}$
	Single-view depth (residual $\mathbf{r}^{svd}$)
3D scene point $\mathbf{X}$	$\begin{aligned} \mathbf{J}^{svd}_{\mathbf{X}} &= \frac{\partial \mathbf{r}^{svd}(\mathbf{X}, \mathbf{T})}{\partial \Delta \mathbf{X}} \\ &= \left.\frac{\partial \mathbf{r}^{svd}(\mathbf{X}_t, \mathbf{T})}{\partial \mathbf{X}_t}\right\vert_{\mathbf{X}=\mathbf{X}_t} \cdot \left.\frac{\partial \mathbf{X}_t}{\partial \Delta \mathbf{X}}\right\vert_{\mathbf{T}=\mathbf{T}_t} \\ &= \begin{bmatrix} 0 & 0 & Z_t^{-2} \end{bmatrix} \mathbf{R}_t \end{aligned}$
Transformation ξ	$\begin{aligned} \mathbf{J}^{svd}_{\xi} &= \frac{\partial \mathbf{r}^{svd}(\mathbf{X}_t, \xi)}{\partial \Delta \xi} \\ &= \left.\frac{\partial \mathbf{r}^{svd}(\mathbf{X}_t, \xi)}{\partial \mathbf{X}_t}\right\vert_{\mathbf{X}=\mathbf{X}_t} \cdot \left.\frac{\partial \mathbf{X}_t}{\partial \Delta \xi}\right\vert_{\mathbf{X}=\mathbf{X}_t, \xi=0} \\ &= \begin{bmatrix} 0 & 0 & Z_t^{-2} \end{bmatrix} \left[\mathbf{I}_{3x3} \mid -\left[\mathbf{X}_t\right]_{\times} \right] \end{aligned}$

Table 4.2: Derivation of the Jacobi matrices to apply the Levenberg-Marquardt method to optimize the energy term w.r.t the 3D scene points $\mathbf{X}$ and the motion variables expressed as Lie-algebra elements ξ. $\mathbf{X}_t$ refers to the 3D position of $\mathbf{X}$ transformed into camera coordinates at time t.

body transformations are given by the method described above (section 4.2.1) and fixed for the dense depth estimation.

A depth cost volume is created, which rates the probability of a set of discretized depth values $\mathcal{D} = \{d_0, d_1, ..., d_N\}$ for each pixel $\mathbf{p}_{i,0}$ in image I_0. The depth values are represented in the inverse space as virtual disparities assuming a virtual baseline even though a monocular setup is used. The representation as virtual disparities is not mandatory but scales the depth to a more intuitively interpretable range of values and allows a discretization as integer values, which could be beneficial for implementation reasons.

The cost entry $E(\mathbf{p}_{i,0}, d_j, \mathbf{T}_k)$ for a given disparity value d_j and pixel $\mathbf{p}_{i,0}$ is defined to prefer a multi-view photometric consistency $\Phi^{photo}(\mathbf{p}', d_j, \mathbf{T}_k)$ and high probabilities in terms of the single-view depth distributions $\Phi^{svd}(\mathbf{p}', d_j)$ provided by ProbDepthNet:

$$E(\mathbf{p}_{i,0}, d_j, \mathbf{T}_k) = \sum_{\mathbf{p}' \in \mathcal{W}^{3\times3}_{\mathbf{p}_{i,0}}} \lambda_4 \cdot \Phi^{photo}(\mathbf{p}', d_j, \mathbf{T}_k) + \lambda_5 \cdot \Phi^{svd}(\mathbf{p}', d_j) \tag{4.20}$$

The transformation $\mathbf{T}_k$ of the corresponding rigid body ($\mathbf{p}_{i,0} \in \mathcal{R}_k$) is given by the motion estimation. The cost values are aggregated in a small window $\mathcal{W}^{3\times3}_{\mathbf{p}_{i,0}}$ and each part is weighted by λ_4 or λ_5.

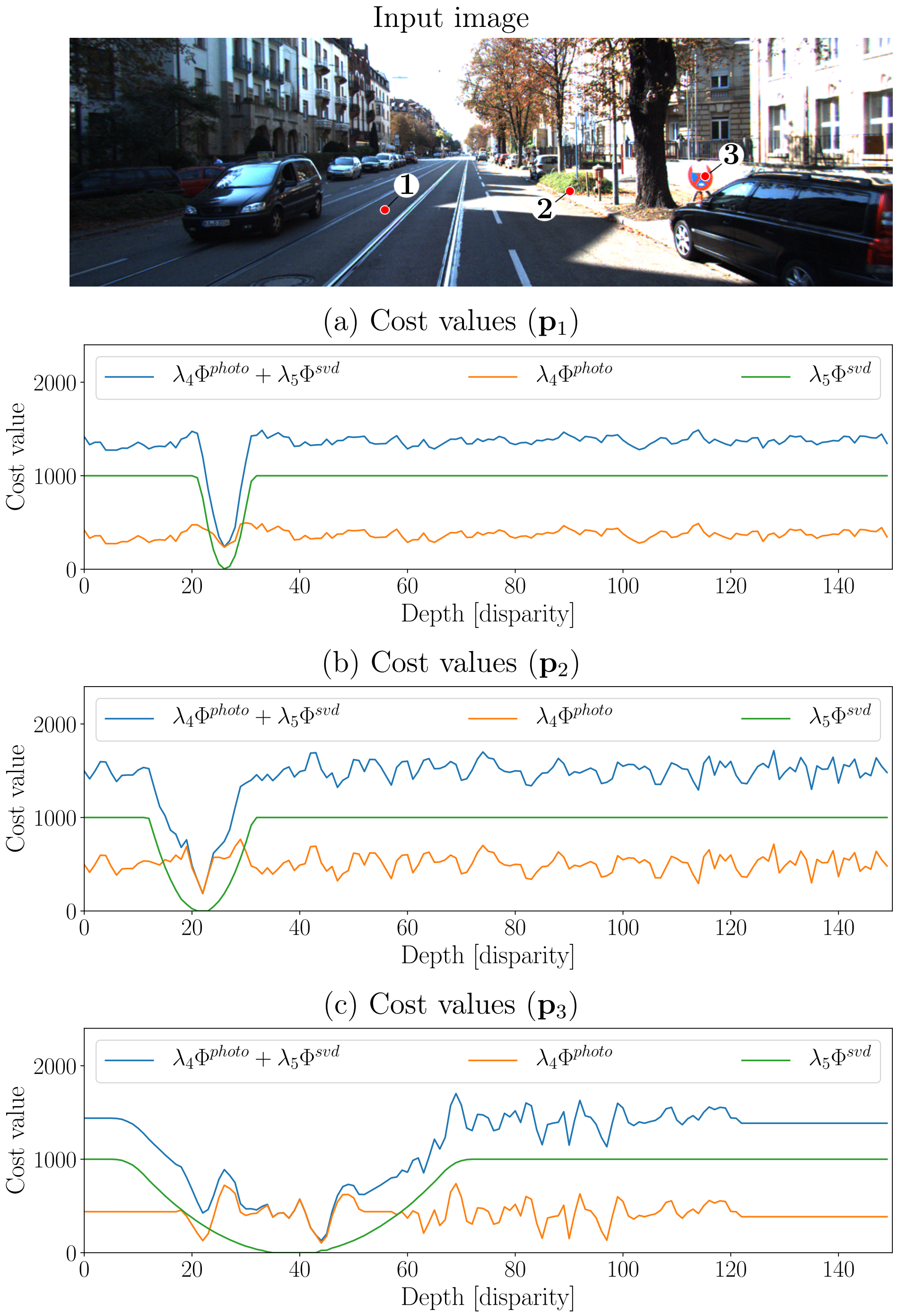
The multi-view geometry part rates the disparity values based on the photometric consistency along the epipolar line in the second image. For each cost entry, the expected image position $\hat{\mathbf{p}}_1$ in the next image is defined by the corresponding image position $\mathbf{p}$, disparity d_j, and transformation $\mathbf{T}_k$ (equation (4.11)). Consequently, the photometric similarity $\Phi^{photo}(\mathbf{p}, d_j, \mathbf{T}_k)$ is defined based on the corresponding image positions $\mathbf{p}$ and $\hat{\mathbf{p}}_1$:

$$\begin{aligned} \Phi^{photo}(\mathbf{p}, d_j, \mathbf{T}_k) = \quad & \lambda_6 \cdot |\mathcal{G}_0(\mathbf{p}, \mathbf{e}(\mathbf{p})) - \mathcal{G}_1(\hat{\mathbf{p}}_1, \mathbf{e}(\mathbf{p}))| \\ & + \lambda_7 \cdot H(\mathcal{B}_0(\mathbf{p}), \mathcal{B}_1(\hat{\mathbf{p}}_1)) \end{aligned} \tag{4.21}$$

Following previous approaches [Yamaguchi et al., 2013, Bai et al., 2016], the first part rates the difference of the image gradients ($\mathcal{G}_0, \mathcal{G}_1$) in epipolar direction $e(\mathbf{p})$. The second part rates the similarity of the image patches as the Hamming distance of the census transforms $H(\mathcal{B}_0, \mathcal{B}_1)$. Both parts are weighted by λ_6 or λ_7.

While the first data term $\Phi^{photo}(\mathbf{p}, d_j, \mathbf{T}_k)$ is defined to prefer a multi-view photometric consistency, the second data term $\Phi^{svd}(\mathbf{p}, d_j)$ integrates the probabilistic single-view depth distributions provided by ProbDepthNet. Therefore, the single-view depth distributions rate the likelihood of each disparity value d_j and pixel $\mathbf{p}$ analogously to equation (4.17).

Figure 4.6 shows the depth cost values at three image positions to illustrate how both kinds of information interact. The first image position (see figure 4.6 (a)) lies on the road, resulting in combined cost values that are dominated by single-view depth information. While the photometric distance is less distinctive in this case, ProbDepthNet provides a depth distribution with low variance. A similar characteristic would also be given in low-textured areas or low-parallax situations (e.g. standstill scenarios or image positions close to the focus of expansion). For the second image position (see figure 4.6 (b)), ProbDepthNet provides a medium uncertainty and a



(a) Cost values ($\mathbf{p}_1$)

(b) Cost values ($\mathbf{p}_2$)

(c) Cost values ($\mathbf{p}_3$)

Figure 4.6: Illustration of depth cost volume at 3 image positions. The corresponding image positions of $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$ are shown in the top image (red dots). The diagrams shows both parts (multi-view photometric distance Φ^{photo} (orange) and single-view depth information Φ^{svd} (green) provided by ProbDepthNet) as well as the combined cost value (blue).

depth estimate that is consistent with the minimum of the photometric distance. Combining both kinds of information results in a sharp global minimum at the correct disparity value. The cost values at the third image position (see figure 4.6 (c)), which corresponds to a traffic sign, are more dominated by the photometric distance due to the high uncertainty of the single-view depth information. The combined global minimum differs from the most likely single-view depth estimate. However, single-view depth information is still beneficial to prefer the correct local minimum of the photometric term. Note that this highly benefits from integrating single-view depth distributions instead of single depth estimate that merely represent the most likely depth value. The photometric term is typically much sharper, but also more ambiguous than the single-view depth information. Consequently, the global minimum often corresponds to a local minimum of the photometric term in a plausible range of disparity values defined by the single-view depth part. It should be briefly mentioned that the truncation of the single-view depth term is just due to the implementation to avoid overflows.

To extract a dense depth image, semi-global-matching (SGM) [Hirschmuller, 2005] and slanted-plane smoothing (SPS) [Yamaguchi et al., 2014] are applied to the depth cost volume. SGM performs smoothing on the depth cost volume and the dense disparities are extracted as the pixel-wise minimum of the smoothed depth cost volume. Based on these dense disparities, SPS performs a superpixel segmentation, where each superpixel corresponds to a planar surface element. These approaches are applied in their standard proposed form – thus, the reader is referred to the corresponding papers [Hirschmuller, 2005, Yamaguchi et al., 2014] for further details.

4.3 Experimental Evaluation of SVD-MSfM Method

SVD-MSfM estimates the scene flow from monocular images focusing on dynamic traffic scenes, which means providing the 3D position and 3D motion of each pixel. The pixel-wise representation of 3D position and 3D motion is defined by the pixel-wise depth estimates and the 6D motion of the corresponding rigid bodies. After the experimental setup is described in section 4.3.1.1, sections 4.3.1.2 and 4.3.1.3 provide qualitative results and quantitative evaluation with respect to several monocular baseline methods. The baselines are represented by SotA methods of the different approaches explained in section 4.1. To the best of my knowledge, it is also the first time that the different kinds of approaches are compared in terms of a monocular scene flow metric. Components and design choices of SVD-MSfM are analyzed in section 4.3.2. The first two ablation studies address the claimed combination of multi-view geometry with single-view depth information. The experiments show that both kinds of information provide a significant improvement and confirm the importance of integrating single-view depth estimates in a probabilistic and well-calibrated form. Additionally, also the proposed extension of the libviso2-based [Geiger et al., 2011] sparse flow estimation presented in section 4.2.1.2 is confirmed to provide an improvement in terms of the scene flow estimation. Finally, the motion estimation is evaluated by itself.

4.3.1 Evaluation of Scene Flow Estimation

The present subsection provides a qualitative and quantitative evaluation of the proposed monocular scene flow estimation starting with a description of the experimental setup and the monocular baseline methods.

4.3.1.1 *Experimental Method and Baselines*

Dense scene flow can be encoded by the 3D position of each pixel at $t = 0$ plus its 3D translational motions from $t = 0$ to $t = 1$. The evaluation follows the equivalent representation of two dense depth images for $t = 0$ and $t = 1$ and an optical flow field [Menze and Geiger, 2015]. The depth and optical flow images are aligned with the image coordinates of the reference image at $t = 0$. For calibrated cameras, the 3D position of one pixel is defined by a depth value in addition to its image coordinates. Consequently, the depth image at $t = 0$ represents the 3D positions at $t = 0$. The optical flow, which defines the corresponding image coordinates at $t = 1$, plus the depth values for $t = 1$ represent the 3D position at $t = 1$.

The quantitative evaluation is based on the KITTI scene flow dataset [Menze and Geiger, 2015], which reports the frequencies of errors for the depth at time $t = 0$ (D1), depth at $t = 1$ (D2), and the optical flow (Fl). All estimates are given in the image coordinates of the reference image at $t = 0$. An estimate is considered as an incorrect estimate if the error in terms of stereo disparity or optical flow endpoint error exceeds a threshold of 3 pixels and 5% relative to the disparity and optical flow vector length. Furthermore, an estimate is only defined as a valid scene flow estimate (SF) if it fulfills all the requirements on the D1, D2, and Fl metrics (disparities and optical flow endpoint error lower than 3 pixel and 5%). All metrics are evaluated separately for moving objects (fg), the static scene (bg), and both combined (all). Four categories of monocular methods are evaluated.

In the first category are the multi-task networks that were presented in section 4.1.3. The evaluation is based on the published code, models, or results of GeoNet [Yin and Shi, 2018], DF-Net [Zou et al., 2018], Struct2Depth [Casser et al., 2019], and Self-Mono-SF [Hur and Roth, 2020] that are found in the internet or stated in their papers. For GeoNet and DF-Net, the estimated optical flow is combined with the depth estimates for both images to define a scene flow following [Schuster et al., 2018]. The optical flow is also used to transform the depth estimates at $t = 1$ to the image coordinates at $t = 0$. However, this transformation can not handle occlusions and pixels that leave the field of view. In contrast to GeoNet and DF-Net, Struct2Depth provides directly 6D transformation matrices for both the static environment and moving objects. Therefore, the depth at $t = 1$ can be derived by applying the respective transformations to the depth estimates at $t = 0$. Struct2Depth assumes an instance segmentation with aligned object identifiers to be given. As this is not provided by the published code, the same instance segmentation as for SVD-MSfM is used and the proposed sparse flow voting scheme is applied to align the object identifiers. These methods are trained in an unsupervised manner and suffer from a global scale ambiguity. Therefore, the estimates are scaled by a factor $s = \text{median}\left(d_i/d_{i,GT}\right)$ to align with the ground truth. Additionally, the results of EPC++ are stated in the corresponding paper [Yang et al., 2018b] for the D1, D2, and Fl metric. This method also integrates stereo supervision during training and provides metric scale-aware estimates. However, the D2 metric is excluded as it seems to be inconsistent. To give a short explanation, the 'D2-all' metric is a weighted mean of the 'D2-fg' and 'D2-bg' metrics weighted by their respective frequency. However, the relation is not consistent for the stated D2 results of EPC++. The scene flow outputs of the Self-Mono-SF method are publicly available and taken for evaluation. Self-Mono-SF is also considered as the most representative method of this group for qualitative evaluation as it explicitly addresses a scene flow estimation problem.

Instead of a multi-task network that provides both optical flow and depth estimates (e.g. GeoNet or DF-Net), individual methods could be used that address the respective tasks separately. Therefore, as a second category, single-view depth ('LRC [Godard et al., 2017]' or 'DORN [Fu et al., 2018]') and optical flow estimation ('MirrorFlow [Hur and Roth, 2017]' or 'HD^3-F [Yin et al., 2019]') are combined as individual tasks. The individual depth and optical flow estimates are again combined to a scene flow following [Schuster et al., 2018].

The third group comprises the MSfM-based methods described in section 4.1.2. First, DMDE [Ranftl et al., 2016] corresponds to a method that follows an optical flow-based motion segmentation with subsequent reconstruction of each rigid body. The scale ambiguity is solved by scene model assumptions, which is basically the assumption that moving objects are in contact with the ground plane. Second, the results of S.Soup [Kumar et al., 2017] and S.Rel. [Di et al., 2019] are provided, which corresponds to a MSfM-approach formulated as a '3D jigsaw puzzle'. These methods were evaluated on their mean absolute relative error (MRE) in terms of depth estimates capped at 50 meters. The results are taken from their respective papers.

The fourth category consists of the methods that combine single-view depth estimates with multi-view geometry. MFA [Kumar et al., 2019] is added to this group as it uses single-view depth estimates for initializing an approach similar to S.Soup. The results of MFA are also provided in terms of the MRE metric. However, this group is mainly represented by the proposed SVD-MSfM method, which explicitly combines both kinds of information.

4.3.1.2 *Qualitative Results of Scene Flow Estimation*

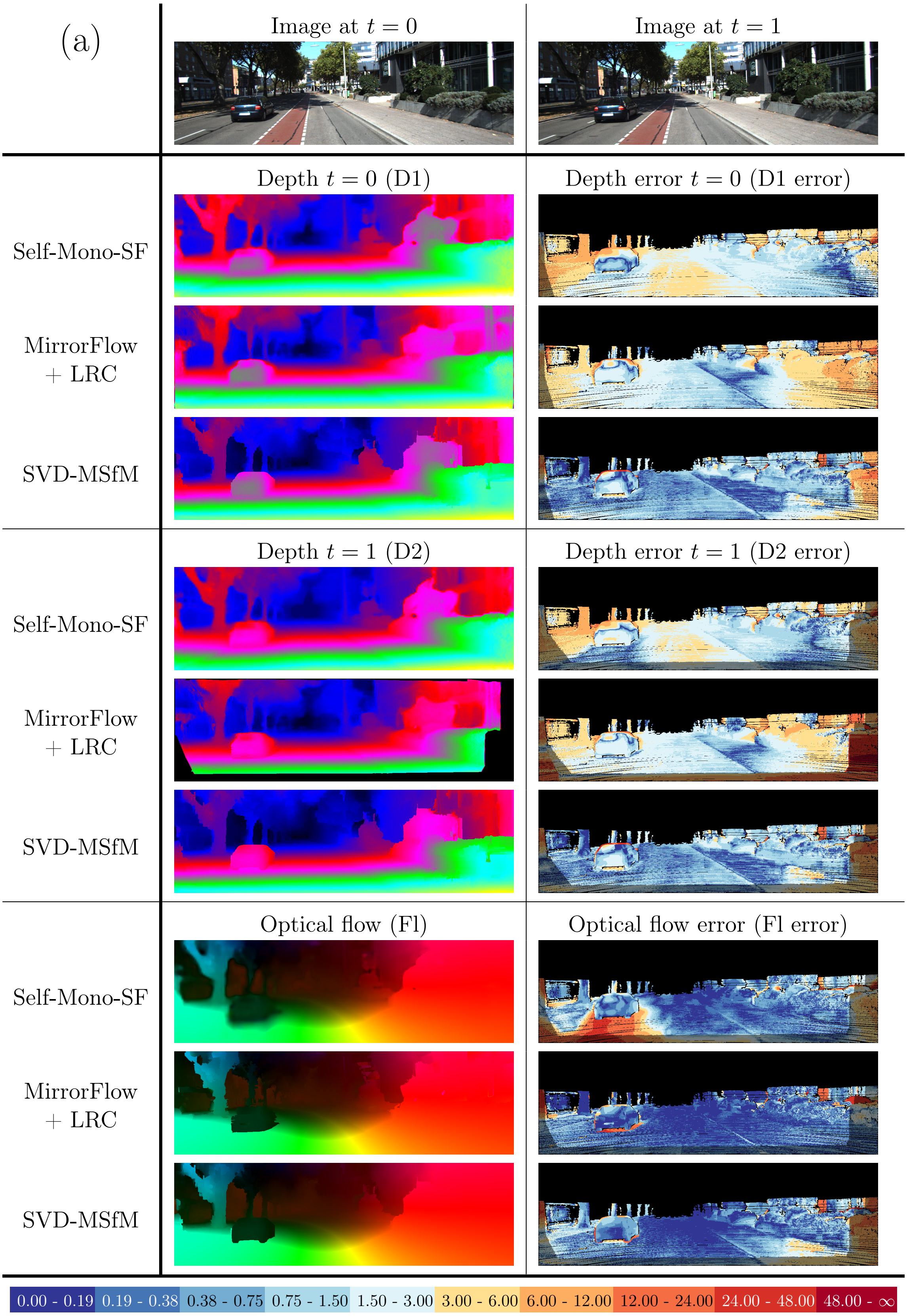
In the present subsection, qualitative results of monocular scene flow estimation methods are provided for the KITTI scene flow training set [Menze and Geiger, 2015]. Figure 4.11 shows qualitative results and errors in terms of the disparity at $t = 0$ (D1), the disparity at $t = 1$ (D2), and the optical flow (Fl). The scene flow error (SF error) is defined as the maximum of the disparity and optical flow errors. The visualizations are generated by using the KITTI scene flow evaluation tools provided by Menze and Geiger [2015]. All results are provided at their image coordinates in the first frame at $t = 0$. The error color-coding follows a logarithmic scale, where errors above 3 pixels are colored in red shades and errors below 3 pixels are colored in blue shades.

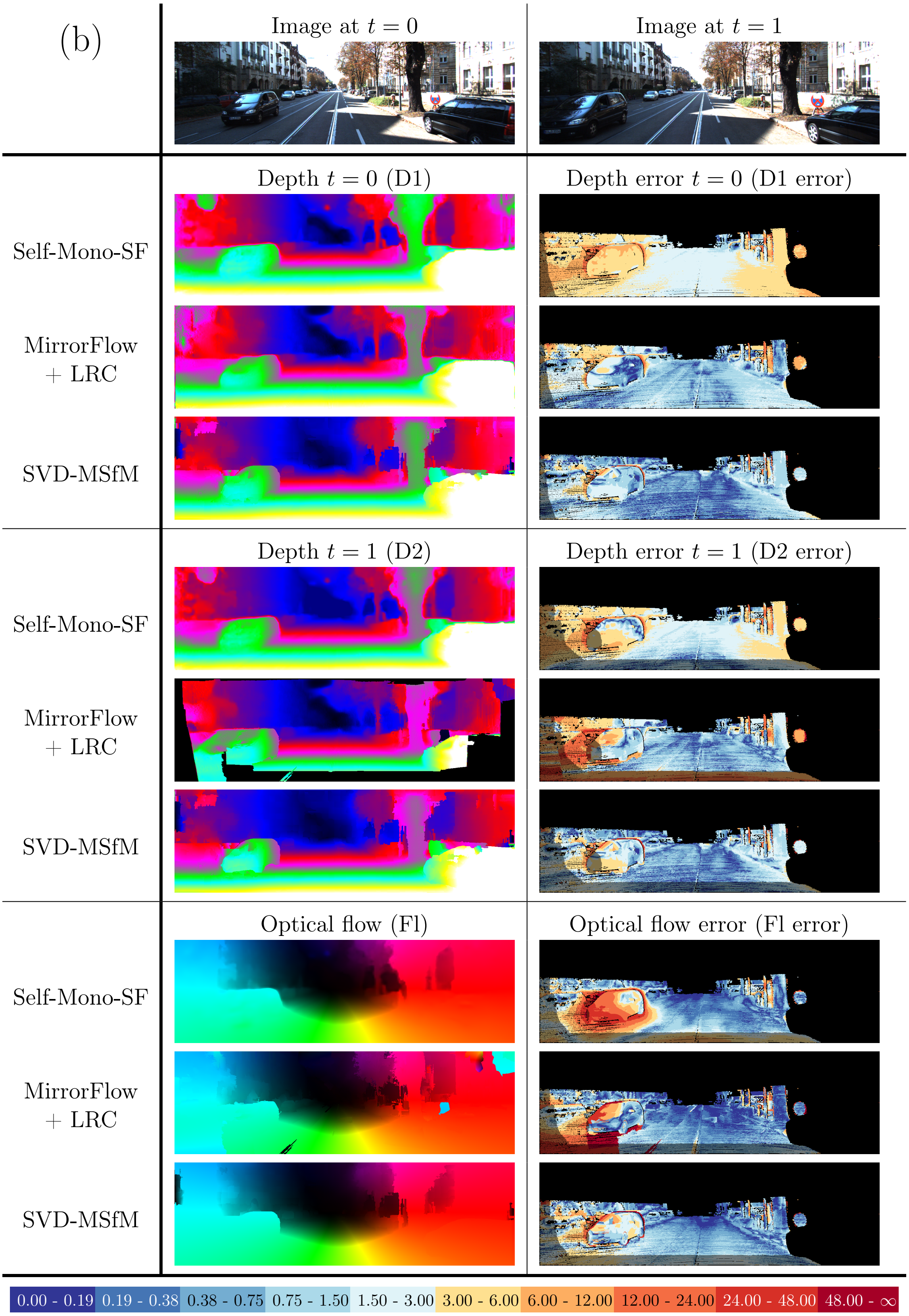
In addition to SVD-MSfM, the results are shown for 'Self-Mono-SF [Hur and Roth, 2020]', which represents the first group, and 'MirrorFlow [Hur and Roth, 2017] + LRC [Godard et al., 2017]', which represents the second group of baseline methods.

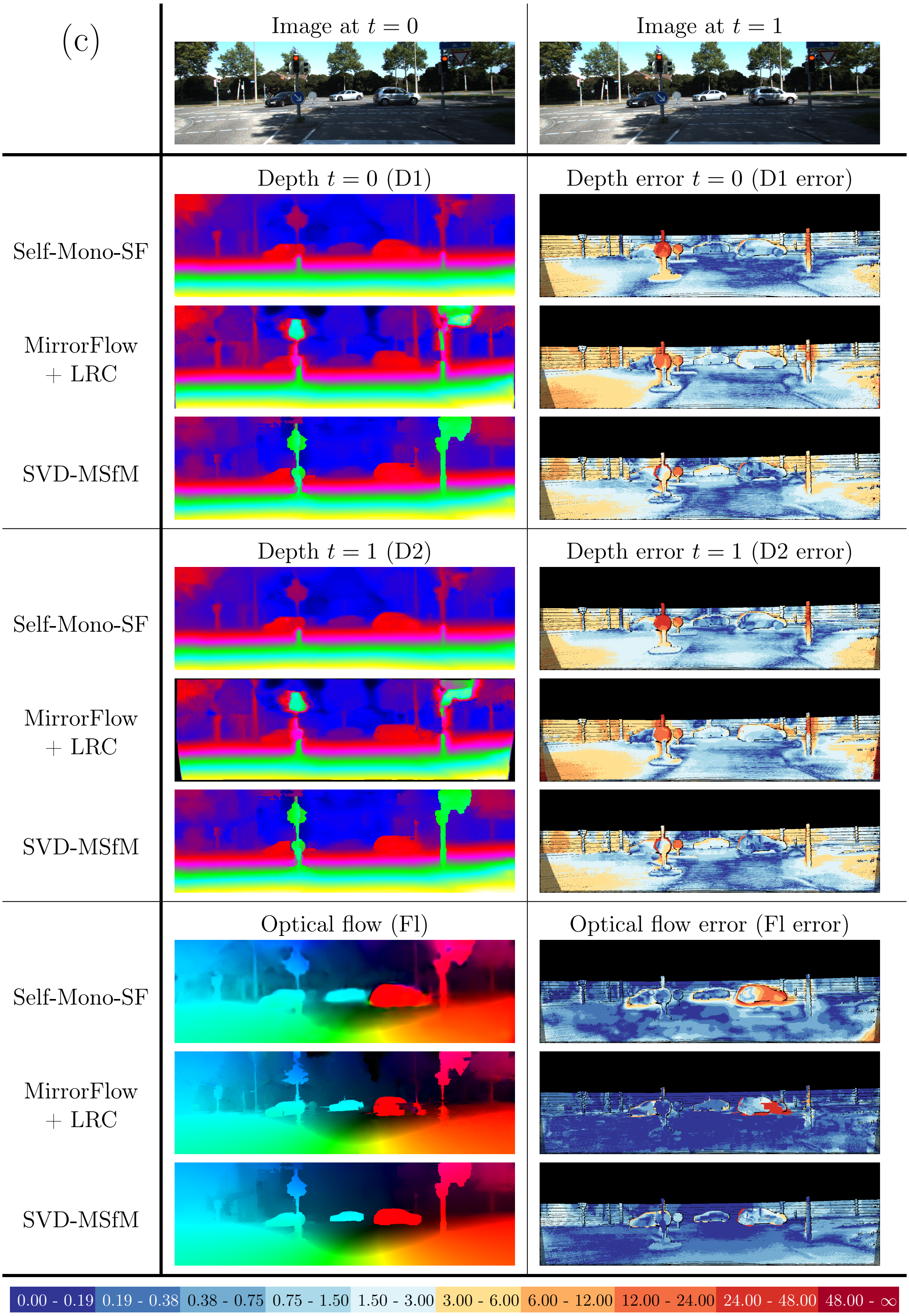
The examples show many vehicles covering several motions: (1) oncoming vehicles (see figure 4.11 (b,c)), (2) preceding vehicles (see figure 4.11 (a,d)), (3) crossing vehicles (see figure 4.11 (c,e,f)), and (4) standing vehicles (see figure 4.11 (b,f)). SVD-MSfM is able to cover all these motions and to provide suitable reconstructions.

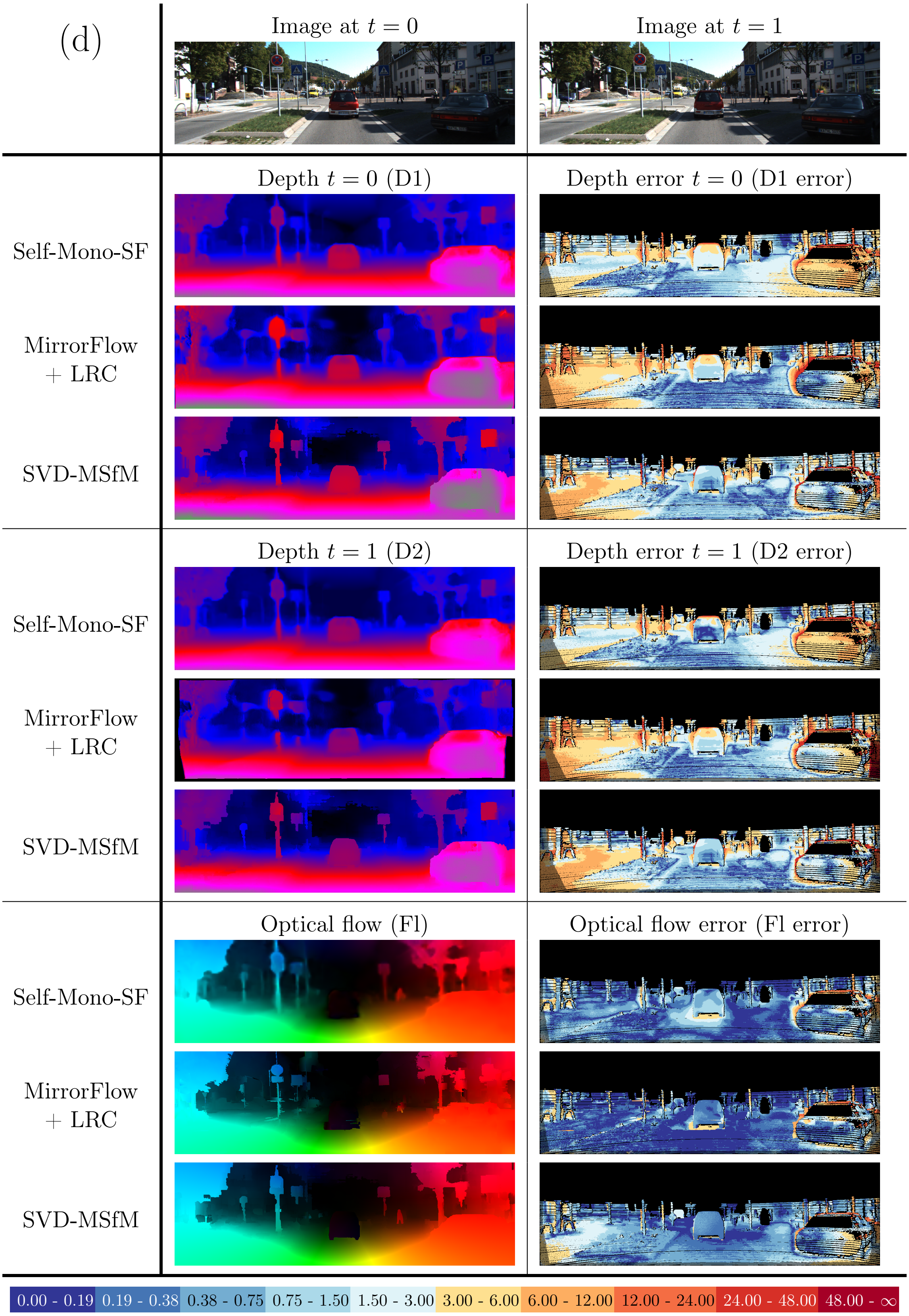
Compared to the baseline methods, SVD-MSfM shows especially an improvement in terms of depth estimation. The examples also show that SVD-MSfM is able to reconstruct thin objects in many cases (e.g. pole and sign in figure 4.11 (c,d)). Even low-textured objects, which are challenging for photometric matching and optical flow estimation, are reconstructed comparatively well (see the white wall in figure 4.11 (e)). SVD-MSfM generalizes to standstill scenarios, whereas the depth estimation degenerates to a single-view depth estimation (see figure 4.11 (f)).

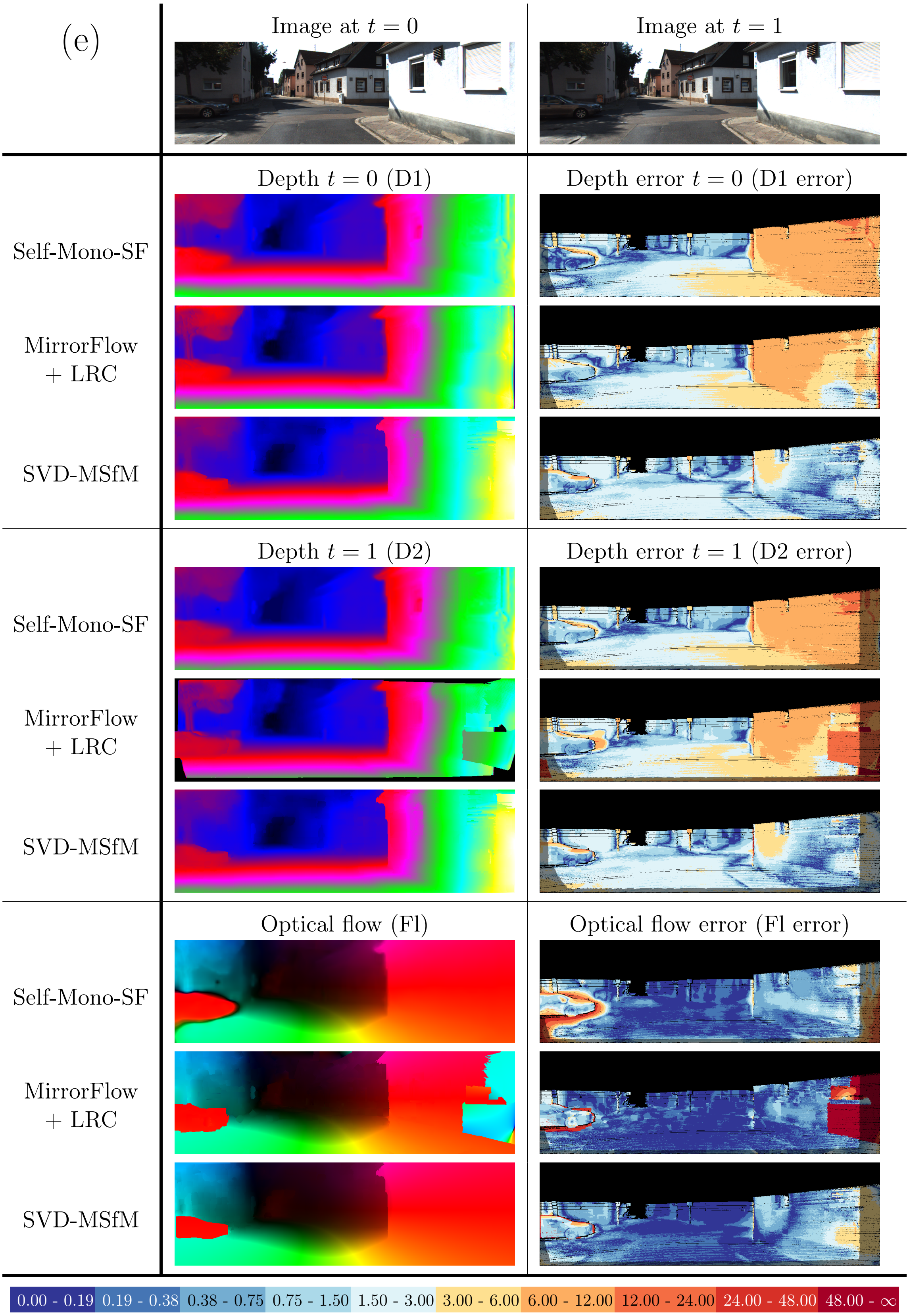
Even though the estimation of Self-Mono-SF is based on an optical flow cost volume, the depth results mainly follow the characteristics of a single-view depth estimation such as higher errors for poles and vegetation (sections 3.2 and 3.4). In contrast to that, SVD-MSfM is also able to handle many of these parts. This implies that the explicit combination of photometric distance and probabilistic single-view depth in a depth cost volume is better suited to integrate and exploit multi-view geometric information.


(a)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Optical flow (Fl)
Optical flow error (Fl error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞


(b)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Optical flow (Fl)
Optical flow error (Fl error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞


(c)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Optical flow (Fl)
Optical flow error (Fl error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞


(d)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Optical flow (Fl)
Optical flow error (Fl error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞


(e)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
Optical flow (Fl)
Optical flow error (Fl error)
Self-Mono-SF
MirrorFlow + LRC
SVD-MSfM
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞

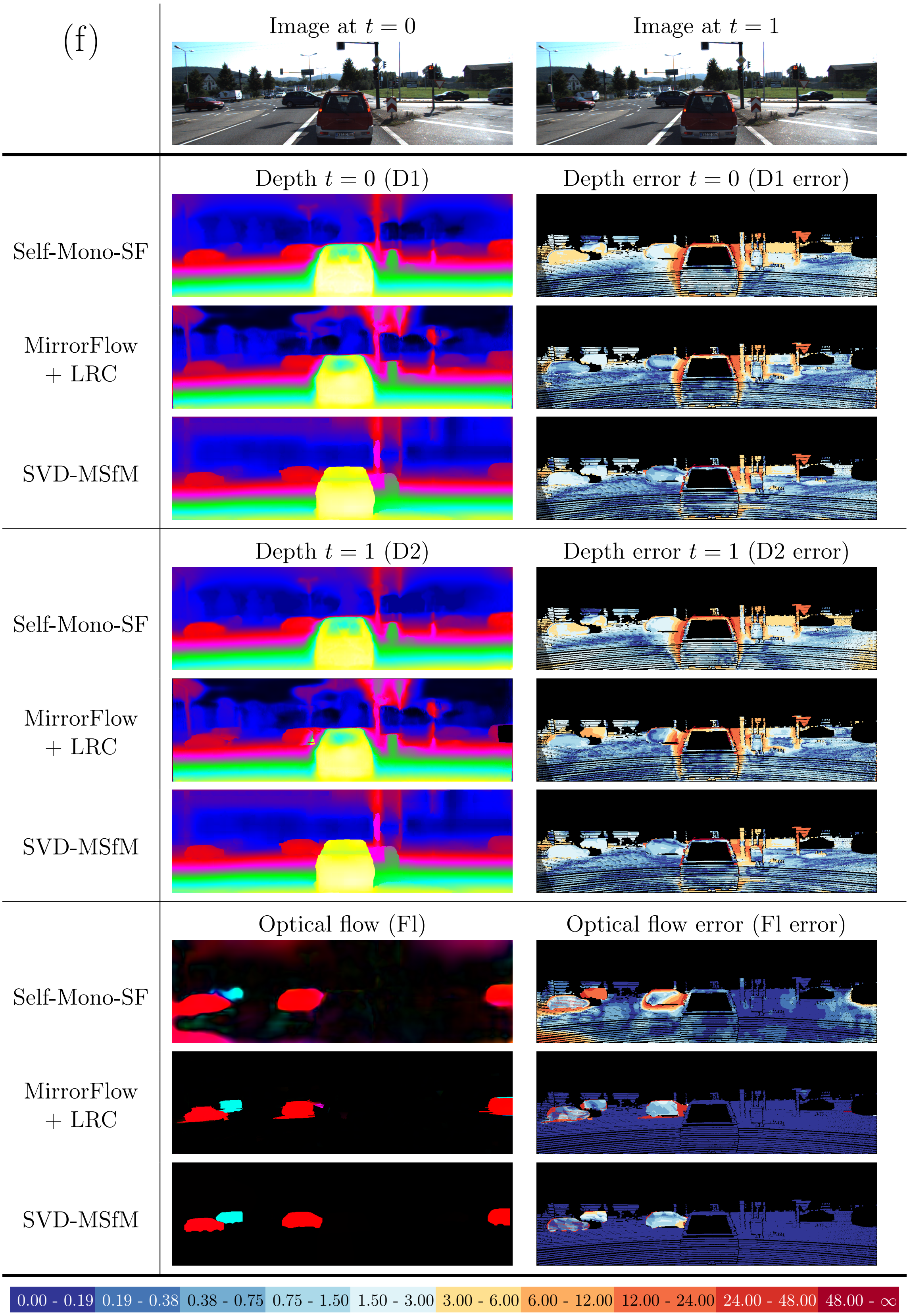


Figure 4.11: Qualitative results of SVD-MSfM in comparison to monocular scene flow baseline methods ('Self-Mono-SF' and 'MirrorFlow + LRC') on the KITTI scene flow training set [Menze and Geiger, 2015]. The color coding represent the estimated depth (from close (warm) to far (cool)), the optical flow (Middlebury color coding [Baker et al., 2011]) or the disparity/ optical flow endpoint error (color coding shown in the legend).

The results also visualize some challenges and limitations. The accuracy mainly decreases and is more prone to errors, where the multi-view photometric consistency does not provide powerful information. First, errors are visible in figure 4.11 (b) for parts that leave the field of view in the subsequent image. Second, the SGM-based smoothing gets more dominant in low-parallax situations (e.g. objects at far distances and situations with low translational motion) and tends to smooth out small or thin objects (see figure 4.11 (c,f)).

4.3.1.3 *Quantitative Evaluation of Scene Flow Estimation*

The quantitative evaluation is based on the KITTI scene flow dataset [Menze and Geiger, 2015], which reports the frequencies of errors for the depth at time $t = 0$ (D1) and $t = 1$ (D2) and the optical flow (Fl) as described in section 4.3.1.1.

Additionally, the mean relative error of the depth estimation capped at 50 meters is reported to be comparable to some baselines. The results are shown in table 4.3. Note that the models of 'HD^3-F [Yin et al., 2019]' and 'DORN [Fu et al., 2018]' used parts of the dataset for training. Therefore, these methods are disregarded for ranking.

Even though the unsupervised multi-task networks (GeoNet [Yin and Shi, 2018], DF-Net [Zou et al., 2018], and Struct2Depth [Casser et al., 2019]) provide reasonable results, integrating stereo or ground truth supervision still improves the accuracy in terms of depth estimation (EPC++ [Luo et al., 2019], Self-Mono-SF [Hur and Roth, 2020], LRC [Godard et al., 2017], and DORN [Fu et al., 2018]). Furthermore, the multi-task networks are not able to outperform depth and optical flow estimation addressed as individual tasks, neither on their respective tasks nor on the combined scene flow metric. The individual single-view depth estimation, LRC, also shows higher accuracy on the depth estimation compared to the MSfM-based methods (DMDE [Ranftl et al., 2016], S.Soup [Kumar et al., 2017], and S.Rel. [Di et al., 2019]) in terms of the MRE metric. As a short side note, the relatively poor accuracy of the DORN method for moving objects is due to the used ground truth of the KITTI depth prediction benchmark [Uhrig et al., 2017].

The proposed approach, SVD-MSfM, shows the best rating on most of the metrics. The evaluation reveals the following characteristics: While the depth estimation of the static environment ('D1-bg') significantly outperforms previous methods, the depth estimation for moving objects ('D1-fg') does not. Moving objects are comparable small to the whole static environment and the depth estimates highly depend on the single-view depth estimates. To illustrate this fact, note that for a single point the depth estimates at both times and the optical flow is a minimal scene flow representation, which disentangles both subtasks. The coupling is based on the fact that many pixels undergo the same projected rigid body motion. Even though a

Method	**MRE**	**D1**			**D2**			**Fl**			**SF**		
		bg	fg	all	bg	fg	all	bg	fg	all	bg	fg	all
GeoNet∗ [Yin and Shi, 2018]	19.41	34.59	62.72	38.90	47.30	65.40	50.07	32.43	67.69	37.83	58.77	90.92	63.69
DF-Net∗ [Zou et al., 2018]	18.90	33.38	59.20	37.34	48.41	57.59	49.81	25.66	37.45	27.47	56.40	77.67	59.66
Struct2Depth∗ [Casser et al., 2019]	14.92	27.29	56.58	31.77	33.25	66.12	38.29	37.86	71.96	43.08	49.98	91.39	56.32
EPC++ [Luo et al., 2019]	-	22.76	26.63	23.84	-	-	-	17.58	26.89	19.64	-	-	-
Self-Mono-SF [Hur and Roth, 2020]	9.98	28.75	45.07	31.25	33.00	45.15	34.86	23.06	25.92	23.49	44.27	62.40	47.05
MirrorFlow [Hur and Roth, 2017] + LRC [Godard et al., 2017]	9.68	25.33	**19.82**	24.48	35.82	**26.15**	34.34	**9.39**	**14.22**	**10.13**	40.55	35.17	39.72
HD^3-F† [Yin et al., 2019] + DORN† [Fu et al., 2018]	11.18	17.02	37.54	20.16	30.08	40.47	31.67	4.01	6.76	4.43	32.57	46.89	34.76
DMDE [Ranftl et al., 2016]	14.6	-	-	-	-	-	-	-	-	-	-	-	-
S.Soup [Kumar et al., 2017]	12.68	-	-	-	-	-	-	-	-	-	-	-	-
S.Rel. [Di et al., 2019]	10.23	-	-	-	-	-	-	-	-	-	-	-	-
MFA [Kumar et al., 2019]	11.82	-	-	-	-	-	-	-	-	-	-	-	-
SVD-MSfM	**8.55**	**17.84**	23.94	**18.77**	**20.37**	26.72	**21.35**	15.31	15.55	15.34	**24.50**	**35.01**	**26.11**

MRE*: mean relative depth error in %;* ***D1, D2****: disparity* ($t = 0, 1$)*;* ***Fl****: optical flow;* ***SF****: scene flow*
D1, D2, Fl, SF*: percentage[%] of estimates that exceed an error threshold* ($> 3px$ *and* $> 5\%$ *of length)*
fg*: foreground (moving objects) ;* ***bg****: background (static environment);* ***all****: bg + fg*
†*: parts of dataset used for training (disregarded for ranking);* ∗*: scaled to align the ground truth*

Table 4.3: Quantitative evaluation of SVD-MSfM with respect to several monocular methods on the KITTI scene flow training set [Menze and Geiger, 2015]. The methods are divided into four groups: First, multi-task CNNs; second, combining optical flow and single-view depth estimation as individual tasks; third, MSfM-based approaches; fourth, combining single-view depth information with multi-view geometry. The separation of the methods belonging to different groups is indicated by two horizontal lines.

moving object is not a single point, it shows the effect that for smaller objects the subtasks are more strongly decoupled with the consequence that the reconstructed depth highly follows the single-view depth estimates. However, SVD-MSfM shows higher accuracy for moving objects on the scene flow metric, which motivates to use the proposed approach also for moving objects.

While the accuracy of the depth estimates predicted to the image at $t = 1$ ('D2') significantly decreases for the baseline methods, it only decreases slightly for SVD-MSfM. This is most likely due to the explicit estimation of motions, which for example could also handle occlusion problems. The optical flow estimation cannot compete with individual SotA optical flow methods. Overall, on the targeted combined

Method	MRE	D1			D2			Fl			SF		
		bg	fg	all	bg	fg	all	bg	fg	all	bg	fg	all
SVD-MSfM (w/o single-view)	73.90	38.98	39.88	39.12	41.65	42.05	41.71	19.09	16.99	18.77	42.98	47.93	43.74
SVD-MSfM (w/o multi-view)	10.19	27.73	**23.04**	27.01	31.17	**26.64**	30.48	26.03	23.24	25.60	37.11	38.68	37.35
SVD-MSfM	**8.55**	**17.84**	23.94	**18.77**	**20.37**	26.72	**21.35**	**15.31**	**15.55**	**15.34**	**24.50**	**35.01**	**26.11**

MRE*: relative depth;* ***D1****,* ***D2****: disparity ($t = 0, 1$);* ***Fl****: optical flow;* ***SF****: scene flow errors*
fg*: foreground (moving objects) ;* ***bg****: background (static environment);* ***all****: bg + fg; all errors in %*

Table 4.4: Ablation study on combining multi-view geometry and single-view depth information for SVD-MSfM. The approach is also performed without integrating single-view depth information ('SVD-MSfM (w/o single-view)') or multi-view photometric consistency ('SVD-MSfM (w/o multi-view)') for scene flow estimation. The results show that both parts contribute to the final performance.

scene flow metric, SVD-MSfM outperforms previous methods by a large margin and reduces the number of errors by around 8 percentage points.

4.3.2 Evaluation of SVD-MSfM Components

The previous section 4.3.1 provides an evaluation of monocular scene flow estimation methods and shows that SVD-MSfM outperforms previous methods in terms of scene flow estimation. The present subsection presents an analysis of SVD-MSfM components and design choices. The first ablation experiment gives evidence for the claimed combination of multi-view geometry and single-view depth information, supports the proposed integration of single-view depth estimation in a probabilistic and well-calibrated form, and shows that the proposed extension of the sparse flow improves scene flow accuracy. The second ablation study additionally evaluates the motion estimation by itself in terms of camera motion estimation and confirms that the proposed approach provides high robustness on short image sequences.

4.3.2.1 *Ablation Studies on Scene Flow Estimation*

Combining multi-view geometry and single-view depth information: The first ablation study is performed to analyze the proposed combination of multi-view photometric distance and probabilistic single-view depth estimation to derive the depth cost volume. Table 4.4 provides the results of the proposed method with a depth cost volume merely based on the multi-view photometric term ('SVD-MSfM (w/o single-view)') or based on the single-view depth distributions ('SVD-MSfM (w/o multi-view)'). The motion estimation is still based on both. The results show that both parts contribute to the final performance. The only exceptions are the 'D1-fg' and 'D2-fg' metrics, where the accuracy with and without the multi-view

Method	**MRE**	**D1**			**D2**			**Fl**			**SF**		
		bg	**fg**	**all**	**bg**	**fg**	**all**	**bg**	**fg**	**all**	**bg**	**fg**	**all**
SVD-MSfM (LRC)	9.44	26.93	24.35	26.53	31.88	28.37	31.34	18.20	19.07	18.33	37.65	37.44	37.62
SVD-MSfM (w/o prob.depth)	9.68	29.99	24.92	29.22	33.89	28.92	33.13	18.34	18.59	18.38	39.99	38.05	39.67
SVD-MSfM (w/o recalib.)	9.04	23.07	**22.91**	23.05	26.21	27.22	26.37	21.80	18.56	21.30	32.16	36.77	32.86
SVD-MSfM	**8.55**	**17.84**	23.94	**18.77**	**20.37**	**26.72**	**21.35**	**15.31**	**15.55**	**15.34**	**24.50**	**35.01**	**26.11**

MRE*: relative depth;* ***D1****,* ***D2****: disparity ($t = 0, 1$);* ***Fl****: optical flow;* ***SF****: scene flow errors*
fg*: foreground (moving objects) ;* ***bg****: background (static environment);* ***all****: bg + fg; all errors in %*

Table 4.5: Ablation study on ProbDepthNet for SVD-MSfM. For integrating single-view depth information, ProbDepthNet is more suitable than LRC for single-view depth estimation (improvement over '(LRC [Godard et al., 2017])'); especially due to the importance of providing single-view depth prediction in a probabilistic (improvement over '(w/o prob. depth)') and well-calibrated form (improvement over '(w/o recalib.)').

photometric distance is nearly the same. This confirms that the depth estimation of moving objects essentially follows the single-view depth estimates as claimed in section 4.3.1.2. However, the multi-view part is still useful to improve the accuracy of the optical flow ('Fl-fg') and scene flow ('SF-fg') estimates of moving objects. The high MRE error without single-view depth information is due to high errors for the standstill scenarios. In general, the multi-view geometric part provides a substantial improvement in terms of the optical flow metric and the single-view depth part improves significantly the depth estimates. However, both kinds of information are able to improve all, the depth, optical flow, and scene flow metrics. This highly motivates the combination for a monocular scene flow estimation.

Integrating probabilistic single-view depth distributions: ProbDepthNet, presented in chapter 3, is designed to provide single-view depth information in a probabilistic and well-calibrated form. The second ablation study is designed to give evidence for the ProbDepthNet design to integrate single-view depth information. Therefore, the results of four SVD-MSfM variants based on different single-view depth estimations are provided in table 4.5. The two SVD-MSfM variants 'SVD-MSfM (LRC)' and 'SVD-MSfM (w/o prob. depth)' use CNNs that provide only the expected depth values instead of depth distributions. While 'SVD-MSfM (LRC)' is based on the LRC method [Godard et al., 2017] for single-view depth estimation, 'SVD-MSfM (w/o prob. depth)' is based on the maximum likelihood estimates of ProbDepthNet derived by the first moment of the distributions. The depth values are integrated by assuming the same Gaussian distribution (determined on a test set) for all pixels. SVD-MSfM based on the probabilistic ProbDepthNet ('SVD-MSfM') integrating depth distributions outperforms both. This supports the claimed ProbDepthNet design to provide single-view depth estimations in a probabilistic form.

Method	**MRE**	**D1**			**D2**			**Fl**			**SF**		
		bg	**fg**	**all**	**bg**	**fg**	**all**	**bg**	**fg**	**all**	**bg**	**fg**	**all**
SVD-MSfM (libviso2-Flow)	8.75	19.00	**23.82**	19.74	21.76	27.70	22.67	18.31	17.35	18.16	28.95	36.02	30.03
SVD-MSfM	**8.55**	**17.84**	23.94	**18.77**	**20.37**	**26.72**	**21.35**	**15.31**	**15.55**	**15.34**	**24.50**	**35.01**	**26.11**

***MRE**: relative depth; **D1**, **D2**: disparity $(t = 0, 1)$; **Fl**: optical flow; **SF**: scene flow errors*
***fg**: foreground (moving objects) ; **bg**: background (static environment); **all**: bg + fg; all errors in %*

Table 4.6: Ablation study on claimed extensions of libviso2-Flow [Geiger et al., 2011]. The approach based on the extended flow outperforms the approach based on the original libviso2-flow ('SVD-MSfM (libviso2-Flow)').

Furthermore, the improvements compared to a variant based on ProbDepthNet excluding CalibNet 'SVD-MSfM (w/o recalib.)' supports that the recalibration technique is an essential component.

Extension of sparse flow estimation: To confirm the claimed extension of the libviso2-based [Geiger et al., 2011] sparse flow, a third ablation study is performed. Therefore, SVD-MSfM is performed without the extensions using the original libviso2-based flow estimation ('SVD-MSfM (libviso2-Flow)'). The results in table 4.6 support that the proposed extensions improve the accuracy of scene flow estimation. While the first extension of exploiting object-wise flow statistics should basically affect the accuracy of foreground objects, the introduced third stage could improve the accuracy of both, static environment and moving objects. This reveals that the large improvement for the static environment is mainly due to the introduced third stage, which additionally exploits single-view depth as prior information for matching the feature candidates.

4.3.2.2 *Evaluation of Motion Estimation*

The previous section 4.3.2 provides several experiments regarding the components and design choices of SVD-MSfM in terms of scene flow estimation. In the present subsection, the motion estimation is evaluated by itself in terms of camera motion estimation. The experiments are based on the KITTI odometry benchmark [Geiger et al., 2012]. This dataset provides 22 sequences with ground truth camera poses acquired by a GPS and inertial measurement unit, which is publicly available for the first 11 sequences. Therefore, a ProbDepthNet model is trained on the sequences 0 to 8 and the evaluation is based on the sequences 9 and 10.

Due to the fact that moving objects frequently appear and disappear, the proposed motion optimization was designed to provide accurate and robust motion estimates given only two images. To derive the full camera pose trajectory for evaluation, these independent two-frame pose estimates are concatenated.

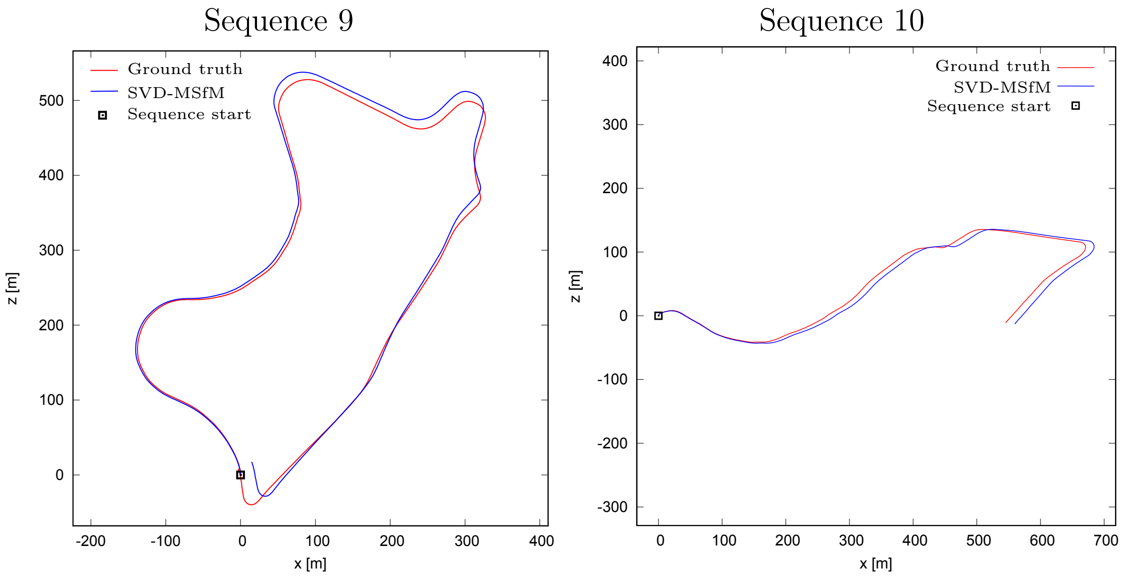


Figure 4.12: Qualitative results of SVD-MSfM's motion estimation in terms of camera pose estimation for the sequences 9 and 10 of the KITTI odometry training set [Geiger et al., 2012].

Even though the pose estimations are performed independently for every two consecutive frames, the results in figure 4.12 still show reasonably high accuracy of the whole camera trajectory.

The quantitative evaluation with respect to SotA monocular methods is shown in table 4.7. The evaluation follows the metric of the KITTI odometry benchmark [Geiger et al., 2012]. For the baseline methods, the results are taken from the corresponding papers, whereby D3VO [Yang et al., 2020] only evaluated the translational error. The pose estimation error is evaluated after driven distances of 100,200,...,800 meters and the results are averaged. The translational error corresponds to the

Method	**Seq. 09**		**Seq. 10**	
	Trans.[%]	**Rot.**[deg/100m]	**Trans.**[%]	**Rot.**[deg/100m]
VISO2-M [Geiger et al., 2011]	4.04	1.43	25.20	3.88
MLM-SFM [Frost et al., 2016]	1.76	0.47	2.12	0.85
PMO [Fanani et al., 2017]	1.31	0.31	2.06	0.46
ORB-SR [Yin et al., 2017]	4.14	0.19	1.70	0.29
DVSO [Yang et al., 2018a]	0.83	0.21	0.74	0.21
D3VO [Yang et al., 2020]	0.78	-	0.62	-
SVD-MSfM	1.65	0.30	2.45	0.45

***Trans.**: relative translational error;* ***Rot.**: rotational error per 100m*
Pose estimation error evaluated and averaged after driven distances of 100,200,...,800m.

Table 4.7: Quantitative evaluation of SVD-MSfM and baseline methods for monocular visual odometry on KITTI odometry benchmark [Geiger et al., 2012].

endpoint error, while the rotation error corresponds to the pose orientations. All baseline methods exploit the long history of camera poses by pose filtering, bundle adjustment (BA), or feature tracking based on prediction (section 4.1). However, the pose estimation of SVD-MSfM is still comparative and only provides slightly lower accuracy than the best approaches.

The following experiments are designed to reveal the performance in situations, where the methods are frequently confronted with initializations and can not exploit a long temporal context. These two characteristics are important to use the motion estimation also for moving objects. Therefore, the pose evaluation follows the proposed metric by Zhou et al. [2017]. The whole sequence is divided into 5-frame snippets. Analogously to the metric described above, the translational and rotational pose error is evaluated and averaged for the 5 poses of the snippet.

SVD-MSfM is evaluated with respect to two categories of monocular baseline methods for camera pose estimation. First, the accuracy of traditional SLAM methods is provided (section 4.1.1): ORB-SLAM [Mur-Artal et al., 2015], LDSO [Gao et al., 2018], and VISO2-M [Geiger et al., 2011]. The accuracy of these methods is stated for the whole trajectory given as input as well as the accuracy if only the 5-frames are given as input. Second, the results of deep learning-based approaches are stated (section 4.1.3): Godard [Godard et al., 2019], SFMLearner [Prasad and Bhowmick, 2019], DF-Net [Zou et al., 2018], Mahjourian [Mahjourian et al., 2017], GeoNet [Yin and Shi, 2018], CC [Ranjan et al., 2019], BeyondPhoto [Shen et al., 2019], and Struct2Depth [Casser et al., 2019]. The results of Godard, DF-Net, Mahjourian, CC, and Struct2Depth are taken from the published papers. For the other methods, the published code or pose estimates are used for evaluation. The baseline methods, apart from VISO2-M [Geiger et al., 2011], only provide pose estimates up to an unknown scale. To provide a fair comparison, the 5-frame trajectories of all methods are scaled to align with the ground truth.

The accuracy of the methods is shown as a histogram in figure 4.13 with the corresponding detailed results in table 4.8. The evaluation reveals that the direct method LDSO and ORB-SLAM, which tracks features based on their prediction, do not provide reasonable results for the 5-frame snippet as input. Similar to the method proposed here, VISO2-M is an indirect method, which builds upon a general sparse flow for pose estimation. The VISO2-M method shows nearly the same accuracy for the whole sequence as for the 5-frame snippet as input, which supports the design choice of using an indirect method and general sparse flow. The deep learning-based methods provide accurate estimates of the translational motion. However, the rotational motion estimation is worse than for SVD-MSfM. The comparison to VISO2-M without aligning the scale to the ground truth is also provided in table 4.8.

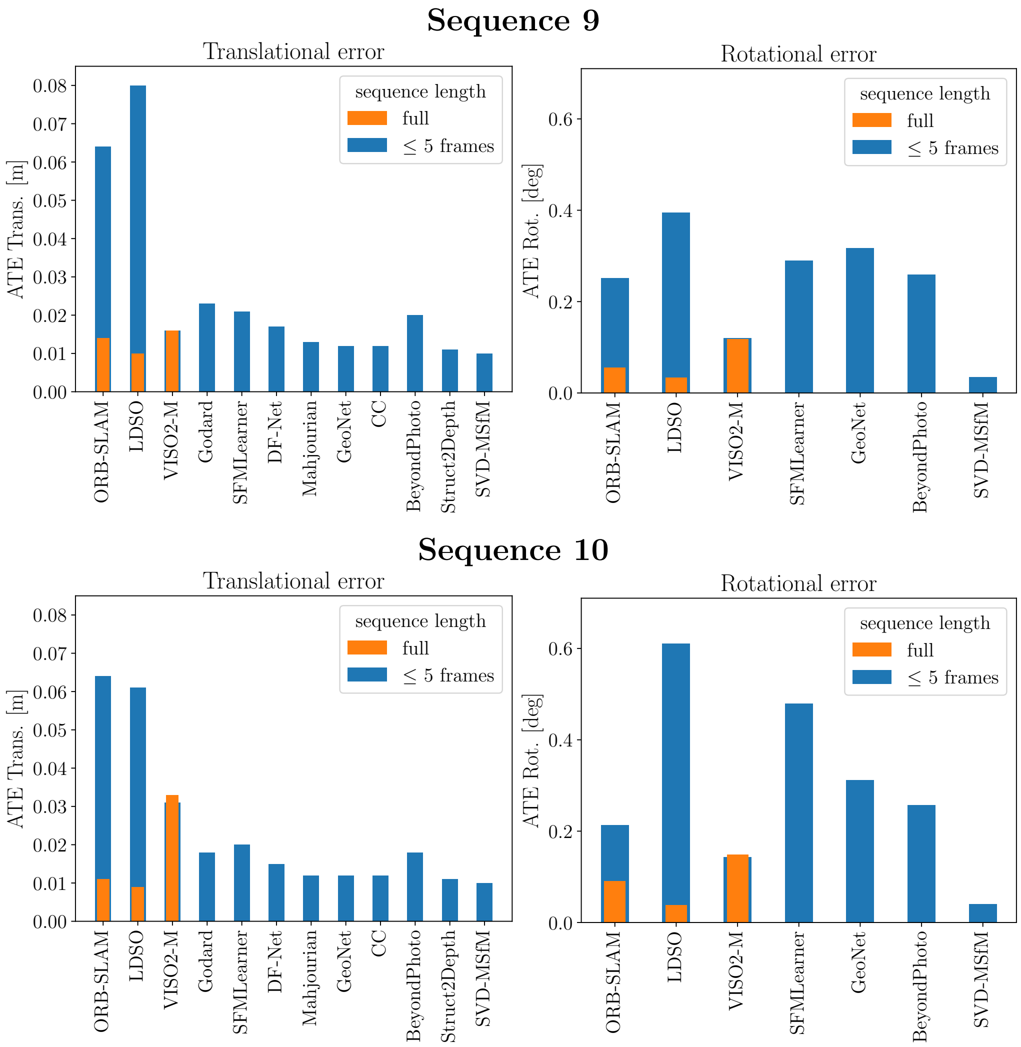


Figure 4.13: Overview of quantitative evaluation of SVD-MSfM in terms of 5-frame pose estimation based on the sequences 09 and 10 of the KITTI odometry dataset [Geiger et al., 2012]. First, the results are shown for the SLAM methods, ORB-SLAM [Mur-Artal and Tardós, 2017], LDSO [Gao et al., 2018], and VISO2-M [Geiger et al., 2011]. These methods are evaluated given the full image sequence as input (orange) and given only the 5-frame snippet (blue). Second, the accuracy of deep learning methods is shown using the following methods: Godard [Godard et al., 2019], SfMLearner [Zhou et al., 2017], DF-Net [Zou et al., 2018], Mahjourian [Mahjourian et al., 2017], GeoNet [Yin and Shi, 2018], CC [Ranjan et al., 2019], BeyondPhoto [Shen et al., 2019], and Struct2Depth [Casser et al., 2019]. The papers originally only evaluated the translational error. Thus, the rotational error is only shown for methods that provided their estimates or code.

These results support a high accuracy of the proposed scale estimation – significantly better than the ground plane-based scale estimation of VISO2-M.

Method	Input	Seq. 09		Seq. 10	
		Trans.[m]	**Rot.**[deg]	**Trans.**[m]	**Rot.**[deg]
VISO2-M [Geiger et al., 2011]	full	0.016 ± 0.008	0.118 ± 0.077	0.033 ± 0.056	0.148 ± 0.193
	5 frames	0.016 ± 0.008	0.120 ± 0.075	0.031 ± 0.049	0.143 ± 0.183
ORB-SLAM [Mur-Artal et al., 2015]	full	0.014 ± 0.007	0.055 ± 0.241	0.011 ± 0.008	0.090 ± 0.027
	5 frames	0.064 ± 0.141	0.251 ± 0.733	0.064 ± 0.130	0.213 ± 0.539
LDSO [Gao et al., 2018]	full	0.010 ± 0.006	0.033 ± 0.022	0.009 ± 0.008	0.038 ± 0.022
	5 frames	0.080 ± 0.180	0.395 ± 1.042	0.061 ± 0.137	0.610 ± 1.562
Godard [Godard et al., 2019]	2 frames	0.023 ± 0.013	n.a.	0.018 ± 0.014	n.a.
SfMLearner [Zhou et al., 2017]	5 frames	0.021 ± 0.017	0.289 ± 0.422	0.020 ± 0.015	0.479 ± 0.603
DF-Net [Zou et al., 2018]	5 frames	0.017 ± 0.007	n.a.	0.015 ± 0.009	n.a.
Mahjourian [Mahjourian et al., 2017]	3 frames	0.013 ± 0.010	n.a.	0.012 ± 0.011	n.a.
GeoNet [Yin and Shi, 2018]	5 frames	0.012 ± 0.007	0.317 ± 0.166	0.012 ± 0.009	0.311 ± 0.165
CC [Ranjan et al., 2019]	5 frames	0.012 ± 0.007	n.a.	0.012 ± 0.008	n.a.
BeyondPhoto [Shen et al., 2019]	3 frames	0.020 ± 0.010	0.259 ± 0.217	0.018 ± 0.014	0.257 ± 0.257
Struct2Depth [Casser et al., 2019]	3 frames	0.011 ± 0.006	n.a.	0.011 ± 0.010	n.a.
SVD-MSfM	2 frames	**0.010 ± 0.006**	**0.034 ± 0.015**	**0.010 ± 0.008**	**0.040 ± 0.020**
VISO2-M∗ [Geiger et al., 2011]	full	0.062 ± 0.060	0.118 ± 0.077	0.306 ± 0.948	0.148 ± 0.193
	5 frames	0.062 ± 0.061	0.119 ± 0.075	0.288 ± 0.895	0.143 ± 0.183
SVD-MSfM∗	2 frames	**0.036 ± 0.024**	**0.034 ± 0.015**	**0.039 ± 0.029**	**0.040 ± 0.020**

***Trans.**: average translational error; **Rot.**: average rotational error*

∗: Method provide scale-aware estimates and is therefore not scaled to align the ground truth

Pose estimation error evaluated and averaged for all poses of 5-frame snippet

Results that take the full trajectory as input are disregarded for ranking

Table 4.8: Quantitative evaluation of SVD-MSfM and baseline methods for camera pose estimation based on the sequences 09 and 10 of the KITTI odometry dataset [Geiger et al., 2012]. The evaluation is performed for short camera trajectories of 5-frames. The results show the mean and variance of the absolute trajectory error (ATE) in terms of rotation (in degree) and translation (in meters). While SLAM-methods formulated as an optimization or filtering problem are considered as the first group of baseline methods, the second group of baseline methods corresponds to deep learning-based approaches. The separation of the methods belonging to different groups is indicated by two horizontal lines. The trajectories of all methods in the upper part are scaled to align the ground truth, while the methods in the lower part provide scale-aware estimates by itself (indicated by an ∗).

4.4 Conclusion

The present chapter presented a novel method, SVD-MSfM, which combines multi-view geometry with single-view depth estimation in a MSfM-based concept using a joint depth cost volume, which leads to new SotA accuracy for monocular scene flow estimation. The motion estimation of SVD-MSfM is confirmed to perform robustly on short image sequences. Even more, it provides a novel concept of providing scale-aware motion estimates also for moving objects by integrating single-view depth information. A depth cost volume, which comprises multi-view photometric consistency with probabilistic single-view depth distributions, is used as a basis to derive dense depth estimates. The experiments clearly motivate the claimed combination of multi-view geometry with single-view depth distributions because both provide powerful information for a scene flow estimation task. Additionally, the experiments show how to integrate single-view depth information in a suitable way and confirm the claimed importance of integrating single-view depth as well-calibrated distributions – the main aspects ProbDepthNet is designed for.

5 MONOCULAR INSTANCE SCENE FLOW

CONTENTS

This chapter extends parts of the work that has been published previously in [Brickwedde et al., 2019].

The proposed approach, SVD-MSfM, presented in chapter 4 is divided into the consecutive steps of motion and depth estimation. However, the tasks of motion and depth estimation highly depend on each other. For example, stereo-based scene flow methods [Menze and Geiger, 2015, Menze et al., 2018, Behl et al., 2017] show the advantage of a joint optimization.

Therefore, I propose in the present chapter a monocular scene flow estimation, called *Mono-SF*, formulated as a *joint optimization* of the motion and depth structure combining multi-view geometry with probabilistic single-view depth estimates provided by ProbDepthNet. Following previous stereo-based scene flow methods, the scene is represented by a set of 3D planar surface elements and 6D motions of rigid bodies. A rigid body is either the background or a potentially moving object.

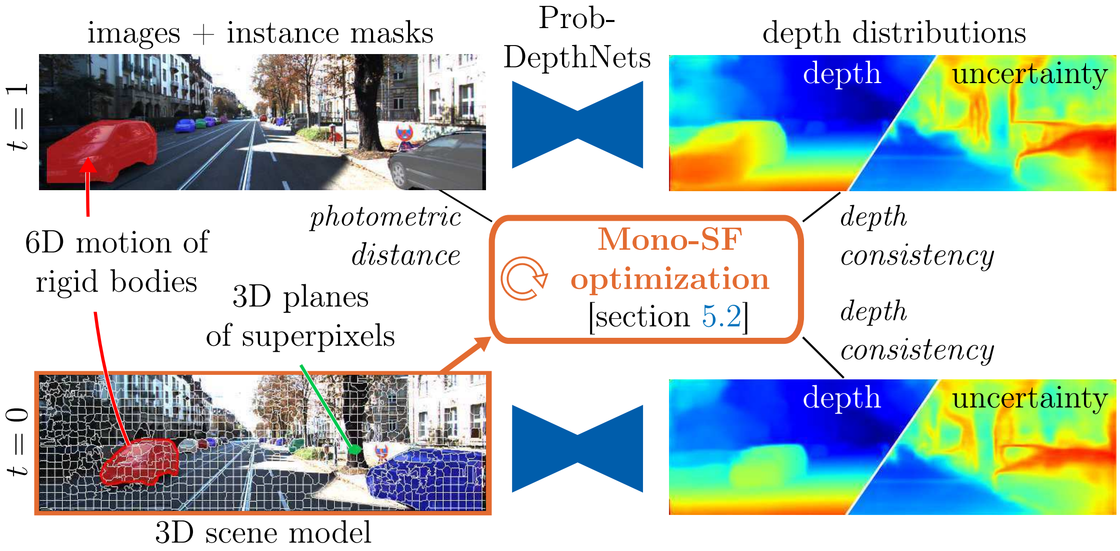


Figure 5.1: Overview of Mono-SF for monocular scene flow estimation. Mono-SF jointly optimizes the 3D geometry of a set of planes and the 6D motion of rigid bodies considering (1) a photometric distance by warping the reference image into the subsequent image, (2) single-view depth distributions provided by ProbDepthNet, and (3) scene model smoothness priors.

Mono-SF jointly optimizes the 3D geometry of each plane and the 6D motion of each rigid body considering (1) the multi-view geometry by warping the reference image into the subsequent image, (2) probabilistic single-view depth estimates, and (3) scene model smoothness priors (see figure 5.1). The scene flow estimation is formulated as a non-linear energy minimization problem, which is optimized in an iterative scheme and initialized with the outputs of SVD-MSfM.

The Mono-SF approach is evaluated with respect to several state of the art (SotA) monocular baselines and an ablation study confirms the importance of the individual components of the proposed optimization framework. The joint optimization formulated as a scene flow estimation problem provides a significant improvement compared to SVD-MSfM, which is used for initialization. The suitability of ProbDepthNet for integrating single-view depth information in Mono-SF is confirmed, especially due to the importance of providing single-view depth information as well-calibrated depth distributions. Furthermore, Mono-SF was the first monocular method published on the KITTI scene flow dataset [Menze and Geiger, 2015].

5.1 Related Work

The Mono-SF method is a monocular scene flow estimation combining multi-view geometry with probabilistic single-view depth. The reader is referred to section 4.1 for an overview of monocular scene reconstruction and to section 3.1 for the related works in terms of probabilistic single-view depth estimation. This chapter additionally covers *stereo-based scene flow* methods, which inspired the Mono-SF scene model and formulation as a joint optimization of motion and depth structure.

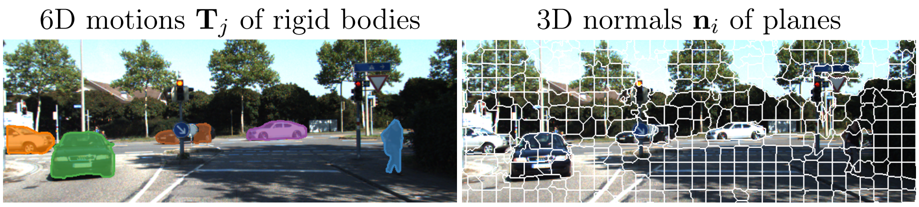
Scene flow estimation was introduced by Vedula et al. [1999, 2005] as a joint optimization of 3D geometry and motion of the scene based on a sequence of stereo images. Mostly variational approaches were used subsequently to extend the scene flow concept [Huguet and Devernay, 2007, Pons et al., 2007, Wedel et al., 2008, 2011, Valgaerts et al., 2010, Basha et al., 2013, Herbst et al., 2013].

However, Vogel et al. [2013] were the first that significantly outperformed individual stereo and optical flow methods on their respective tasks for dynamic traffic scenes. They represented the dynamic scene as a collection of rigid moving planar surface elements, each one comprising its 3D scaled normal to represent the geometry and its 6D motion parameters. Considering two consecutive stereo image pairs, four images are given. Initially, the reference image is divided into surface elements using a superpixel segmentation. The geometry and the motion of each plane are optimized using energy minimization. The plane parameters are optimized to minimize a photometric distance by warping each plane of the reference image into the other images. Additionally, scene model priors prefer a smooth structure in terms of depth, orientation, and motion.

A traffic scene, in particular, consists of a few independent motions by vehicles and other objects. Instead of individual motion estimation of each plane, Menze and Geiger [2015] formulated the problem by a set of rigid moving objects. Each plane is associated with one rigid moving object, which is represented by its 6D motion parameters. Consequently, the scene flow estimation is formulated as a joint optimization of the 6D rigid body motion parameters, 3D plane parameters and association of planes to objects. This representation is particularly beneficial if the association of planes to objects is supported by an instance segmentation as proposed in [Behl et al., 2017].

The currently leading[1] approach [Ma et al., 2019] on the KITTI scene flow benchmark [Menze and Geiger, 2015] follows the decomposition of the scene into rigid moving objects. In contrast to previous methods, the estimation is based on an end-to-end trainable convolutional neural network (CNN) architecture, which directly optimizes the depth of each pixel instead of the geometry of planar surface elements. Based on initial deep learning-based disparity, optical flow, and instance mask estimates, a Gauss-Newton solver jointly optimizes the depth of each pixel and the rigid body motions as a scene flow optimization problem. The Gauss-Newton solver is implemented as a recurrent neural network, which enables to train the network in an end-to-end fashion.

[1] Benchmark on January 06, 2020. Methods with a publication are considered.



6D motions $\mathbf{T}_j$ of rigid bodies

3D normals $\mathbf{n}_i$ of planes

Figure 5.2: Variables of Mono-SF model and energy minimization problem are the 6D rigid body motions $\mathbf{T}_j$ of moving objects (colored in the left image) and the background as well as the 3D scaled normals $\mathbf{n}_i$ of superpixel planes (boundaries in the right image).

Mono-SF corresponds to the *object* or *instance scene flow* [Menze and Geiger, 2015, Behl et al., 2017] model formulated as an energy minimization problem. In contrast to these methods, Mono-SF uses only monocular images by integrating probabilistic single-view depth estimates instead of the right stereo images.

5.2 Mono-SF Method

The present section presents the *Mono-SF* optimization framework structured as follows: First, the decomposition of the scene into piecewise planar surface elements and rigid bodies is described. Second, the optimization is formulated as an energy minimization problem combining (1) multi-view geometry-based photometric distance, (2) the probabilistic single-view depth estimates of ProbDepthNet, and (3) scene model smoothness priors. Finally, the inference process and initialization of the optimization problem are explained.

5.2.1 Scene Model

Following previous object scene flow approaches [Menze and Geiger, 2015, Behl et al., 2017, Menze et al., 2018], the main assumption is that a traffic scene can be approximated by a set of piecewise planar surface elements to represent the structure of the scene and a set of rigid bodies to represent the motion (see figure 5.2). The reference image is divided into a set of superpixels each one representing a 3D plane. Each 3D plane is defined by its normal $\mathbf{n}_i \in \mathbb{R}^3$, scaled by the inverse distance of the plane to the camera to encode the 3D position $\mathbf{X}$ of each point on the plane by $\mathbf{n}_i^T\mathbf{X} = 1$. The set of rigid bodies consists of the background as well as other traffic participants such as pedestrians or vehicles detected by an instance segmentation. Even though a pedestrian does not undergo a rigid body motion, at a certain scale, it can be approximated by its dominant rigid body transformation as motivated by

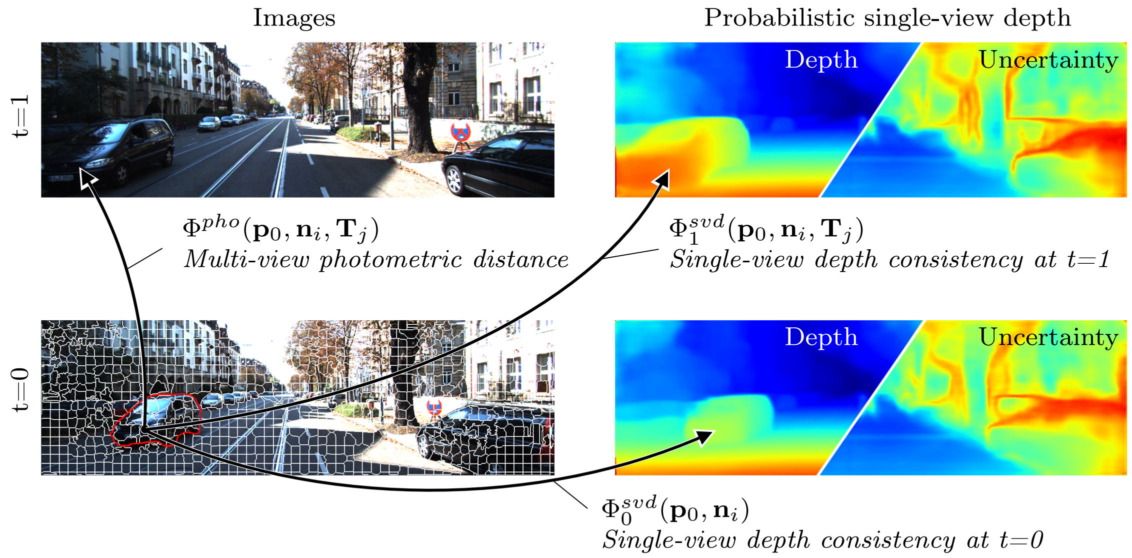

Images
Probabilistic single-view depth
t=1
t=0
Depth
Uncertainty
Depth
Uncertainty
$\Phi^{pho}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$
Multi-view photometric distance
$\Phi_1^{svd}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$
Single-view depth consistency at t=1
$\Phi_0^{svd}(\mathbf{p}_0, \mathbf{n}_i)$
Single-view depth consistency at t=0

Figure 5.3: Illustration of unary data terms of Mono-SF energy minimization problem. The unary terms comprises for each pixel $\mathbf{p}_0$ an appearance-based photometric distance Φ^{pho} and the consistency to the probabilistic single-view depth estimates for both images Φ_0^{svd} and Φ_1^{svd}. The normal vector $\mathbf{n}_i$ is defined by the corresponding plane (white boundaries). The transformation $\mathbf{T}_j$ is defined by the corresponding rigid body (red boundary).

Menze et al. [2018]. Each rigid body is represented by its 6D motion $\mathbf{T}_j \in SE(3)$. Additionally, each superpixel is associated with one rigid body and with the pixels $\mathcal{R}_i$ of the corresponding superpixel.

5.2.2 Energy Minimization Problem

The main idea of Mono-SF is that the scene geometry and motion should be consistent in terms of warping the reference image I_0 in the target image I_1 and consistent to the inverse depth distributions $p(\rho \mid I_0)$ and $p(\rho \mid I_1)$ provided by ProbDepthNet. Formally, Mono-SF jointly optimizes the 6D motion of each rigid body $\mathbf{T}_j$ and 3D normal of each plane $\mathbf{n}_i$ as an energy minimization problem, which corresponds to a maximum a posteriori probability estimation. The energy term E consists of unary data terms $\Phi(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$ for each pixel $\mathbf{p}_0$ and pairwise smoothness terms $\Psi(\mathbf{n}_k, \mathbf{n}_l)$ for each two planes $\mathbf{n}_k$ and $\mathbf{n}_l$ adjacent in the image $k, l \in \mathcal{N}$:

$$E = \sum_{\mathbf{n}_i} \sum_{\mathbf{p}_0 \in \mathcal{R}_i} \Phi(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) + \sum_{k,l \in \mathcal{N}} \Psi(\mathbf{n}_k, \mathbf{n}_l) \tag{5.1}$$

The transformation $\mathbf{T}_j$ corresponds to the rigid body assigned to the plane $\mathbf{n}_i$.

The unary terms $\Phi(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$ consist of two parts as shown in figure 5.3. First, $\Phi^{pho}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$ minimizes an appearance-based photometric distance between pixel $\mathbf{p}_0$ and its projected position in the subsequent image. Second, $\Phi_t^{svd}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$ prefers a 3D position consistent to the depth probabilities of ProbDepthNet at time $t = 0$ and $t = 1$:

$$\Phi(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) = \lambda_1 \cdot \Phi^{pho}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) + \lambda_2 \cdot \sum_{t \in \{0,1\}} \Phi_t^{svd}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) \tag{5.2}$$

The terms are weighted by λ_1 or λ_2, respectively. The photometric distance $\Phi^{pho}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$ rates the similarity of the two corresponding image positions $\mathbf{p}_0$ and $\mathbf{p}_1$ as the Hamming distance of their respective 5 × 5 Census descriptors [Zabih and Woodfill, 1994] truncated at τ_1. The corresponding image coordinates $\mathbf{p}_1$ in the second image I_1 are defined by a homography (section 2.1.3) considering the 3D normal $\mathbf{n}_i$ and the motion of the corresponding rigid body $\mathbf{T}_j$:

$$\mathbf{p}_1 = \mathbf{K}(\mathbf{R}_j - \mathbf{t}_j \mathbf{n}_i^T)\mathbf{K}^{-1}\mathbf{p}_0 \tag{5.3}$$

The rotation matrix $\mathbf{R}_j$ and translation vector $\mathbf{t}_j$ refer to the decomposition of $\mathbf{T}_j$. The matrix $\mathbf{K}$ is the intrinsic camera matrix.

The term $\Phi_t^{svd}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$ rates the consistency of the depth of pixel $\mathbf{p}_0$ based on the ProbDepthNet estimates. While the inverse depth $\rho_0(\mathbf{p}_0, \mathbf{n}_i)$ at time $t = 0$ is directly defined by the corresponding scaled normal vector $\mathbf{n}_i$, the motion of the corresponding rigid body $\mathbf{T}_j$ needs to be considered to derive the inverse depth $\rho_1(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$ at time $t = 1$. Both depth values are rated by the negative log-likelihood (NLL) of the probability provided by ProbDepthNet for their respective image I_t and image coordinates $\mathbf{p}_t$:

$$\Phi_t^{svd}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) = -\log p_{\mathbf{p}_t}\left(\rho_t(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) \mid I_t\right) \tag{5.4}$$

The image coordinates $\mathbf{p}_1$ are again defined as in equation (5.3).

The previous data terms include the single-view depth information and multi-view geometry-based photometric distance. Additionally, scene model priors visualized in figure 5.4 are integrated similar to [Menze and Geiger, 2015] as pairwise smoothness terms $\Psi(\mathbf{n}_k, \mathbf{n}_l)$ preferring a smooth structure in terms of depth $\Psi^d(\mathbf{n}_k, \mathbf{n}_l)$ and orientation $\Psi^{ori}(\mathbf{n}_k, \mathbf{n}_l)$, each part weighted by λ_3 or λ_4:

$$\Psi(\mathbf{n}_k, \mathbf{n}_l) = \lambda_3 \cdot \Psi^d(\mathbf{n}_k, \mathbf{n}_l) + \lambda_4 \cdot \Psi^{ori}(\mathbf{n}_k, \mathbf{n}_l) \tag{5.5}$$

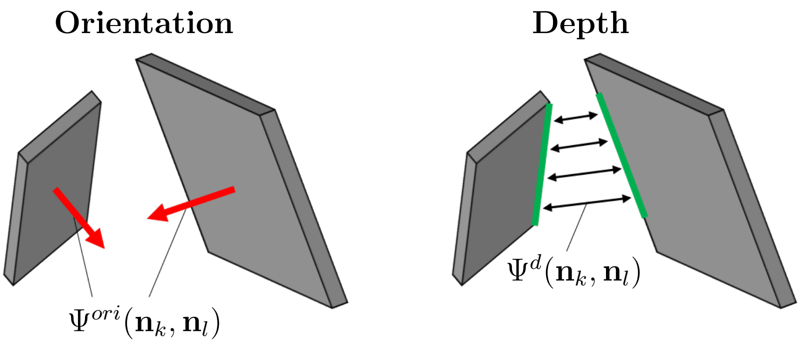

Orientation
Depth
$\Psi^{ori}(\mathbf{n}_k, \mathbf{n}_l)$
$\Psi^{d}(\mathbf{n}_k, \mathbf{n}_l)$

Figure 5.4: Illustration of Mono-SF smoothness priors. The images show two adjacent surface elements projected into the 3D space based on the normal vectors $\mathbf{n}_k$ and $\mathbf{n}_l$. The first prior $\Psi^{ori}(\mathbf{n}_k, \mathbf{n}_l)$ is based on the direction of the normal vectors (red arrows in left picture) and rates the similarity of the orientations of two adjacent planes as a cosine similarity of the normal vectors. The second prior $\Psi^{d}(\mathbf{n}_k, \mathbf{n}_l)$ is based on the parts of the boundaries, which correspond to shared boundary pixels of both planes in the 2D image, e.g. the green boundary part in the right picture. This prior is defined as the sum of differences of the inverse depths values of corresponding boundary pixels. Both priors are defined to achieve a smooth structure in terms of orientation and depth.

For each shared boundary pixel $\mathbf{p}_0 \in \mathcal{B}_{k,l}$ of plane $\mathbf{n}_k$ and $\mathbf{n}_l$, a difference in depth is penalized:

$$\Psi^{d}(\mathbf{n}_k, \mathbf{n}_l) = \sum_{\mathbf{p}_0 \in \mathcal{B}_{k,l}} \min\left(|\rho_0(\mathbf{p}_0, \mathbf{n}_k) - \rho_0(\mathbf{p}_0, \mathbf{n}_l)|, \tau_2\right) \tag{5.6}$$

Analogously, a smooth orientation of planes adjacent in the image is preferred by measuring the cosine similarity of the normal vectors $\mathbf{n}_k$ and $\mathbf{n}_l$:

$$\Psi^{ori}(\mathbf{n}_k, \mathbf{n}_l) = \min\left(1 - \frac{|\mathbf{n}_k \mathbf{n}_l|}{||\mathbf{n}_k|| ||\mathbf{n}_l||}, \tau_3\right) \tag{5.7}$$

Both smoothness terms are truncated by τ_2 or τ_3 to regard discontinuities in the depth or orientation, for example between different objects. The hyper-parameters λ and τ are defined differently according to the rigid body type, background or object, and differently for adjacent planes belonging to different rigid bodies. These dependencies are neglected in the previous equations for ease of reading.

5.2.3 Initialization and Inference

The Mono-SF model is related to the formulation and scene model assumptions of SVD-MSfM. The main difference lies in the joint optimization. SVD-MSfM proposed the motion and depth estimation as two consecutive steps. A representation as planar surface elements is achieved by subsequently applying the slanted-plane smoothing (SPS) method [Yamaguchi et al., 2014] as part of SVD-MSfM. In contrast to that, Mono-SF jointly optimizes the motions and depth using directly a representation of surface elements. This also allows to directly integrate respective scene model priors.

The resulting non-linear optimization problem defined in equation (5.1) requires a suitable initialization to apply an iterative optimization approach. Therefore, SVD-MSfM is still essential to provide this initialization. Following the proposed optimization of the object scene flow methods [Menze and Geiger, 2015, Behl et al., 2017], sequential tree-reweighted message passing (section 2.2.2) is applied for optimizing the energy minimization problem. The continuous optimization problem is converted into a discrete labeling problem by creating samples for each optimized variable. The set of planes and rigid bodies as well as the association of planes to rigid bodies remain fixed during optimization. 5 particles for each 6D rigid body motion are generated by Gaussian sampling around the current solution. For the 3D normal vectors, 5 particles are also derived by Gaussian sampling and another 5 particles are defined by a neighboring surface element such that they represent the same scene plane, which would minimize the smoothness priors. This optimization based on the sequential tree-reweighted message passing is iteratively repeated for 10 times with newly generated sampled based on the current solution.

5.3 Experimental Evaluation of Mono-SF Method

Mono-SF optimizes the 3D position and motion of a dynamic traffic scene formulated as a joint scene flow estimation problem. The first section 5.3.1 provide qualitative results and a quantitative evaluation with respect to several monocular methods including SVD-MSfM described in chapter 4. Additionally, the second section 5.3.2 presents analyzes regarding Mono-SF components and design choices. First, the experiments give evidence for the importance of providing single-view depth information as well-calibrated distributions as provided by ProbDepthNet. Second, each part of the energy minimization problem is confirmed to provide a contribution to the final accuracy.

5.3.1 Evaluation of Scene Flow Estimation

The present subsection evaluates Mono-SF in terms of monocular scene flow estimation. The experimental setup is similar to the experiments regarding SVD-MSfM. It is briefly summarized here as a reminder. For a more detailed description the reader is referred to section 4.3.1.

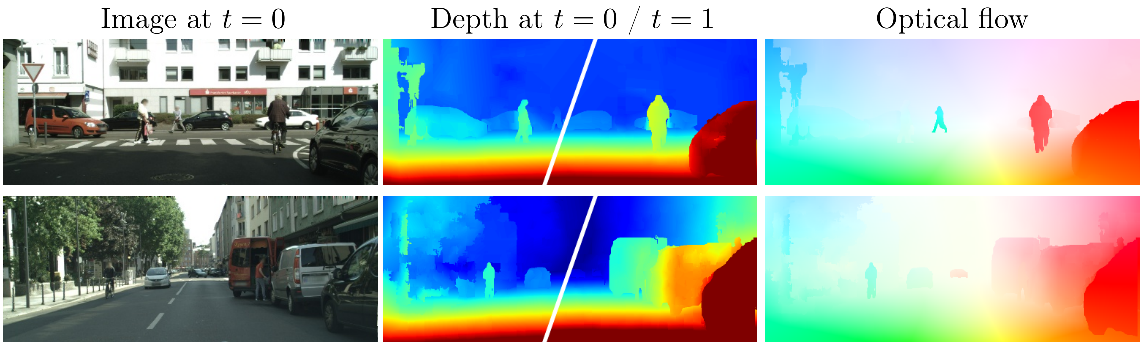


Figure 5.5: Qualitative results of Mono-SF on a crop of Cityscapes (removing car hood); left: first input image, middle: estimated depth values at time $t = 0$ (left half) and $t = 1$ (right half), right: estimated optical flow

The scene flow results and evaluations are based on the equivalent representation as the depth of each pixel at both times ($t = 0$, $t = 1$) and the optical flow. Thereby, the 3D position and the ability of the approaches to predict a 3D point from $t = 0$ to $t = 1$ based on its 3D motion is evaluated. The quantitative evaluation is based on the KITTI scene flow dataset [Menze and Geiger, 2015], which reports the frequencies of errors for the depth at time $t = 0$ (D1) and $t = 1$ (D2) and the optical flow (Fl). An estimate is considered as a valid scene flow estimate (SF) if it fulfills all the D1, D2, and Fl metrics. All metrics are evaluated separately for moving objects (fg), the static scene (bg), and both combined (all).

The results are compared to four categories of methods. First, the results of multi-task networks (section 4.1.3) are described. Second, the combination of single-view depth and optical flow estimation as individual tasks is considered as an additional group. Third, the evaluation covers multi-body structure from motion (MSfM)-based methods (section 4.1.2). Fourth, methods that combine single-view depth information with multi-view geometry represents the fourth group of methods including the proposed methods SVD-MSfM and Mono-SF.

5.3.1.1 *Qualitative Results of Scene Flow Estimation*

In the present subsection, results of Mono-SF are provided for data samples of the Cityscapes dataset [Cordts et al., 2016] and KITTI scene flow training set [Menze and Geiger, 2015]. The figure 5.5 shows the results on the Cityscapes dataset. In addition to the static environment and vehicles, also pedestrians and cyclists, which do not undergo an ideal rigid body motion, are reconstructed reasonably.

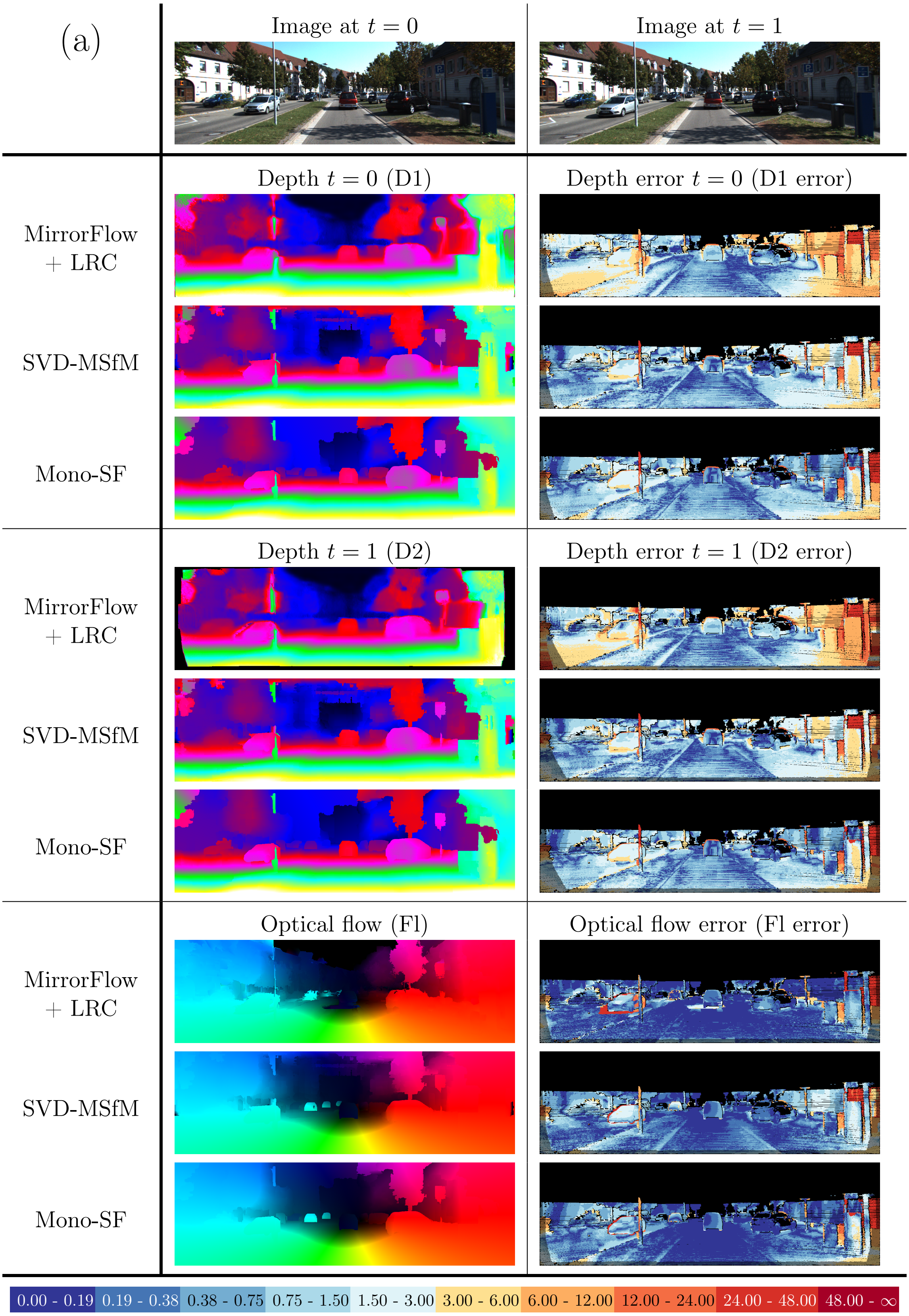
The qualitative results on the KITTI dataset are shown in figure 5.7 in comparison to the baseline method by combining MirrorFlow [Hur and Roth, 2017] for optical flow estimation with LRC [Godard et al., 2017] for single-view depth estimation and SVD-MSfM (chapter 4). The estimates and errors are visualized in terms of disparity at $t = 0$ (D1), disparity at $t = 1$ (D2), and optical flow (Fl) using the KITTI scene flow evaluation tools provided by Menze and Geiger [2015]. All estimates are represented at their image coordinates in the first frame at $t = 0$. The error color coding follows a logarithmic scale, where errors above 3px are colored in red shades and errors below 3px are colored in blue shades.

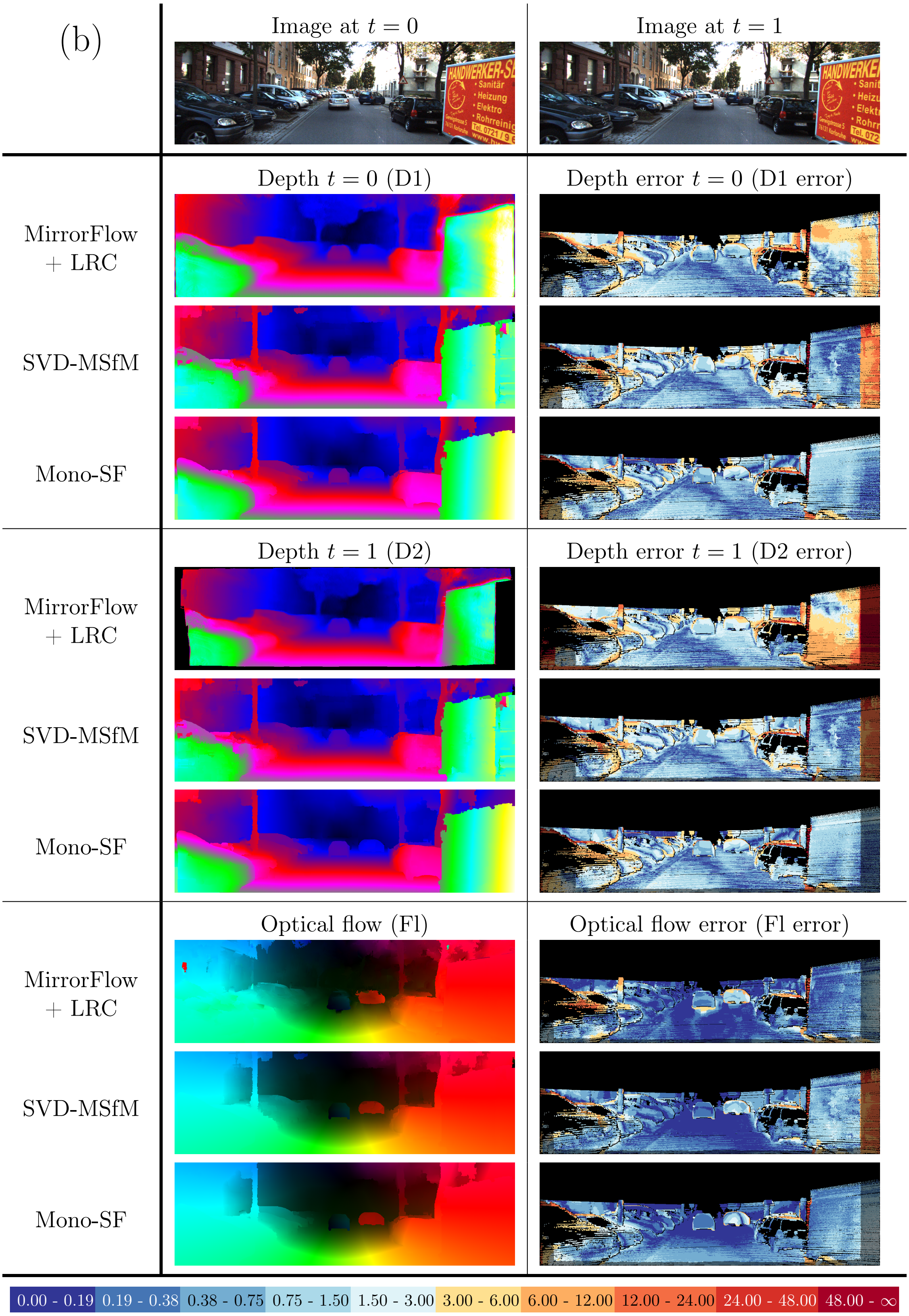
In comparison to SVD-MSfM, Mono-SF shows a significant improvement for some parts (e.g. right building in the background of figure 5.7 (a)) and a fine-tuning in general. The general characteristic is similar to SVD-MSfM.

For example, Mono-SF is also able to handle a wide range of object motions such as (1) oncoming vehicles (see figure 5.7 (a)), (2) preceding vehicles (see figure 5.7 (a,b)), and (3) crossing vehicles (see figure 5.7 (c)).

Mono-SF shows similar limitations as SVD-MSfM (described in section 4.3.1.2) and is not capable of compensating significant failures of SVD-MSfM, which is used as an initialization. For example, the missing part of the pole in figure 5.7 (a) is not recovered by Mono-SF and the smoothing prior gets also dominant in low-parallax situations (e.g. standstill scenario in figure 5.7 (c)). One reason is that Mono-SF requires a good initialization, which is not given for those parts. Furthermore, the segmentation in 3D planar surface elements stays fixed during optimization. Consequently, failures in the initial segmentation cannot be corrected.

The example in figure 5.7 (c) covers an additional situation that was not presented in the previous evaluation of SVD-MSfM. The bottom part of the white van is completely occluded. Therefore, a method that relies on detecting the ground contact point would fail. Even though also the accuracy of SVD-MSfM and Mono-SF is lower compared to other vehicles, the methods still provide reasonable results.


(a)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
MirrorFlow + LRC
SVD-MSfM
Mono-SF
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
MirrorFlow + LRC
SVD-MSfM
Mono-SF
Optical flow (Fl)
Optical flow error (Fl error)
MirrorFlow + LRC
SVD-MSfM
Mono-SF
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞


(b)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
MirrorFlow + LRC
SVD-MSfM
Mono-SF
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
MirrorFlow + LRC
SVD-MSfM
Mono-SF
Optical flow (Fl)
Optical flow error (Fl error)
MirrorFlow + LRC
SVD-MSfM
Mono-SF
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞

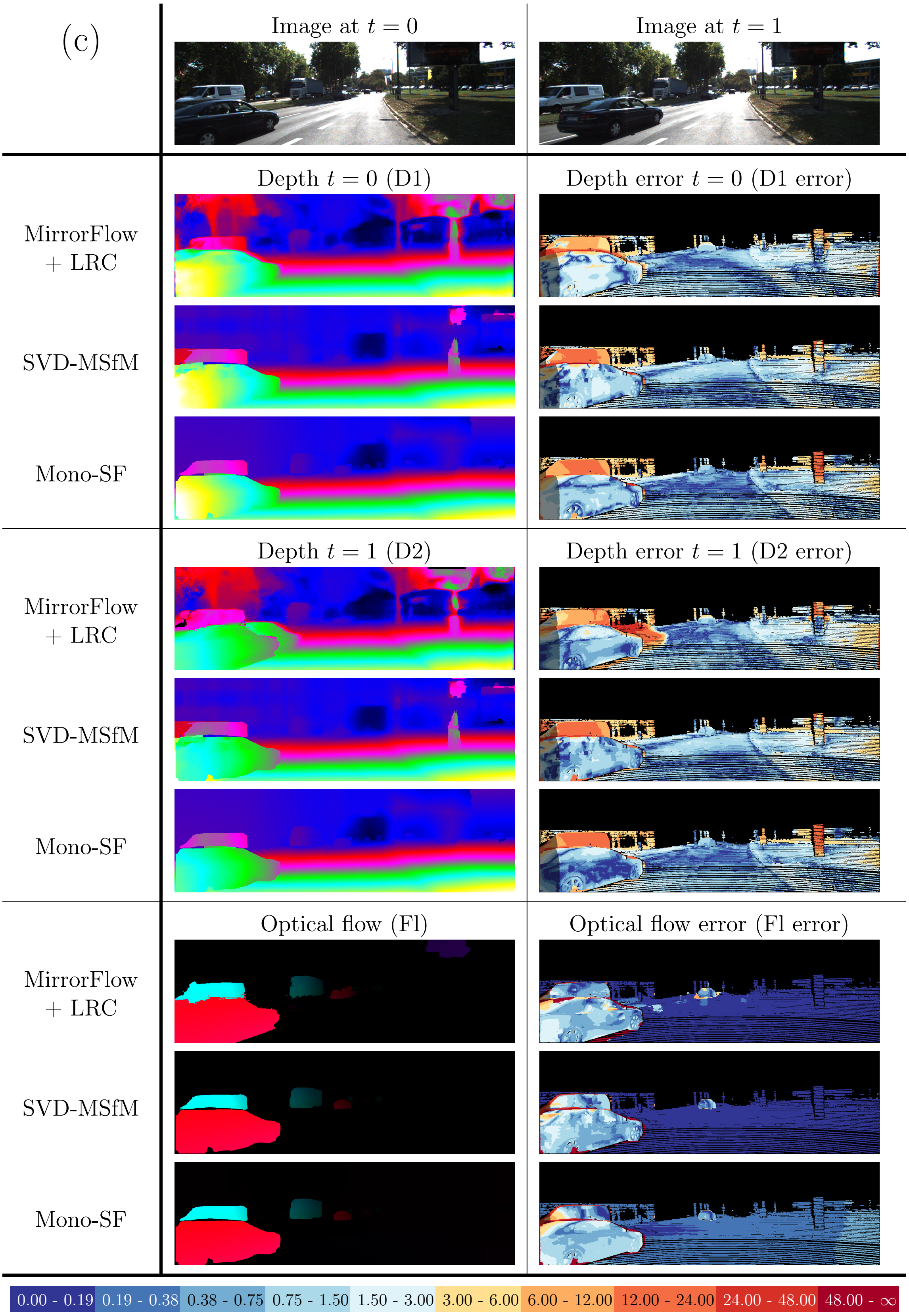


Figure 5.7: Qualitative results of Mono-SF in comparison to monocular scene flow baseline methods ('MirrorFlow + LRC' and 'SVD-MSfM') on the KITTI scene flow training set [Menze and Geiger, 2015]. The color coding represent the estimated depth (from close (warm) to far (cool)), the optical flow (Middlebury color coding [Baker et al., 2011]) or the disparity/ optical flow endpoint error (color coding shown in the legend).

Method	MRE	D1			D2			Fl			SF		
		bg	fg	all	bg	fg	all	bg	fg	all	bg	fg	all
GeoNet∗ [Yin and Shi, 2018]	20.07	47.03	63.41	49.54	56.24	68.88	58.17	32.42	67.69	37.82	67.69	91.40	71.32
DF-Net∗ [Zou et al., 2018]	18.95	44.42	57.94	46.49	61.55	61.47	61.53	25.66	37.44	27.46	71.62	82.51	73.29
Struct2Depth∗ [Casser et al., 2019]	14.92	27.29	56.58	31.77	33.25	66.12	38.29	37.86	71.96	43.08	49.98	91.39	56.32
EPC++ [Luo et al., 2019]	-	22.76	26.63	23.84	-	-	-	17.58	26.89	19.64	-	-	-
Self-Mono-SF [Hur and Roth, 2020]	9.98	28.75	45.07	31.25	33.00	45.15	34.86	23.06	25.92	23.49	44.27	62.40	47.05
MirrorFlow [Hur and Roth, 2017] + LRC [Godard et al., 2017]	9.68	25.33	**19.82**	24.48	35.82	26.15	34.34	**9.39**	14.22	**10.13**	40.55	35.17	39.72
HD^3-F† [Yin et al., 2019] + DORN† [Fu et al., 2018]	11.18	17.02	37.54	20.16	30.08	40.47	31.67	4.01	6.76	4.43	32.57	46.89	34.76
DMDE [Ranftl et al., 2016]	14.6	-	-	-	-	-	-	-	-	-	-	-	-
S. Soup [Kumar et al., 2017]	12.68	-	-	-	-	-	-	-	-	-	-	-	-
S.Rel. [Di et al., 2019]	10.23	-	-	-	-	-	-	-	-	-	-	-	-
MFA [Kumar et al., 2019]	11.82	-	-	-	-	-	-	-	-	-	-	-	-
SVD-MSfM [chapter 4]	8.55	17.84	23.94	18.77	20.37	26.72	21.35	15.31	15.55	15.34	24.50	35.01	26.11
Mono-SF	**8.14**	**15.64**	22.72	**16.72**	**17.93**	**24.71**	**18.97**	12.20	**9.90**	11.85	**20.19**	**29.40**	**21.60**

***MRE**: mean relative depth error in %; **D1**, **D2**: disparity ($t = 0, 1$); **Fl**: optical flow; **SF**: scene flow*
***D1**, **D2**, **Fl**, **SF**: percentage[%] of estimates that exceed an error threshold ($> 3px$ and $> 5\%$ of length)*
***fg**: foreground (moving objects) ; **bg**: background (static environment); **all**: bg + fg*
†: parts of dataset used for training (disregarded for ranking); ∗: scaled to align the ground truth

Table 5.1: Quantitative evaluation of Mono-SF with respect to several monocular methods on the KITTI scene flow training set [Menze and Geiger, 2015]. The methods are divided into four groups: First, multi-task CNNs; second, combining optical flow and single-view depth estimation as individual tasks; third, MSfM-based approaches; fourth, combining single-view depth information with multi-view geometry.

5.3.1.2 *Quantitative Evaluation of Scene Flow Estimation*

The quantitative evaluation of Mono-SF on the KITTI scene flow training set [Menze and Geiger, 2015] is provided in table 5.1. Following the evaluation of SVD-MSfM, the baseline methods are categorized in four groups represented by respective SotA methods. GeoNet [Yin and Shi, 2018], DF-Net [Zou et al., 2018], Struct2Depth [Casser et al., 2019], EPC++ [Yang et al., 2018b], and Self-Mono-SF [Hur and Roth, 2020] represent the first group of multi-task networks. The second category of combining single-view depth and optical flow estimation as individual tasks is covered by combining LRC [Godard et al., 2017] or DORN [Fu et al., 2018] with Mirror-

Method	D1			D2			Fl			SF		
	bg	fg	all	bg	fg	all	bg	fg	all	bg	fg	all
UberATG-DRISF [Ma et al., 2019]	2.16	4.49	2.55	2.90	9.73	4.04	3.59	10.40	4.73	4.39	15.94	6.31
ISF [Behl et al., 2017]	4.12	6.17	4.46	4.88	11.34	5.95	5.40	10.29	6.22	6.58	15.63	8.08
OSF 2018 [Menze et al., 2018]	4.11	11.12	5.28	5.01	17.28	7.06	5.38	17.61	7.41	6.68	24.59	9.66
PR-Sceneflow [Vogel et al., 2013]	4.74	13.74	6.24	11.14	20.47	12.69	11.73	24.33	13.83	13.49	31.22	16.44
SGM + SF [Hirschmuller, 2005] +[Hornacek et al., 2014]	5.15	15.29	6.84	14.10	23.13	15.60	20.91	25.50	21.67	23.09	34.46	24.98
Self-Mono-SF-ft [Hur and Roth, 2020]	20.72	29.41	22.16	23.83	32.29	25.24	15.51	17.96	15.91	31.51	45.77	33.88
Mono-Expansion [Yang and Ramanan, 2020]	24.85	27.90	25.36	27.69	**31.59**	28.34	**5.83**	**8.66**	**6.30**	29.82	**36.67**	30.96
Mono-SF	**14.21**	**26.94**	**16.32**	**16.89**	33.07	**19.59**	11.40	19.64	12.77	**19.79**	39.57	**23.08**

***MRE**: relative depth; **D1**, **D2**: disparity ($t = 0, 1$); **Fl**: optical flow; **SF**: scene flow errors*
***fg**: foreground (moving objects) ; **bg**: background (static environment); **all**: bg + fg; all errors in %*

Table 5.2: Evaluation of Mono-SF on KITTI scene flow benchmark server [Menze and Geiger, 2015]. The top group represents stereo-based methods with UberATG-DRISF [Ma et al., 2019] as the currently leading approach. The bottom group corresponds to monocular scene flow methods including the proposed Mono-SF approach. Only monocular methods are considered for ranking.

Flow [Hur and Roth, 2017] or HD^3-F [Yin et al., 2019]. Note that the used models for DORN and HD^3-F include parts of the scene flow set for training, which is why these methods are disregarded for ranking. The MSfM-based methods, DMDE [Ranftl et al., 2016], S.Soup [Kumar et al., 2017], and S.Rel [Di et al., 2019] form the third group. The fourth category comprises methods that fuse single-view depth information with multi-view geometry for monocular scene flow estimation. In addition to the baseline method, MFA [Kumar et al., 2019], this category consists of the proposed SVD-MSfM (chapter 4) and Mono-SF methods.

Mono-SF shows the best accuracy on most of the metrics – especially outperforming previous methods on the scene flow metrics. Furthermore, Mono-SF provides a further improvement compared to SVD-MSfM, which is utilized for initialization. Mono-SF reduces the number of errors in terms of scene flow estimation by 4.31% for the static environment and by 5.61% for moving objects – which clearly motivates the joint optimization of depth and motion directly on a superpixel level formulated as a scene flow estimation problem.

Furthermore, Mono-SF is the first monocular method published on the KITTI scene flow benchmark [Menze and Geiger, 2015] as shown in table 5.2. In comparison to the stereo-based methods, Mono-SF would have been ranked on the 13th place with respect to the 21 published stereo methods at the time of submission. Even though Mono-SF is not comparative with the SotA stereo methods, it still shows that also a monocular method could provide reasonable scene flow results. Subsequently to Mono-SF, two further monocular methods are published on the KITTI benchmark[2], namely Self-Mono-SF [Hur and Roth, 2020] and Mono-Expansion [Yang and Ramanan, 2020]. Both methods correspond to an end-to-end neural network approach (section 4.1.3). The proposed Mono-SF method still provides the highest accuracy on the overall scene flow metric and remains the currently leading approach. Mono-Expansion directly uses a SotA optical flow method, which outperforms Mono-SF on an optical flow metric and shows a slightly better accuracy for moving objects on a scene flow metric. Mono-SF outperforms Mono-Expansion and Self-Mono-SF especially on the depth estimation for the background. Analogously to the evaluation on section 5.3.1.2, this supports that the combination of multi-view geometry and single-view depth in a joint optimization benefits more from the scene rigidity knowledge for large rigid bodies such as the background.

Self-Mono-SF and Mono-Expansion are close to real-time capability, at least on a powerful GPU exploiting highly optimized deep learning frameworks. In contrast to that, the Mono-SF approach is currently not focused on runtime and needs around 41 seconds per image on a single CPU-core. Most of the runtime (above 30 seconds) is spent on the non-linear optimization of Mono-SF. Even more important, previous approaches [Gehrig et al., 2009, Mur-Artal and Tardós, 2017] show the real-time capability of methods related to optimizations used in SVD-MSfM. While this at least suggests that SVD-MSfM could be implemented in a real-time capable version, a real-time capability of related stereo-based scene flow optimizations (e.g. [Menze et al., 2018, Behl et al., 2017]) was not shown so far. In summary, there is definitely a trade-off between runtime and accuracy and the improved accuracy of Mono-SF comes at the cost of significant higher computational effort.

5.3.2 Evaluation of Mono-SF Components

Mono-SF energy term: In table 5.3, the individual components of the Mono-SF optimization framework are analyzed by removing some parts of the proposed energy minimization problem (setting their weights to zero). SVD-MSfM, which is used for initialization of Mono-SF is denoted by the row without checkmarks. A scene flow estimation merely based on the photometric distance, improves the accuracy in terms of optical flow estimation compared to the initialization. However, it is not able to cope with all situations such as standstill scenarios and the accuracy of the depth estimates decreases. Adding the probabilistic single-view depth estimates

[2] Referring to methods with a publication submitted to the benchmark until October 02, 2020.

Energy terms			MRE	D1			D2			Fl			SF		
Φ^{pho}	Φ^{svd}	Ψ		bg	fg	all	bg	fg	all	bg	fg	all	bg	fg	all
-	-	-	9.38	17.84	23.94	18.77	20.37	26.72	21.35	15.31	15.55	15.34	24.50	35.01	26.11
✓	-	-	21.73	18.62	35.43	21.19	20.83	37.64	23.41	13.27	16.98	13.84	22.89	43.87	26.11
✓	✓	-	9.40	17.91	22.72	18.65	20.35	25.18	21.09	13.62	11.56	13.30	22.44	30.46	23.65
✓	✓	✓	**8.99**	**15.63**	**22.71**	**16.72**	**17.93**	**24.70**	**18.97**	**12.19**	**9.90**	**11.84**	**20.19**	**29.40**	**21.60**

***MRE**: relative depth; **D1**, **D2**: disparity ($t = 0, 1$); **Fl**: optical flow; **SF**: scene flow errors*

***fg**: foreground (moving objects) ; **bg**: background (static environment); **all**: bg + fg; errors in %*

Table 5.3: Ablation study on Mono-SF approach. Using the Mono-SF optimization improves the scene flow estimation compared to its initialization (denoted by the row without checkmark). Each term of the energy minimization problem (photometric distance(Φ^{pho}), single-view depth (Φ^{svd}), and smoothness prior (Ψ)) contributes to the final performance.

significantly improves the depth accuracy, which leads also to an improvement compared to the initialization. A further improvement is achieved by adding the scene model smoothness priors. Therefore, the ablation study shows that each part of the energy term contributes to the final performance – the multi-view geometry, the single-view depth information, and the scene model smoothness priors.

Integrating probabilistic single-view depth distributions: To analyze the importance of the proposed ProbDepthNet design, the results of four Mono-SF variants based on different single-view depth estimations as proposed in section 4.3.2 are provided in table 5.4. The proposed Mono-SF variant that integrates well-calibrated single-view depth distributions outperforms all other variants. A significant improvement compared to Mono-SF integrating non-probabilistic single-view depth estimates ('w/o prob. depth' and 'LRC [Godard et al., 2017]') is shown, especially for the background. Additionally, the experiments support the claimed importance of

Method	MRE	D1			D2			Fl			SF		
		bg	fg	all	bg	fg	all	bg	fg	all	bg	fg	all
Mono-SF (LRC)	10.05	22.59	**21.02**	22.37	26.73	**23.29**	26.29	15.20	14.53	15.10	31.30	**29.03**	30.96
Mono-SF (w/o prob. depth)	10.60	25.95	22.93	25.48	29.28	26.14	28.80	15.34	13.33	15.03	34.04	31.06	33.58
Mono-SF (w/o recalib.)	9.54	19.80	23.12	20.31	22.60	27.60	23.36	14.80	19.36	15.50	25.16	36.52	26.90
Mono-SF	**8.99**	**15.63**	22.71	**16.72**	**17.93**	24.70	**18.97**	**12.19**	**9.90**	**11.84**	**20.19**	29.40	**21.60**

***MRE**: relative depth; **D1**, **D2**: disparity ($t = 0, 1$); **Fl**: optical flow; **SF**: scene flow errors*

***fg**: foreground (moving objects) ; **bg**: background (static environment); **all**: bg + fg; all errors in %*

Table 5.4: Ablation study on ProbDepthNet for Mono-SF. For integrating single-view depth information, ProbDepthNet is more suitable than LRC for single-view depth estimation (improvement over 'LRC Godard et al. [2017]'); especially due to the importance of providing single-view depth estimates in a probabilistic (improvement over 'w/o prob. depth') and well-calibrated form (improvement over 'w/o recalib.') for Mono-SF.

providing single-view depth distributions in a well-calibrated form (improvement over 'w/o recalib.').

In summary, the ablations studies confirm two main observations also made for SVD-MSfM. First, multi-view geometric information and single-view depth estimation provide powerful and complementary information to the task of monocular scene flow estimation, which highly motivates a method that combines both. Second, single-view depth information provided in a probabilistic and well-calibrated form is beneficial for a suitable integration.

5.4 Conclusion

The present chapter presented Mono-SF for joint optimization of motion and depth formulated as a monocular scene flow problem directly using a superpixel representation. The evaluation confirms the suitability of this joint optimization, which provides a further improvement compared to SVD-MSfM. Additional experiments strengthen the claimed combination of multi-view geometry with single-view depth information and the ProbDepthNet design to provide single-view depth information as well-calibrated depth distributions. Mono-SF was the first monocular method published on the KITTI scene flow benchmark for monocular scene flow estimation, even though better accuracy comes at the cost of a higher computational effort.

6

MONOCULAR STIXEL SCENE FLOW

CONTENTS

This chapter extends parts of the works that have been published previously in [Brickwedde et al., 2018b,a].

The new state of the art (SotA) accuracy of the proposed methods SVD-MSfM (chapter 4) and Mono-SF (chapter 5) for monocular scene flow estimation strongly motivates to integrate such methods in robotics and automotive applications. However, the mere accuracy of the estimates is not the only criterion to be considered for practical application. A scene reconstruction in terms of depth and motion is typically not the ultimately addressed goal, but it is used as a basis for subsequent tasks such as path planning or autonomous braking. The scene representation needs

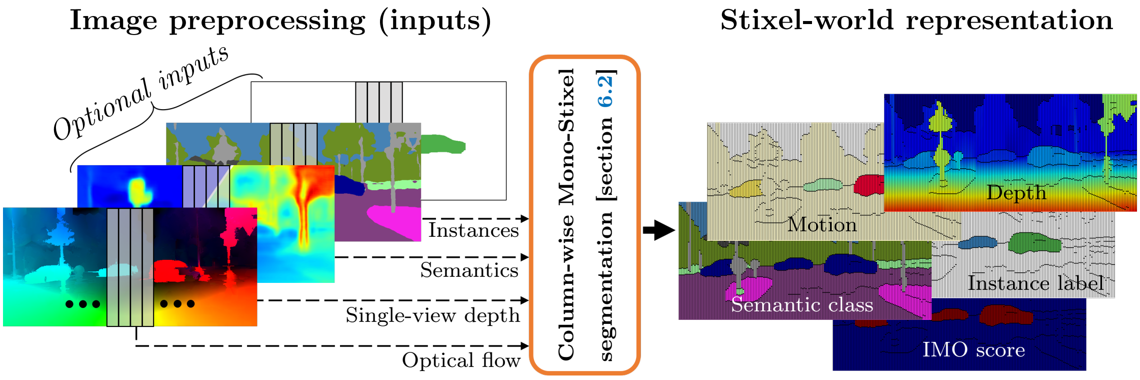


Figure 6.1: Overview of the Mono-Stixel approach and representation. Stixels are defined as thin stick-like elements providing a compact, but-detailed scene representation. Each stixel (see black boundaries) encodes its scene flow (depth and motion), its semantic class, corresponding rigid body, and a score of being an independent moving object (IMO). The segmentation is formulated as a column-wise segmentation problem and follows the essential findings of the previous chapters, how to combine multi-view geometry with deep learning methods such as probabilistic single-view depth estimation, semantic, or instance segmentation.

to be stored, potentially transferred for distributed systems, and processed by the subsequent application. This highlights an additional requirement for a scene representation – it should be *compact*. SVD-MSfM and Mono-SF provide a non-compact scene representation consisting of superpixels of arbitrary form. Street scenes follow special characteristics, which allows using more specialized representations. Such representation will be introduced and used in the present chapter.

One useful compact medium-level representation for street scenes is the so-called *stixel world*, which was introduced by Badino et al. [2009] and extended to a *multi-layer stixel world* by Pfeiffer and Franke [2011b]. The stixel world representation is defined as a column-wise segmentation of the image into thin stick-like and planar elements, the stixels. The stixels are typically estimated based on a dense disparity map in a stereo setup.

In the present chapter, I propose the *Mono-Stixel* method (see figure 6.1), which is a monocular scene flow estimation with stixels as underlying representation. It combines the essential contributions of the previous chapters regarding monocular scene flow estimation with the benefits of using a specialized representation for traffic scenes. In addition to the more specialized and compact scene representation compared to SVD-MSfM (chapter 4) and Mono-SF (chapter 5), further extensions are proposed as part of the Mono-Stixel method.

First, the stixels provided by the Mono-Stixel method encode additional types of information. SVD-MSfM and Mono-SF consider all objects detected by an instance segmentation as potentially moving objects, for example, including parked vehicles. In contrast to that, the Mono-Stixel method additionally detects which parts of the scene are really in motion encoded by an additional independent moving object score. Furthermore, each stixel consists of its semantic class such as road, sidewalk, or building (see figure 6.1).

Second, the moving objects are directly defined by the instance segmentation using SVD-MSfM or Mono-SF. Thus, missing object detections would directly result in erroneous scene flow estimates because the individual object motion is not considered. In contrast to that, the differentiation between static and dynamic objects is part of the stixel segmentation of the Mono-Stixel approach and could potentially overrule a missing detection of the instance segmentation.

Third, the only indispensable input for the Mono-Stixel method is the optical flow. The other inputs, probabilistic single-view depth, instance segmentation, and semantic segmentation (see figure 6.1) are optional. This makes the Mono-Stixel method more flexible, for example, to be adaptable to a certain hardware setup.

The experiments in section 6.3 show that the Mono-Stixel method provides SotA monocular scene flow estimates comparable to the Mono-SF method. Even more, the Mono-Stixel method shows better characteristics than Mono-SF in terms of scene flow estimation for small and thin objects such as poles. While Mono-SF sometimes tends to smooth out thin objects especially at larger distances, the Mono-Stixel method maintains much more details. Additionally, the fact that all inputs expect the optical flow are optional allows analyzing the impact of each input by performing several ablation studies, which are presented in section 6.3.3.

6.1 Related Work

The Mono-Stixel approach is a monocular scene flow estimation approach, which is designed to provide a compact scene representation using the stixel world model. A general overview of monocular scene reconstruction and monocular scene flow estimation have already been provided in section 4.2. The present section places the focus on works related to the stixel world representation.

Stereo-based stixel world methods: The estimation of a stixel world representation is traditionally addressed in a stereo setup based on a dense disparity map. Pioneering, Badino et al. [2009] introduced the term *stixel*. The naming is related to stick-like superpixels and corresponds to the assumption that the objects, which limit the free-space, have vertical surfaces and can be approximated by adjacent rectangular sticks with a certain height and width. The stixels are defined by a column-wise detection of the objects base-point, which limits the free-space, and a column-wise height segmentation [Badino et al., 2009]. However, this concept is not limited to a free-space estimation and was extended to a multi-layer stixel world model [Pfeiffer and Franke, 2011b], which covers the whole image. A visualization of the stixel world representation is show in figure 1.2 in section 1.1.1. The image of size $w \times h$ is divided into columns of a fixed width w_s to split the stixel world segmentation $\mathcal{S}$ into individual stixel columns $\mathcal{S}_u$:

$$\mathcal{S} = \{\mathcal{S}_u \mid 1 \leq u \leq \frac{w}{w_s}\} \tag{6.1}$$

Each stixel column is segmented individually into N_u stixels $\mathbf{s}_i$:

$$\begin{aligned} \mathcal{S}_u &= \{\mathbf{s}_i \mid 1 \leq i \leq N_u \leq h\} \\ \mathbf{s}_i &= \left(v_i^b, v_i^t, m_i, d_i(v)\right) \end{aligned} \tag{6.2}$$

Each stixel $\mathbf{s}_i$ (index stands for i-th stixel in the column) is defined by its base v_i^b and top image position v_i^t, its class or type m_i, and its disparity model $d_i(v)$. Additional constraints ensure that each pixel is exactly assigned to one stixel ($v_1^b = 1$, $v_N^t = h$ and $v_{n-1}^t + 1 = v_n^b$).

Originally, the type m_i distinguishes the three types, ground, object, and sky. Each type defines a geometric model in terms of orientation and distance. Ground stixel have a normal vector parallel to the vehicle's z-axis (pointing upwards), object stixel have a normal vector perpendicular to the vehicle's z-axis, and sky stixels are at infinite distance. Formally, this defines a disparity model $d_i(v)$ depending on the stixel's type m_i:

$$d_i(v) = \begin{cases} \mu_i & \text{, if } m_i = \text{object} \\ \alpha \cdot (v_{hor} - v) & \text{, if } m_i = \text{ground} \\ 0 & \text{, if } m_i = \text{sky} \end{cases} \tag{6.3}$$

The parameters α and v_{hor} are defined by the intrinsic and extrinsic camera parameters. The variable μ_i represents the distance of the object to the camera and is estimated based on the mean disparity values in the corresponding image segment.

For each column, the stixel segmentation $\mathcal{S}_u$ is formulated as a 1D energy minimization problem derived from a maximum a posteriori probability estimation based on a dense disparity map $\mathbf{d}$:

$$p(\mathcal{S}_u \mid \mathbf{d}) = \frac{p(\mathbf{d} \mid \mathcal{S}_u) \cdot p(\mathcal{S}_u)}{p(\mathbf{d})} \tag{6.4}$$

The normalization factor $p(\mathbf{d})$, which expresses the prior probability of the disparity measures, is constant during the optimization of $\mathcal{S}_u$ and thereby neglectable. Switching to the log-domain defines the 1D energy minimization problem $E(\mathcal{S}_u, \mathbf{d})$, which consist of the data term $\Phi(\mathcal{S}_u, \mathbf{d})$ and prior terms $\Psi_{str}(\mathcal{S}_u)$, $\Psi_{mc}(N_u)$:

$$E(\mathcal{S}_u, \mathbf{d}) = \Phi(\mathcal{S}_u, \mathbf{d}) + \Psi_{str}(\mathcal{S}_u) + \Psi_{mc}(N_u) \tag{6.5}$$

One part of the prior term regularizes the model complexity in terms of the number of stixels N_u by adding $\Psi_{mc}(N_u) = \beta_{mc} N_u$ to the energy term. Additionally, a structural prior is integrated to prefer a typical layout of a traffic scene. The structural prior $\Psi_{str}(\mathcal{S}_u)$ rates the likelihood of two neighboring stixels in the column and consist of three terms:

$$\Psi_{str}(\mathcal{S}_u) = \sum_{i=0}^{N_u} \left(\Phi_{do}(\mathbf{s}_i, \mathbf{s}_{i-1}) + \Phi_{grav}(\mathbf{s}_i, \mathbf{s}_{i-1}) + \Phi_{type}(\mathbf{s}_i, \mathbf{s}_{i-1})\right) \tag{6.6}$$

The weighting and definition of each prior term depend on hyperparameters $\alpha_{(\cdot)}, \beta_{(\cdot)}$.

First, an ordering prior $\Phi_{do}(\mathbf{s}_i, \mathbf{s}_{i-1})$ prefers that an object stixel on top of another object is typically behind the bottom stixel in the 3D scene. This term is only non-zero if both adjacent stixels are of the object type and if the upper stixel $\mathbf{s}_i$ is closer than the lower stixel $\mathbf{s}_{i-1}$:

$$\Phi_{do}(\mathbf{s}_i, \mathbf{s}_{i-1}) = \begin{cases} \alpha_{do} + \beta_{do} \cdot (d_i - d_{i-1}) & \text{, if } m_i, m_{i-1} = \text{object } \& \ d_i > d_{i-1} \\ 0 & \text{, otherwise} \end{cases} \tag{6.7}$$

Second, a gravity prior $\Phi_{grav}(\mathbf{s}_i, \mathbf{s}_{i-1})$ favors that objects typically stand on the ground plane. This term is only non-zero if an object stixel $\mathbf{s}_i$ follows a ground stixel $\mathbf{s}_{i-1}$ and rates the difference Δ_d of the object stixel to the disparity of the ground stixel at the segmentation boundary v^t_{i-1}, where both stixels are connected in the image. The gravity prior distinguish if the stixel is above or below the ground plane:

$$\Phi_{grav}(\mathbf{s}_i, \mathbf{s}_{i-1}) = \begin{cases} \alpha^-_{grav} + \beta^-_{grav} \Delta_d & \text{, if } m_i = \text{object } \& \ m_{i-1} = \text{ground } \& \ \Delta_d < 0 \\ \alpha^+_{grav} + \beta^+_{grav} \Delta_d & \text{, if } m_i = \text{object } \& \ m_{i-1} = \text{ground } \& \ \Delta_d > 0 \\ 0 & \text{, otherwise} \end{cases} \tag{6.8}$$

Third, the likelihood of a transition between different stixel types is rated by a third part $\Phi_{type}(\mathbf{s}_i, \mathbf{s}_{i-1})$. For example, an object stixel above a ground stixel is more likely than vice versa. Discrete transition cost values $\gamma(m_i, m_{i-1})$ are predefined for each possible combination of stixel types:

$$\Phi_{type}(\mathbf{s}_i, \mathbf{s}_{i-1}) = \gamma(m_i, m_{i-1}) \tag{6.9}$$

The data term integrates the input measurements to favor a stixel depth structure consistent with the disparity estimates $\mathbf{d}$:

$$\Phi(\mathcal{S}_u, \mathbf{d}) = \sum_{i=0}^{N_u} \sum_{v=v_i^b}^{v_i^t} \Phi(\mathbf{s}_i, v, \mathbf{d}_v) \tag{6.10}$$

A measurement model is used to rate the distance of the measured disparity $\mathbf{d}_v$ at row v to the disparity $d_i(v)$ defined by the corresponding stixel $\mathbf{s}_i$ for each pixel in the stixel's segment. Pfeiffer and Franke [2011b] proposed a mixture model of a uniform and a Gaussian distribution to define the measurement model for disparity estimation. The pixel-wise summation in equation (6.10) refers to the assumption of statistical independence of all pixels.

Based on the defined 1D energy minimization problem and exploiting the described constraints regarding the segmentation, the inference can be expressed as the shortest path problem, which is solved via dynamic programming [Pfeiffer and Franke, 2011b, Cordts et al., 2017]. The approach by Pfeiffer and Franke [2011b] minimizes the energy term globally in terms of segmentation v_i^b, v_i^t and stixel types m_i. The disparity of each stixel $d_i(v)$ is locally approximated for each segment to reduce the computational effort.

Several works described methods to integrate further information or to estimate further attributes of the stixels. Cordts et al. [2014] proposed to incorporate additional object classifier responses. The object bounding boxes of the classifier are used to define priors for the stixel segmentation boundaries. Instead of integrating bounding box-based object detections, also a pixel-wise semantic segmentation is proposed to be integrated as an additional data term [Scharwächter and Franke, 2015, Schneider et al., 2016]. On the one hand, the semantic segmentation supports distinguishing the different stixel types (e.g. the road class is assigned to the ground stixel type). On the other hand, it allows to additionally infer a stixel class label c_i such as road, building, or vegetation. The resulting representation is denoted as *semantic stixels* [Schneider et al., 2016]. A classification of stixels into semantic classes has been proposed before [Scharwächter et al., 2013, 2014]. However, the classification of stixels was formulated as a subsequent step in those papers. Thus, the accuracy of the stixel estimation itself cannot benefit from it.

While originally the orientations of stixels are directly defined by the stixel type, subsequent works extended the stixel world model to represent non-flat roads or slanted objects. The first kind of approaches addresses these tasks by integrating a separate ground surface estimation, which provides a column-wise polynomial [Saleem et al., 2017] or B-spline approximation [Xu et al., 2018] of the ground surface. Hernandez-Juarez et al. [2017] proposed the so-called *slanted stixels* approach, which additionally estimates a slope for each stixel by adapting the disparity model to $d_i(v) = b_i v + a_i$. The parameters a_i and b_i are part of the estimation for ground and object stixels. The fact that, for example, a ground stixel is typically horizontal is integrated as a Gaussian prior over the parameters a_i, b_i.

The presented approaches represent the depth structure of a scene given a static stereo image pair. The present thesis is focused on scene flow estimation, which means providing depth and motion. However, the stixel world representation has been also extended to represent the motion. Pfeiffer and Franke [2011a] proposed to integrate the temporal component using the 6D vision principle [Franke et al., 2005], which provides the 3D position and 3D motion for every feature individually. The longitudinal and lateral stixel motion is inferred using a Kalman Filter. Additionally, a classification of the stixel's motion state (static, oncoming, forward-moving, left-moving, right-moving) was proposed by Erbs et al. [2012].

In contrast to these methods, which use a stereo setup, the Mono-Stixel method proposed in the present thesis addresses a stixel estimation in a monocular setup by combining optical flow measurements, single-view depth estimates, semantic segmentation, and instance segmentation. While the motion of an object does not matter for a stixel estimation using a static stereo image pair, the individual motion of moving objects needs to be considered for integrating optical flow. Therefore the stixel model of the Mono-Stixel method explicitly distinguishes static and dynamic objects and defines respective motion constraints, which makes the Mono-Stixel approach applicable to dynamic scenes in a monocular camera setup. Furthermore, global optimization following Mono-SF is proposed to improve the accuracy of the initial segmentation, which infers each column separately.

Monocamera-based stixel world methods: The estimation of a stixel world representation is typically addressed in a stereo setup. However, there are a few works using a monocular camera. Wolcott and Eustice [2016] proposed a column-wise partitioning of the image in ground, obstacle, and background based on a prior appearance ground map and optical flow. Levi et al. [2015] and Garnett et al. [2017] proposed a convolutional neural network (CNN), called StixelNet, which predicts the segment and depth of the closest stixel in each column. Since the input of StixelNet is a single image, this method is highly related to a stixel estimation in terms of single-view depth estimation. A multi-view geometry-based approach has been proposed by Suhr and Jung [2019]. This method follows the stereo-based concept of Badino et al. [2009] but replaces the disparity map with a 3D point cloud derived by structure from motion (SfM). However, using SfM point clouds limits this approach to the static parts of the scene. Furthermore, all these methods represent only the first row of the closest stixels. Thus, these methods are more related to free-space estimation.

In contrast to the existing monocular stixel methods, the Mono-Stixel method presented here provides several contributions: (1) the Mono-Stixel method is the first monocular multi-layer stixel world estimation, (2) both kinds of information, multi-view geometry and single-view depth, are combined, and (3) the Mono-Stixel method provides a full scene flow representation applicable to dynamic scenes.

Two works use a monocular camera instead of a stereo camera setup – but integrate light detection and ranging (LiDAR) measurements instead. Thereby, the estimated depth structure relies on the LiDAR point cloud. The monocular image is exploited for depth completion using color and texture information [Saleem et al., 2018] or to integrate semantic information [Piewak et al., 2018].

Applications based on the stixel world representation: Several higher-level vision tasks or applications have been proposed that build upon the stixel world representation. This supports the usability of the stixel world representation. A brief overview of these works is given in the following.

The first kind of tasks is to derive an object-centric representation from the stixel world representation. Especially the clustering of stixels to vehicles represented as bounding boxes including the motion parameters has been addressed [Erbs et al., 2011, 2013, 2014]. The clustering could be formulated as hypothesis testing [Erbs et al., 2011], a labeling problem in a conditional random field (CRF) [Erbs et al., 2013], or solved via dynamic programming based on a tree-structured graph, which connects adjacent stixels [Erbs et al., 2014]. These approaches base their clustering on the geometric information provided by stixels. Alternatively, Hehn et al. [2019] proposed to integrate a CNN for instance segmentation. The integrated instance segmentation follows the Box2Pix concept [Uhrig et al., 2018], which provides pixel-wise 2D offsets pointing to the center of its respective instance. Analogously, the stixel labels are extended by their corresponding 2D instance coordinates and clustered afterward.

The stixel world representation has been exploited for classification tasks such as pedestrian detection [Benenson et al., 2012, 2011] or vehicle detection [Enzweiler et al., 2012]. The main benefit of using a stixel world representation is to derive a reduced set of candidate windows based on the stixels' geometry. This can speed-up the classification task and increase the robustness.

Furthermore, applications have been based on the stixel world representation. For example, the autonomous driving research project presented in [Franke et al., 2013] uses the stixel world for environment representation.

Applications based on the stixel world representation are out of the scope of the present thesis. However, due to the similarity of the stereo-based stixel world representation and the representation provided by the Mono-Stixel method, the mentioned applications can possibly also based on the output of the Mono-Stixel method.

6.2 Mono-Stixel Method

The present section describes the Mono-Stixel approach starting with the definition of the scene model. The subsequent sections present the column-wise stixel segmentation, independent moving object (IMO) detection, and the global optimization following the Mono-SF concept (chapter 5) with stixels as underlying representation.

6.2.1 Scene Model

The image of size $w \times h$ is divided into w/w_s columns of a fixed width w_s. Each column is considered as an individual 1D segmentation $\mathcal{S}_u$ into stixel segments as defined in [Pfeiffer and Franke, 2011b] and equations (6.1) and (6.2). The index u represents the u-th column. Each stixel segment $\mathbf{s}_i$ is defined as a thin stick-like planar and rigid moving element in the scene. The following stixel labels are defined to encode the segmentation, type, geometry, and motion of a stixel segment:

- The segmentation labels v_i^b, v_i^t define the top and bottom row of the stixel in the corresponding column to define the segmentation boundaries.

- The semantic class label c_i represents the class of the stixel segment such as road, building, or vegetation.

- The type label m_i assigns the stixel segment to one stixel type: ground, static object, dynamic object, or sky. Each type defines specific model constraints and groups several semantic classes into one type.

- The distance label ρ_i stands for the distance of the stixel segment to the camera defined as an inverse depth.

- The rigid body label o_i stands for the association of a stixel segment to one rigid body. A set of rigid bodies $\mathcal{O}$ including their scale-aware 6D motions $\mathbf{T}_j$ is assumed to be given as an input. The motion estimation described in section 4.2.1 is used to provide such a set of rigid bodies, which provides a set of motion estimates for the static environment o_{BG} and potentially moving objects $\mathcal{O} = \{o_{BG}, o_2, ..., o_M\}$. This association serves as a representation of the relative motion of the stixel.

- The individual 2D translational stixel motion label $\mathbf{t}_i$ allows representing an individual stixel motion in the case that an association to a rigid body is not possible because the corresponding rigid body is not part of the set $\mathcal{O}$.
- The score γ_i represents the likelihood to be an independent moving object (IMO) and expresses which objects are really in motion (e.g distinguishing parking and driving vehicles).

Consequently, a stixel segment can be considered as an element described by the labels

$$\mathbf{s}_i = (v_i^b, v_i^t, m_i, c_i, o_i, \gamma_i, \rho_i, \mathbf{t}_i). \tag{6.11}$$

Four stixel types m_i are introduced, each one defining special stixel model constraints. The stixel model constraints are defined to regard the specific prior knowledge applicable to dynamic traffic scenes. The definition of the stixel types is shown in table 6.1. A more detailed description of the stixel types and their model constraints is given in the following.

Ground stixel $m_i = \mathbb{G}$: The first stixel type is defined to represent the ground. Typical classes in traffic scenes are road, sidewalks or terrain (e.g. lawn), which are considered as the three possible semantic classes $c_i \in \mathcal{C}_{\mathbb{G}} = \{\text{road, sidewalk, terrain}\}$ of the stixel.

The surface orientation of a ground plane is nearly horizontal. This characteristic is regarded by fixing the normal vector $\mathbf{n}_i$ of the stixel to be parallel to the vehicle's z-axis. Formally, the normal vector is defined as $\mathbf{n}_i^v = (0\ 0\ 1)^T$ in vehicle coordinates (x-forward, y-left, z-upward). Based on the extrinsic camera calibration, the normal vector could also be expressed in camera coordinates by $\mathbf{n}_i^c = \mathbf{R}^{v2c}\mathbf{n}_i^v$. The height is encoded by the inverse distance $\rho_i \in \mathbb{R}$ of the stixel's plane to the camera, which also allows an approximation of slanted surfaces in a stepwise manner. The 3D position $\mathbf{X}$ of each point on the plane fulfills $\rho_i \mathbf{n}_i^T \mathbf{X} = 1$. The ground belongs to the static environment without any individual motion. Therefore, the association to a rigid body is constrained to be $o_i = o_{BG}$ and the relative translational motion is defined by the camera motion estimate of the rigid body $\mathbf{t}_i = \mathbf{t}(o_{BG})$.

Static object stixel $m_i = \mathbb{SO}$: The second stixel type comprises the static objects in the scene excluding those that are potentially moving. There are several classes of static objects in a traffic scene, which are clustered in the three main classes $c_i \in \mathcal{C}_{\mathbb{SO}} = \{\text{building, poles-signage, vegetation}\}$. The semantic class definition can also be done in more detail and include more classes, especially if an application needs a more finely detailed distinction of classes.

The geometric and motion constraints are similar to the ground stixels with the main difference that the orientation of static object stixels is defined to be upright.

Stixel type	Semantic	Geometry	Motion
ground	ground class	lying	static
$m_i = \mathbb{G}$	$c_i \in \mathcal{C}_{\mathbb{G}} =$ *{road, sidewalk, terrain}*	$\mathbf{n}_i^v = \begin{pmatrix}0\\0\\1\end{pmatrix}$ $\rho_i \in \mathbb{R}$	$o_i = o_{BG}$ $\mathbf{t}_i = \mathbf{t}(o_i)$
static object	static object class	upright	static
$m_i = \mathbb{SO}$	$c_i \in \mathcal{C}_{\mathbb{SO}} =$ *{building, poles-signage, vegetation}*	$\mathbf{n}_i^v = \begin{pmatrix}1\\0\\0\end{pmatrix}$ $\rho_i \in \mathbb{R} \geq 0$	$o_i = o_{BG}$ $\mathbf{t}_i = \mathbf{t}(o_i)$
dynamic object	dynamic object class	upright	potentially moving
$m_i = \mathbb{DO}$	$c_i \in \mathcal{C}_{\mathbb{DO}} =$ *{vehicle, two-wheeler, person}*	$\mathbf{n}_i^v = \begin{pmatrix}1\\0\\0\end{pmatrix}$ $\rho_i \in \mathbb{R} \geq 0$	case 1: *"known rigid body"* $\hookrightarrow o_i \in \{\mathcal{O} \setminus o_{BG}\}$, $\mathbf{t}_i = \mathbf{t}_i(o_i)$ case 2: *"unknown rigid body"* $\hookrightarrow o_i \notin \mathcal{O}$, $\mathbf{t}_i = \mathbf{R}^{v2c}\mathbf{t}_i^v$ with $\mathbf{t}_i^v = (\mathbf{t}_x^v\ \mathbf{t}_y^v\ 0)^T$
sky	sky class	infinite distance	static
$m_i = \mathbb{S}$	$c_i = \mathcal{C}_{\mathbb{S}} =$ *sky*	$\rho_i = 0$	$o_i = o_{BG}$ $\mathbf{t}_i = \mathbf{t}(o_i)$

m_i: type; c_i class; o_i: rigid body; ρ_i: inverse depth; $\mathbf{t}_i$: 2D translational motion; γ_i: IMO score
$\mathbf{n}_i^v$: normal vector defined in vehicle coordinates (x-forward, y-left, z-upward)

Table 6.1: Definition of stixel types and their model constraints of the Mono-Stixel method. The four stixel types, ground, static object, dynamic object, and sky define model constraints in terms of the semantic, geometry, and motion. The top row of each type represents the model assumption and the bottom row the mathematical consideration (see text for details).

This defines the normal vector as $\mathbf{n}_i^v = (1\ 0\ 0)^T$ in vehicle coordinates or $\mathbf{n}_i^c = \mathbf{R}^{v2c}\mathbf{n}_i^v$ in camera coordinates. The distance of the object to the camera is encoded by its inverse depth $\rho_i \in \mathbb{R}$ with the constraint that the object needs to be in front of the camera $\rho_i \geq 0$. Due to the constraint of the stixel type to be a static object, the associated rigid body is $o_i = o_{BG}$ and the relative translation is directly defined by the camera motion $\mathbf{t}_i = \mathbf{t}(o_{BG})$.

Dynamic object stixel $m_i = \mathbb{DO}$: The static object stixel explicitly excludes all objects that are potentially moving. These kinds of objects are comprised in the dynamic object stixel type. Focusing on dynamic traffic scenes, typical moving traffic participants are vehicles, two-wheelers, or persons, which are expressed by the potential semantic classes $c_i \in \mathcal{C}_{\mathbb{DO}} = \{\text{vehicle, two-wheeler, person}\}$.

The geometry of a dynamic object stixel in terms of its orientation and depth is identical to the static object stixel type. However, a dynamic object stixel can have an individual motion. There are 2 cases of how the motion is represented. In the first case, the stixel is associated with one rigid body $o_i \in \{\mathcal{O} \setminus o_{BG}\}$ with known 6D motion $\mathbf{T}(o_i)$. Thereby, the relative motion is directly given by the association $\mathbf{t}_i = \mathbf{t}(o_i)$. In the second case, if this association does not exist $o_i \notin \mathcal{O}$, the motion is represented as an individual 2D translational motion $\mathbf{t}_i^v = (\mathbf{t}_{i,x}^v \ \mathbf{t}_{i,y}^v \ 0)^T$ parallel to the ground plane. Considering the typical characteristics of traffic participants, an upward translational motion, as well as a rotational roll or pitch movement, are unusual or negligibly small. Furthermore, the yaw-rotation is negligible due to the small horizontal extent of a stixel. Based on the extrinsic camera calibration, the translation could be transformed into camera coordinates $\mathbf{t}_i = \mathbf{R}^{v2c}\mathbf{t}_i^v$. Introducing individual stixel motion makes this representation more generic because a motion representation is still possible even if the rigid body is not detected. Thereby, the motion estimated for an object (section 4.2.1) becomes an optional input and misdetections can be compensated. It is important to note that the dynamic object stixel type covers all potentially moving objects including e.g. parking vehicles. The differentiation between moving and standing objects is given by the additional IMO score γ_i. How to derive such a score is explained in section 6.2.4.

Sky stixel $m_i = \mathbb{S}$: The fourth stixel type represents the sky, which is also the corresponding semantic class $c_i = \mathcal{C}_\mathbb{S} =$ sky. A sky stixel is constrained to be at an infinite distance $\rho_i = 0$ and static.

6.2.2 Column-wise Stixel Segmentation

The stixel world representation is defined as a column-wise segmentation $\mathcal{S}_u$ into stixel segments $\mathbf{s}_i$. The present subsection describes how to estimate such a stixel representation. This means finding a good segmentation, differentiation of the stixel types, and estimating the stixel labels. Following the original stereo-based segmentation approach [Pfeiffer and Franke, 2011b], the segmentation is formulated as a 1D energy minimization problem (equation (6.5)), which is solved for each column independently:

$$E(\mathcal{S}_u) = \Phi(\mathcal{S}_u, \mathbf{f}, \mathbf{c}, \mathbf{d}, \mathbf{o}, \mathcal{O}) + \Psi(\mathcal{S}_u) \tag{6.12}$$

The energy term contains a data likelihood $\Phi(\mathcal{S}_u, \mathbf{f}, \mathbf{c}, \mathbf{d}, \mathbf{o}, \mathcal{O})$ integrating

1. an optical flow field $\mathbf{f}$ (e.g. provided by MirrorFlow [Hur and Roth, 2017]),
2. pixel-wise semantic class scores $\mathbf{c}$ (e.g. provided by FCN [Long et al., 2015]),
3. probabilistic single-view depth $\mathbf{d}$ (provided by ProbDepthNet),
4. instance labels $\mathbf{l}$ (e.g. provided by a Mask R-CNN [He et al., 2017]), and

5. a set of rigid body motion estimates $\mathcal{O}$ (provided as described in chapter 4).

The optical flow field and camera motion estimate, which is equal to the relative rigid body motion of the static environment, are mandotory, all other inputs are optional.

Additionally, a pairwise prior term $\Psi(\mathcal{S}_u)$ incorporates prior knowledge about the typical structure of street scenes. The components of the energy term are explained in the following sections 6.2.2.1 and 6.2.2.2 in more detail. Subsequently, the inference of the energy minimization problem is explained.

6.2.2.1 *Scene Model Priors*

The prior term prefers a stixel segmentation with a geometric layout that is plausible for the typical structure of street scenes $\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1})$ and further regularizes the model complexity by adding a constant value Θ_1 for each new stixel.

$$\Psi(\mathcal{S}_u) = \sum_{i=1}^{N} \left(\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1}) + \Theta_1\right) \tag{6.13}$$

The structural prior term follows the general idea of Pfeiffer and Franke [2011b] in a slightly different definition and adds a prior which prefers a flat ground surface. The applied structural prior $\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1})$ depends on the transition of two consecutive stixels and their stixel types. The segmentation is performed from bottom to top, which means that the stixel $\mathbf{s}_i$ is above the stixel $\mathbf{s}_{i-1}$ in the image.

$$\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1}) = \begin{cases} \Psi^{grav}(\Delta h_i) & \text{, if } m_i \in \{\mathbb{SO}, \mathbb{DO}\}, m_{i-1} \in \{\mathbb{G}, \mathbb{S}\} \\ \min(\Psi^{ord}(\rho_{i-1}, \rho_i), \Psi^{grav}(\Delta h_i)) & \text{, if } m_i, m_{i-1} \in \{\mathbb{SO}, \mathbb{DO}\} \\ \Psi^{flat}(\Delta h_i) & \text{, if } m_i = \mathbb{G} \\ 0 & \text{, if } m_i = \mathbb{S} \end{cases} \tag{6.14}$$

The three different structural terms, the gravity prior Ψ^{grav}, the ordering prior Ψ^{ord}, and flat ground prior Ψ^{flat}, are visualized in figure 6.2 and formally defined as follows. The prior terms are paramterizable by hyperparameters $\Theta_{(\cdot)}$.

Gravity prior Ψ^{grav}: As a first prior, the gravity constraint prefers objects standing on the ground plane. Thus, if an object stixel $\mathbf{s}_i \in \{\mathbb{SO}, \mathbb{DO}\}$ succeeds

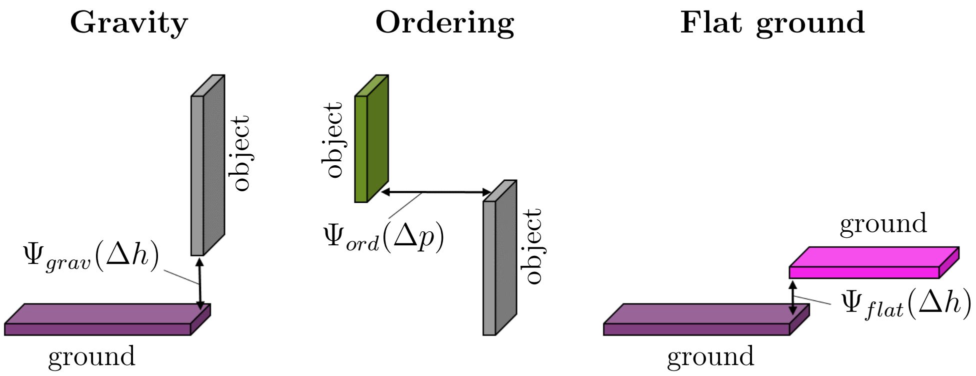


Figure 6.2: Scene model priors of Mono-Stixel method: gravity, ordering and flat ground. The gravity prior expresses that objects typically stand on the ground plane by rating the height of an object stixel over the ground plane. The ordering prior is defined to regard the characteristic that an object on top of another object is typically behind the bottom stixel. The flat ground prior prefers small deviations in the relative height of two consecutive ground stixels. Small deviations are plausible for example between the road and sidewalk or to represent a slanted road in a step-wise manner.

(from bottom to top) a ground or sky stixel $\mathbf{s}_{i-1} \in \{\mathbb{G}, \mathbb{S}\}$, the structural energy term is defined as $\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1}) = \Psi^{grav}(\Delta h_i)$ with

$$\Psi^{grav}(\Delta h_i) = \begin{cases} \Theta_2 + \Theta_3 \Delta h_i & \text{, if } \Delta h_i < 0 \\ \Theta_4 + \Theta_5 \Delta h_i & \text{, if } \Delta h_i > 0 \end{cases}. \tag{6.15}$$

The height Δh_i of the stixel $\mathbf{s}_i$ over the reference ground is defined as the height above the last previous ground stixel $\mathbf{s}_{\mathbb{G},ref}$ in the column. Formally, Δh_i is defined as follows:

$$\Delta h_i = \rho^{-1}_{\mathbb{G},ref} - \frac{\mathbf{n}^T_{\mathbb{G}} \mathbf{K}^{-1} \mathbf{p}_{v^b_i}}{\rho_i \mathbf{n}^T_i \mathbf{K}^{-1} \mathbf{p}_{v^b_i}} \tag{6.16}$$

The pixel coordinates $\mathbf{p}_{v^b_i}$ correspond to the bottom of the stixel $\mathbf{s}_i$ centered in the middle of column u:

$$\mathbf{p}_{v^b_i} = \begin{bmatrix} (u + 0.5) w / w_s & v^b_i & 1 \end{bmatrix}^T \tag{6.17}$$

If there is no previous ground stixel, the height of the reference ground $\rho^{-1}_{\mathbb{G},ref}$ is defined as the height of the camera mounting position.

Ordering prior Ψ^{ord}: If the previous object is also an object stixel $\mathbf{s}_i, \mathbf{s}_{i-1} \in \{\mathbb{SO}, \mathbb{DO}\}$, the bottom of the stixel might not be the bottom of the object due to occlusion or a depth discontinuity inside the object. Therefore, the structural prior term is defined as the minimum of the gravity and an ordering term $\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1}) = \min(\Psi^{grav}, \Psi^{ord})$ in this case:

$$\Psi^{ord}(\rho_{i-1}, \rho_i) = \begin{cases} \Theta_6 + \Theta_7 \cdot \left(\rho_i^{-1} - \rho_{i-1}^{-1}\right) & \text{, if } \rho_i > \rho_{i-1} \\ 0 & \text{, otherwise} \end{cases} \tag{6.18}$$

The ordering prior prefers that an object stixel on top of another object is behind or close to the bottom stixel in the 3D scene.

Flat ground prior Ψ^{flat}: Furthermore, small discontinuities in the height of the ground plane are preferred, e.g. caused by a slanted street or a sidewalk. The structural prior for ground stixels $\mathbf{s}_i = \mathbb{G}$ is defined as $\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1}) = \Psi^{flat}$ with

$$\Psi^{flat}(\Delta h_i) = \Theta_8 + \Theta_9 \cdot \Delta h_i^2. \tag{6.19}$$

The height difference Δh_i between the ground stixel $\mathbf{s}_i$ and reference ground is defined by the last previous ground stixel or camera height as in equation (6.16).

6.2.2.2 *Data Likelihood*

The unary term or data likelihood $\Phi(\mathcal{S}_u, \mathbf{f}, \mathbf{c}, \mathbf{d}, \mathbf{l}, \mathcal{O})$ rates the consistency of an individual stixel hypotheses $\mathbf{s}_i$ based on the semantic segmentation $\Phi^{ss}(\mathbf{s}_i, \mathbf{c}_v)$, optical flow field $\Phi^{f}(\mathbf{s}_i, \mathbf{f}_v, v)$, probabilistic single-view depth estimation $\Phi^{svd}(\mathbf{s}_i, \mathbf{d}_v, v)$, and instance segmentation $\Phi^{is}(\mathbf{s}_i, \mathbf{l}_v)$. The data likelihoods are modeled to be independent across the pixels and therefore independent across the rows v in the column.

$$\begin{aligned}\Phi(\mathcal{S}_u, \mathbf{f}, \mathbf{c}, \mathbf{d}, \mathbf{o}, \mathcal{O}) = \sum_{i=1}^{N} \sum_{v=v_i^b}^{v_i^t} (&\lambda_1 \cdot \Phi^{ss}(\mathbf{s}_i, \mathbf{c}_v) + \lambda_2 \cdot \Phi^{f}(\mathbf{s}_i, \mathbf{f}_v, v) + \\ &\lambda_3 \cdot \Phi^{svd}(\mathbf{s}_i, \mathbf{d}_v, v) + \lambda_4 \cdot \Phi^{is}(\mathbf{s}_i, \mathbf{l}_v))\end{aligned} \tag{6.20}$$

Each part of the data term is weighted by $\lambda_{(\cdot)}$. The following paragraphs describe each data term in more detail.

Semantic segmentation Φ^{ss}: The semantic segmentation provides for each row v a class score $\mathbf{c}_v$ to belong to one of the semantic classes defined in table 6.1. For example, a FCN [Long et al., 2015] could be trained to provide such class scores for the specified classes. The data term is defined to prefer stixels $\mathbf{s}_i$ having a semantic class c_i with high class scores $\mathbf{c}_v(c_i)$ at the corresponding rows $v \in (v_i^b, v_i^t)$. Due to the known issues of overconfident scores $\mathbf{c}$ (section 3.1.2.1), the provided scores should not be directly interpreted as probabilities. A recalibration technique could be applied to get well-calibrated probabilities, which was shown to be beneficial for the single-view depth estimates (sections 4.3.2.1 and 5.3.2). However, here it is assumed that calibrated probabilities are not given and the data term is defined as follows:

$$\Phi^{ss}(\mathbf{s}_i, \mathbf{c}_v) = \begin{cases} \min\left(\tau_1, -\log(\mathbf{c}_v(c_i))\right) & \text{, if } c_i \in \mathcal{C}_{m_i} \\ \infty & \text{, otherwise} \end{cases} \tag{6.21}$$

The parameter τ_1 serves as a truncation of the energy term and allows to overrate the class scores, even if the scores are overconfident. For highly overconfident scores, this data term corresponds to a voting scheme considering that the provided class is erroneous with a certain probability.

The condition $c_i \in \mathcal{C}_{m_i}$ reflects the constraint that the semantic classes need to be consistent with the stixel types. Note that there is a unique association from semantic class to stixel type as defined in table 6.1. Thus, semantic segmentation is also exploited to distinguish the different types and which model constraints are applicable.

Optical flow Φ^f: The optical flow input provides for each pixel the displacement $\mathbf{f}_v$ that defines the corresponding image position in the next image $\mathbf{f}_v = \mathbf{p}_{v,1} - \mathbf{p}_{v,0}$. A dense optical flow field could, for example, be estimated using the MirrorFlow method [Hur and Roth, 2017].

The term $\Phi^f(\mathbf{s}_i, \mathbf{f}_v, v)$ rates the consistency of the optical flow at row v for a stixel hypothesis $\mathbf{s}_i$. Due to the definition of a stixel to be a planar part of the scene, the expected optical flow $\hat{\mathbf{f}}_v$ at row v can be derived by the homography $\mathbf{H}_i$ (section 2.1.3) of stixel $\mathbf{s}_i$:

$$\begin{gathered} \hat{\mathbf{f}}_v = \mathbf{H}_i \mathbf{p}_{v,0} - \mathbf{p}_{v,0} \\ \text{with } \mathbf{H}_i = \mathbf{K}\left(\mathbf{R}_i - \rho_i \mathbf{t}_i \mathbf{n}_i^T\right)\mathbf{K}^{-1} \end{gathered} \tag{6.22}$$

The pixel coordinates $\mathbf{p}_{v,0}$ at row v are defined as in equation (6.17) and $\mathbf{K}$ is the intrinsic camera matrix. The other parts are directly defined by the stixel $\mathbf{s}_i$ and its model constraints: $\mathbf{n}_i$ is the normal vector defined by the stixel type (see table 6.1) and ρ_i is the inverse depth label. The motion is either defined by the associated rigid body with $\mathbf{R}_i = \mathbf{R}(o_i)$ and $\mathbf{t}_i = \mathbf{t}(o_i)$ or by the camera rotation $\mathbf{R}_i = \mathbf{R}(o_{BG})$ and the individual stixel motion $\mathbf{t}_i$. The individual stixel motion is defined by the 2D translation over the ground plane in vehicle coordinates.

For sky stixels, there is the special case that the inverse depth is zero $\rho_i = 0$, which simplifies the homography to $\mathbf{H}_i = \mathbf{K}\mathbf{R}_i\mathbf{K}^{-1}$. The stixel's homography serves as the common description of the optical flow for all static and dynamic stixel types.

The consistency between the measured $\mathbf{f}_v$ and expected optical flow $\hat{\mathbf{f}}_v$ is rated as a reprojection error assuming a mixture of a Gaussian and uniform distribution as the underlying measurement model of the optical flow.

$$\Phi^f(s_i, \mathbf{f}_v, v) = \min\left(\tau_2, \log\left(|\Sigma_v|\right) + \frac{1}{2}(\mathbf{f}_v - \hat{\mathbf{f}}_v)^T \Sigma_v^{-1} (\mathbf{f}_v - \hat{\mathbf{f}}_v)\right) \tag{6.23}$$

The parameter τ_2 truncates the energy and is related to the Uniform distribution of the measurement model. Σ_v corresponds to the covariance matrix of the Gaussian distribution of the measurement model. This covariance is defined as $\Sigma_v = diag(\sigma^2_{flow})$, but could be exchanged with a more distinctive covariance matrix. Defining the data term as the minimum of the uniform and Gaussian distribution can be seen as an approximation as shown in [Pfeiffer, 2012, p. 40].

Probabilistic single-view depth Φ^{svd}: The data term $\Phi^{svd}(\mathbf{s}_i, \mathbf{d}_v, v)$ integrates the probabilistic single-view depth estimates provided by ProbDepthNet. The estimate $\mathbf{d}_v$ stands for the parameter $\mu_i, \lambda_i, \sigma_i$ that define the distribution $p_v(\rho \mid I)$ of the inverse depth (equation (3.9) in section 3.3). The provided probability distribution allows to rate the inverse depth of the stixel ρ_i as the negative log-likelihood:

$$\Phi^{svd}(\mathbf{s}_i, \mathbf{d}_v, v) = -\log\left(p_v(\hat{\rho}_{i,v} | I)\right) \tag{6.24}$$

While the depth provided by ProbDepthNet is encoded as $\rho = Z^{-1}$ with Z being the Z-coordinate of the 3D point in camera coordinates, the stixel depth ρ_i is defined as the inverse distance of the camera to the plane. However, the stixel depth ρ_i could be transformed to $\hat{\rho}_{i,v} = \hat{Z}_{i,v}^{-1}$ by

$$\hat{\rho}_{i,v} = \rho_i \frac{\mathbf{n}_i^T \mathbf{K}^{-1} \mathbf{p}_v}{(0\ 0\ 1)^T \mathbf{K}^{-1} \mathbf{p}_v}. \tag{6.25}$$

The pixel coordinates $\mathbf{p}_v$ at row v are defined as in equation (6.17).

Instance segmentation Φ^{is}: The instance segmentation provides labels $\mathbf{l}_v$, which associate each pixel to one rigid body in $\mathcal{O}$. Analogously to the semantic segmentation, it is not assumed that calibrated probabilities are given for the association to a rigid body. Instead of this, it is considered that the instance segmentation provides a single label without any additional probability or score measure. Furthermore, it is assumed that the class of each rigid body is known (e.g. $c(o_i)$ = vehicle). These definitions regarding the output of instance segmentation could be provided e.g. by a Mask R-CNN [He et al., 2017].

The data term $\Phi^{is}(\mathbf{s}_i, \mathbf{l}_v)$ is defined to prefer a stixel $\mathbf{s}_i$ with an object label o_i consistent to the provided instance label $\mathbf{l}_v$:

$$\Phi^{is}(\mathbf{s}_i, \mathbf{l}_v) = \begin{cases} 0 & \text{, if } o_i = \mathbf{l}_v \text{ and } c_i = c(o_i) \\ \infty & \text{, if } o_i = \mathbf{l}_v \text{ and } c_i \neq c(o_i) \\ \tau_3 & \text{, otherwise} \end{cases} \tag{6.26}$$

Setting the energy to infinity for a mismatch of the stixel class c_i and the class of the associated rigid body $c(o_i)$ ensures consistent class and object labels. For the static environment rigid body o_{BG}, all static classes are accepted.

In contrast to SVD-MSfM (chapter 4) and Mono-SF (chapter 5), the truncation τ_3 allows to overrule erroneous instance labels. The truncation τ_3 corresponds to a measurement model, which considers an erroneous instance label with a certain probability.

6.2.3 Solving the Stixel Segmentation Problem

The previous section 6.2.2 described the stixel scene model of the Mono-Stixel method and the segmentation problem for one column formulated as a 1D energy minimization problem (equation (6.12)). A 1D energy minimization problem is solvable via dynamic programming, e.g. by using the Viterbi algorithm. However, even with dynamic programming, the run time grows quadratic with all possible labels for a stixel hypothesis, which results in a high computational effort. Therefore, only the stixel types m_i and segmentations v_i^b, v_i^t are solved globally using dynamic programming. The other stixel labels (c_i, o_i, ρ_i, and $\mathbf{t}_i$) are derived using a local approximation.

6.2.3.1 *Inference as Shortest Path Problem*

The 1D segmentation problem is represented as a hidden semi-Markov model (section 2.2.2). The column represents the sequence of states, where the stixel type corresponds to the hidden variable. The observations at each pixel in the column are defined by the optical flow, single-view depth, semantic segmentation, and instance segmentation. Each stixel covers a certain part of the column, which is modeled in the semi-Markov model by remaining in the same state for a certain duration. Additional constraints for the segmentation ($v_1^b = 1$, $v_N^t = h$ and $v_{n-1}^t + 1 = v_n^b$) ensure that the whole column is covered with non-overlapping stixels. The segmentation problem can be expressed by the directed graph illustrated in figure 6.3, which represents a trellis diagram for hidden semi-Markov models. The explicit modeling to

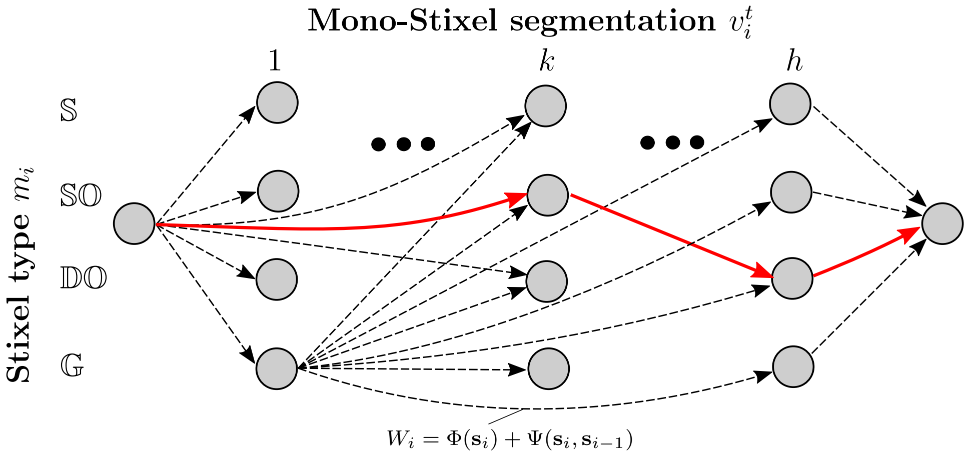

Mono-Stixel segmentation v_i^t
1
k
h
Stixel type m_i
$\mathbb{S}$
$\mathbb{SO}$
$\mathbb{DO}$
$\mathbb{G}$
$W_i = \Phi(\mathbf{s}_i) + \Psi(\mathbf{s}_i, \mathbf{s}_{i-1})$

Figure 6.3: Illustration of Mono-Stixel inference as shortest path problem (illustration following [Cordts et al., 2017]). Each edge of the directed graph corresponds to a new stixel $\mathbf{s}_i$. The target node defines the stixel type m_i (vertical position). The segmentation is defined by the horizontal position of the source (v_i^b) and target node (v_i^b). The source node additionally corresponds to the type of the previous stixel $\mathbf{s}_{i-1}$. A weight W_i is associated with each edge and defined by the data $\Phi(\mathbf{s}_i)$ and prior $\Psi(\mathbf{s}_i, \mathbf{s}_{i-1})$ terms. A path (e.g. red edges) from the source (left) to the sink (right) corresponds to one stixel segmentation $\mathcal{S}_u$. The shortest path in terms of the sum of the weights corresponds to the optimal stixel segmentation. Only a subset of edges are shown in the figure for visualization purposes.

remain in the same state introduces edges that skip certain parts of the sequence, but allows representing the segmentation boundaries. Each edge corresponds to one stixel segment $\mathbf{s}_i$. While the horziontal position of the target node defines the upper segmentation boundary v_i^t, the vertical position of the target node corresponds to the stixel type m_i. The source node defines the bottom v_i^b and the type of the previous stixel m_{i-1}. A path from the source (left) to the sink (right) defines one possible stixel segmentation $\mathcal{S}_u$. For example, the red path in figure 6.3 would correspond to the following segmentation:

$$\begin{aligned}\mathcal{S}_u = \{&\mathbf{s}_1 = (v_i^b = 1, v_i^t = k, m_i = \mathbb{SO}, ...),\\ &\mathbf{s}_2 = (v_i^b = k+1, v_i^t = h, m_i = \mathbb{DO}, ...)\}\end{aligned} \tag{6.27}$$

While the graph structure defines the possible segmentations $\mathcal{S}_u$ represented as paths, weights are introduced to define the costs along the path in terms of the energy term in equation (6.12). Each weight W_i is associated with one edge and defined as:

$$W_i = \underbrace{\sum_{v=v_i^b}^{v_i^t} \left(\lambda_1 \cdot \Phi^{ss}(\mathbf{s}_i, \mathbf{c}_v) + \lambda_2 \cdot \Phi^{f}(\mathbf{s}_i, \mathbf{f}_v, v) + \lambda_3 \cdot \Phi^{svd}(\mathbf{s}_i, \mathbf{d}_v, v) + \lambda_4 \cdot \Phi^{is}(\mathbf{s}_i, \mathbf{l}_v)\right)}_{\text{data term: } \Phi(\mathbf{s}_i)} + \underbrace{\Psi^{str}(\mathbf{s}_i, \mathbf{s}_{i-1}) + \Theta_1}_{\text{prior term: } \Psi(\mathbf{s}_{i-1}, \mathbf{s}_i)} \tag{6.28}$$

Therefore, finding the shortest path in terms of the sum of the weights along the path provide the global optimal stixels $\mathcal{S}_u$ regarding the stixel types m_i and segmentation v_i^b, v_i^t. The inference of the shortest path problem is performed via the Viterbi algorithm (section 2.2.2). The asymptotic runtime complexity of the shortest path problem is $\mathcal{O}(h^2)$.

6.2.3.2 *Local Stixel Estimation*

The formulation as a shortest path problem solves globally the stixel types and segmentation. To estimate all other stixel labels (c_i, o_i, ρ_i, and $\mathbf{t}_i$), which are also needed to define the weights of the edges, a local approximation is proposed. This means finding the semantic class c_i, object instance label o_i, inverse depth ρ_i and motion $\mathbf{t}_i$ for one stixel $\mathbf{s}_i$ given its segmentation v_i^b, v_i^t and type m_i.

The general approach proposed here is divided into three steps, which are related to the MLESAC [Torr and Zisserman, 2000] method. First, a set of stixel hypothesis $\mathbf{s}_i \in \mathcal{H} = \{\mathbf{s}_{h1}, \mathbf{s}_{h2}, ..., \mathbf{s}_{hN}\}$ is generated. Second, each stixel hypothesis is rated by its data term $\Phi(\mathbf{s}_{hj})$ (equation (6.28)). Third, the best stixel hypothesis with the lowest cost value is selected by

$$\mathbf{s}_i = \underset{\mathbf{s}_{hj} \in \mathcal{H}}{\arg\min}\, \Phi(\mathbf{s}_{hj}). \tag{6.29}$$

The following part presents how to define the set of stixel hypothesis $\mathcal{H}$ depending on the given stixels type m_i and segmentation v_i^b, v_i^t.

Sky stixel $m_i = \mathbb{S}$: The simplest case is a sky stixel $m_i = \mathbb{S}$ because all labels are still defined based on the model constraints in table 6.1: $c_i = \mathbb{S}$, $\rho_i = 0$, $o_i = o_{BG}$, and $\mathbf{t}_i = \mathbf{t}(o_i)$. Thus, for a sky stixel, there exists only one stixel hypothesis, which defines the weight of the corresponding edge.

Static object and ground stixel $m_i \in \{\mathbb{G}, \mathbb{SO}\}$: The approach to define a set of stixel hypotheses $\mathcal{H}$ is the same for both static types, ground $\mathbb{G}$ and static object $\mathbb{SO}$. Based on the model constraints defined in table 6.1, the object label is defined as $o_i = o_{BG}$ with the corresponding motion $\mathbf{t}_i = \mathbf{t}(o_i)$. Thus, only the semantic class c_{hj} and the inverse depth ρ_{hj} needs to be estimated.

The set of possible semantic class hypotheses is defined in table 6.1 as $\mathcal{C}_{m_i}$. Merely the data term of the semantic segmentation $\Phi^{ss}(c_{hj}, \mathbf{c}_v)$ depends on the semantic class, which allows minimizing the semantic class c_i individually:

$$c_i = \underset{c_{hj} \in \mathcal{C}_{m_i}}{\arg\min} \sum_{v=v_i^b}^{v_i^t} \Phi^{ss}(c_{hj}, \mathbf{c}_v) \tag{6.30}$$

For the inverse depth ρ_i, each optical flow estimate $\mathbf{f}_v$ and each expected value of the single-view depth distribution parameterized by $\mathbf{d}_v$ provide one hypothesis $\mathcal{HD} = \{\rho_{h1}, \rho_{h2}, ..., \rho_{hM}\}$. Referring to the proposed approach, the inverse depth hypothesis ρ_{hj} minimizing the optical flow $\Phi^f(\mathbf{s}_i, \mathbf{f}_v, v)$ and single-view depth $\Phi^{svd}(\mathbf{s}_i, \mathbf{d}_v, v)$ data term is selected by

$$\rho_i = \underset{\rho_{hj} \in \mathcal{HD}}{\arg\min} \sum_{v=v_i^b}^{v_i^t} \left(\lambda_2 \cdot \Phi^f(\mathbf{s}_i, \mathbf{f}_v, v) + \lambda_3 \cdot \Phi^{svd}(\mathbf{s}_i, \mathbf{d}_v, v) \right). \tag{6.31}$$

Note that the costs for the semantic c_{hj} and inverse depth ρ_{hj} hypotheses can be stored in an integral table and does not need to be computed for each stixel segment individually.

An inverse depth hypothesis ρ_{hj} based on the expected value $\mu_{\rho,v}$ of the single-view depth distribution parameterized by $\mathbf{d}_v$ is defined by rearranging equation (6.25):

$$\rho_{hj} = \mu_{\rho,v} \frac{(0\ 0\ 1)^T \mathbf{K}^{-1} \mathbf{p}_v}{\mathbf{n}_i^T \mathbf{K}^{-1} \mathbf{p}_v} \tag{6.32}$$

An inverse depth hypothesis ρ_{hj} based on an optical flow measure $\mathbf{f}_v$ is derived as a direct linear transform for the stixel's homography estimation. The optical flow-based data term Ψ^f is minimal if the expected optical flow is identical to the measured optical flow, which defines the following equations:

$$\begin{aligned} s\,\underbrace{\mathbf{K}^{-1}\mathbf{p}_{v,1}}_{\mathbf{x}_1} &\overset{!}{=} (\mathbf{R}_i - \rho_i \mathbf{t}_i \mathbf{n}_i^T)\,\underbrace{\mathbf{K}^{-1}\mathbf{p}_{v,0}}_{\mathbf{x}_0} \\ s\mathbf{x}_1 &= \underbrace{\mathbf{R}_i \mathbf{x}_0}_{\mathbf{x}_0^r} - \rho_i \mathbf{t}_i \mathbf{n}_i^T \mathbf{x}_0 \end{aligned} \tag{6.33}$$

The pixel position $\mathbf{p}_{v,1}$ in the next frame at $t = 1$ is defined by $\mathbf{p}_{v,1} = \mathbf{f}_v + \mathbf{p}_{v,0}$. A homography is only defined up to an unknown scale, which is made explicity by the arbitrary scale s. Dividing by the third equation cancels out the unknown scale s and results in the following two equations:

$$\begin{bmatrix} \mathbf{x}_{1,x} \cdot (\mathbf{x}^r_{0,z} - \rho_i \mathbf{t}_z \mathbf{n}^T \mathbf{x}_0) \\ \mathbf{x}_{1,y} \cdot (\mathbf{x}^r_{0,z} - \rho_i \mathbf{t}_z \mathbf{n}^T \mathbf{x}_0) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{1,z} \cdot (\mathbf{x}^r_{0,x} - \rho_i \mathbf{t}_x \mathbf{n}^T \mathbf{x}_0) \\ \mathbf{x}_{1,z} \cdot (\mathbf{x}^r_{0,y} - \rho_i \mathbf{t}_y \mathbf{n}^T \mathbf{x}_0) \end{bmatrix} \tag{6.34}$$

Rearranging these equations provides a linear system of equations for the inverse depth hypothesis ρ_i:

$$\underbrace{\begin{bmatrix} \mathbf{n}^T \mathbf{x}_0 \cdot (\mathbf{t}_z \mathbf{x}_{1,x} - \mathbf{t}_x \mathbf{x}_{1,z}) \\ \mathbf{n}^T \mathbf{x}_0 \cdot (\mathbf{t}_z \mathbf{x}_{1,y} - \mathbf{t}_y \mathbf{x}_{1,z}) \end{bmatrix}}_{\mathbf{A}_\rho} \rho_i = \underbrace{\begin{bmatrix} \mathbf{x}_{1,x} \mathbf{x}^r_{0,z} - \mathbf{x}_{1,z} \mathbf{x}^r_{0,x} \\ \mathbf{x}_{1,y} \mathbf{x}^r_{0,z} - \mathbf{x}_{1,z} \mathbf{x}^r_{0,y} \end{bmatrix}}_{\mathbf{b}_\rho} \tag{6.35}$$

One way to solve this system is the formulation as a normal equation solved using the pseudo inverse. The solution serves as one inverse depth hypothesis ρ_{hj}.

$$\rho_{hj} = (\mathbf{A}_\rho^T \mathbf{A}_\rho)^{-1} \mathbf{A}_\rho^T \mathbf{b}_\rho \tag{6.36}$$

A solution based on a normal equation minimizes the quadratic error of the defined algebraic error $||\mathbf{b}_\rho - \mathbf{A}_\rho \rho_{hj}||^2$. The presented approach to derive an inverse depth hypothesis ρ_{hj} follows a direct linear transform for homography estimation – but with directly estimating the one degree of freedom of the stixel's homography considering the model constraints defined in table 6.1.

Dynamic object stixel $m_i = \mathbb{DO}$: In contrast to the static stixel types, the motion and associated rigid body are not directly defined by the model assumptions for a dynamic object stixel and needs to be estimated as well. The dynamic stixel model distinguishes two motion cases (see table 6.1). First, the corresponding rigid body is part of the given set of rigid bodies $\mathcal{O}$. Second, the motion is expressed as an individual stixel motion $\mathbf{t}_i$.

In the first case, each rigid body is considered as one hypothesis $o_{hj} \in \mathcal{O} \setminus o_{BG}$. This defines also the motion and class $c_{hj} = c(o_{hj})$ of the stixel. The inverse depth for each rigid body is derived in the same way as for the static stixels (equations (6.32), (6.32), and (6.36)) by replacing the camera motion with the relative motion of the corresponding rigid body.

In the second case, the corresponding rigid body of the stixel is not part of the set $\mathcal{O}$. Consequently, the semantic class c_i is not constrained by the rigid body and is estimated analogously to the static stixels (equation (6.41)). The inverse depth ρ_i is derived by equation (6.31) – but the set of inverse depth hypothesis $\mathcal{H}$ is only defined by the single-view depth estimates (equation (6.32)) due to the missing rigid body motion. However, the optical flow estimates are exploited to derive a motion hypothesis using a direct linear transform for the stixel's homography estimation. Based on the model constraints of a dynamic object stixel, the corresponding stixel homography has two degrees of freedom, which are denoted as $\tilde{\mathbf{t}}_i$. These two degrees of freedom are the 2D translational motion of the stixel over the ground plane scaled by the inverse depth ρ_i:

$$\rho_i \mathbf{t}_i = \mathbf{R}^{v2c}\tilde{\mathbf{t}}_i = \mathbf{R}^{v2c}\begin{bmatrix}\tilde{\mathbf{t}}_x\\ \tilde{\mathbf{t}}_y\\ 0\end{bmatrix} = \begin{bmatrix}\mathbf{R}^{v2c}_{0,0}\tilde{\mathbf{t}}_x + \mathbf{R}^{v2c}_{0,1}\tilde{\mathbf{t}}_y\\ \mathbf{R}^{v2c}_{1,0}\tilde{\mathbf{t}}_x + \mathbf{R}^{v2c}_{1,1}\tilde{\mathbf{t}}_y\\ \mathbf{R}^{v2c}_{2,0}\tilde{\mathbf{t}}_x + \mathbf{R}^{v2c}_{2,1}\tilde{\mathbf{t}}_y\end{bmatrix} \tag{6.37}$$

Analagously to equation (6.34) (substitution of $\rho_i\mathbf{t}_i$ with the equations defined above), the following two equations are defined:

$$\begin{bmatrix}\mathbf{x}_{1,x}\cdot(\mathbf{x}^r_{0,z} - (\mathbf{R}^{v2c}_{2,0}\tilde{\mathbf{t}}_x + \mathbf{R}^{v2c}_{2,1}\tilde{\mathbf{t}}_y)\mathbf{n}^T\mathbf{x}_0)\\ \mathbf{x}_{1,y}\cdot(\mathbf{x}^r_{0,z} - (\mathbf{R}^{v2c}_{2,0}\tilde{\mathbf{t}}_x + \mathbf{R}^{v2c}_{2,1}\tilde{\mathbf{t}}_y)\mathbf{n}^T\mathbf{x}_0)\end{bmatrix} = \begin{bmatrix}\mathbf{x}_{1,z}\cdot(\mathbf{x}^r_{0,x} - (\mathbf{R}^{v2c}_{0,0}\tilde{\mathbf{t}}_x + \mathbf{R}^{v2c}_{0,1}\tilde{\mathbf{t}}_y)\mathbf{n}^T\mathbf{x}_0)\\ \mathbf{x}_{1,z}\cdot(\mathbf{x}^r_{0,y} - (\mathbf{R}^{v2c}_{2,0}\tilde{\mathbf{t}}_x + \mathbf{R}^{v2c}_{2,1}\tilde{\mathbf{t}}_y)\mathbf{n}^T\mathbf{x}_0)\end{bmatrix} \tag{6.38}$$

Rearranging these equations results in a linear system of equations with $\tilde{\mathbf{t}}_i$ as the free variables:

$$\underbrace{\begin{bmatrix}\mathbf{n}^T\mathbf{x}_0\cdot(\mathbf{x}_{1,x}\mathbf{R}^{v2c}_{2,0} - \mathbf{x}_{1,z}\mathbf{R}^{v2c}_{0,0}) & \mathbf{n}^T\mathbf{x}_0\cdot(\mathbf{x}_{1,x}\mathbf{R}^{v2c}_{2,1} - \mathbf{x}_{1,z}\mathbf{R}^{v2c}_{0,1})\\ \mathbf{n}^T\mathbf{x}_0\cdot(\mathbf{x}_{1,y}\mathbf{R}^{v2c}_{2,0} - \mathbf{x}_{1,z}\mathbf{R}^{v2c}_{1,0}) & \mathbf{n}^T\mathbf{x}_0\cdot(\mathbf{x}_{1,y}\mathbf{R}^{v2c}_{2,1} - \mathbf{x}_{1,z}\mathbf{R}^{v2c}_{1,1})\end{bmatrix}}_{\mathbf{A}_{\tilde{\mathbf{t}}}}\begin{bmatrix}\tilde{\mathbf{t}}_x\\ \tilde{\mathbf{t}}_y\end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{x}_{1,x}\mathbf{x}^r_{0,z} - \mathbf{x}_{1,z}\mathbf{x}^r_{0,x}\\ \mathbf{x}_{1,y}\mathbf{x}^r_{0,z} - \mathbf{x}_{1,z}\mathbf{x}^r_{0,y}\end{bmatrix}}_{\mathbf{b}_{\tilde{\mathbf{t}}}} \tag{6.39}$$

Based on the normal equations and pseudo inverse matrix, a hypothesis for $\tilde{\mathbf{t}}_{hj}$ can be derived by

$$\tilde{\mathbf{t}}_{hj} = (\mathbf{A}^T_{\tilde{\mathbf{t}}}\mathbf{A}_{\tilde{\mathbf{t}}})^{-1}\mathbf{A}^T_{\tilde{\mathbf{t}}}\mathbf{b}_{\tilde{\mathbf{t}}}. \tag{6.40}$$

The best hypothesis $\tilde{\mathbf{t}}_{hj}$ defined by one optical flow estimate $\mathbf{f}_v$ in the stixel segment is selected by

$$\tilde{\mathbf{t}}_i = \underset{\tilde{\mathbf{t}}_{hj}\in\mathcal{HT}}{\arg\min}\sum_{v=v_i^b}^{v_i^t}\Phi^f(\tilde{\mathbf{t}}_i, \mathbf{f}_v, v). \tag{6.41}$$

Analagously, this represents a direct linear transform for homography estimation constrained by the stixel model for dynamic stixel. The estimation of $\tilde{\mathbf{t}}_i$ is independent of the estimation of the inverse depth ρ_i. After estimating both, the scale-aware translation $\mathbf{t}_i$ is computed based on ρ_i and $\tilde{\mathbf{t}}_i$ using equation (6.37).

The stixel hypotheses for the different rigid bodies and the stixel hypothesis with an individual motion are compared by equation (6.29). This provides finally the stixel $\mathbf{s}_i$ for the dynamic object stixel type and allows computing the weight of the corresponding edge.

Based on these local approximations, the stixel hypotheses and weights of each edge are defined, which allows performing the stixel inference as the shortest path problem for each column. The local approximation results in a computational effort of $\mathcal{O}\left(h^3\right)$ per column. To segment the whole image, w/w_s columns need to be processed and the overall asymptotic runtime complexity is $\mathcal{O}\left(wh^3\right)$.

6.2.4 Independent Moving Object Detection

The dynamic stixels are defined as potentially moving, which also includes e.g. parking vehicles. To represent which objects are really in motion, an additional independent moving object (IMO) score γ_i is estimated for each stixel $\mathbf{s}_i$. The IMO score is determined based on a statistical hypothesis test comparing the hypothesis of being static $\mathcal{H}^{stat}$ with the hypothesis of being in motion $\mathcal{H}^{mov}$.

The general definition of a generalized likelihood ratio test is

$$\gamma = \frac{\max_{\Theta_0} p(\mathbf{X} \mid \Theta_0, \mathcal{H}_0)}{\max_{\Theta_1} p(\mathbf{X} \mid \Theta_1, \mathcal{H}_1)}. \tag{6.42}$$

The hypothesis $\mathcal{H}_0$ is tested against the hypothesis $\mathcal{H}_1$ based on the observations $\mathbf{X}$. The best model parameters that explain each hypothesis are $\Theta_{(\cdot)}$.

The generalized likelihood ratio test is applied to the IMO detection of a given stixel $\mathbf{s}_i$. The hypotheses are $\mathcal{H}_0 = \mathcal{H}^{mov}$ and $\mathcal{H}_1 = \mathcal{H}^{stat}$ with the corresponding stixel hypotheses $\Theta_0 = \mathbf{s}_i^{mov}$ and $\Theta_1 = \mathbf{s}_i^{stat}$. The observations $\mathbf{X}$ are the measurements of the data terms such as the optical flow and single-view depth estimates. Intuitively, the hypothesis test should rate the likelihood that an individual rigid body motion $o_i \in \{\mathcal{O} \setminus o_{BG}\}$ or individual stixel motion $\mathbf{t}_i$ is needed to explain the measurements in the respective stixel segment.

Switching to the logarithmic space defines the following equation for the IMO score γ_i:

$$\begin{aligned}
\gamma_i &= \log \frac{\max_{s_i^{mov}} p(\mathbf{X} \mid \mathbf{s}_i^{mov}, \mathcal{H}^{mov})}{\max_{\mathbf{s}_i^{stat}} p(\mathbf{X} \mid \mathbf{s}_i^{stat}, \mathcal{H}^{stat})} \\
&= \max_{\mathbf{s}_i^{mov}} \log\left(p(\mathbf{X} \mid \mathbf{s}_i^{mov}, \mathcal{H}^{mov})\right) - \max_{\mathbf{s}_i^{stat}} \log\left(p(\mathbf{X} \mid \mathbf{s}_i^{stat}, \mathcal{H}^{stat})\right) \\
&= -\left(\min_{\mathbf{s}_i^{mov}} \Phi(\mathbf{s}_i^{mov}) - \min_{\mathbf{s}_i^{stat}} \Phi(\mathbf{s}_i^{stat})\right)
\end{aligned} \tag{6.43}$$

The data term $\Phi(\mathbf{s}_i)$ of one stixel is defined in equation (6.28).

While either $\mathbf{s}_i^{stat}$ or $\mathbf{s}_i^{mov}$ is identical to the estimated stixel $\mathbf{s}_i$ of the segmentation, the other stixel hypothesis corresponds to the most likely stixel with a different motion state. The alternative stixel hypothesis is estimated based on the same stixel model as in table 6.1 and local estimation described in section 6.2.3.2. The only difference is that a static stixel can be also of a semantic class of dynamic stixels (vehicle, two-wheeler, person).

6.2.5 Global Stixel Scene Flow Optimization

Chapter 5 presents the Mono-SF approach for a joint scene flow optimization. The proposed optimization is also applicable to a stixel world scene model. The Mono-SF approach is briefly summarized here and the adaption to the stixel world representation is described.

Originally, the underlying scene model of Mono-SF is defined by a set of rigid bodies with their 6D motions $\mathbf{T}_j$ and a set of superpixels represented by their scaled normals $\mathbf{n}_i$. First, the definition of rigid bodies is identical to the set of rigid bodies $\mathcal{O}$ of the stixel world (provided for both approaches by SVD-MSfM as described in chapter 4). Additionally, each stixel that undergoes an individual motion is considered as an additional rigid body – but only the 2D translations $\mathbf{t}_i^v$ over the ground plane are considered as variables for the optimization. Second, each stixel $\mathbf{s}_i$ is a more specialized and constrained kind of superpixel that represents a planar surface element. The scaled normal vector $\mathbf{n}_i$ of each stixel is derivable by its type ($\mathbf{n}_i$ in table 6.1) and inverse depth ρ_i. Mono-SF optimizes all three parameters of the scaled normal, which allows estimating stixels that represent slanted objects and road surfaces. The other labels of the stixels (e.g. segmentation and semantic class) remain fixed during optimization.

The energy minimization problem (equation (5.1)) of Mono-SF is directly applicable to optimize the stixel world representation in terms of the represented scene flow. However, two adaptions are proposed. First, the stixel world representation includes the semantic class c_i of each stixel. Therefore, instead of one general smoothing weight, the smoothing weights in equation (5.5) are defined differently depending on whether the adjacent stixels belong to the same or different classes. Thereby, a smoothing effect between different objects is reduced and, for example, a pole is not smoothed into the background. Second, the Mono-Stixel approach is designed to perform its optimization based on the optical flow measurements instead of a photometric distance that needs direct access to the images. The data term of Mono-SF (equation (5.2)) is adapted by replacing the photometric error with a reprojection error $\Phi^f(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)$:

$$\begin{aligned}\Phi(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) = \lambda_4 \cdot \Phi^f(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j) \\ + \lambda_5 \cdot \sum_{t \in \{0,1\}} \Phi_t^{svd}(\mathbf{p}_0, \mathbf{n}_i, \mathbf{T}_j)\end{aligned} \tag{6.44}$$

The reprojection error is defined as in equation (6.23), which approximates the optical flow measurement model as a mixture of Gaussian and uniform distribution.

In contrast to Mono-SF, the final accuracy additionally depends on the accuracy of the provided optical flow. However, the experiments in section 5.3.1 reveal that current SotA methods (e.g. HD^3-F [Yin et al., 2019]) are superior to Mono-SF in terms of optical flow estimation.

The optimization is iteratively performed for 10 times using sequential tree-reweighted message passing (section 2.2.2). Converting the continuous variables into a discrete labeling problem follows the same strategy as for Mono-SF (section 5.2.3) by generating 5 particles for each motion variable and 10 particles for each 3D normal vector.

While the stixel segmentation of the Mono-Stixel method initially treats each column separately, applying Mono-SF couples different columns to optimize the stixel segments globally in terms of the scene flow.

6.3 Experimental Evaluation of Mono-Stixel Method

One motivation of the proposed Mono-Stixel algorithm is providing a scene representation, in particular providing the scene flow, in a compact form using the stixel world representation. The first section 6.3.1 provides qualitative results of the stixel segmentation in comparison to its inputs. The results show a significant improvement of the provided stixel representation compared to the respective inputs. In addition to the compact representation, the second section 6.3.2 validates quantitatively that the provided scene flow accuracy is competitive to Mono-SF. Even more, the qualitative results show better characteristics of the stixels for thin objects, especially in low-parallax situations. One additional advantage of the Mono-Stixel approach is that merely the optical flow and camera motion are mandatory. This enables analyzing the impact of the deep learning inputs in terms of scene flow estimation and IMO detection. The results are presented in sections 6.3.3 and 6.3.4.

The experimental results are based on the Mono-Stixel approach using the following inputs: (1) MirrorFlow [Hur and Roth, 2017] is implemented to provide a dense optical flow field, (2) the probabilistic single-view depth estimates are provided by ProbDepthNet (section 3.3), (3) instance labels are provided by a Mask R-CNN [He et al., 2017] (implementation of [Wang, 2018]), (4) a FCN [Long et al., 2015] is trained to provide the class scores for the semantic classes defined in table 6.1, and (5) the set of rigid bodies and their scale-aware 6D motions are provided by SVD-MSfM (chapter 4). ProbDepthNet is trained on KITTI [Geiger et al., 2013] with pretraining on Cityscapes [Cordts et al., 2016] and Mask R-CNN is only trained on Cityscapes [Cordts et al., 2016]. FCN is pretrained on Cityscapes [Cordts et al., 2016] and fine-tuned on a dataset of 470 KITTI images [Geiger et al., 2013], which are collected from the labeled subsets in [Kundu et al., 2014, Ros et al., 2015, Sengupta et al., 2013, Xu et al., 2016, Ošep et al., 2016, Upcroft et al., 2014]. For all networks, it is ensured that the training images are from different sequences than the KITTI scene flow training dataset [Menze and Geiger, 2015], which is the dataset used for evaluation.

6.3.1 Qualitative Results of Stixel Segmentation

Qualitative results of the stixel segmentation are shown in figure 6.4 using a stixel width of $w_s = 5$ pixel. The figures show additionally the inputs of the Mono-Stixel method to illustrate the impact of the Mono-Stixel segmentation compared to its given inputs. The comparison to the inputs supports the following characteristics. The Mono-Stixel method can provide an improvement for noisy optical flow estimates, e.g. visible in figure 6.4 (a). Additionally, significant failures of the optical flow such as on the crossing vehicles in figure 6.4 (b,d) can be corrected by the Mono-Stixel method. Compared to the single-view depth estimates, the provided depth by the Mono-Stixel method is sharper, for example, for the signage in figure 6.4 (a,b). Another characteristic is that the optical flow of the Mono-Stixel method corresponds to the projection of the scene flow instead of an appearance-based optical flow. This difference is, for example, visible for the moving shadow of the vehicle in figure 6.4 (c). While the instance segmentation is visually similar for Mask R-CNN and stixel segmentation, the semantic segmentation benefits from applying the Mono-Stixel method, so that the incorrect classification as a sidewalk in the front-left of the vehicle in figure 6.4 (d) is corrected.

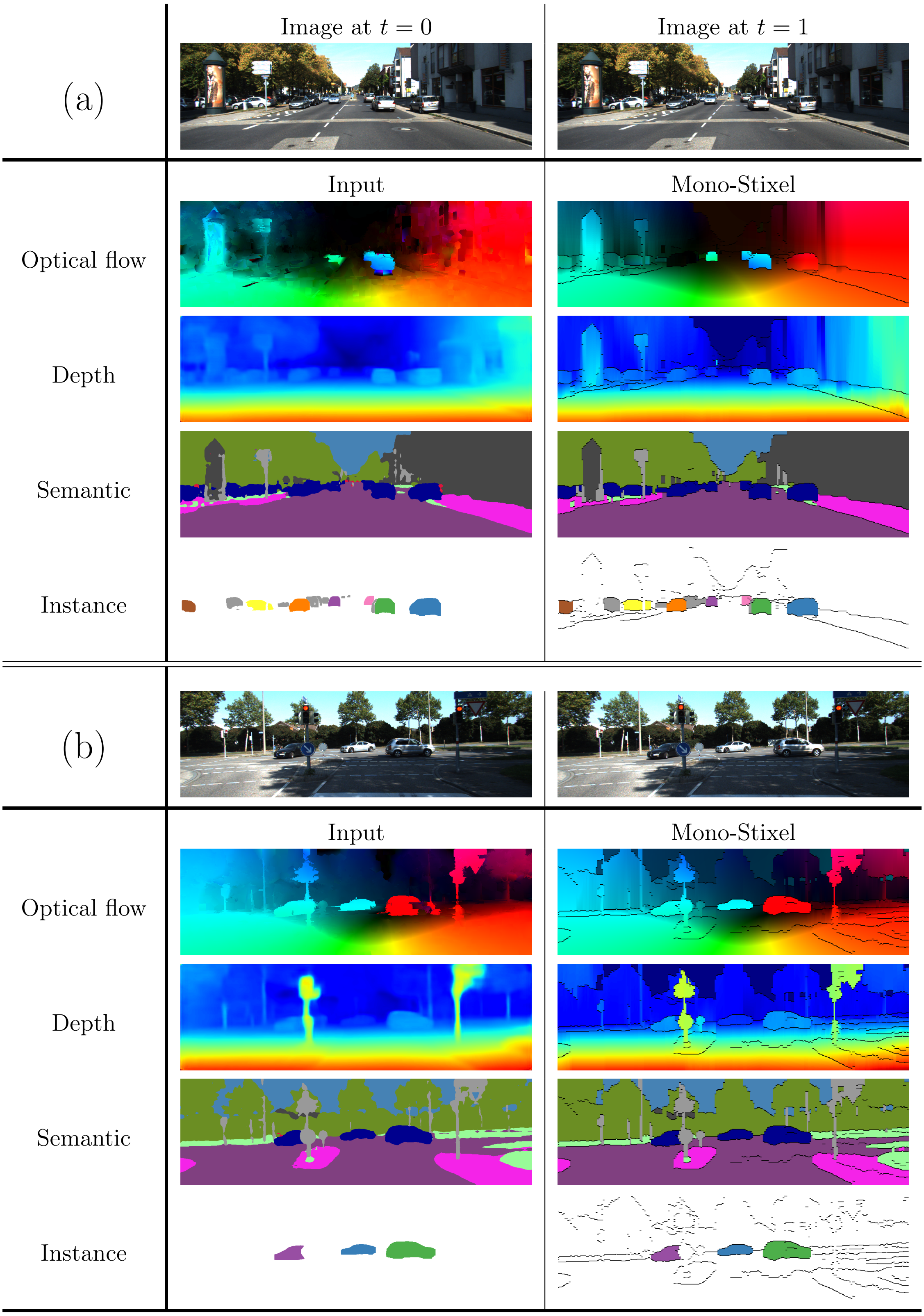


Image at $t = 0$
Image at $t = 1$
(a)
Input
Mono-Stixel
Optical flow
Depth
Semantic
Instance
(b)
Input
Mono-Stixel
Optical flow
Depth
Semantic
Instance

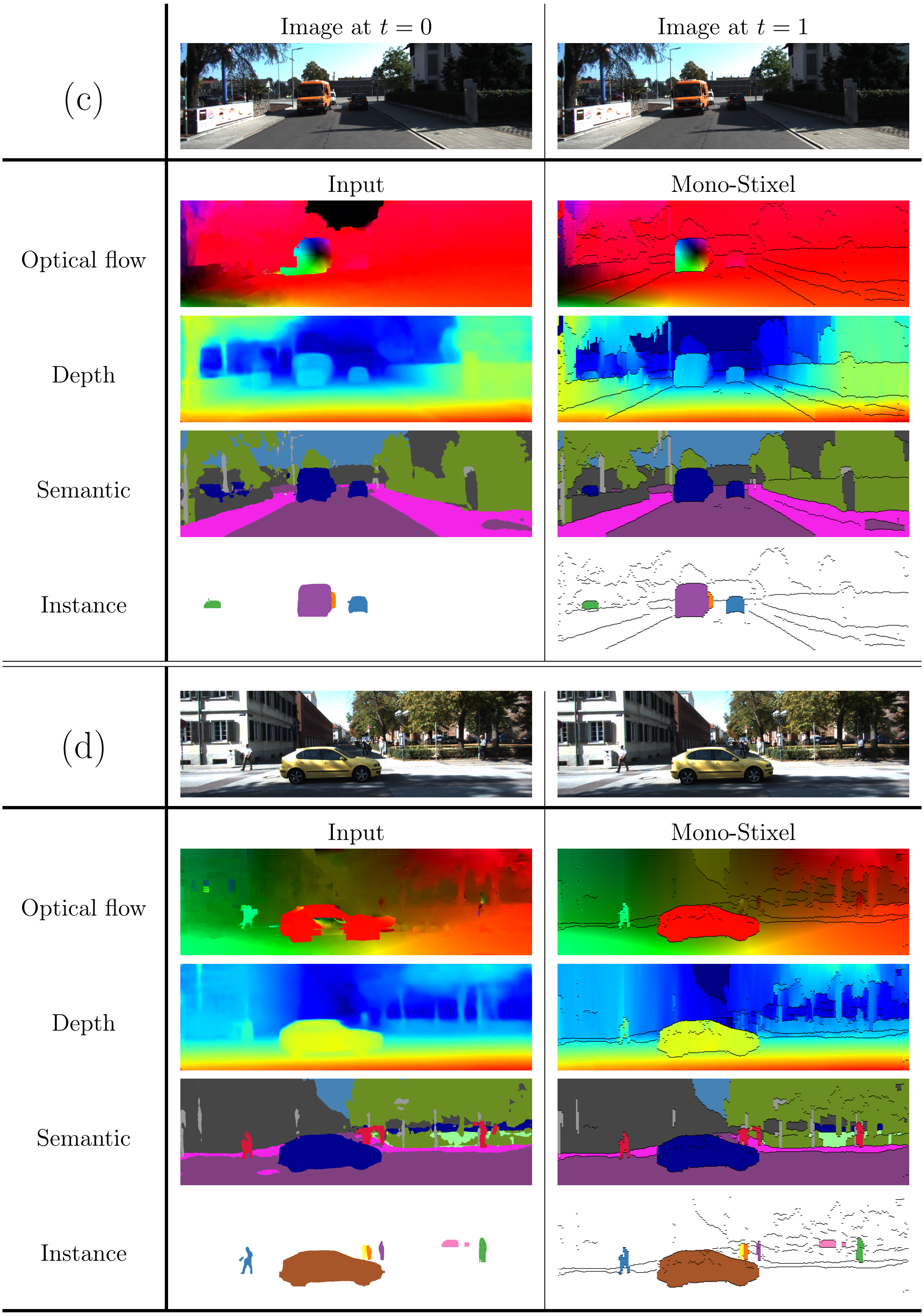


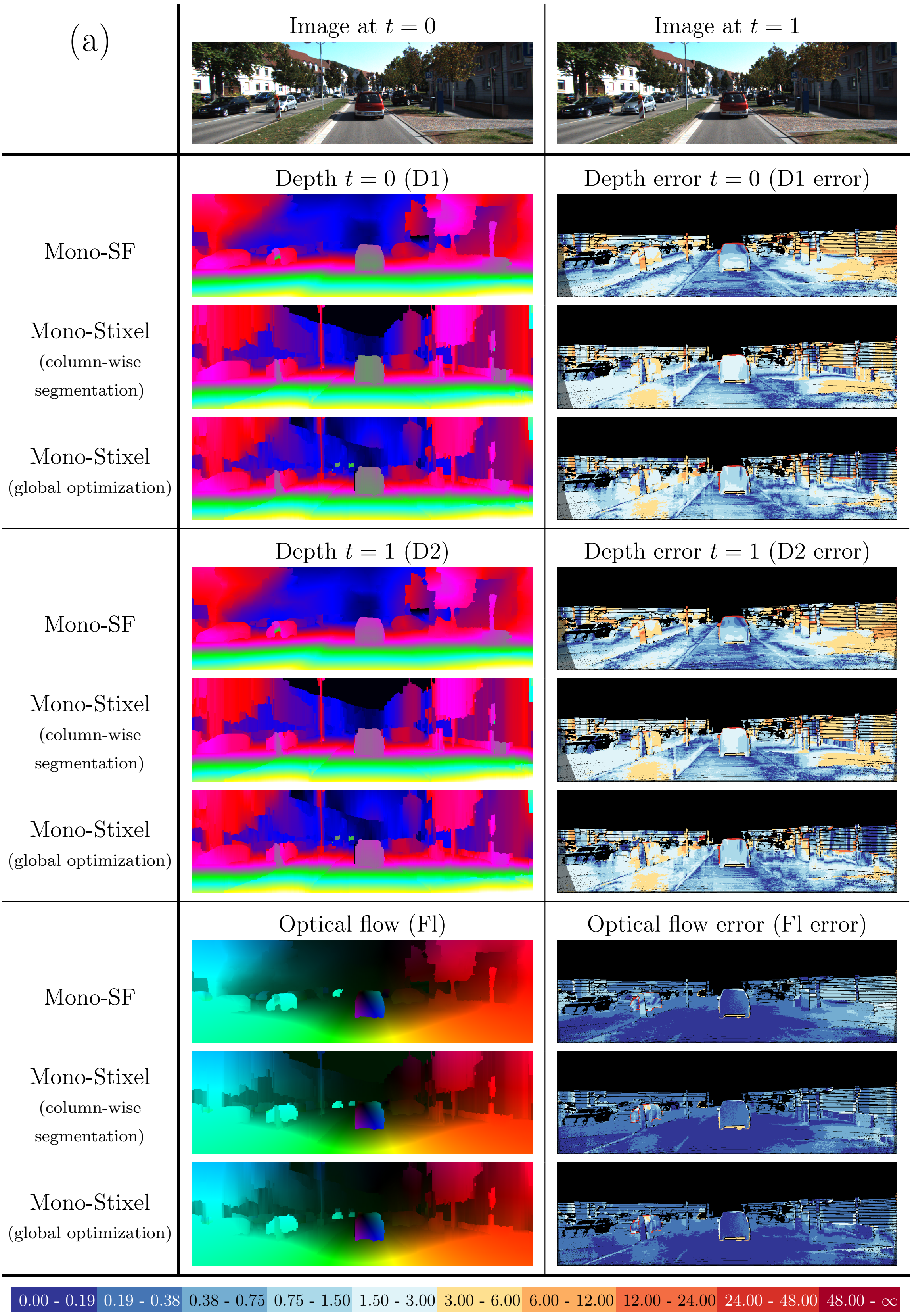
Figure 6.4: Qualitative results of the Mono-Stixel method in comparison to its inputs on the KITTI scene flow training dataset [Menze and Geiger, 2015]. The color coding represent the estimated depth (from close (warm) to far (cool)), the optical flow (Middlebury color coding [Baker et al., 2011]) or semantic class (color coding of [Cordts et al., 2016]). The stixel's top and bottom are visualized in black, the vertical boundaries (columns) every 5 pixels are not shown.

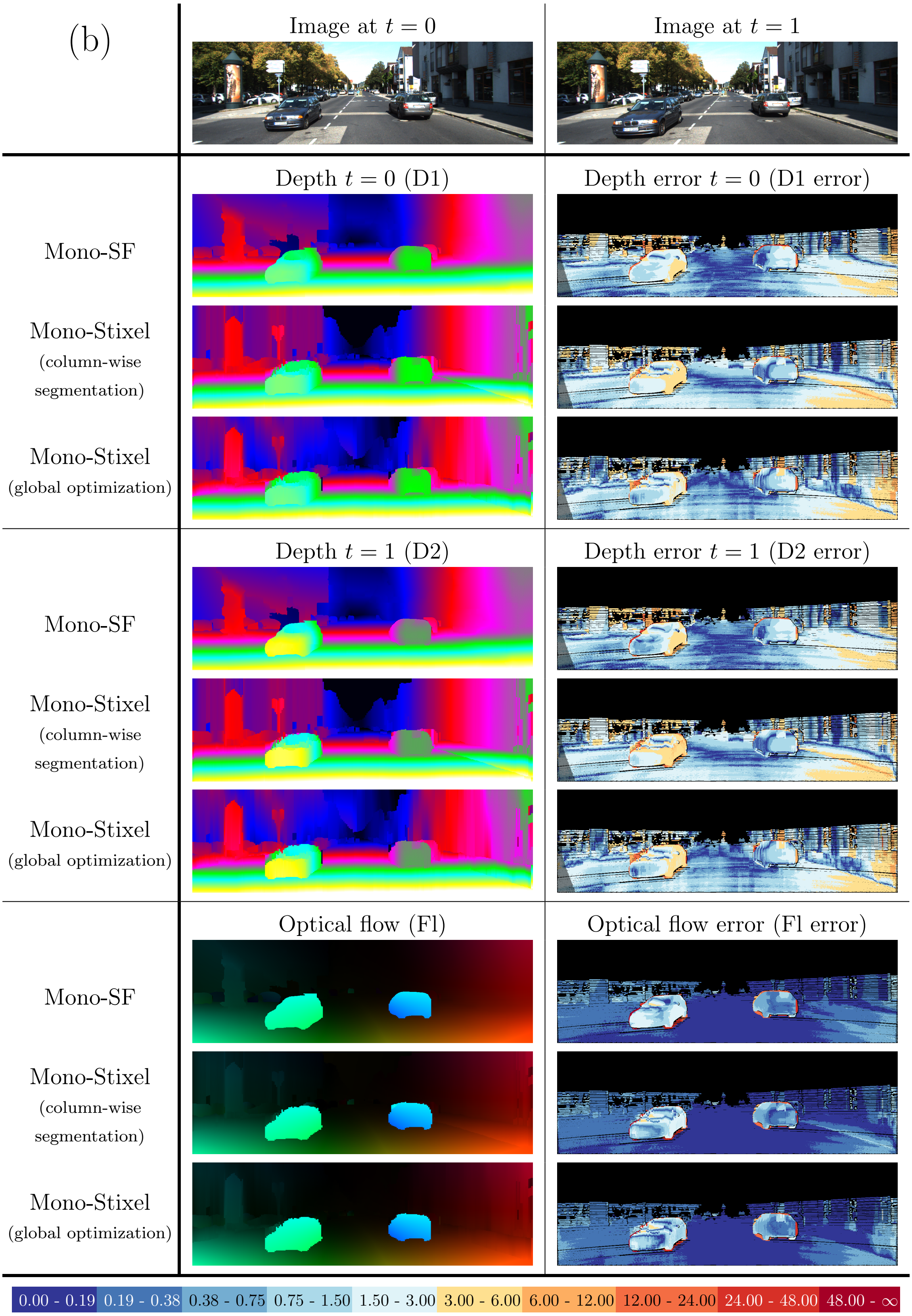
6.3.2 Evaluation of Scene Flow Estimation

The evaluation of the Mono-Stixel method in terms of monocular scene flow estimation follows the same setup as for Mono-SF (section 5.3.1) and SVD-MSfM (section 4.3.1). As a summary, the scene flow results are evaluated based on the equivalent representation as the depth of each pixel at both times ($t = 0$, $t = 1$) and the optical flow using the KITTI scene flow dataset [Menze and Geiger, 2015]. The metric reports the frequencies of errors for the depth at time $t = 0$ (D1) and $t = 1$ (D2) and the optical flow (Fl). A valid scene flow estimate (SF) is defined as fulfilling all the D1, D2, and Fl metrics. The results are stated for the moving objects (fg), the static scene (bg), and both combined (all). The baseline methods represent four categories of monocular scene flow approaches. Multi-task networks (section 4.1.3) that provide depth and optical flow or depth and motion estimates are considered as the first group. The second group comprises the combination of single-view depth and optical flow estimation as individual tasks. While the third category exploits multi-view geometry such as multi-body structure from motion (MSfM)-based methods (section 4.1.2), the fourth category covers methods that combine single-view depth information with multi-view geometry. The fourth category is especially represented by the proposed methods, SVD-MSfM, Mono-SF, and the Mono-Stixel method.

6.3.2.1 *Qualitative Results of Scene Flow Estimation*

Figure 6.6 shows qualitative results of monocular scene flow estimation for Mono-SF, column-wise Mono-Stixel segmentation (section 6.2.2), and global Mono-Stixel scene flow optimization (section 6.2.5). The Mono-Stixel method provides reasonable results for the static environment and for moving objects, which cover various motions such as oncoming (see figure 6.6 (a-c)), preceding (see figure 6.6 (a,b)), or crossing (see figure 6.6 (c)). The global Mono-Stixel optimization serves as a fine-tuning, which reduces the discretization effects for representing a slanted road in a stepwise manner (see figure 6.6 (c)), but also corrects some erroneous estimates (see right advertising sign in figure 6.6 (c)). Comparing the Mono-SF and Mono-Stixel methods shows a similar accuracy in general – but the stixels provides sharper estimates and better reconstructions for thin objects. While the pole in figure 6.6 (a) is smoothed out using Mono-SF, it is reconstructed reasonably using the Mono-Stixel method. Even in standstill scenarios (see figure 6.6 (c)), the stixels preserve more details.


(a)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
Mono-SF
Mono-Stixel (column-wise segmentation)
Mono-Stixel (global optimization)
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
Mono-SF
Mono-Stixel (column-wise segmentation)
Mono-Stixel (global optimization)
Optical flow (Fl)
Optical flow error (Fl error)
Mono-SF
Mono-Stixel (column-wise segmentation)
Mono-Stixel (global optimization)
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞


(b)
Image at $t = 0$
Image at $t = 1$
Depth $t = 0$ (D1)
Depth error $t = 0$ (D1 error)
Mono-SF
Mono-Stixel
(column-wise
segmentation)
Mono-Stixel
(global optimization)
Depth $t = 1$ (D2)
Depth error $t = 1$ (D2 error)
Mono-SF
Mono-Stixel
(column-wise
segmentation)
Mono-Stixel
(global optimization)
Optical flow (Fl)
Optical flow error (Fl error)
Mono-SF
Mono-Stixel
(column-wise
segmentation)
Mono-Stixel
(global optimization)
0.00 - 0.19
0.19 - 0.38
0.38 - 0.75
0.75 - 1.50
1.50 - 3.00
3.00 - 6.00
6.00 - 12.00
12.00 - 24.00
24.00 - 48.00
48.00 - ∞

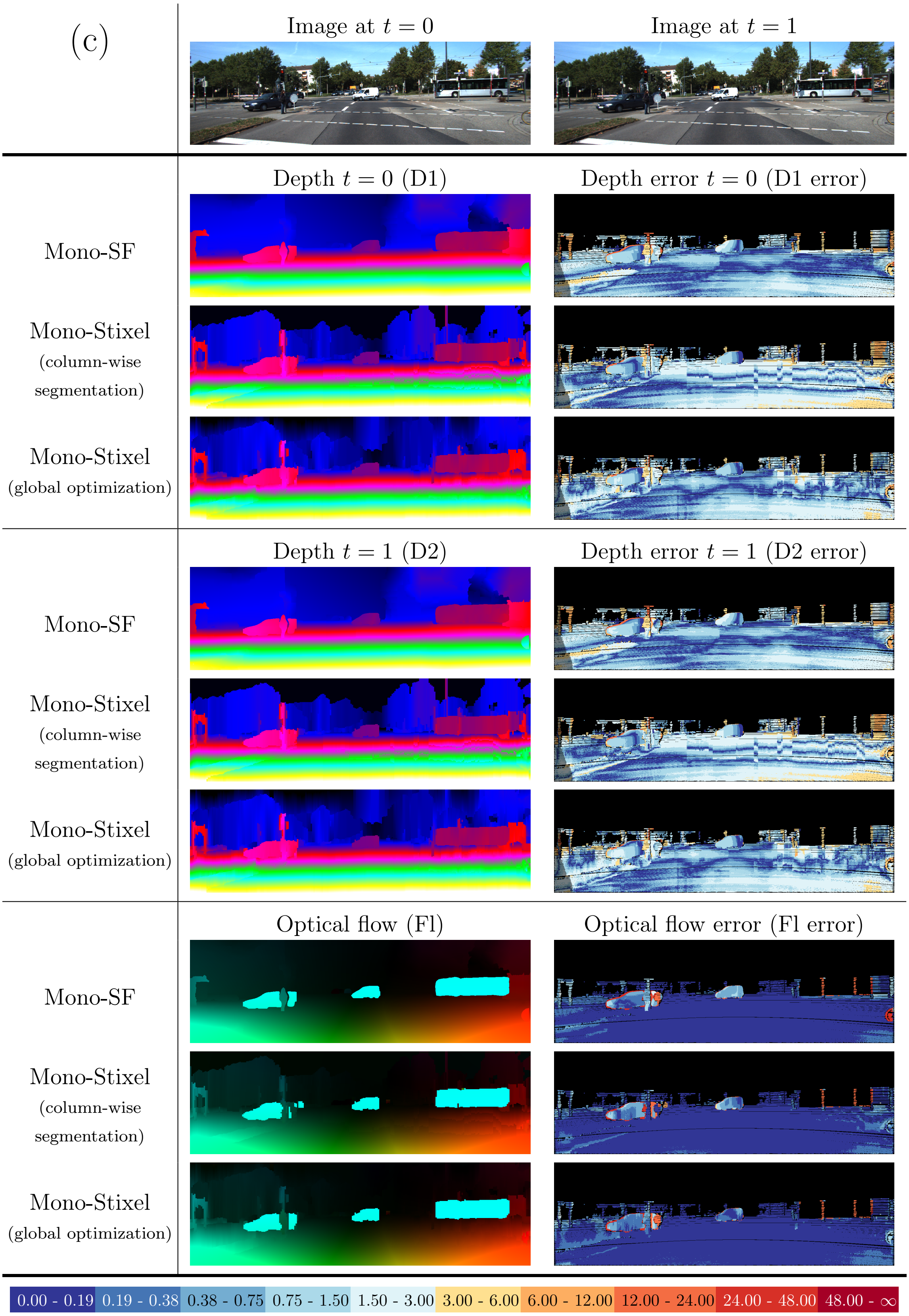


Figure 6.6: Qualitative results of the Mono-Stixel method in comparison to Mono-SF on the KITTI scene flow training set [Menze and Geiger, 2015]. The third rows show the Mono-Stixel results including the global optimization. The color coding represent the estimated depth (from close (warm) to far (cool)), the optical flow (Middlebury color coding [Baker et al., 2011]) or the disparity/ optical flow endpoint error (color coding shown in the legend).

Method	MRE	D1			D2			Fl			SF		
		bg	fg	all	bg	fg	all	bg	fg	all	bg	fg	all
GeoNet$*$ [Yin and Shi, 2018]	20.07	47.03	63.41	49.54	56.24	68.88	58.17	32.42	67.69	37.82	67.69	91.40	71.32
DF-Net$*$ [Zou et al., 2018]	18.95	44.42	57.94	46.49	61.55	61.47	61.53	25.66	37.44	27.46	71.62	82.51	73.29
Struct2Depth$*$ [Casser et al., 2019]	14.92	27.29	56.58	31.77	33.25	66.12	38.29	37.86	71.96	43.08	49.98	91.39	56.32
EPC++ [Luo et al., 2019]	-	22.76	26.63	23.84	-	-	-	17.58	26.89	19.64	-	-	-
Self-Mono-SF [Hur and Roth, 2020]	9.98	28.75	45.07	31.25	33.00	45.15	34.86	23.06	25.92	23.49	44.27	62.40	47.05
MirrorFlow [Hur and Roth, 2017] + LRC [Godard et al., 2017]	9.68	25.33	**19.82**	24.48	35.82	26.15	34.34	9.39	14.22	10.13	40.55	35.17	39.72
HD^3-F† [Yin et al., 2019] + DORN† [Fu et al., 2018]	11.18	17.02	37.54	20.16	30.08	40.47	31.67	4.01	6.76	4.43	32.57	46.89	34.76
DMDE [Ranftl et al., 2016]	14.6	-	-	-	-	-	-	-	-	-	-	-	-
S. Soup [Kumar et al., 2017]	12.68	-	-	-	-	-	-	-	-	-	-	-	-
S.Rel. [Di et al., 2019]	10.23	-	-	-	-	-	-	-	-	-	-	-	-
MFA [Kumar et al., 2019]	11.82	-	-	-	-	-	-	-	-	-	-	-	-
SVD-MSfM [chapter 4]	8.55	17.84	23.94	18.77	20.37	26.72	21.35	15.31	15.55	15.34	24.50	35.01	26.11
Mono-SF [chapter 5]	**8.14**	15.64	22.72	16.72	17.93	**24.71**	18.97	12.20	**9.90**	11.85	20.19	**29.40**	21.60
Mono-Stixel (column-wise DP)	8.34	17.13	28.22	18.83	20.45	32.50	22.30	9.44	10.50	9.60	22.60	36.64	24.75
Mono-Stixel (global optimization)	8.59	**14.22**	24.18	**15.74**	**17.45**	27.10	**18.93**	**9.23**	10.94	**9.49**	**19.25**	30.84	**21.03**

MRE*: mean relative depth error in %;* ***D1, D2****: disparity ($t = 0, 1$);* ***Fl****: optical flow;* ***SF****: scene flow*
D1, D2, Fl, SF*: percentage[%] of estimates that exceed an error threshold (> 3px and > 5% of length)*
fg*: foreground (moving objects) ;* ***bg****: background (static environment);* ***all****: bg + fg*
$\dagger$: parts of dataset used for training (disregarded for ranking); $*$: scaled to align the ground truth

Table 6.2: Quantitative evaluation of the Mono-Stixel method with respect to several monocular methods on the KITTI scene flow training set [Menze and Geiger, 2015]. The methods are divided into four groups: First, multi-task CNNs; second, combining optical flow and single-view depth estimation as individual tasks; third, MSfM-based approaches; fourth, fusing single-view depth information with multi-view geometry. The groups are seperated by two horizontal lines.

6.3.2.2 *Quantitative Evaluation of Scene Flow Estimation*

The quantitative evaluation of the Mono-Stixel method in comparison to monocular baseline methods including SVD-MSfM and Mono-SF is shown in table 6.2. In addition to the targeted compact representation form, the results show that the Mono-Stixel method provides SotA monocular scene flow estimates. Also in terms of optical flow estimation, the Mono-Stixels method provides an improvement over the MirrorFlow [Hur and Roth, 2017] method, which is used as input. While the accuracy of the Mono-Stixel method with global optimization is comparative to Mono-SF, the accuracy of the column-wise stixel segmentation slightly outperforms SVD-MSfM. This validates, on the one hand, that the global Mono-SF improves the accuracy of both, SVD-MSfM and column-wise stixel segmentation. On the other hand, it highlights the trade-off between accuracy and runtime. While similar approaches to SVD-MSfM and the column-wise stixel segmentation have been shown to be real-time capable, the global optimization of Mono-SF increases the computational effort.

6.3.3 Ablation Study on Mono-Stixel Inputs

The Mono-Stixel segmentation is designed to be flexible in terms of the used inputs. All inputs, except the optical flow and camera motion estimation, are optional. Deactivating one input could be considered as setting its weight to zero in equation (6.20). However, the local stixel estimation (section 6.2.3.2) differs slightly. The necessary adaptions for deactivating a certain input and the corresponding impacts are described in the following parts. The scene flow results of the Mono-Stixel variants with different input configurations (denoted by checkmarks) are stated in table 6.3. The experiments are based on the column-wise segmentation without global optimization.

Semantic segmentation: The different semantic classes c_i of each type are derived based on the semantic segmentation input. However, the types are still distinguishable by their geometric constraints and the general stixel segmentation is applicable.

The results in table 6.3 show an improvement in all metrics compared to its counterpart without semantic segmentation (respective counterpart one row above). Even though the semantic segmentation does not directly provide depth or motion information, it supports the segmentation and distinguishing the stixel types, which is important to apply the correct model constraints.

Instance segmentation: The instance segmentation is exploited to define the set of rigid bodies $\mathcal{O}$ to derive their motion estimates (section 4.2.1) and to support the association of stixels to rigid bodies. Without instance segmentation merely the rigid body of the background o_{BG} is given and the stixel motion of dynamic objects is always expressed by the individual 2D translation, which corresponds to the second case of dynamic object motion in table 6.1. However, the stixel motion is still representable and the Mono-Stixel algorithm applicable.

Semantic Seg.	**Instance Seg.**	**Single-View Depth**	**MRE**	**D1-bg**	**D1-fg**	**D1-all**	**D2-bg**	**D2-fg**	**D2-all**	**Fl-bg**	**Fl-fg**	**Fl-all**	**SF-bg**	**SF-fg**	**SF-all**
-	-	-	33.01	26.07	72.96	33.26	29.38	75.97	36.52	9.81	14.47	10.53	31.29	78.63	38.55
✓	-	-	14.84	22.15	50.74	26.53	25.87	57.27	30.68	9.02	14.30	9.83	27.67	62.18	32.96
-	✓	-	36.00	24.90	54.62	29.45	28.39	55.26	32.51	9.40	14.54	10.19	29.92	59.67	34.48
✓	✓	-	18.37	21.86	49.70	26.13	25.53	49.84	29.26	**8.93**	13.77	9.67	27.24	54.49	31.41
-	-	✓	9.07	18.78	30.11	20.52	22.23	39.96	24.95	9.94	18.83	11.31	24.60	48.22	28.22
✓	-	✓	**8.19**	17.29	**27.72**	18.89	20.71	37.64	23.30	9.52	14.29	10.25	22.91	45.94	26.44
-	✓	✓	8.97	17.39	29.55	19.26	20.68	34.11	22.73	9.50	13.38	10.10	22.80	39.30	25.33
✓	✓	✓	8.34	**17.13**	28.22	**18.83**	**20.45**	**32.50**	**22.30**	9.44	**10.50**	**9.60**	**22.60**	**36.64**	**24.75**

MRE: *relative depth;* ***D1***, ***D2***: *disparity* ($t = 0, 1$)*;* ***Fl***: *optical flow;* ***SF***: *scene flow errors*

fg: *foreground (moving objects) ;* ***bg***: *background (static environment);* ***all***: *bg + fg; all errors in %*

Table 6.3: Evaluation of column-wise stixel segmentation of the Mono-Stixel method using different input configurations. The metric rates the scene flow accuracy. The used inputs in terms of single-view depth estimation, semantic segmentation, and instance segmentation are denoted by the checkmarks. Additionally, a dense optical flow and at least a camera motion estimate is given as input. The highest accuracy is achieved by integrating all inputs, which supports that each input provides an improvement and contributes to the final accuracy.

Referring to the results stated in table 6.3, integrating the instance segmentation shows the following effects. The counterpart without instance segmentation is shown in the respective two rows above. For the static environment, the instance segmentation provides a very small improvement, which is likely related to the supported distinction of static and dynamic objects. For moving objects, the depth at $t = 0$ benefits not or only slightly from the instance segmentation. However, integrating the instance segmentation improves the depth at $t = 1$, the optical flow, and scene flow for moving objects significantly. These metrics need to consider the motion and show the ability to predict moving objects – thus benefiting the most from the scale-aware object motion estimates and rigid body prior.

Single-view depth estimation: The motion estimation exploits the single-view depth estimates to derive scale-aware motions. Therefore, it is considered that the scale-ware motion estimates are not given except for the static environment o_{BG}. Actually, the camera motion estimation of o_{BG} also derives its metric scale from the single-view depth. However, for the camera motion, different approaches can provide scale-aware estimates also without single-view depth estimates as described in section 4.1.1.

The depth of the static stixel types is estimated analogously to the proposed approach, whereby the depth hypotheses are defined by the optical flow. Due to the missing scale-aware motion estimates, the motion of dynamic stixels is always represented by the individual 2D translation (second case in table 6.1). Originally, the optical flow defines the hypotheses for the 2 degrees of freedom of the homography $\tilde{\mathbf{t}}_i$ and the single-view depth estimates provide the inverse depth ρ_{hj} hypotheses for dynamic objects. Due to the missing single-view depth estimation, the depth is alternatively derived by minimizing the structural prior Ψ^{str}. For a single stixel, this is achieved by that inverse depth ρ_i that leads the stixel to stand on the reference ground $\rho_{\mathbb{G},ref}$. For stixels that belong to the same instance, the median of the scales defined by minimizing the structural prior is taken to exploit that these stixels undergo the same rigid body motion.

Integrating single-view depth information significantly improves the depth and scene flow estimates for the static environment and moving objects, which is shown in table 6.3 (respective counterparts four rows above). The optical flow does not benefit significantly and is in some cases even slightly worse. Single-view depth estimates improve the depth estimates in general, but are also needed to handle standstill and low-parallax situations. This is also supported by the high mean relative error (MRE) without single-view depth estimation, which is highly affected by the outliers during standstill. Additionally, the improvement for moving objects supports that integrating the single-view depth estimates is superior to the depth estimation based on minimizing the structural prior.

Optical flow: The optical flow is not considered as an optional input. The present paragraph presents the special case of using only the optical flow as input and deactivating all others. Thus, the rigid body and motion estimation are merely given for the static environment o_{BG} and the semantic classes c_i are not distinguishable. The inverse depth hypotheses ρ_{hj} are defined by the optical flow for the static stixel types. The motion and depth estimation for dynamic stixel follows the described variant above without single-view depth estimation. Due to the less restrictive model constraints of a dynamic stixel in terms of the motion parameters, the Mono-Stixel approach would tend to represent all objects as dynamic objects based on the optical flow. Therefore, the stixel segmentation is initially performed over the three stixel types, ground, dynamic object, and sky. The proposed IMO detection is exploited to distinguish static and moving objects. Stixels detected as potentially static are subsequently replaced with the stixel $\mathbf{s}_i^{stat}$ as defined in section 6.2.4.

The results of the Mono-Stixel approach only based on the optical flow are stated by the row without checkmarks in table 6.3. In general, the accuracy is lower than for variants integrating additional inputs. A significant drop in the accuracy is visible for moving objects. This is also because some object motions such as oncoming or preceding objects are not detectable based on the optical flow as described in section 4.1.2, which results in wrong depth estimates.

Overall, the highest accuracy is achieved by integrating all input, which supports that each input provides an improvement and contributes to the final accuracy.

6.3.4 Evaluation of Independent Moving Object Detection

The Mono-Stixel approach provides an addition independent moving object (IMO) score γ_i, which defines the probability to be in motion. The KITTI scene flow dataset [Menze et al., 2018] provides labels for moving parts of the image, which are the basis for distinguishing the foreground and background classes for the scene flow evaluation. By introducing a threshold, a set of pixels with $\gamma_i > \gamma^{mov}$ is defined as moving and pixels with $\gamma_i \leq \gamma^{mov}$ are defined as static. The number of pixels (1) correctly detected as moving are the true positives (TPs), (2) erroneously detected as moving are the false positives (FPs), (3) correctly detected as static are the true negatives (TNs), and (4) erroneously detected as static are the false negatives (FNs). Based on these values the following metrics are defined:

$$\begin{aligned}
\text{Precision (P)} &= \frac{TP}{TP + FP} \\
\text{Recall (R)} &= \frac{TP}{TP + FN} \\
\text{Average Precision (AP)} &= \int_{R=0}^{1} P(R) dR \\
\text{Intersection over Union (IoU)} &= \max_{\gamma^{mov}} \left(\frac{TP}{TP + FP + FN} \right)
\end{aligned} \tag{6.45}$$

By variating the threshold γ^{mov} a receiver operating characteristic (ROC) curve is derived, which shows the relation of the precision (P) to the recall (R). Integrating over this curve defines the average precision (AP) to rate the accuracy of the independent moving object (IMO). Alternatively, the intersection over union (IoU) selects the best threshold in terms of the defined metric.

Figure 6.7 shows the ROC curves for the different Mono-Stixel variants based on the different input configurations described in section 6.3.3. Additionally, the results of MODNet [Siam et al., 2018] are stated as a baseline method. MODNet is a CNN that provides moving object masks using a raw image and an optical flow field as inputs. The Mono-Stixel variant integrating all inputs provides the highest accuracy in terms of the AP and IoU metric and significantly outperforms MODNet. The lowest accuracy is achieved for the Mono-Stixel approach merely based on the optical flow. On the one hand, some object motions such as oncoming

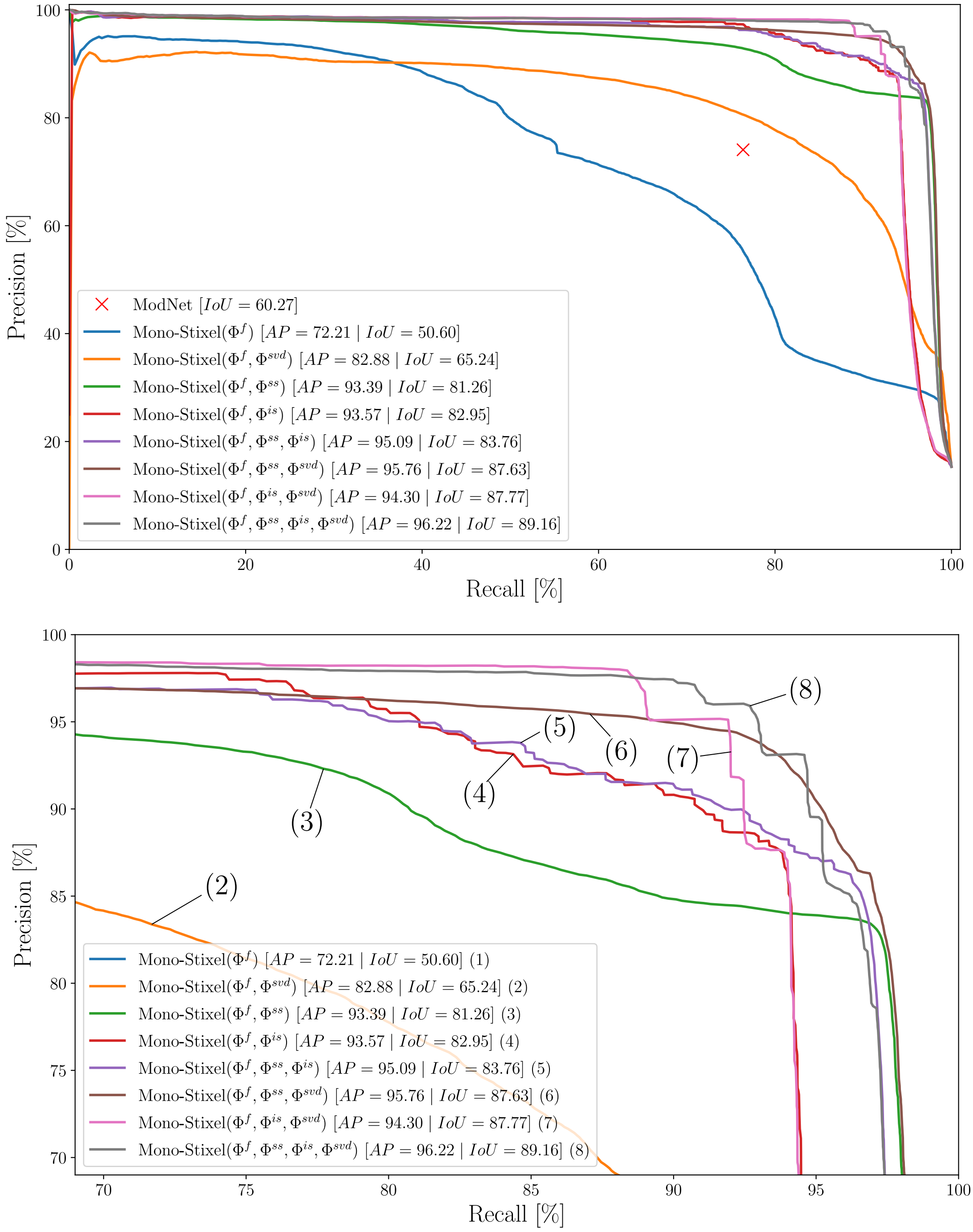


Figure 6.7: Evaluation of Mono-Stixel variants for independent moving object (IMO) detection on KITTI scene flow dataset [Menze et al., 2018]. The ROC curves, AP, and IoU are shown for Mono-Stixel variants with different input configurations and for the baseline method, MODNet [Siam et al., 2018]. The integrated inputs are indicated by the corresponding data term (Φ^f for optical flow, Φ^{ss} for semantic segmentation, Φ^{is} for instance segmentation, and Φ^{svd} for single-view depth). The highest accuracy is achieved for the Mono-Stixel variant integrating all inputs. Especially integrating a classificaton in the form of a semantic or instance segmentation provides a significant improvement. Both figures show the same ROC curves, whereby the bottom diagram is a zoomed-in view of the upper right corner of the top diagram.

or preceding objects are not detectable in this case. On the other hand, the moving object detection is sensitive to erroneous optical flow estimates. Integrating single-view depth estimates shows improvement and allows detecting all kinds of motions. However, the highest improvement is obtained by adding a classification in the form of a semantic or instance segmentation, which helps to restrict the parts that are potentially moving such as vehicles. Either integrating semantic segmentation or an instance segmentation shows a similar improvement in terms of IMO detection.

6.4 Conclusion

The present chapter presented the Mono-Stixel method, an approach for monocular scene flow estimation providing a scene representation in the form of a compact stixel world representation. The evaluation confirms that the Mono-Stixel method provides SotA monocular scene flow estimation and IMO detection. Even more, the results illustrate better characteristics especially for thin objects such as poles. The improvement and usefulness of integrating the individual inputs are validated by the experiments.

The formulation as a 1D energy minimization problem for column-wise segmentation allows an efficient computation. A variant of the Mono-Stixel algorithm based on a sparse optical flow including simplifications and approximations is implemented in real-time on embedded hardware. Even though this Mono-Stixel variant is out of the scope of the present thesis, it highlights the applicability of the Mono-Stixel model and approach for driver assistance systems or autonomous driving.

7 CONCLUSION AND OUTLOOK

The present thesis is focused on monocular scene flow estimation and driven by the core question: *'How to combine the principles of multi-view geometry with deep learning-based perception for scene flow estimation in a monocular camera setup focusing on multi-rigid-object dynamic scenes?'* In the concluding chapter, the essential contributions are summarized and an outlook is given for the presented approaches and monocular scene flow estimation in general.

7.1 Summary of Essential Contributions

Several contributions and details are provided in terms of the addressed core questions, which are summarized into five essential statements.

(1) *Monocular scene flow estimation formulated as an optimization combining multi-view geometric information with probabilistic single-view depth estimates achieve new state of the art (SotA) accuracy. Ablation studies confirm that both kinds of information, multi-view geometry and deep learning-based single-view depth, significantly contribute to the final accuracy.*

The approaches in chapters 4 to 6 address the task of monocular scene flow estimation. The methods combine a multi-view geometry-based part such as a photometric or reprojection error with probabilistic single-view depth estimates. The experiments validate that the multi-view part and the single-view depth information are essential to achieve the new SotA accuracy. The highest accuracy is achieved for a global energy minimization jointly optimizing the motion of rigid bodies and the structure of planar surface elements.

(2) *The integration of single-view depth information into a monocular scene flow estimation highly benefits from a probabilistic and well-calibrated representation as depth distributions.*

The monocular scene flow approaches are formulated as energy minimization problems. The experiments in chapters 4 and 5 show the importance that the single-view depth estimates are provided in a probabilistic and well-calibrated form. This is made possible by the proposed ProbDepthNet presented in chapter 3 including a novel recalibration method.

(3) *Calibrated distributions are achievable for neural regression networks by additional subsequent layers trained on a hold-out split of the training data.*

Most of the regression networks for single-view depth estimation merely provide a single maximum likelihood estimate without an uncertainty quantification. The ProbDepthNet presented in chapter 3 provides pixel-wise depth distributions by estimating the parameters of a distribution. While the estimation of overconfident scores due to overfitting effects is a well-known problem for classification problems, ProbDepthNet presents a recalibration technique for regression problems. A few additional layers, the CalibNet, are trained on a hold-out data split and reshape the parameter of the distribution such that they are well-calibrated.

(4) *Integrating probabilistic single-view depth estimates into a rigid-object motion estimation results in metric scale-aware motion estimates also for moving objects.*

The scale ambiguity is a well-known problem for pose and motion estimation in a monocular camera setup. While previous methods exploit special scene constraints such as a known camera height, these methods typically lack generalizability to moving objects. The motion estimation proposed in chapter 4 is formulated as an energy minimization problem minimizing the poses and a set of 3D scene points. The energy term captures a multi-view photometric distance and rates the 3D scene points by the single-view depth estimates. The results show that the proposed methods provide accurate scale-aware motion estimates – also being applicable to moving objects.

(5) *A specialized representation for dynamic traffic scenes such as the stixel world can be used as underlying scene model for monocular scene flow estimation.*

Applications that are designed to work in a specific domain of scenes such as dynamic traffic scenes can exploit additional prior scene knowledge. The presented Mono-Stixel method (chapter 6) formulates the scene flow estimation using a stixel world representation. This results in better characteristics such as that thin objects are smoothed out less and that the representation can be decoded in a compacter way.

7.2 Outlook

The present thesis provides essential contributions in terms of probabilistic single-view depth estimation and monocular scene flow estimation. The present section describes potential directions of further research for the presented methods and in general.

Probabilistic single-view depth estimation: ProbDepthNet presented in chapter 3 provides depth distribution designed to capture the measurement uncertainty. The quantification of the model uncertainty or out of distribution (OOD) detection is out of the scope of the ProbDepthNet model.

For the model uncertainty, methods such as Monte Carlo Dropout are also applied to regression problems and could be integrated into the ProbDepthNet.

For the OOD detection, the experiments in section 3.4.3 analyzed the generalization capabilities of ProbDepthNet and illustrate the limitations for images that are too different to the training data, e.g. close-up views of buildings. In such a case, neither the depth estimates nor the uncertainties are guaranteed to be reasonable. Such samples are considered as OOD data, which require special treatment to be detected. The current SotA methods for OOD are focused on classification problems. Methods such as analyzing the neural activation pattern are not directly applicable for regression problems. Thus, one direction of research would be OOD detection as part of the ProbDepthNet and OOD detection for regression problems in general.

Furthermore, the recalibration technique requires ground truth data and the hold-out split reduces the number of training samples for the main network. Both can be disadvantageous especially if the ground truth data needs a high effort to be collected and only a few training data samples exist. The uncertainty quantification for unsupervised training and recalibration without needing a hold-out split of training data are open challenges and a further direction of research.

Monocular scene flow estimation: The chapters 4 to 6 addresses the task of monocular scene flow estimation problems. The experiments show the importance of integrating the single-view depth estimates in a probabilistic and well-calibrated form. This probably holds for all inputs and it could be therefore advantageous to integrate also the motion estimates with an uncertainty measure and to apply recalibration techniques to the semantic and instance segmentation.

This thesis is tailored to monocular scene flow estimation in a two-view setup. The qualitative results of all methods show that the accuracy decreases for parts that are occluded in one image or leave the field of view and for low-parallax situations such as standstill scenarios. Typically, most parts of the scene have already been observed for a longer time before they are occluded or leave the field of view. Furthermore, even if there is a standstill situation, the camera previously was in motion. This strongly motivates to extend the estimation to a longer temporal context to increase the accuracy and overcome the described limitations. For example, the estimation could be directly formulated on more than two views such as a bundle adjustment. This would probably increase the computational effort significantly and a prediction and tracking scheme might be practically more useful.

Furthermore, this thesis is focused on multi-rigid body scenes, especially on dynamic traffic scenes. Many of the findings and general concepts probably also apply to other scenes, which could be analyzed in subsequent works. The single-view depth estimation benefits from strong model constraints such as a road surface or that vehicle stand on the ground plane. Thus, it is not ensured that the single-view depth information is such powerful information for all kinds of scenes. The assumption that the scene is decomposable into a set of rigid bodies is also a traffic scene-specific constraint. For other scenes, such as close-up views of persons, the approaches need to be extended to handle non-rigid motions.

The presented methods for monocular scene flow estimation are not real-time capable. For the Mono-Stixel algorithm, a real-time capable version for embedded hardware is implemented, which corresponds to the column-wise segmentation based on an optical flow field with some further simplifications and approximations of the segmentation algorithm. This Mono-Stixel variant is out of the scope of the present thesis but still highlights that the approaches can reach real-time at least with some simplifications. However, reaching real-time also including the deep learning parts remains a challenge, especially on embedded systems.

Finally, I would like to discuss a more general direction, in which the methods for monocular scene flow could lead. While the proposed methods are based on a traditional energy minimization problem, end-to-end neural networks showed to outperform traditional methods for many tasks. The multi-task networks that provide depth and motion estimates could be considered as end-to-end networks. However, so far these networks are not able to fully exploit the multi-view information for depth estimation during inference. These methods are still significantly outperformed by the proposed methods in the present thesis. While the convolutional kernel of a convolutional neural network (CNN) is invariant to the image position, multi-view geometry principles depends on the image position. This gives a plausible explanation, why CNNs in its standard form cannot represent tasks such as triangulation.

However, if suitable special layers and architectures can be found that integrate all relevant information such as the proposed energy minimization problems, end-to-end neural networks could be the next promising improvement also for monocular scene flow estimation.

Overall, the present thesis is the first work explicitly focussed on monocular scene flow estimation, especially in terms of combining geometric principles with deep learning approaches. Essential contributions are presented and new SotA methods are provided. There are still interesting directions of research and I am very curious to see future works in the field of monocular scene flow estimation.

BIBLIOGRAPHY

Abdulla, W. (2017). Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. https://github.com/matterport/Mask_RCNN.

Aleotti, F., Tosi, F., Poggi, M., and Mattoccia, S. (2018). Generative adversarial networks for unsupervised monocular depth prediction. In *Proc. of European Conference on Computer Vision Workshops (ECCVW)*.

Almalioglu, Y., Saputra, M. R. U., d. Gusmão, P. P. B., Markham, A., and Trigoni, N. (2019). GANVO: Unsupervised Deep Monocular Visual Odometry and Depth Estimation with Generative Adversarial Networks. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5474–5480.

Azizpour, H., Teye, M., and Smith, K. (2018). Bayesian Uncertainty Estimation for Batch Normalized Deep Networks. In *Proc. of International Conference on Machine Learning (ICML)*.

Badino, H., Franke, U., and Pfeiffer, D. (2009). The Stixel World - A Compact Medium Level Representation of the 3D-World. In *Joint Pattern Recognition Symposium*, pages 51–60. Springer.

Bai, M., Luo, W., Kundu, K., and Urtasun, R. (2016). Exploiting Semantic Information and Deep Matching for Optical Flow. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 154–170. Springer.

Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M. J., and Szeliski, R. (2011). A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31.

Barnes, D., Maddern, W., Pascoe, G., and Posner, I. (2018). Driven to Distraction: Self-Supervised Distractor Learning for Robust Monocular Visual Odometry in Urban Environments. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1894–1900. IEEE.

Basha, T., Moses, Y., and Kiryati, N. (2013). Multi-view Scene Flow Estimation: A View Centered Variational Approach. *International Journal of Computer Vision*, 101(1):6–21.

Behl, A., Jafari, O. H., Mustikovela, S. K., Alhaija, H. A., Rother, C., and Geiger, A. (2017). Bounding Boxes, Segmentations and Object Coordinates: How Important is Recognition for 3D Scene Flow Estimation in Autonomous Driving Scenarios? In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2574–2583.

Bendale, A. and Boult, T. E. (2016). Towards Open Set Deep Networks. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1563–1572.

Benenson, R., Mathias, M., Timofte, R., and Van Gool, L. (2012). Fast Stixel Computation for Fast Pedestrian Detection. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 11–20. Springer.

Benenson, R., Timofte, R., and Van Gool, L. (2011). Stixels estimation without depth map computation. In *Proc. of IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 2010–2017. IEEE.

Bergmann, P., Wang, R., and Cremers, D. (2018). Online Photometric Calibration of Auto Exposure Video for Realtime Visual Odometry and SLAM. *IEEE Robotics and Automation Letters (RA-L)*, 3:627–634.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* Springer.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight Uncertainty in Neural Network. In Bach, F. and Blei, D., editors, *Proc. of International Conference on Machine Learning (ICML)*, volume 37 of *Proceedings of Machine Learning Research*, pages 1613–1622, Lille, France. PMLR.

Brickwedde, F., Abraham, S., and Mester, R. (2018a). Exploiting Single Image Depth Prediction for Mono-Stixel Estimation. In *Proc. of European Conference on Computer Vision Workshops (ECCVW)*. IEEE.

Brickwedde, F., Abraham, S., and Mester, R. (2018b). Mono-Stixels: Monocular Depth Reconstruction of Dynamic Street Scenes. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE.

Brickwedde, F., Abraham, S., and Mester, R. (2019). Mono-SF: Multi-View Geometry Meets Single-View Depth for Monocular Scene Flow Estimation of Dynamic Traffic Scenes. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

Bullinger, S., Bodensteiner, C., Arens, M., and Stiefelhagen, R. (2017). 3D Trajectory Reconstruction of Dynamic Objects Using Planarity Constraints. *arXiv preprint arXiv:1711.06136.*

Bullinger, S., Bodensteiner, C., Arens, M., and Stiefelhagen, R. (2018). 3D Vehicle Trajectory Reconstruction in Monocular Video Data Using Environment Structure Constraints. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 35–50.

Casser, V., Pirk, S., Mahjourian, R., and Angelova, A. (2019). Depth Prediction without the Sensors: Leveraging Structure for Unsupervised Learning from Monocular Videos. In *Proc of AAAI Conference on Artificial Intelligence (AAAI).*

Chen, Y., Schmid, C., and Sminchisescu, C. (2019). Self-Supervised Learning With Geometric Constraints in Monocular Video: Connecting Flow, Depth, and Camera. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 7063–7072.

Chhaya, F., Reddy, D., Upadhyay, S., Chari, V., Zia, M. Z., and Krishna, K. M. (2016). Monocular reconstruction of vehicles: Combining SLAM with shape priors. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5758–5765.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223.

Cordts, M., Rehfeld, T., Schneider, L., Pfeiffer, D., Enzweiler, M., Roth, S., Pollefeys, M., and Franke, U. (2017). The Stixel World: A medium-level representation of traffic scenes. *Image and Vision Computing.*

Cordts, M., Schneider, L., Enzweiler, M., Franke, U., and Roth, S. (2014). Object-Level Priors for Stixel Generation. In *Proc. of German Conference on Pattern Recognition (GCPR)*, pages 172–183. Springer.

Coughlan, J. M. and Yuille, A. L. (1999). Manhattan world: compass direction from a single image by bayesian inference. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 941–947. IEEE.

Cutting, J. E. and Vishton, P. M. (1995). Perceiving Layout and Knowing Distances: The Integration, Relative Potency, and Contextual Use of Different Information about Depth. In *Perception of Space and Motion*, pages 69–117. Elsevier.

Delaunay, B. et al. (1934). Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(793-800):1–2.

Dellaert, F., Kaess, M., et al. (2017). Factor Graphs for Robot Perception. *Foundations and Trends® in Robotics*, 6(1-2):1–139.

DeVries, T. and Taylor, G. W. (2018). Learning Confidence for Out-of-Distribution Detection in Neural Networks. *arXiv preprint arXiv:1802.04865*.

Di, Y., Morimitsu, H., Gao, S., and Ji, X. (2019). Monocular Piecewise Depth Estimation in Dynamic Scenes by Exploiting Superpixel Relations. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 4363–4372.

Dijk, T. v. and Croon, G. d. (2019). How Do Neural Networks See Depth in Single Images? In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2183–2191.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, pages 2366–2374.

Engel, J., Koltun, V., and Cremers, D. (2017). Direct Sparse Odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Engel, J., Schöps, T., and Cremers, D. (2014). LSD-SLAM: Large-Scale Direct Monocular SLAM. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 834–849. Springer.

Engel, J., Usenko, V., and Cremers, D. (2016). A Photometrically Calibrated Benchmark For Monocular Visual Odometry. *arXiv preprint arXiv:1607.02555*.

Engel, J.-J. (2017). *Large-Scale Direct SLAM and 3D Reconstruction in Real-Time*. PhD thesis, Technische Universität München.

Enzweiler, M., Hummel, M., Pfeiffer, D., and Franke, U. (2012). Efficient Stixel-based object recognition. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 1066–1071. IEEE.

Erbs, F., Barth, A., and Franke, U. (2011). Moving vehicle detection by optimal segmentation of the Dynamic Stixel World. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 951–956. IEEE.

Erbs, F., Schwarz, B., and Franke, U. (2012). Stixmentation - Probabilistic Stixel based Traffic Scene Labeling. In *Proc. of British Machine Vision Conference (BMVC)*, pages 71.1–71.12. BMVA Press.

Erbs, F., Schwarz, B., and Franke, U. (2013). From Stixels to objects - A conditional random field based approach. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 586–591. IEEE.

Erbs, F., Witte, A., Scharwaechter, T., Mester, R., and Franke, U. (2014). Spider-based Stixel object segmentation. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 906–911. IEEE.

Fácil, J. M., Concha, A., Montesano, L., and Civera, J. (2017). Single-View and Multi-View Depth Fusion. *IEEE Robotics and Automation Letters (RA-L)*, 2(4):1994–2001.

Fanani, N., Ochs, M., and Mester, R. (2018). Detecting parallel-moving objects in the monocular case employing CNN depth maps. In *Proc. of European Conference on Computer Vision Workshops (ECCVW).*

Fanani, N., Ochs, M., Sturck, A., and Mester, R. (2018). CNN-based multi-frame IMO detection from a monocular camera. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 957–964.

Fanani, N., Stürck, A., Ochs, M., Bradler, H., and Mester, R. (2017). Predictive monocular odometry (PMO): What is possible without RANSAC and multiframe bundle adjustment? *Image and Vision Computing.*

Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). SVO: Fast semi-direct monocular visual odometry. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22. IEEE.

Franke, U., Pfeiffer, D., Rabe, C., Knoeppel, C., Enzweiler, M., Stein, F., and Hertwich, R. (2013). Making Bertha See. In *Proc. of IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 214–221.

Franke, U., Rabe, C., Badino, H., and Gehrig, S. (2005). 6D-Vision: Fusion of Stereo and Motion for Robust Environment Perception. In *Joint Pattern Recognition Symposium*, pages 216–223. Springer.

Frost, D. P., Kähler, O., and Murray, D. W. (2016). Object-aware bundle adjustment for correcting monocular scale drift. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 4770–4776. IEEE.

Fu, H., Gong, M., Wang, C., Batmanghelich, K., and Tao, D. (2018). Deep Ordinal Regression Network for Monocular Depth Estimation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2002–2011.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proc. of International Conference on Machine Learning (ICML)*, pages 1050–1059.

Gan, Y., Xu, X., Sun, W., and Lin, L. (2018). Monocular Depth Estimation with Affinity, Vertical Pooling, and Label Enhancement. In *Proc. of European Conference on Computer Vision (ECCV)*.

Gao, X., Wang, R., Demmel, N., and Cremers, D. (2018). LDSO: Direct Sparse Odometry with Loop Closure. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2198–2204. IEEE.

Garg, R., Carneiro, G., and Reid, I. (2016). Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 740–756. Springer.

Garnett, N., Silberstein, S., Oron, S., Fetaya, E., Verner, U., Ayash, A., Goldner, V., Cohen, R., Horn, K., and Levi, D. (2017). Real-Time Category-Based and General Obstacle Detection for Autonomous Driving. In *Proc. of IEEE International Conference on Computer Vision Workshop (ICCVW)*.

Gast, J. and Roth, S. (2018). Lightweight Probabilistic Deep Networks. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3369–3378.

Gehrig, S. K., Eberli, F., and Meyer, T. (2009). A Real-Time Low-Power Stereo Vision Engine Using Semi-Global Matching. In *International Conference on Computer Vision Systems*, pages 134–143. Springer.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)*.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Geiger, A., Ziegler, J., and Stiller, C. (2011). StereoScan: Dense 3d reconstruction in real-time. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968.

Godard, C., Mac Aodha, O., and Brostow, G. (2019). Digging Into Self-Supervised Monocular Depth Estimation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised Monocular Depth Estimation With Left-Right Consistency. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Goodale, M. A. and Milner, A. D. (1992). Separate Visual Pathways for Perception and Action. *Trends in Neurosciences*, 15(1):20–25.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014a). Generative Adversarial Nets. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, pages 2672–2680.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014b). Explaining and Harnessing Adversarial Examples. *arXiv preprint arXiv:1412.6572.*

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. In *Proc. of International Conference on Machine Learning (ICML)*, pages 1321–1330.

Guo, X., Li, H., Yi, S., Ren, J., and Wang, X. (2018). Learning Monocular Depth by Distilling Cross-domain Stereo Networks. In *Proc. of European Conference on Computer Vision (ECCV).*

Gupta, A., Hebert, M., Kanade, T., and Blei, D. M. (2010). Estimating Spatial Layout of Rooms using Volumetric Reasoning about Objects and Surfaces. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, pages 1288–1296.

Hartley, R. and Zisserman, A. (2003). *Multiple View Geometry in Computer Vision.* Cambridge university press.

Hartley, R. I. (1997). Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision*, 22(2):125–140.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask R-CNN. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988. IEEE.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hedau, V., Hoiem, D., and Forsyth, D. (2010). Thinking Inside the Box: Using Appearance Models and Context Based on Room Geometry. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 224–237. Springer.

Hehn, T. M., Kooij, J. F. P., and Gavrila, D. M. (2019). Instance Stixels: Segmenting and Grouping Stixels into Objects. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 2542–2549.

Hendrycks, D. and Gimpel, K. (2017). A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks.

Herbst, E., Ren, X., and Fox, D. (2013). RGB-D flow: Dense 3-D motion estimation using color and depth . In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 2276–2282.

Hernandez-Juarez, D., Schneider, L., Espinosa, A., Vázquez, D., López, A. M., Franke, U., Pollefeys, M., and Moure, J. C. (2017). Slanted Stixels: Representing San Francisco's Steepest Streets. In *Proc. of British Machine Vision Conference (BMVC)*.

Hirschmuller, H. (2005). Accurate and efficient stereo processing by semi-global matching and mutual information. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 807–814.

Hoiem, D., Efros, A. A., and Hebert, M. (2005). Automatic Photo Pop-up. In *Proc. of ACM transactions on graphics (TOG)*, volume 24, pages 577–584. ACM.

Horn, B. K. and Brooks, M. J. (1986). The variational approach to shape from shading. *Computer Vision, Graphics, and Image Processing*, 33(2):174 – 208.

Hornacek, M., Fitzgibbon, A., and Rother, C. (2014). SphereFlow: 6 DoF Scene Flow from RGB-D Pairs. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3526–3533.

Huang, P.-Y., Hsu, W.-T., Chiu, C.-Y., Wu, T.-F., and Sun, M. (2018). Efficient Uncertainty Estimation for Semantic Segmentation in Videos. In *Proc. of European Conference on Computer Vision (ECCV)*.

Huber, P. J. (1992). Robust estimation of a location parameter. In *Breakthroughs in Statistics*, pages 492–518. Springer.

Huguet, F. and Devernay, F. (2007). A Variational Method for Scene Flow Estimation from Stereo Sequences. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1–7. IEEE.

Hur, J. and Roth, S. (2017). MirrorFlow: Exploiting Symmetries in Joint Optical Flow and Occlusion Estimation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

Hur, J. and Roth, S. (2020). Self-Supervised Monocular Scene Flow Estimation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Ilg, E., Cicek, O., Galesso, S., Klein, A., Makansi, O., Hutter, F., and Brox, T. (2018). Uncertainty Estimates and Multi-Hypotheses Networks for Optical Flow. In *Proc. of European Conference on Computer Vision (ECCV).*

Jähne, B. (2005). *Digital Image Processing.* Engineering online library. Springer Berlin Heidelberg.

Jiang, H., Learned-Miller, E., Larsson, G., Maire, M., and Shakhnarovich, G. (2018). Self-Supervised Relative Depth Learning for Urban Scene Understanding. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 19–35.

Karsch, K., Liu, C., and Kang, S. B. (2012). Depth Extraction from Video Using Non-parametric Sampling. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 775–788. Springer.

Karsch, K., Liu, C., and Kang, S. B. (2014). Depth Transfer: Depth Extraction from Video Using Non-Parametric Sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2144–2158.

Kendall, A. and Gal, Y. (2017). What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, pages 5574–5584.

Kingma, D. P. and Ba, J. L. (2014). Adam: A Method for Stochastic Optimization. In *Proc. of International Conference for Learning Representations (ICLR).*

Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes.

Kirillov, A., He, K., Girshick, R., Rother, C., and Dollár, P. (2019). Panoptic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9404–9413.

Klappstein, J. (2008). *Optical-Flow Based Detection of Moving Objects in Traffic Scenes.* PhD thesis.

Klein, G. and Murray, D. (2007). Parallel Tracking and Mapping for Small AR Workspaces. In *Proc. of IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 1–10. IEEE Computer Society.

Klodt, M. and Vedaldi, A. (2018). Supervising the new with the old: learning SFM from SFM. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 698–713.

Kolmogorov, V. (2006). Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568–1583.

Konrad, J., Wang, M., Ishwar, P., Wu, C., and Mukherjee, D. (2013). Learning-Based, Automatic 2D-to-3D Image and Video Conversion. *IEEE Transactions on Image Processing*, 22(9):3485–3496.

Krueger, D., Huang, C.-W., Islam, R., Turner, R., Lacoste, A., and Courville, A. (2018). Bayesian hypernetworks.

Kuleshov, V., Fenner, N., and Ermon, S. (2018). Accurate Uncertainties for Deep Learning Using Calibrated Regression. In *Proc. of International Conference on Machine Learning (ICML)*, pages 2801–2809.

Kumar, A. C., Bhandarkar, S. M., and Prasad, M. (2018). Monocular Depth Prediction Using Generative Adversarial Networks. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 413–4138. IEEE.

Kumar, S., Dai, Y., and Li, H. (2017). Monocular Dense 3D Reconstruction of a Complex Dynamic Scene From Two Perspective Frames. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 4649–4657.

Kumar, S., Ghorakavi, R. S., Dai, Y., and Li, H. (2019). A Motion Free Approach to Dense Depth Estimation in Complex Dynamic Scene. *arXiv preprint arXiv:1902.03791*.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g^{2}o: A General Framework for Graph Optimization. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 3607–3613. IEEE.

Kundu, A., Krishna, K. M., and Jawahar, C. (2010). Realtime Motion Segmentation based Multibody Visual SLAM. In *Proc. of Indian Conference on Computer Vision, Graphics and Image Processing*, pages 251–258. ACM.

Kundu, A., Krishna, K. M., and Jawahar, C. (2011). Realtime multibody visual SLAM with a smoothly moving monocular camera. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2080–2087. IEEE.

Kundu, A., Li, Y., Dellaert, F., Li, F., and Rehg, J. M. (2014). Joint semantic segmentation and 3D reconstruction from monocular video. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 703–718. Springer.

Kuznietsov, Y., Stückler, J., and Leibe, B. (2017). Semi-Supervised Deep Learning for Monocular Depth Map Prediction. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6647–6655.

Ladicky, L., Shi, J., and Pollefeys, M. (2014). Pulling Things out of Perspective. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 89–96.

Laidlow, T., Czarnowski, J., and Leutenegger, S. (2019). DeepFusion: Real-Time Dense 3D Reconstruction for Monocular SLAM using Single-View Depth and Gradient Predictions. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 4068–4074.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, pages 6402–6413.

Lee, J. H., Han, M.-K., Ko, D. W., and Suh, I. H. (2019). From Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation. *arXiv preprint arXiv:1907.10326*.

Lee, K., Lee, H., Lee, K., and Shin, J. (2018a). Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples. In *Proc. of International Conference for Learning Representations (ICLR)*.

Lee, K., Lee, K., Lee, H., and Shin, J. (2018b). A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, pages 7167–7177.

Levi, D., Garnett, N., Fetaya, E., and Herzlyia, I. (2015). StixelNet: A Deep Convolutional Network for Obstacle Detection and Road Segmentation. In *Proc. of British Machine Vision Conference (BMVC)*, pages 109–1.

Liang, S., Li, Y., and Srikant, R. (2018). Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks.

Lis, K., Nakka, K., Fua, P., and Salzmann, M. (2019). Detecting the Unexpected via Image Resynthesis. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2152–2161.

Liu, C., Gu, J., Kim, K., Narasimhan, S. G., and Kautz, J. (2019). Neural RGB->D Sensing: Depth and Uncertainty From a Video Camera. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, F., Shen, C., and Lin, G. (2015). Deep Convolutional Neural Fields for Depth Estimation From a Single Image. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5162–5170.

Liu, F., Shen, C., Lin, G., and Reid, I. (2016). Learning Depth from Single Monocular Images Using Deep Convolutional Neural Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2024–2039.

Liu, M., Salzmann, M., and He, X. (2014). Discrete-Continuous Depth Estimation from a Single Image. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 716–723. IEEE.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440.

Luo, C., Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R., and Yuille, A. (2019). Every Pixel Counts ++: Joint Learning of Geometry and Motion with 3D Holistic Understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Luo, Y., Ren, J., Lin, M., Pang, J., Sun, W., Li, H., and Lin, L. (2018). Single View Stereo Matching. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 155–163.

Ma, H., Ding, Y., Wang, L., Zhang, M., and Li, D. (2018). Depth Estimation from Monocular Images Using Dilated Convolution and Uncertainty Learning. In Hong, R., Cheng, W.-H., Yamasaki, T., Wang, M., and Ngo, C.-W., editors, *Proc. of Advances in Multimedia Information Processing (PCM)*, pages 13–23, Cham. Springer International Publishing.

Ma, W.-C., Wang, S., Hu, R., Xiong, Y., and Urtasun, R. (2019). Deep Rigid Instance Scene Flow. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Mahjourian, R., Wicke, M., and Angelova, A. (2017). Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5667–5675.

Malik, J. and Rosenholtz, R. (1997). Computing Local Surface Orientation and Shape from Texture for Curved Surfaces. *International Journal of Computer Vision*, 23(2):149–168.

Malinin, A. and Gales, M. (2018). Predictive Uncertainty Estimation via Prior Networks. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, pages 7047–7058.

Malinin, A., Ragni, A., Knill, K., and Gales, M. (2017). Incorporating Uncertainty into Deep Learning for Spoken Language Assessment. In *Proc. of the Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 45–50.

Masana, M., Ruiz, I., Serrat, J., van de Weijer, J., and López, A. M. (2018). Metric Learning for Novelty and Anomaly Detection. In *Proc. of British Machine Vision Conference (BMVC)*, page 64.

Menze, M. and Geiger, A. (2015). Object Scene Flow for Autonomous Vehicles. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Menze, M., Heipke, C., and Geiger, A. (2018). Object Scene Flow. *ISPRS Journal of Photogrammetry and Remote Sensing*, 140:60 – 76. Geospatial Computer Vision.

Milner, A. D. (2017). How do the two visual streams interact with each other? *Experimental Brain Research*, 235(5):1297–1308.

Mishkin, M. and Ungerleider, L. G. (1982). Contribution of striate inputs to the visuospatial functions of parieto-preoccipital cortex in monkeys. *Behavioural brain research*, 6(1):57–77.

Mitiche, A., Mathlouthi, Y., and Ben Ayed, I. (2015). Monocular Concurrent Recovery of Structure and Motion Scene Flow. *Frontiers in ICT*, 2:16.

Mou, Y., Gong, M., Fu, H., Batmanghelich, K., Zhang, K., and Tao, D. (2019). Learning Depth from Monocular Videos Using Synthetic Data: A Temporally-Consistent Domain Adaptation Approach. *arXiv preprint arXiv:1907.06882*.

Mundt, M., Pliushch, I., Majumder, S., and Ramesh, V. (2019). Open Set Recognition Through Deep Neural Network Uncertainty: Does Out-of-Distribution Detection Require Generative Classifiers? In *Proc. of IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 0–0.

Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163.

Mur-Artal, R. and Tardós, J. D. (2017). Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262.

Murray, R. M., Li, Z., Sastry, S. S., and Sastry, S. S. (1994). *A mathematical introduction to robotic manipulation.* CRC press.

Naeini, M. P., Cooper, G., and Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. In *Proc of AAAI Conference on Artificial Intelligence (AAAI).*

Nalisnick, E., Matsukawa, A., Teh, Y. W., Gorur, D., and Lakshminarayanan, B. (2019). Do Deep Generative Models Know What They Don't Know? *Proc. of International Conference for Learning Representations (ICLR).*

Namdev, R. K., Krishna, K. M., and Jawahar, C. V. (2013). Multibody VSLAM with relative scale solution for curvilinear motion reconstruction. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5732–5739.

Neubeck, A. and Van Gool, L. (2006). Efficient Non-Maximum Suppression. In *Proc. of International Conference on Pattern Recognition (ICPR)*, volume 3, pages 850–855. IEEE.

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011). DTAM: Dense tracking and mapping in real-time. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2320–2327. IEEE.

Nguyen, A., Yosinski, J., and Clune, J. (2015). Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 427–436.

Niculescu-Mizil, A. and Caruana, R. (2005). Predicting good probabilities with supervised learning. In *Proc. of International Conference on Machine Learning (ICML)*, pages 625–632. ACM.

Ooi, T. L., Wu, B., and He, Z. J. (2001). Distance determined by the angular declination below the horizon. *Nature*, 414(6860):197.

Ošep, A., Hermans, A., Engelmann, F., Klostermann, D., Mathias, M., and Leibe, B. (2016). Multi-scale object candidates for generic object tracking in street scenes. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 3180–3187. IEEE.

Ozden, K. E., Cornelis, K., Van Eycken, L., and Van Gool, L. (2004). Reconstructing 3D trajectories of independently moving objects using generic constraints. *Computer Vision and Image Understanding*, 96(3):453–471.

Pawlowski, N., Brock, A., Lee, M. C., Rajchl, M., and Glocker, B. (2017). Implicit Weight Uncertainty in Neural Networks. *Advances in Neural Information Processing Systems Workshops (NeurIPSW).*

Pfeiffer, D. and Franke, U. (2011a). Modeling Dynamic 3D Environments by Means of The Stixel World. *IEEE Intelligent Transportation Systems Magazine*, 3(3):24–36.

Pfeiffer, D. and Franke, U. (2011b). Towards a Global Optimal Multi-Layer Stixel Representation of Dense 3D Data. In *Proc. of British Machine Vision Conference (BMVC)*, volume 11, pages 51–1.

Pfeiffer, D.-I. D. (2012). *The Stixel World.* PhD thesis, Humboldt-Universität zu Berlin.

Piewak, F., Pinggera, P., Enzweiler, M., Pfeiffer, D., and Zöllner, M. (2018). Improved Semantic Stixels via Multimodal Sensor Fusion. In *Proc. of German Conference on Pattern Recognition (GCPR)*, pages 447–458. Springer.

Platt, J. (1999). Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. *Advances in Large Margin Classifiers*, 10(3):61–74.

Pons, J.-P., Keriven, R., and Faugeras, O. (2007). Multi-View Stereo Reconstruction and Scene Flow Estimation with a Global Image-Based Matching Score. *International Journal of Computer Vision*, 72(2):179–193.

Postels, J., Ferroni, F., Coskun, H., Navab, N., and Tombari, F. (2019). Sampling-free Epistemic Uncertainty Estimation Using Approximated Variance Propagation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 2931–2940.

Prasad, V. and Bhowmick, B. (2019). SfMLearner++: Learning Monocular Depth & Ego-Motion using Meaningful Geometric Constraints. In *Proc. of IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2087–2096. IEEE.

Qi, X., Liao, R., Liu, Z., Urtasun, R., and Jia, J. (2018). GeoNet: Geometric Neural Network for Joint Depth and Surface Normal Estimation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 283–291.

Ranftl, R., Vineet, V., Chen, Q., and Koltun, V. (2016). Dense Monocular Depth Estimation in Complex Dynamic Scenes. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4058–4066.

Ranjan, A., Jampani, V., Balles, L., Kim, K., Sun, D., Wulff, J., and Black, M. J. (2019). Competitive Collaboration: Joint Unsupervised Learning of Depth, Camera Motion, Optical Flow and Motion Segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12240–12249.

Ros, G., Ramos, S., Granados, M., Bakhtiary, A., Vazquez, D., and Lopez, A. M. (2015). Vision-based offline-online perception paradigm for autonomous driving. In *Proc. of IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 231–238. IEEE.

Ruo Zhang, Ping-Sing Tsai, Cryer, J. E., and Shah, M. (1999). Shape-from-shading: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706.

Sabzevari, R. and Scaramuzza, D. (2016). Multi-body Motion Estimation from Monocular Vehicle-Mounted Cameras. *IEEE Transactions on Robotics*, 32(3):638–651.

Saleem, N. H., Griffin, A., and Klette, R. (2018). Monocular Stixels: A LIDAR-guided Approach. In *Proc. of International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6.

Saleem, N. H., Rezaei, M., and Klette, R. (2017). Extending the stixel world using polynomial ground manifold approximation . In *Proc. of International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pages 1–6. IEEE.

Saxena, A., Sun, M., and Ng, A. Y. (2009). Make3D: Learning 3D Scene Structure from a Single Still Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):824–840.

Scharwächter, T., Enzweiler, M., Franke, U., and Roth, S. (2013). Efficient Multi-cue Scene Segmentation. In *Proc. of German Conference on Pattern Recognition (GCPR)*, pages 435–445. Springer.

Scharwächter, T., Enzweiler, M., Franke, U., and Roth, S. (2014). Stixmantics: A Medium-Level Model for Real-Time Semantic Scene Understanding. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 533–548. Springer.

Scharwächter, T. and Franke, U. (2015). Low-level fusion of color, texture and depth for robust road scene understanding. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 599–604. IEEE.

Schneider, L., Cordts, M., Rehfeld, T., Pfeiffer, D., Enzweiler, M., Franke, U., Pollefeys, M., and Roth, S. (2016). Semantic Stixels: Depth is not enough. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 110–117. IEEE.

Schubert, D., Demmel, N., Usenko, V., Stueckler, J., and Cremers, D. (2018). Direct Sparse Odometry With Rolling Shutter. In *Proc. of European Conference on Computer Vision (ECCV)*.

Schultheiss, A., Käding, C., Freytag, A., and Denzler, J. (2017). Finding the Unknown: Novelty Detection with Extreme Value Signatures of Deep Neural Activations. In *Proc. of German Conference on Pattern Recognition (GCPR)*, pages 226–238. Springer.

Schuster, R., Bailer, C., Wasenmüller, O., and Stricker, D. (2018). Combining Stereo Disparity and Optical Flow for Basic Scene Flow. In *Commercial Vehicle Technology 2018*, pages 90–101. Springer.

Schwing, A. G. and Urtasun, R. (2012). Efficient Exact Inference for 3D Indoor Scene Understanding. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 299–313. Springer.

Segù, M., Loquercio, A., and Scaramuzza, D. (2020). A General Framework for Uncertainty Estimation in Deep Learning. *IEEE Robotics and Automation Letters*.

Sengupta, S., Greveson, E., Shahrokni, A., and Torr, P. H. (2013). Urban 3D semantic modelling using stereo vision. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 580–585. IEEE.

Shen, T., Luo, Z., Zhou, L., Deng, H., Zhang, R., Fang, T., and Quan, L. (2019). Beyond Photometric Loss for Self-Supervised Ego-Motion Estimation. *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*.

Siam, M., Mahgoub, H., Zahran, M., Yogamani, S., Jagersand, M., and El-Sallab, A. (2018). MODNet: Motion and Appearance based Moving Object Detection Network for Autonomous Driving. In *Proc. of IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 2859–2864. IEEE.

Song, S. and Chandraker, M. (2014). Robust Scale Estimation in Real-Time Monocular SFM for Autonomous Driving. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Song, S. and Chandraker, M. (2015). Joint SFM and detection cues for monocular 3D localization in road scenes. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3734–3742.

Suhr, J. K. and Jung, H. G. (2019). Rearview Camera-Based Stixel Generation for Backing Crash Prevention. *IEEE Transactions on Intelligent Transportation Systems*.

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. (2018). PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8934–8943.

Tateno, K., Tombari, F., Laina, I., and Navab, N. (2017). CNN-SLAM: Real-Time Dense Monocular SLAM With Learned Depth Prediction. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2.

Teng, Q., Chen, Y., and Huang, C. (2018). Occlusion-Aware Unsupervised Learning of Monocular Depth, Optical Flow and Camera Pose with Geometric Constraints. *Future Internet*, 10(10):92.

Torr, P. H. and Zisserman, A. (2000). MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156.

Tosi, F., Aleotti, F., Poggi, M., and Mattoccia, S. (2019). Learning Monocular Depth Estimation Infusing Traditional Stereo Knowledge. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Uhrig, J., Rehder, E., Fröhlich, B., Franke, U., and Brox, T. (2018). Box2Pix: Single-Shot Instance Segmentation by Assigning Pixels to Object Boxes. In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pages 292–299. IEEE.

Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., and Geiger, A. (2017). Sparsity Invariant CNNs. In *Proc. of IEEE International Conference on 3D Vision (3DV)*.

Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., and Brox, T. (2017). DeMoN: Depth and Motion Network for Learning Monocular Stereo. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Upcroft, B., McManus, C., Churchill, W., Maddern, W., and Newman, P. (2014). Lighting invariant urban street classification. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1712–1718. IEEE.

Valgaerts, L., Bruhn, A., Zimmer, H., Weickert, J., Stoll, C., and Theobalt, C. (2010). Joint Estimation of Motion, Structure and Geometry from Stereo Sequences. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 568–581. Springer.

Vedula, S., Baker, S., Rander, P., Collins, R., and Kanade, T. (1999). Three-dimensional scene flow. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 722–729. IEEE.

Vedula, S., Rander, P., Collins, R., and Kanade, T. (2005). Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):475–480.

Vidal, R. and Hartley, R. (2008). Three-View Multibody Structure from Motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):214–227.

Vogel, C. (2015). *Robust and Accurate 3D Motion Estimation under Adverse Conditions*. PhD thesis, ETH Zurich.

Vogel, C., Schindler, K., and Roth, S. (2013). Piecewise Rigid Scene Flow. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1377–1384.

Vyas, A., Jammalamadaka, N., Zhu, X., Das, D., Kaul, B., and Willke, T. L. (2018). Out-of-Distribution Detection Using an Ensemble of Self Supervised Leave-out Classifiers. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 550–564.

Wainwright, M. J., Jaakkola, T. S., and Willsky, A. S. (2005). MAP estimation via agreement on trees: message-passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717.

Wang, C., Buenaposada, J. M., Zhu, R., and Lucey, S. (2018). Learning Depth from Monocular Videos using Direct Methods. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2022–2030.

Wang, N. (2018). An MXNet implementation of Mask R-CNN. https://github.com/TuSimple/mx-maskrcnn. [accessed June, 25 2018].

Wang, R., Pizer, S. M., and Frahm, J.-M. (2019). Recurrent Neural Network for (Un-)Supervised Learning of Monocular Video Visual Odometry and Depth. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wedel, A., Brox, T., Vaudrey, T., Rabe, C., Franke, U., and Cremers, D. (2011). Stereoscopic Scene Flow Computation for 3D Motion Understanding. *International Journal of Computer Vision*, 95(1):29–51.

Wedel, A., Rabe, C., Vaudrey, T., Brox, T., Franke, U., and Cremers, D. (2008). Efficient Dense Scene Flow from Sparse or Dense Stereo Data. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 739–751. Springer.

Wolcott, R. W. and Eustice, R. M. (2016). Probabilistic Obstacle Partitioning of Monocular Video for Autonomous Vehicles. In *Proc. of British Machine Vision Conference (BMVC)*.

Wu, Z., Wu, X., Zhang, X., Wang, S., and Ju, L. (2019). Spatial Correspondence With Generative Adversarial Network: Learning Depth From Monocular Videos. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 7494–7504.

Xia, Z., Sullivan, P., and Chakrabarti, A. (2020). Generating and Exploiting Probabilistic Monocular Depth Estimates. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xiao, D., Yang, Q., Yang, B., and Wei, W. (2017). Monocular scene flow estimation via variational method. *Multimedia Tools and Applications*, 76(8):10575–10597.

Xiong, Y., Liao, R., Zhao, H., Hu, R., Bai, M., Yumer, E., and Urtasun, R. (2019). UPSNet: A Unified Panoptic Segmentation Network. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xu, B., Wang, X., and Song, M. (2018). Extending the Dynamic Stixel World with B-Spline based Road Estimation for Obstacle Detection. In *Chinese Automation Congress (CAC)*, pages 2973–2978. IEEE.

Xu, P., Davoine, F., Bordes, J.-B., Zhao, H., and Denœux, T. (2016). Multimodal information fusion for urban scene understanding. *Machine Vision and Applications*, 27(3):331–349.

Yamaguchi, K., McAllester, D., and Urtasun, R. (2013). Robust Monocular Epipolar Flow Estimation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1862–1869.

Yamaguchi, K., McAllester, D., and Urtasun, R. (2014). Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Proc. of European Conference on Computer Vision (ECCV)*, pages 756–771, Cham. Springer International Publishing.

Yang, G. and Ramanan, D. (2020). Upgrading Optical Flow to 3D Scene Flow through Optical Expansion. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yang, N., von Stumberg, L., Wang, R., and Cremers, D. (2020). D3VO: Deep Depth, Deep Pose and Deep Uncertainty for Monocular Visual Odometry. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yang, N., Wang, R., Stückler, J., and Cremers, D. (2018a). Deep Virtual Stereo Odometry: Leveraging Deep Depth Prediction for Monocular Direct Sparse Odometry. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 835–852. Springer.

Yang, Z., Wang, P., Wang, Y., Xu, W., and Nevatia, R. (2018b). Every Pixel Counts: Unsupervised Geometry Learning with Holistic 3D Motion Understanding. In *Proc. of European Conference on Computer Vision Workshops (ECCVW)*.

Yin, X., Wang, X., Du, X., and Chen, Q. (2017). Scale Recovery for Monocular Visual Odometry Using Depth Estimated With Deep Convolutional Neural Fields. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 5870–5878.

Yin, Z., Darrell, T., and Yu, F. (2019). Hierarchical Discrete Distribution Decomposition for Match Density Estimation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yin, Z. and Shi, J. (2018). GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2.

Yu, Q. and Aizawa, K. (2019). Unsupervised Out-of-Distribution Detection by Maximum Classifier Discrepancy. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 9518–9526.

Yuan, C. and Medioni, G. (2006). 3D Reconstruction of Background and Objects Moving on Ground Plane Viewed from a Moving Camera. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2261–2268. IEEE.

Zabih, R. and Woodfill, J. (1994). Non-parametric local transforms for computing visual correspondence. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 151–158. Springer.

Zadrozny, B. and Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699. ACM.

Zhan, H., Garg, R., Weerasekera, C. S., Li, K., Agarwal, H., and Reid, I. (2018). Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 340–349.

Zhang, J.-N., Su, Q.-X., Liu, P.-Y., Ge, H.-Y., and Zhang, Z.-F. (2019). MuDeepNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose Using Multi-view Consistency Loss. *International Journal of Control, Automation and Systems*.

Zhongfei Zhang, Weiss, R., and Hanson, A. R. (1997). Obstacle detection based on qualitative and quantitative 3d reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):15–26.

Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised Learning of Depth and Ego-Motion from Video. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhuo, W., Salzmann, M., He, X., and Liu, M. (2015). Indoor Scene Structure Analysis for Single Image Depth Estimation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 614–622.

Zou, Y., Luo, Z., and Huang, J.-B. (2018). DF-Net: Unsupervised Joint Learning of Depth and Flow using Cross-Network Consistency. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 36–53.