Institute of Visualization and Interactive Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Master's Thesis

A Global Adversarial Attack on Scene Flow

Marcel Hasenbalg

Course of Study:	Informatik
Examiner:	Prof. Dr.-Ing. Andrés Bruhn
Supervisor:	Jenny Schmalfuß, M.Sc.
Commenced:	September 8, 2022
Completed:	March 8, 2023

Abstract

In the field of computer vision deep neural networks are proven and tested solutions for complex problems. Depth reconstruction from stereo images and estimation of optical flow from image sequences is solved most accurate by deep learning based methods. The extension of optical flow to three spatial dimensions is called scene flow. Deep neural networks for scene flow estimation outperform classical energy functional minimisation methods. The application of neural network solutions for scene flow estimation in security-critical applications such as autonomous driving or robot-assisted surgery calls for an in-depth evaluation of these systems. The manipulation of network outputs with adversarial attacks was first uncovered for object classification networks. Adversarial attacks aim at introducing imperceptible perturbations to input images to cause erroneous network outputs. Recent research could reveal the low adversarial robustness of state-of-the-art stereo matching and optical flow neural network solutions.

In this thesis a framework to generate a targeted constrained global adversarial attack on scene flow neural networks (GSFA) is developed. Multiple different attack types which add perturbations to specific types of inputs or at different stages of the networks processing pipeline are introduced. The attack types are applied to the state-of-the-art scene flow estimation network RAFT-3D. The effects of GSFA regarding scene flow estimation accuracy and perturbation size of inputs is analysed using RAFT-3D and two scene flow benchmark datasets. The results of various experiments proof that RAFT-3D shows the same vulnerabilities to adversarial attacks as optical flow and stereo matching networks. Constraints on perturbation sizes effectively keep perturbations imperceptible or hardly perceptible, while scene flow estimations approach a defined zero scene flow target.

Kurzfassung

Im Bereich Computer Vision haben sich tiefe neuronale Netze als Lösung für komplexe Probleme bewährt. Die Rekonstruktion von Tiefeninformation aus Stereo-Bildern und die Schätzung des optischen Flusses aus Bildsequenzen wird am genauesten von Methoden, welche auf tiefem maschinellem Lernen basieren gelöst. Die Erweiertung des optischen Flusses auf drei räumliche Dimensionen wird Scene Flow genannt. Tiefe neuronale Netze für die Schätzung von Scene Flow übertreffen die Genauigkeit klassischer Methoden, welche auf Minimierung von Energie Funktionen basieren. Die Anwendung von neuronalen Netzen für die Schätzung von Scene Flow in sicherheitskritischen Bereichen wie autonomes Fahren oder roboter-assistierter Chirurgie erfordert eine gründliche Evaluation dieser Systeme. Die Manipulation der Ausgabeschicht von Neuronalen Netzwerken mithilfe von Adversarial Attacks wurde zunächst für Objekt-Klassifizierungsnetzwerke entdeckt. Das Ziel von Adversarial Attacks ist den Eingabe-Bildern minimale Störungen hinzuzufügen, welche letztendlich zu fehlerhaften Ausgaben führen.

Aktuelle Forschungsergebnisse konnten zeigen, dass moderne Netzwerke zur Schätzung des optischen Flusses oder der Tiefenrekonstruktion eine geringe Robustheit gegen Adversarial Attacks aufweisen.

In dieser Thesis wird ein Framework entwickelt, welches eine gezielte beschränkte globale Adversarial Attack (GSFA) auf neuronale Netze zur Scene Flow Schätzung ermöglicht. Mehrere verschiedene Arten von Attacken, welche Störungen auf bestimmte Eingaben oder an verschiedenen Stellen der Vearbeitung hinzufügen, werden eingeführt. Diese Arten von Adversarial Attacks werden angewendet um die Ausgaben des modernen Scene Flow Netzwerks RAFT-3D zu manipulieren. Die Auswirkungen von GSFA auf die Genaugikeit der Scene Flow Schätzung und die Größe der Störung der Eingabebilder wird mit RAFT-3D und zwei Scene Flow Benchmark Datensätzen überprüft. Die Ergebnisse vielseitiger Experimente beweisen, dass RAFT-3D dieselben Schwächen gegen Adversarial Attacks aufzeigt, wie Netzwerke zur Tiefenrekonstruktion oder zur Schätzung des optischen Flusses. Einschränkungen der Größe der Störungen sind effektiv darin die Störungen der Eingabebilder im nicht wahrnehmbaren oder kaum wahrnehmbaren Bereich zu halten, während die Ausgabe sich dem definierten Ziel des Zero Scene Flow annähert.

Contents

1 Introduction 7

2 Foundations and Related Work 11
2.1 Stereo Matching . . . 11
2.2 Optical Flow . . . 19
2.3 Scene Flow . . . 23
2.4 Adversarial Attacks on Flow Networks . . . 30

3 Applied Techniques in Detail 39
3.1 GA-Net: Guided Aggregation Net . . . 39
3.2 RAFT-3D: Scene Flow Estimation . . . 42
3.3 PCFA: Perturbation-Constrained Flow Attack . . . 45

4 A Global Adversarial Attack on Scene Flow 47
4.1 Targets for Scene Flow . . . 47
4.2 Perturbations for Scene Flow . . . 48
4.3 Loss Functions for Adversarial Examples . . . 50
4.4 Attack Types . . . 51
4.5 Adversarial Attacks on RAFT-3D . . . 52
4.6 Implementation Details . . . 56

5 Experiments and Results 61
5.1 Benchmarks for Flow Problems . . . 61
5.2 Parameter Refinement . . . 64
5.2.1 Penalty Weights of GSFAC . . . 64
5.2.2 Learning Rates for GSFAC . . . 66
5.3 Global Scene Flow Attack on RAFT-3D . . . 68
5.3.1 GSFA: Unconstrained Attack . . . 69
5.3.2 GSFAC: Perturbation Constrained Attack . . . 72
5.3.3 GSFAC-D and GSFAC-I: Perturbation of Input Types . . . 75
5.3.4 GSFAC-coupled: Adversarial Attack including GA-Net . . . 80
5.3.5 Comparison of Attack Types . . . 82

6 Conclusion and Outlook 85
6.1 Outlook . . . 85

6.2 Conclusion 86

A Appendix 89
A.1 Penalty Weights Refinement Experiments 89
A.2 Learning Rates Refinement Experiments 90

Bibliography 91

1 Introduction

Applications of scene flow include autonomous vehicle navigation, robotics, video analysis and augmented reality. In the domain of autonomous driving scene flow helps the vehicle to navigate and understand their environment. The motion estimation of other vehicles, pedestrians and environment objects is critical to make sound driving decisions and avoid collisions [1]. Robots equipped with a stereoscopic camera are able to provide doctors with important information about motion of organs during minimally invasive surgeries [2]. Extracted scene flow information can be used to analyze video footage to extract the motion information of objects in the scene. For instance, this is applied in football sports analysis to track players or to enhance spectator experience with augmented reality [3].

Scene flow is the three-dimensional motion of real-word or virtual objects in a three-dimensional scene. It can be derived from subsequent images captured by multiple cameras at different points in time. A dense scene flow maps each three-dimensional point of a scene to a three-dimensional motion vector, which indicates the motion of the point for a certain period in each direction. Optical flow is a projection of the three-dimensional scene flow onto the two-dimensional image plane. Stereo matching recovers three-dimensional structures from multiple two-dimensional images which are captured by multiple cameras. By combining the solutions of optical flow and stereo matching problems for a scene one can derive scene flow.

The formulation and first solutions of the scene flow problem predate the extensive usage of deep neural network solutions for complex problems [4]. However, since improved hardware and GPU-based computing made neural network solutions feasible in the last decade, all current state-of-the-art scene flow estimation methods use deep learning techniques. On the one hand, deep learning based methods excel at solving complex problems and perform well given solid training data. On the other hand, they cannot provide interpretable solutions and are therefore also referred to as black box. With the rise of deep learning based methods a major weakness called adversarial attacks was identified by Szegedy et al. [5]. Minimal perturbations of input data, which are hardly perceptible for human observers, can lead to erroneous output classifications and predictions. Therefore, in recent years the robustness against adversarial attacks has become another important quality characteristic of deep learning based methods besides prediction accuracy.

Adversarial attacks in the scope of computer vision flow problems are not extensively covered in current literature yet. Optical flow estimation neural networks were attacked by Ranjan et al. [6] using patch-based adversarial attacks. Schrodi et al. [7] and Schmalfuss et al. [8]

used global adversarial attacks to examine the robustness of optical flow deep neural network methods. The depth estimation of stereo matching neural networks was attacked by Berger et al. [9] and Wong et al. [10]. All of them could proof certain vulnerabilities in the analysed state-of-the-art methods. To the best of the author's knowledge, no adversarial attacks on scene flow estimation neural networks have been conducted yet. The experiments in this work show, that scene flow neural networks are affected by the same vulnerabilities to adversarial attacks as optical flow and stereo matching networks. As scene flow relies on solutions to optical and stereo flow problems, it is interesting to analyse the effects of adversarial attacks on different inputs or at different stages of the network.

In this work:

- the lack of knowledge about the vulnerability of scene flow deep learning based methods to adversarial attacks is addressed.
- a framework to perform different types of unconstrained and constrained attacks on scene flow estimation neural networks is introduced.
- different adversarial attacks are carried out on the state-of-the-art scene flow estimation network RAFT-3D.
- the robustness of RAFT-3D against these adversarial attacks is compared on different benchmark datasets.

Outline

This thesis is structured as follows:

Chapter 2 – Foundations and Related Work: In this chapter existing solutions and practices from literature to computer vision flow problems as well as to adversarial attacks are presented. Furthermore, the fundamentals of optical, stereo and scene flow problems are explained in detail.

Chapter 3 – Applied Techniques in Detail covers the applied methods used for the adversarial attacks in this work. These methods include neural networks for scene flow estimation (RAFT-3D) and stereo matching estimation (GA-Net). The global constrained adversarial attack developed in this work is based on the perturbation constrained flow attack (PCFA), whose operation is explained in Section 3.3.

Chapter 4 – A Global Adversarial Attack on Scene Flow: The framework to perform different types of attacks on scene flow estimation neural networks is introduced. The RAFT-3D scene flow network is attacked using different combinations of attack inputs and parameters, as well as attacking the network at different stages of the processing pipeline to identify the most effective attack types.

Chapter 5 – Experiments and Results of the different experiments performed in Chapter 4 are presented in this chapter. Therefore, error metrics from literature as well as new metrics for scene flow and adversarial attacks are defined. Ground truth data from two different scene flow benchmarks is used to compare the adversarial attack strength and robustness of RAFT-3D.

Chapter 6 – Conclusion and Outlook concludes the results of this work and presents promising follow-up experiments and future research topics in this domain.

2 Foundations and Related Work

The following sections focus on related work from literature and the fundamental definitions of scene flow and adversarial attacks. The scene flow problem (Section 2.3), can be subdivided into stereo flow and optical flow which are discussed in Section 2.1 and 2.2 respectively. Basic principles of adversarial attacks and relevant publications in this domain are discussed in Section 2.4.

Inherent to all flow problems is the change of scene properties in space and time. In stereo flow or stereo matching the position of real world objects changes in the image space, resulting in a scalar disparity between the left and right image of a stereo image pair captured at a fixed point in time. In optical flow the real world motion of objects over time is projected to the image space, where the apparent motion of objects can be used to infer a displacement vector field from an ordered sequence of images. Consequently, scene flow preserves the three-dimensional real world motion of objects by combining stereo flow and optical flow inferred from a sequence of stereo image pairs.

In the following the three different flow problems are defined and proven solutions are presented. Standard solutions to these problems from the past include local and variational methods, which involve the minimisation of an energy functional. However, auspicious results of deep-learning based methods for computer vision problems in recent years led to the development of deep neural networks to solve these flow problems. Classical methods and state-of-the-art neural networks for scene flow rely on principles from stereo matching and optical flow.

2.1 Stereo Matching

The goal of the stereo matching or stereo flow problem is to reconstruct the three-dimensional structure of a scene using multiple two-dimensional images. Human binocular vision is able to perceive depth with two eyes, which are offset from one another. Both eyes view the same object from a different perspective. Hence, disparity information processed by the visual cortex enables humans to construct a three-dimensional representation of the scene [11]. The human brain is capable of performing such tasks automatically, but computer systems can use the same information for stereo matching algorithms. However, systems may need calibration to enable the translation of two-dimensional image data to three-dimensional world data.

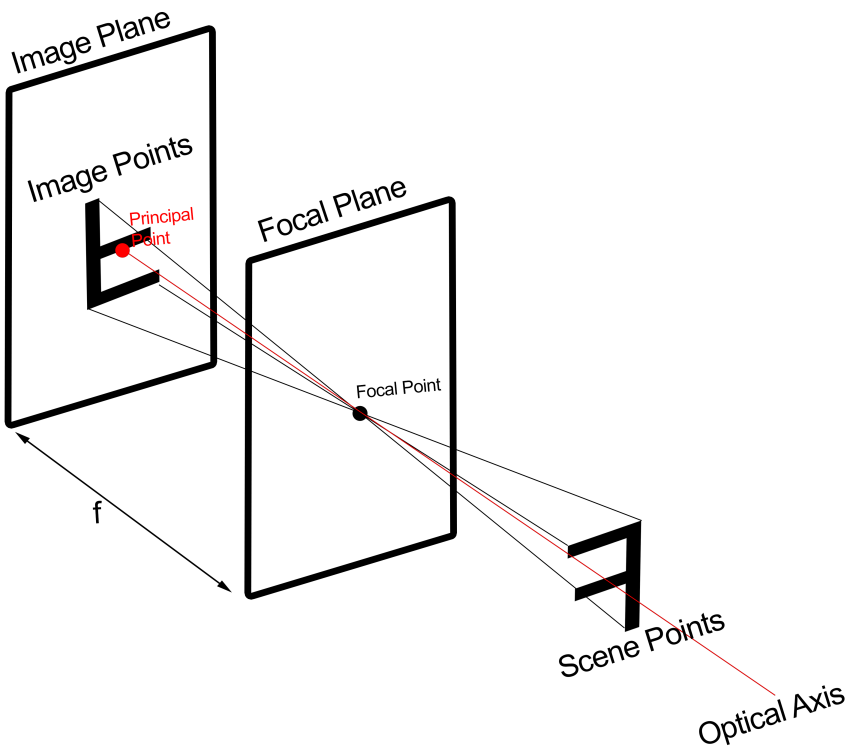


Figure 2.1: Objects captured with the pinhole camera model appear upside down on the image plane. The distance between focal and image plane is called focal distance f.

Required parameters regarding camera models and calibration as well as techniques to find correspondences are explained in the following.

2.1.1 Camera Model and Parameters

The pinhole camera model can be used to model the perspective projection from three-dimensional space onto the two-dimensional image plane [12]. A three-dimensional scene point $M = (X, Y, Z)$ is mapped to the two-dimensional image point $m = (x, y)$. As Figure 2.1 shows, the camera model uses the focal plane F and the image plane I, where the distance between these planes is called focal distance f. The order of image plane and focal plane leads to objects appearing upside down in the resulting image.

From a scene point $M = (X, Y, Z)$ to an image pixel $p = (u, v)$ three transitions between different coordinate systems are conducted:

1. **Transition from world to camera coordinates:** The six extrinsic parameters, three parameters for translation and rotation each, characterise the position and orientation of the camera in relation to the world coordinates [13]. These parameters are used to transform points from the world coordinates to the camera coordinates. The 3×3 rotation matrix R with entries $r_{i,j}$ is part of the upper left corner of the extrinsic parameters A_{ext}. It is defined as the product of three rotation matrices with a specific rotation angle of the camera around each axis, called roll, pitch and yaw. The location of the camera in the world coordinate system is defined by the translation vector t. In the end the extrinsic matrix can be constructed, if the camera's roll, pitch and yaw angles and translation in each direction are known.

$$A_{ext} = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.1}$$

   This matrix notation of combined translation and rotation of an object is also called SE(3), the special Euclidean group of three-dimensional rigid body displacement [14].

2. **Transition from camera coordinates to image coordinates:** The projective matrix P_f, where f denotes the focal length, transitions points from the camera coordinate system to the image plane. Homogeneous coordinates are a remedy for the non-uniqueness of multiple points on the optic ray mapping to the same pixel. An additional coordinate is added by scaling each point by a factor $\lambda \neq 0$:

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} \lambda x \\ \lambda y \\ \lambda \end{pmatrix}. \tag{2.2}$$

   Division by λ transforms homogeneous coordinates back to non-homogenous coordinates. With the use of the intercept theorem [15] and homogenous coordinates the transition from camera to image coordinates can be expressed with the projection matrix P_f:

$$\begin{pmatrix} \lambda x \\ \lambda y \\ \lambda \end{pmatrix} = \begin{pmatrix} Zx \\ Zy \\ Z \end{pmatrix} = \underbrace{\begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{P_f} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{2.3}$$

3. **Transition from camera coordinates to image coordinates:** Finally, image coordinates are translated to pixel coordinates using the intrinsic camera parameters. The principal point c of the image plane is located at (u_0, v_0). These parameters can be put together in

the intrinsic camera parameters matrix A_{int}. For quadratic pixels with density k aligned in a square grid the intrinsic matrix becomes:

$$A_{int} = \begin{pmatrix} k & -k & u_0 \\ 0 & k & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

By ordered multiplication of the three coordinate transformation matrices one can calculate the pixel coordinates from real world coordinates:

$$\underbrace{\begin{pmatrix} k & -k & u_0 \\ 0 & k & v_0 \\ 0 & 0 & 1 \end{pmatrix}}_{A_{int}} \underbrace{\begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{P_f} \underbrace{\begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{A_{ext}} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{2.5}$$

The product of the intrinsic matrix A_{int}, projection matrix P_f and extrinsic matrix A_{ext} is called full projection matrix P [13]. Estimating the intrinsic and extrinsic camera parameters of a stereo camera rig is called camera calibration. In practice, the stereo camera calibration can be determined by analysis of captured images of an object of known dimensions.

2.1.2 Stereo geometry for multiple cameras

The use of multiple calibrated cameras enables the depth reconstruction of a scene, which is important for estimating three-dimensional scene flow. The basic setup of a stereo flow problem consists of a set of at least two cameras (left and right). If more than two cameras are used the problem is referred to as multi-view stereo (MVS) [16]. Both cameras capture an image of the scene at the same time, resulting in images I_0 and I_1. Both cameras are located at different positions and the distance between their focal points is called baseline b. If the optical axes of the cameras are parallel, the setup is called ortho-parallel, as illustrated in Figure 2.2.

The stereo problem can be subdivided in two phases:

1. **Disparity estimation:** Let $M = (X, Y, Z)$ be a three-dimensional scene point displayed in the left image I_0 and the right image I_1. Because both cameras are located at different positions, the pixels representing M will be located at different positions in the images. Identifying the conjugated points in I_0 and I_1 means finding the pixels representing M in each image. The plane spanned by the scene point M and the focal points of the cameras C_0 and C_1 is called epipolar plane [13] (see Figure 2.2). All epipolar lines intersect the epipol of a camera, which is a mapping of the focal point of the other camera. Intersecting the epipolar plane with the image plane of C_0 yields the first epipolar line e_0. In the same way e_1 is defined by the intersection of the epipolar plane with the image

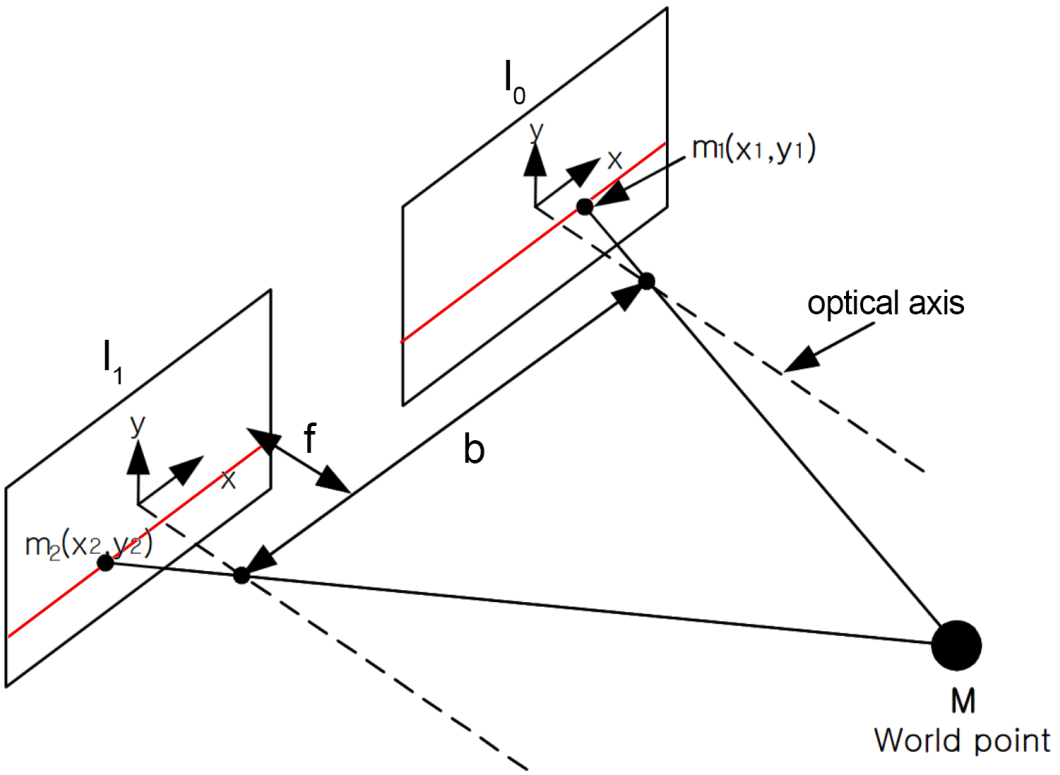


Figure 2.2: Stereo geometry for ortho-parallel cameras resulting in disparity $x_1 - x_2$ for scene point M. The epipolar lines are marked in red. Illustration adapted from Kim et al. [17].

plane of C_1. The points representing M are located on the epiploar lines e_0 and e_1. By superposition of I_0 and I_1 the displacement between conjugated points can be identified. This displacement is defined as disparity d and can be visualised by a gray-scale image, where each pixel is assigned its disparity value (see Figure 2.3). In practice the process of stereo rectification ensures that corresponding points are located on the same image scan line (y-axis). Therefore, the disparity becomes the displacement in x-direction and the distance can be calculated by subtraction of the x-coordinates x_1 and x_2 of the corresponding points.

2. **Depth reconstruction:** In the ortho-parallel case the depth Z of the scene point M can be reconstructed using the disparity $x_1 - x_2$:

$$Z = \frac{bf}{x_1 - x_2} \tag{2.6}$$

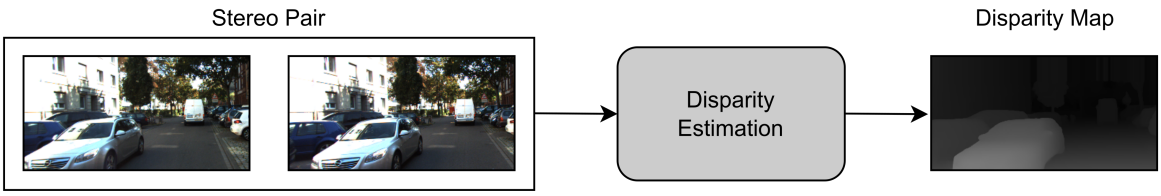


Figure 2.3: A disparity map encodes the disparity value for each pixel of the rectified stereo image pair.

2.1.3 Stereo Rectification

In the ortho-parallel case the epipolar lines are parallel to the x-axis of the image plane, which is convenient as it reduces the search space for correspondences to one dimension. However, this does not hold for a setup of converging cameras. The process of rectification transforms a stereo image pair in such a way, that the epipolar lines become parallel to the x-axis of the rectified image plane. This can be achieved by projective transforms, which map the epipoles to points that are infinitely far away from the camera. Consequently, corresponding epipolar lines become parallel. The algorithm proposed by Hartley and Zisserman [13] performs stereo rectification by finding a projective transform that maps epipoles to infinity and minimizes image distortions between the warped images.

Finding conjugated points in stereo images is a correspondence problem and the most complex part of the stereo problem discussed in the following sections.

2.1.4 Local Methods for Stereo Matching

Local methods search the correspondences in both images locally by matching local image regions of a stereo pair. Corresponding points only appear along epiploar lines and within a maximum displacement distance. Zhang et al. [18] could show that recovery of the epipolar geometry from images without camera calibration is possible by estimation of correspondences only.

The intensity values of the local window around corresponding points in both images are assumed to be similar. Local methods compare the similarity of box- or ball-shaped windows around two points (x, y) and $(x+u, y+v)$ where $(u, v)^\top$ denotes the displacement vector along the epipolar line. The intensity consistency assumption for stereo matching in Equation (2.7) states this similarity:

$$I_0(x, y) = I_1(x + u, y + v) \tag{2.7}$$

These methods compare all window pair candidates in the search space, which is limited by the epipolar lines and the maximum displacement. Hirschmüller and Scharstein [19] list

and compare different measures used for window comparison, including the window mean value, normalized cross-correlation (NCC), sum of absolute differences (SAD) and mutual information (MI). Local methods that use these similarity measures were proposed by Zhang et al. [20] (NCC) or Super and Klarquist [21] (mean and standard deviation). Illumination intensity, direction or color changes can violate the intensity consistency assumption, hindering the matching of local windows. Window similarity measures that are invariant under certain illumination changes are desirable and were proposed by Heo et al. [22] in form of adaptive normalized cross-correlation (ANCC).

The two windows with the highest similarity score are selected as corresponding points. However, in flat image regions or regions including repetitive textures the window similarity score is not discriminable, leading to non-dense disparity maps of local methods. To sum up, local methods are simple and robust but cannot always yield a dense representation of correspondences.

2.1.5 Variational Methods for Stereo Matching

Variational methods examine the images globally to estimate the disparity. Two assumptions are fundamental for these kinds of methods [23]:

1. **Intensity consistency assumption:** Conjugated points have the same pixel values (see Equation (2.7)).
2. **Smoothness assumption:** Disparity maps vary smoothly in space.

A disparity map which satisfies these constraints best can be computed by minimisation of a multi-term energy functional. Equation (2.8) shows such a simplified energy functional for the ortho-parallel case. In an orthoparallel camera setup, no displacements along the y-axis occur. The first term enforces the intensity consistency assumption between corresponding points. The second term ensures a smooth disparity map u with a constant penalty factor α.

$$E(u) = \int_{\Omega} (I_0(x, y) - I_1(x + u, y))^2 + \alpha |\nabla u|^2 \, dx \, dy \tag{2.8}$$

Alvarez et al. [24] implemented a similar energy functional which is not limited to the ortho-parallel case. The minimisation of the energy functional can be achieved by a numerical scheme using finite differences, primal-dual algorithms [25] or gradient descent algorithms [26].

2.1.6 Semi-Global Matching

As global methods are computationally expensive, semi-global matching reduces complexity by providing an alternative approach to the smoothness constraint. Hirschmüller [27] proposes

semi-global matching, which replaces the global approximation of two-dimensional smoothness by a combination of multiple one-dimensional constraints. The stereo reconstruction in classical methods consists of three stages: feature extraction, matching cost aggregation and disparity estimation. In the first step image features are extracted, which can then be used to calculate the cost of matching pixels $C(p, d)$ between the left and right image in the second step. Finally, depending on the matching cost the pixel correspondences can be established and the disparity map is generated. Using only pixel-wise and feature-based matching can lead to incorrect correspondences, as occlusions, noise or reflections can lead to lower matching costs. Cost aggregation, as implemented in semi-global matching, counteracts this problem by extending the matching cost computation to respect certain constraints, such as local and global smoothness assumptions. The matching cost between pixels and the smoothness assumptions are formulated in an energy function $E(d)$, where d is the output disparity map [27]:

$$E(d) = \sum_{p} C(p, d_p) + \sum_{q \in N_p} P_1 \cdot \delta(|d_p - d_q| = 1) + \sum_{q \in N_p} P_2 \cdot \delta(|d_p - d_q| > 1). \tag{2.9}$$

The first term sums up the matching cost $C(p, d_p)$ of each pixel p with assigned disparity d_p. The second and third term are penalty terms, which handle discontinuities of the disparity around pixel p in a neighbourhood N_p. Disparity differences of 1 in the neighbourhood are counted by function δ and penalised by a constant factor P_1. Larger disparity discontinuities in N_p are penalised by the third term with factor P_2.

Instead of ensuring smoothness of disparity values in all directions, semi-global matching aggregates the costs in C along specific paths starting from pixel p. This eases computational complexity, by reducing a two-dimensional problem down to multiple one-dimensional problems. Hirschmüller [27] proposes to aggregate costs along 8 to 16 paths in equally spaced distinct directions. The cost of all these paths is summed up, resulting in the lowest cost $C(p, d)$ for pixel p if disparities along all these paths are smooth. After the cost aggregation C is completed, the disparity can be calculated by minimising the energy function in Equation (2.9).

2.1.7 Deep learning-based Methods for Stereo Matching

Deep learning-based methods are able to learn features from training data instead of relying on handcrafted solutions used in local and global stereo matching methods. Therefore, neural networks for disparity estimation have become the most promising and accurate solutions [28]. These methods rely on (labeled) training data, which is provided by publicly available benchmarks [29, 1, 30]. In actual fact, there is no need to learn stereo, as depth estimation can be solved exactly if no assumptions are violated and the camera system is calibrated [10]. Nevertheless, deep learning is used to regularise the depth reconstruction in areas where these assumptions are violated, which is the case in many real world scenarios.

Mayer et al. [31] introduced the concept of cost volumes in end-to-end stereo matching neural networks. The cost volume represents the costs associated with matching pixels between the left and right images of a stereo pair. It is implemented as multidimensional array, which stores the similarity measure for the pixel pairs. Group-wise correlation was introduced by GwcNet [32] to further improve disparity estimation. Instead of comparing single pixels for a possible match, groups of pixels are compared while searching for correspondences similar to window comparison in local methods. This technique improved robustness of deep-learning based methods for noisy input images. Multiscale cost volumes enable networks to extract image information on multiple scales and were introduced by Shen et al. [33]. The information on the coarse level of a multiscale space allow for more precise disparity estimation on the finer levels.

The idea of semi-global matching is revisited by Zhang et al. [34], who propose GA-Net with local guided aggregation layers (LGA) and semi-global guided aggregation layers (SGA). SGA layers act as a differentiable version of the semi-global matching algorithm, whose parameters and penalty coefficients are learned instead of handcrafted. A modified version of GA-Net, which uses adaptive cross-entropy loss is currently the top ranked method for stereo matching in the KITTI 2015 stereo benchmark [1].

As neural networks grow more complex, neural architecture search (NAS) is used to find optimal network architectures for a given task. NAS algorithms explore the space of possible network architectures. The application of NAS to end-to-end stereo matching neural networks by Cheng et al. [35] resulted in state-of-the-art disparity estimation performance. Current literature proposes a multitude of network architectures and modifications to further improve disparity estimation.

For the experiments with adversarial attacks in Chapter 4 disparity estimation is realised with GA-Net. Therefore, GA-Net is additionally covered in detail in Chapter 3.1.

2.2 Optical Flow

Optical flow describes the relative motion of objects between an observer and an environment over time. In the context of flow estimation, the projection of three-dimensional scene flow onto the image plane is called optical flow. Additional depth information from disparity estimations in combination with the optical flow estimation enables the expression of three-dimensional scene flow. In the field of computer vision optical flow belongs to the class of correspondence problems. The goal of optical flow is to estimate the movement of objects in a scene using a sequence of images. The key idea is, that the apparent motion of objects in subsequent images is related to their actual motion in the real world. Whereas stereo flow uses multiple cameras at the same point in time, optical flow inputs are image sequences generated by a single camera at different points in time. Each pixel (x, y) of an image is mapped to a displacement vector (u, v) resulting in a two-dimensional vector field. Given two images taken at time $t = 0$ and

$t = 1$, function $f(x, y, t)$ retrieves the pixel value of an image for a certain position (x, y) and point in time t.

In general, methods to solve to the optical flow problem resort to the optical flow constraint (OFC): The intensity consistency assumption states, that corresponding points in both images have the same pixel values:

$$f(x, y, t) = f(x + u, y + v, t + 1). \tag{2.10}$$

The differential formulation of the intensity consistency assumption leads to the OFC: If the displacement vector (u, v) is small and the flow varies smoothly in space one can rewrite Equation (2.10) as linearised optic flow constraint:

$$\begin{aligned} f(x, y, t) - f(x + u, y + v, t + 1) &= 0 \\ &\approx f_x u + f_y v + f_t \\ &= (u, v) \begin{pmatrix} f_x \\ f_y \end{pmatrix} + f_t. \end{aligned} \tag{2.11}$$

The OFC in Equation (2.11) has two unknowns, thus no unique solution. Only the flow component parallel to $(f_x, f_y)^\top$ is considered. Therefore, it is possible to add flow components orthogonal to $(f_x, f_y)^\top$ without violating the constraint. Optical flow methods need to introduce additional assumptions as remedy for the so-called aperture problem.

2.2.1 Local Methods for Optical Flow

Multiple methods that use additional local assumptions exist. The approach of Lucas et al. [36] introduces the assumption, that the optic flow is approximately constant in a ball-shaped neighbourhood B_p with radius p. The energy functional in Equation (2.12) is minimised to adequately fulfill the optic flow constraint in a ball-shaped neighbourhood:

$$E(u, v) = \frac{1}{2} \int_{B_p} (f_x u + f_y v + f_t)^2 \, dx \, dy \,. \tag{2.12}$$

The performance of this approach suffers in regions that violate the local constancy assumption. This assumption is violated if the flow is discontinuous or the motion is non-translatory. These problems prohibit the calculation of dense optical flow fields.

2.2.2 Global Methods for Optical Flow

Horn and Schunck [37] introduce a variational method with more flexible model assumptions than the local methods. The intensity consistency assumption in form of the OFC is the same assumption used in local methods (see Equation (2.11)). The second assumption is a smooth

flow field, because neighbouring pixels are expected to belong to the same object and hence move similarly. The second global constraint tries to maximise the smoothness of the flow field in both directions and is a remedy for the aperture problem. Adding the constraint to the energy functional [37] results in Equation (2.13), where the second term penalises non-smooth flow fields:

$$E(u,v) = \int_{\Omega} ((f_x u + f_y v + f_t)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2))\, dx\, dy\,. \tag{2.13}$$

Schnörr [38] could show that this optimization has a unique solution. The variational model can handle non-translatory motion and achieves dense flow fields due to filling in effects. Flow information from dense regions propagates to flat regions, filling in missing flow information. The variational model of Horn and Schunck [37] can be improved by modifying or appending terms. Bartolini and Piva [39] reduce the weight of the smoothness constraint to preserve sharpness of the flow field at motion boundaries by using median filters.

To gain the advantages of both types of optical flow estimation methods a combination of the local Lucas-Kanade and global Horn and Schunck method was proposed by Bauer et al. [40].

2.2.3 Deep-learning Methods for Optical Flow

The research of recent years identified deep-learning based methods as appropriate technique to solve computer vision problems. For instance object classification by ImageNet [41] or semantic segmentation using convolutional neural networks (CNN) [42, 43] show accurate results. In the same way as stereo flow estimation was improved by deep-learning based methods, current state-of-the-art optical flow methods are based on trained neural networks. End-to-end dense optical flow field estimation networks were first introduced with FlowNet by Dosovitskiy et al. [44] using a classical encoder-decoder architecture [45]. The purpose of the encoder part of the network is to extract feature representations of the input data in form of multiple feature maps while reducing spatial resolution. Encoding is realised by a series of convolutional and pooling layers. The decoder part on the other hand, uses the low-resolution feature representations generated by the encoder as input and increases spatial resolution of the networks final output with transposed convolutional layers. FlowNet showed promising results but could not compete with standard variational methods on real-world data including small displacement vectors. Ilg et al. [46] proposed FlowNet 2.0 to alleviate these issues, combining multiple stacked FlowNet modules and introducing a subnetwork specializing on small motions.

SPyNet [47] uses a coarse-to-fine spatial pyramid structure in which separate optical flow neural networks are trained for each pyramid level. At each level of the pyramid the input resolution is halved, resulting in a decrease of neural network parameters and optical flow displacement vector magnitude. This architecture proposed by Ranjan and Black [47] is simpler than FlowNet and has 96% less model parameters, while keeping up with the flow

estimation accuracy of FlowNet. However, SPyNet performs worse than FlowNet 2.0 indicating a trade-off between model size and accuracy. The inference of optical flow by SPyNet starts at the lowest resolution, where the optic flow estimate is initialized as zero-flow. Consecutively the optical flow estimation of the coarser pyramid level is passed as additional input for the next level that estimates flow at double the resolution.

Sun et al. [48] propose PWCNet which refines the SPyNet architecture by introducing a cost volume layer, which stores the matching cost for each pixel's potential correspondence pixel. Whereas SPyNet directly works on input images, PWCNet uses a learned feature representation created from input images at each layer which further reduces the amount of model parameters. Resolution of the feature representation in PWCNet is halved at each pyramid level in the same way as in SPyNet. PWCNet outperforms SPyNet and FlowNet 2.0 but uses fewer parameters.

Zhang et al. [49] propose Separable Flow to replace the correlation cost volume with a separable cost volume to improve optic flow estimation in occluded areas or areas without texture. Memory requirements and computational complexity are improved by separation and compression of the 4D cost volume into two 3D feature tensors. Guided aggregation layers were identified as valuable improvement for stereo matching networks [34]. Therefore, Zhang et al. [49] introduce semi-global matching realised by semi-global aggregation layers to incorporate non-local information. Finally, the authors include a motion regression module which learns a low-resolution but high-quality flow prediction as initial input to the upscaling module in contrast to zero-flow initialisations of the other methods.

Recurrent All-Pairs Field Transforms (RAFT) by Teed and Deng [50] introduces a network architecture for end-to-end optical flow estimation that is different to FlowNet, SPyNet and PWCNet. The network does not use a pyramidal coarse-to-fine scheme but keeps a single high-resolution flow field which is iteratively refined by an update module. The abandonment of the coarse-to-fine scheme is a remedy for error recovery at coarse resolutions, long training times and the tendency to miss small and fast objects [50]. RAFT consists of three main components: feature and context encoders (1), a correlation volume at different scales (2) and an iterative update operator (3). The iterative update module mimics the behaviour of a first-order optimization algorithm, whose parameters are learned instead of handcrafted. At the time of its publication in 2020 RAFT ranked top for the KITTI 2015 optical flow benchmark [1]. Numerous modified and improved versions of optical flow neural networks based on the RAFT architecture were proposed by different authors, all resulting in competitive benchmark results [51, 52, 53, 54]. As the adversarial attacks in this work are applied on the RAFT based scene flow estimation network RAFT-3D [55], this network is discussed in detail in Section 3.2.

The aforementioned deep-learning based approaches are supervised during learning and rely on labeled ground truth data. Unsupervised learning methods can train end-to-end optical flow neural networks without ground truth data [56, 57]. Zhu et al. [57] proposed an unsupervised learning method which uses a proxy ground truth generated by Flow Fields [58], which is a dense correspondence field approach and not a deep-learning based method. SelFlow [59] is another self-supervised approach for deep-learning based optical flow estimation. Ground

truth data is reliably generated from non-occluded pixels. Liu et al. [59] propose to manipulate this ground truth data with artificial occlusions to enable training of optical flow estimation for occluded image regions.

In summary, since the introduction of FlowNet [44] in 2015 the deep-learning based methods were refined multiple times and enhanced with new innovations to improve performance. Recent deep-learning based methods achieve the highest accuracy for optical flow estimation.

2.3 Scene Flow

Scene Flow methods estimate the three-dimensional motion of objects using images or point clouds of a dynamic scene. In literature methods using stereo image sequences or point cloud data as input are presented. In images the three-dimensional world is projected onto the two-dimensional image plane. Furthermore, images depict the world at a specific point in time. To deduct three-dimensional movement of objects in a scene over time, single images are not sufficient and image pairs or sequences are required. Stereo matching recovers the depth of a scene using a stereo image pair. Thus, position in all three dimensions is recovered. Optical flow reconstructs the motion of a scene using an image sequence. The motion is recovered and projected onto the two-dimensional image plane.

Scene flow estimation with a single camera is an ill-posed problem, that requires additional constraints. A stereo camera system renders the task feasible introducing the required constraints with a sequence of stereo image pairs. In summary, scene flow is the three-dimensional version of optical flow. However, scene flow estimation has similar ambiguities in certain cases that also affect human perception [60]:

1. **Unknown camera motion**
   If the motion of the camera is not known, scene flow estimates only retrieve the motion relative to the camera. The question if stars revolve around the earth or the earth is rotating is an example that human perception is also not able to interpret this ambiguity by simple observation.
2. **Occlusions**
   All points along the optic ray of a camera are projected to the same image pixel resulting in occlusions. Scene flow cannot reliably estimate motion of occluded image regions. Human observers are also not able to detect occluded motions, for instance a person walking behind an object.
3. **Aperture problem**
   For flat image areas with missing structure, the aperture problem mentioned in Section 2.2 persists. For instance, motion in uniform color scenes cannot be detected by humans.

2.3.1 Decoupled Variational Approach for Scene Flow Estimation

As scene flow can be derived from stereo matching and optical flow, Wedel and Cremers [60] propose a decoupled approach, where position (disparity) and motion (optical flow and disparity change) are computed separately. Choosing an optimal technique to solve each sub-problem enables selection of appropriate algorithms for each problem and parallel computing. Nonetheless, both problems are coupled and the consistency between optical flow and disparities has to be ensured. In the following a solution by a variational model [60] is used to outline the scene flow problem in detail.

Scene flow estimation is realised by a combined approach of stereo matching and optical flow estimation. The input to the scene flow is an image sequence of stereo pairs. Consecutive stereo pairs at times $t = 0$ and $t = 1$ are captured by a left and right camera ($i \in 0, 1$), resulting in four images $I_{t,i} \in \{I_{0,0}, I_{0,1}, I_{1,0}, I_{1,1}\}$.

Figure 2.4 illustrates, that the scene flow problem is constrained by two stereo matchings d_0 and d_1 and two optical flows f_l and f_r which couple the four images. Let $I_{0,0}(x, y)$ denote the intensity value of the left image at position (x, y) and time $t = 0$. Likewise, $I_{t,1}(x, y)$ retrieves intensity values for the right image for the specified position and time.

- **Optical flows**
  The optical flow of the left camera using the images $I_{0,0}$ and $I_{1,0}$ is defined as $f_l(x, y) = (u_l, v_l)$.
  The optical flow of the right camera using the images $I_{0,1}$ and $I_{1,1}$ is defined as $f_r(x, y) = (u_r, v_r)$.
- **Stereo matchings**
  The stereo matching at time $t = 0$ using images $I_{0,0}$ and $I_{0,1}$ is defined as d_0.
  Respectively, d_1 is the stereo matching at time $t = 1$ using images $I_{1,0}, I_{1,1}$.
  Note that d_0 is registered to the reference frame $I_{0,0}$, whereas d_1 is registered to $I_{0,1}$.

The four-dimensional scene flow can be uniquely defined using one optical flow and two disparity maps. Let $f_l = (u, v)^\top$ be the left optical flow of the image sequence $I_{0,0}, I_{1,0}$. The first disparity between $I_{0,0}$ and $I_{0,1}$ is defined as $d = d_0$. The second disparity d' cannot be adopted in the same manner using the images at time $t = 1$, because it has to be registered to the same reference image. This also implies, that the disparity change Δd cannot be calculated by subtraction of the separate disparities d_0 and d_1 alone, as d_1 references $I_{0,1}$. Therefore, d' denotes the disparity at $t = 1$ registered to the reference image $I_{0,0}$. In this way both disparities are registered to $I_{0,0}$. The disparity change $\Delta d = d' - d$ can be used to determine d'. The combination of disparity maps d and d' and the left optical flow $f_l = (u, v)^\top$ result in the

four-dimensional scene flow vector field. The scene flow function s maps a pixel to its temporal change of flow and disparity:

$$s(x,y) = \begin{pmatrix} u \\ v \\ d \\ d' \end{pmatrix} = \begin{pmatrix} u_l \\ v_l \\ d_0 \\ \Delta d + d_0 \end{pmatrix}. \tag{2.14}$$

Applying the intensity consistency assumption for stereo matchings and optical flows results in three constraints:

1. **Left Optic Flow Constraint**
   The left flow constraint ensures the intensity consistency assumption for f_l:

   $$E_{f_l} : I_{1,0}(x + u_l, y + v_l) - I_{0,0}(x, y) = 0. \tag{2.15}$$

2. **Right Optic Flow Constraint**
   Stereo rectification (see Section 2.1.2) ensures that corresponding points of the left and right images are aligned on the scan lines in y-direction. Therefore, the x-component of the right image is only shifted by the disparity d_0. The same principle holds for the optical flow, where the u component is shifted by the disparity change Δd, so $u_r = u_l + \Delta d$. This leads to the right optical flow constraint:

   $$E_{f_r} : I_{1,1}(x + d + \Delta d + u_l, y + v_l) - I_{0,1}(x + d, y) = 0. \tag{2.16}$$

3. **Disparity Constraint**
   The disparity constraint ensures the intensity consistency assumption for the stereo pair at $t = 1$:

   $$E_d : I_{1,1}(x + d + \Delta d + u_l, y + v_l) - I_{1,0}(x + u_l, y + v_l) = 0. \tag{2.17}$$

Variational Model for Scene Flow

Similar to the variational optical flow method of Horn and Schunck [37], the linearised versions of these three constraints are used as data term E_D in an energy functional E_{SF} in the method of Wedel and Cremers [60]. A simplified version of the data term is described in Equation (2.18):

$$E_D = E_{l_f} + E_{r_f} + E_d. \tag{2.18}$$

As in the optic flow variational approach, Wedel and Cremers [60] introduce a smoothness term, which penalises non-smooth flow or disparity changes by analysing the magnitude of the

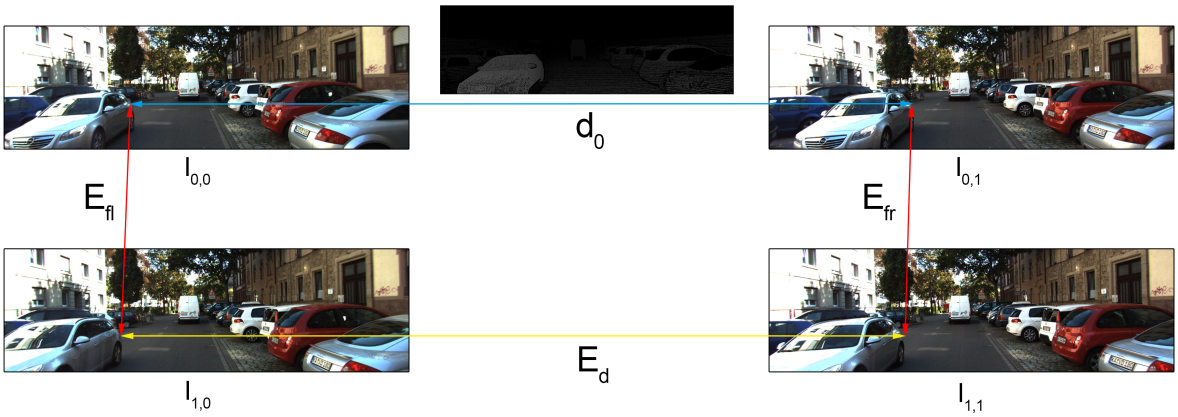


Figure 2.4: The scene flow problem is constrained by the left and right optic flow constraint E_{f_l}, E_{f_r} (red) and the disparity constraint E_d (yellow). The disparity d_0 is given (blue).

spatial gradients. Equation (2.19) shows the weights λ for penalising non-smooth flow and γ for disparity change respectively.

$$E_S = \lambda|\nabla u|^2 + \lambda|\nabla v|^2 + \gamma|\nabla(\Delta d)|^2 \tag{2.19}$$

The final energy functional is minimised in the entire image domain Ω:

$$E_{SF}(u, v, d, d') = \int_\Omega (E_D + E_S)\, dx\, dy\,. \tag{2.20}$$

Minimisation is achieved by computing the Euler-Lagrange equations, which are then solved by a fixed point iteration scheme [61].

2.3.2 3D World Scene Flow

A solved scene flow problem $s(x, y) = (u, v, d, d')^\top$ enables the computation of three-dimensional real world motion vectors $(X', Y', Z')^\top$, which start from the world point (X, Y, Z) at time $t = 0$ [60]. A visualisation of the real-world three-dimensional scene flow vectors is depicted in Figure 2.5. The focal point of the left camera is located at (x_0, y_0), f is the focal length and b the baseline:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \frac{b}{d} \begin{pmatrix} x - x_0 \\ y - y_0 \\ f \end{pmatrix} \tag{2.21}$$

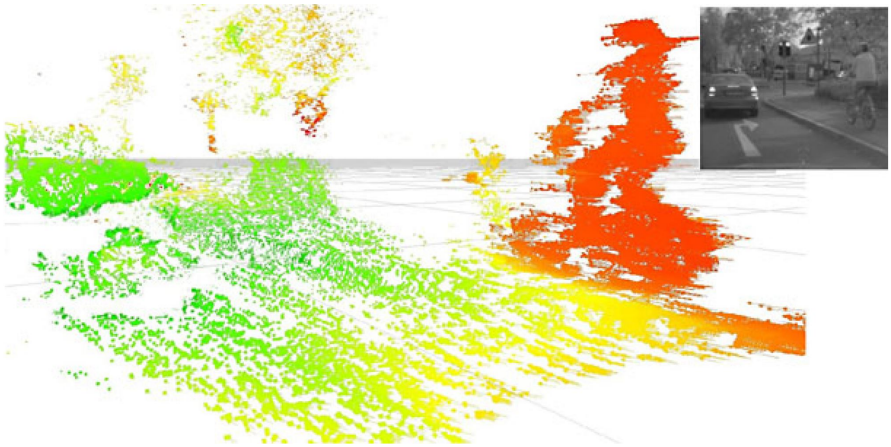


Figure 2.5: Visualisation of scene flow by Wedel and Cremers [60]. The three-dimensional vectors point in the direction of the scene flow. A large displacement vector translates to high velocity of the object (color-coded in red).

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \frac{b}{d + \Delta d} \begin{pmatrix} x - x_0 + u_l \\ y - y_0 + v_l \\ f \end{pmatrix} - \frac{b}{d} \begin{pmatrix} x - x_0 \\ y - y_0 \\ f \end{pmatrix} \tag{2.22}$$

Besides the demonstrated method of Wedel and Cremers [60], different approaches to scene flow estimation exist in literature. In 1999 Vedula et al. [4] coined the term scene flow and propose a method for dense scene flow estimation from optical flow for three different scenarios. The scenarios distinguish between different levels of available stereo information, where either a stereo matching, stereo correspondences or no stereo information is given. Later variational methods use different data terms, for instance Vogel et al. [62] introduce a model, which uses local rigidity instead of smoothness for regularisation to further improve scene flow estimation. The local rigidity constraint assumes that the scene flow in small regions of the scene can be described by a locally rigid motion. Huguet and Devernay [63] propose a variational model, which abstains from linearised constraints to improve performance [61]. This approach couples disparity and flow estimation and optimises $(u, v, d, d')^\top$ collectively, whereas the decoupled method described in Section 2.3.1 first computes d which is then used to optimise $(u, v, d')^\top$. Energy minimisation of variational methods for scene flow can also be realised by multi-resolution pyramid approaches combined with fixed point iteration [63] or primal-dual algorithms [64].

The methods mentioned so far all rely on stereo image sequences, however there exists variational methods that use point cloud data [65, 66]. Point cloud data from the real world can be gathered using the Lidar (*Light detection and ranging*) technology. Lidar infers surface

information by emitting laser beams into the real world and measuring the time it takes for the reflection to reach the sensor.

As outlined in Section 2.2.2 and 2.3.1 it could be shown, that variational models are able to solve the stereo matching and optical flow problem. Hence, the scene flow problem can be solved by creation of variational models that combine approaches from the underlying sub-problems. In literature coupled and decoupled methods for stereo image sequences and point cloud data are available.

2.3.3 Deep-learning based methods for Scene Flow Estimation

In the same way deep-learning based methods for stereo matching and optical flow could show their superiority in contrast to local and variational models, recent deep-learning based scene flow methods score the highest in relevant benchmarks [29, 1, 30]. Mayer et al. [31] introduced SceneFlowNet, the first convolutional neural network for end-to-end scene flow estimation. The authors follow the encoder-decoder architecture of FlowNet [44]. For disparity estimation Mayer et al. [31] introduce DispNet and modify the FlowNet architecture by adding convolutional layers in the decoder to increase the smoothness of disparity maps. In the DispNetCorr architecture the two images of the stereo pair are processed separately for the first two layers, before the resulting features are correlated. This transforms the two-dimensional correlation of images proposed by Dosovitskiy et al. [44] to a one-dimensional correlation of feature vectors, which decreases computational effort and allows the network to cover large disparities. Finally, scene flow prediction is enabled by combination of the modified FlowNet for optical flow and two DispNet modules for disparities at both points in time. The network is trained on the datasets FlyingThings [31] and KITTI [1]. SceneFlowNet showed promising results but could not compete with state-of-the-art variational methods. However, the introduction of the architecture further motivated the use of deep neural networks for scene flow estimation.

Ilg et al. [67] revisited the approach of SceneFlowNet [31] and produced the first competitive results for scene flow estimation by neural networks in the KITTI scene flow benchmark [31]. The proposed architecture is based on FlowNet [44] and FlowNet 2.0 [46]. To compute the disparity change for scene flow, the authors propose to compute the disparity at $t = 1$ and warp this disparity to $t = 0$ to obtain the disparity change.

The aforementioned methods [31, 67] are non-rigid models, which estimate scene flow for each scene point individually. Object-rigid model use semantic segmentation to detect the objects of a scene. Accordingly, a single motion vector is assigned to a rigid object. The scene flow problem is approached with an object-rigid model by Yang and Ramanan [68], who combine the concepts of semantic object segmentation and rigid motion. Depth and optical flow are precomputed using off-the-shelf networks. Remarkable is the limitation to monocular vision as the modular network is trained with only two input images. The two-stream architecture

uses a U-Net module [45] for background segmentation and a CenterNet [69] for instance segmentation. The resulting rigid body motions are fitted to the initial precomputed flow and depth estimates. Compared to other proposed methods, the segmentation priors add meaningful information for scene flow estimation.

Sommer et al. [70] further refine the object-rigid approach. The authors also estimate the scene flow using off-the-shelf networks for optical flow [50] and disparity [35] estimation from a sequence of stereo images. The objective of the proposed iterative algorithm is to estimate a collection of rigid objects including their position and motion. Scene flow estimation by this method uses deep-learning methods to estimate priors which are used by a handcrafted algorithm to estimate scene flow.

RAFT-3D [55] is an end-to-end neural network for scene flow estimation, based on the ideas of RAFT [50]. Instead of an iterative refinement of a two-dimensional optical flow field, a SE(3) rigid body displacement field is optimised for scene flow (see Section 2.1.1 for SE(3)). As previous object-rigid methods showed, information about rigid objects in the scene are advantageous. Therefore, the authors introduce rigid-motion embeddings, which encode the membership of pixels to rigid objects. As RAFT-3D is attacked in this work using adversarial examples, the network is described in detail in Section 3.2.

The standard scene flow problem only considers inputs at two points in time. In real-world applications, such as autonomous driving, more temporal information is available. Multi-frame scene flow methods add additional stereo pairs to the sequence of input images to exploit this information. Besides variational multi-frame methods, Mehl et al. [71] proposed a deep-learning based multi-frame scene flow method that achieves top accuracy rankings [1]. This multi-frame scene flow network trains an improved RAFT-3D [55] version as baseline method. It utilises three stereo pairs at times t_{-1}, t_0, t_1 and uses the improved RAFT-3D for initial estimation of forward ($t_0 \rightarrow t_1$) and backward flow ($t_0 \rightarrow t_{-1}$). Each flow estimate is enriched with rigid motion embeddings and correlation cost volumes and used as input to train a fusion module, which learns the final scene flow estimate ($t_0 \rightarrow t_1$).

Scene flow can be derived from a sequence of stereo pairs or point cloud data. Deep-learning based methods that are based on point clouds are FlowNet3D [72] or PointFlowNet [73]. Both authors could show, that deep-learning based approaches for scene flow estimation using point clouds are able to compete with image based methods. Instead of focusing on only stereo image sequences or Lidar data as network inputs alone, Liu et al. [74] introduce CamLiFlow, a method using synchronized stereo frames and Lidar data as inputs. In this approach the connection between image pixels and Lidar points is learned by a fusion module. The network consists of two connected branches and estimates optical flow and scene flow with state-of-the-art accuracy.

2.4 Adversarial Attacks on Flow Networks

Neural networks are approved solutions for a variety of computer vision tasks, such as object classification, segmentation, stereo matching, optical flow and scene flow [41, 34, 50, 55]. In a mathematical sense, end-to-end neural networks output layers can be interpreted as highly non-linear functions of the input layer. The assessment if this non-linear function is actually artificially intelligent seems difficult, regarding the discovery of adversarial attacks. Szegedy et al. [5] found, that hardly perceptible perturbations to input images can severely impact the accuracy of object classification neural networks for images. The authors introduce a method to train these small perturbations, which are applied to the input images. These modified images are called adversarial examples. Attackers that know how to craft efficient adversarial examples may be able to manipulate machine-learning systems to their advantage in the real world.

While human intelligence succeeds in the classification of adversarial examples, deep neural networks fail in various problem settings [75, 76, 5]. Identifying reasons for this vulnerability is hard, as deep neural networks are complex and operate in a black box manner. It was found, that individual network units do not contribute to the semantics of networks, which means that separate layers are not necessarily responsible for certain semantics. Instead, Szegedy et al. [5] found that the semantics are spread across the entire space of activations. Recent methods try to visualize neural network properties to better understand how and why certain outputs are generated [77]. Simonyan et al. [77] generate class appearance models, which are images that encode all class aspects and output maximal score for a certain object class. Class saliency maps explain for single images, which regions contributed the most to the classification decision. Discovering and explaining neural network semantics as remedy for the black box problem is an open research topic.

The counter-intuitive properties identified by Szegedy et al. [5] led to a multitude of adversarial attacks in various problem domains. These attacks are not limited to noise or synthetic input manipulation but can also be applied in the real world. The adversarial attack of Brown et al. [78] is based on adversarial patches that are placed in the real world in the form of stickers. For instance, a captured image of a banana including this sticker is incorrectly classified as toaster. Real world adversarial examples are not limited to 2D, as 3D-printed objects also contributed to effective adversarial attacks [79]. Simulated real world phenomena include adversarial snow, which was used to attack optical flow networks [80].

2.4.1 Perturbation of Inputs

Let I be the input image for a neural network ϕ, which classifies objects into classes in C. The correct label for I is $c \in C$, while $d \in C$ is an incorrect class label. The adversarial example caused by perturbation δ is then defined as $I + \delta$ leading to an adversarial attack:

$$\begin{aligned} \phi(I) &= c \\ \phi(I + \delta) &= d \end{aligned} \tag{2.23}$$

Besides manipulation of the output, adversarial attacks aim to minimise the perturbation size $|\delta|$, while keeping the perturbed image $I + \delta$ in a valid range of values. In other words, minimal input change should lead to maximal output change of the network. This instability property can be analysed with the Lipschitz constant L. The Lipschitz constant L states, that the distance between two arbitrary function outputs is at most L times the distance of their corresponding inputs $|\delta|$ [81]. Szegedy et al. [5] describe how to analyse a deep neural network ϕ using the Lipschitz constant. The output of each network layer k with weights W_k is represented by the function $\phi_k(I, W_k)$. The upper Lipschitz constant of a layer is then defined as L_k by:

$$\forall I, \delta : ||\phi_k(I, W_k) - \phi_k(I + \delta, W_k)|| \leq L_k ||\delta|| \tag{2.24}$$

The Lipschitz constant L of the entire network ϕ can be obtained by calculating the product of each layers constant L_k:

$$\forall I, \delta : ||\phi(I) - \phi(I + \delta)|| \leq L ||\delta|| \text{ where } L = \prod_{k}^{K} L_k \tag{2.25}$$

In general, small upper bounds of L for a neural network guarantee robustness against adversarial examples [5]. Therefore, Szegedy et al. [5] propose to penalise the upper Lipschitz bounds of each layer during parameter training to defend against adversarial attacks.

2.4.2 Constraints on Perturbations

Perturbation constraints enforce certain limitations on perturbations. Patch-based perturbations limit the shape, location, rotation and scale of a perturbation [78, 6]. For instance, these patch regions can be defined by an origin point and radius. Karmon et al. [82] could show that even small patches that cover at most 2% of the input image area can generate effective adversarial examples. Additionally, perturbed input values have to remain in the valid range of the input domain.

Another important optimisation criteria is the magnitude of perturbations. Ideally, small perturbations which are not perceptible by humans should trigger large output changes. Therefore, a similarity measure between original and perturbed image is required. In practice

the p-norm distance between original and perturbed values L_p is constrained to remain below a certain bound ϵ.

$$||I - (I + \delta)||_p \leq \epsilon \tag{2.26}$$

The L_0-norm counts the number of modified pixels by the perturbation. The L_1-norm is called Manhattan distance and accounts for the total variation of the perturbation:

$$||I||_1 = \sum_{(i,j)} |I(i,j)|. \tag{2.27}$$

The L_2-norm, which is also known as euclidean distance, is an adequate metric to enforce certain perturbation constraints in adversarial attacks [83]:

$$||I||_2 = \sum_{(i,j)} \sqrt{|I(i,j)|^2}. \tag{2.28}$$

The maximum perturbation magnitude between any two pixels is called L_∞-norm. This norm is used in global attacks to ensure all perturbed pixels stay within a specified perturbation range [84]:

$$||I||_\infty = \max(|I(i,j)|). \tag{2.29}$$

2.4.3 Generation of Adversarial Examples

The goal of an adversarial attack is to manipulate the input data I with a perturbation δ to cause the network ϕ output to become incorrect. The specified incorrect network output is called target t. Adversarial examples are generated by minimising a loss function [85]. Carlini et al. [86] formulate an objective function f with:

$$f(I + \delta) \leq 0 \iff \phi(I + \delta) = t \tag{2.30}$$

The resulting Carlini-Wagner (C&W) attack is realised by minimisation of the loss function defined by Equation (2.31). The constant $c > 0$ is empirically chosen to enable simultaneous minimisation of both terms.

$$l(\delta, t) = ||\delta||_p + c \cdot f(I + \delta) \tag{2.31}$$

The output of this loss function becomes minimal if the specified L_p-norm distance between original image and perturbed image is small (1) and the network output matches the target $t = \phi(I + \delta)$ (2). Proposed adversarial attacks in literature use the L_1, L_2 or L_∞-norm [87, 83, 84]. Finally, the loss function is minimized using a solver, such as gradient descent, gradient descent with momentum or Adam [86]. Szegedy et al. [5] introduced a similar method which uses the L_2-norm and the L-BFGS algorithm [88] for minimisation.

The fast gradient sign method (FGSM) uses the gradients of a network to generate adversarial examples [75]. Let ϕ be a trained network with fixed parameters Θ and loss function $J(\Theta, I, y)$, where I is an input image with the ground-truth label y. Then $\nabla_I J$ is the gradient of the network's loss function with respect to the input image. The perturbation δ can then be calculated using Equation (2.32):

$$\delta = \epsilon * \text{sgn}(\nabla_I J(\Theta, I, y)) \tag{2.32}$$

The sign function returns the sign of the gradient with respect to the input data. The scaling factor ϵ is used to limit the perturbations. Additive modification of the input image in this signed direction creates an adversarial example $(I + \delta)$, which maximises the loss:

$$I + \delta = I + \epsilon * \text{sgn}(\nabla_I J(\Theta, I, y)) \tag{2.33}$$

FGSM is a simple method and can be computed fast, however adversarial examples crafted with this method yield a lower attack strength [89]. Kurakin et al. [76] refine the FGSM method with an iterative approach. Instead of a single large step in direction of the maximum loss gradient, multiple small steps are chained while recomputing the gradient using the current adversarial example. This leads to stronger attacks and finer perturbations compared to the non-iterative methods. Additionally, the iterative method does not always select perturbation values in the vicinity of the ϵ-border.

Elastic-Net Attacks (EAD) were proposed by Chen et al. [87]. The authors refine the Carlini-Wagner attack by introducing a regularisation parameter β, which controls the trade-off between the L_1 and L_2 norms. Pixel perturbations that exceed β are shrunken, while pixel deviations smaller β remain unchanged. Consequently, increasing β leads to less L_1 distorted adversarial examples and sparsity of the perturbation. For $\beta = 0$ their attack is equal to the Carlini-Wagner attack.

2.4.4 Attacking Flow Networks

The work of Szegedy et al. [5] demonstrated the effectiveness of adversarial attacks on neural networks which perform object classification in images. However, adversarial attacks are not limited to the object classification problem. Deep-learning based methods perform well for the stereo matching and optical flow problem. Recent research highlighted the vulnerability of neural networks for stereo matching and optical flow estimation to adversarial attacks [6, 7, 8, 9, 10]. For these kinds of computer vision problems the aforementioned approaches were modified to craft adversarial examples.

The network output is not a class label anymore but a scalar or vector-valued image for disparity or optical flow respectively. Therefore, the target of an adversarial attack has to be a

disparity map or optical flow. Let $f(x, y) = (u, v)^\top$ be the optical flow of an image sequence. The negative flow $\tilde{f}$ is defined by inverting the direction of each flow vector:

$$\tilde{f}(x, y) = -\begin{pmatrix} u \\ v \end{pmatrix} \tag{2.34}$$

The zero flow is defined by f_0. The semantic interpretation of f_0 is a static scene.

$$f_0(x, y) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{2.35}$$

Alternatively, the target of an adversarial attack can be a specific but arbitrary target flow $f_t(x, y)$.

For adversarial attacks in the context of optical flow problems three relations between flows are important. These quantitative properties can be used to compare accuracy (1) and robustness (2) of neural networks or the strength of adversarial attacks (3) [8]. Let f_{gt} be the ground-truth flow. Then f is the output of a optical flow neural network using input data which was not attacked. The flow resulting from adversarial example inputs is the adversarial flow f^*, whereas f_t denotes the fixed target flow of this adversarial attack. To compare different networks or adversarial attacks, quality measures based on similarity, such as the mean-squared error MSE or average endpoint-error AEE [90] can be used. Let $m \in \{\text{MSE, AEE}\}$ be one of these measures, where low values correspond to high similarity.

1. **Baseline accuracy**
   A performant deep-learning based flow estimation method achieves high accuracy. The estimated flow and the ground truth flow are similar:

   $$\arg\min_f m(f_{gt}, f) \tag{2.36}$$

2. **Adversarial robustness**
   A robust deep-learning based flow estimation method cannot be manipulated by adversarial examples. Given small perturbations, the adversarial flow and the unattacked flow are still similar:

   $$\arg\min_{f^*} m(f, f^*) \tag{2.37}$$

   The ground truth flow f_{gt} is not considered for the adversarial robustness assessment. No ground truth is known for the perturbed inputs and the baseline (see Equation (2.36)) already measures the flow accuracy.

3. **Attack strength**
   A robust method allows only weak attacks. A strong adversarial attack achieves a high

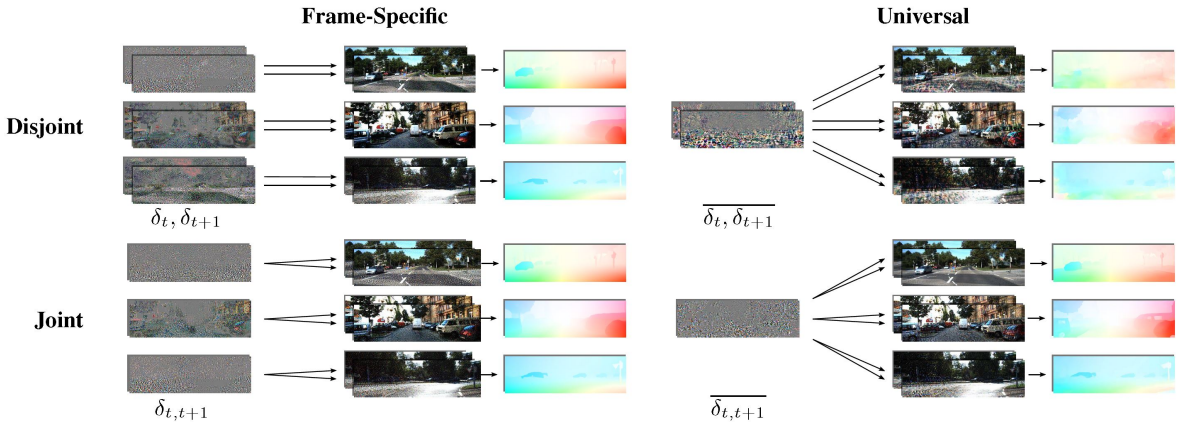


Figure 2.6: Classification of adversarial attack types on optical flow networks of PCFA by Schmalfuss et al. [8].

similarity between adversarial flow and target flow. The attacker seeks to achieve a high attack strength, which translates to high similarity between adversarial and target flow:

$$\arg\min_{f^*} m(f^*, f_t) \tag{2.38}$$

For adversarial attacks on object classification networks a single input image is attacked. In stereo matching and optical flow a pair or sequence of input images can be subject to adversarial attacks. Schmalfuss et al. [8] define different types of attacks that are applied to different subsets of inputs (see Figure 2.6). Frame-specific attacks train perturbations for each input image of an instance of the dataset. Universal attacks apply the same optimised perturbation to all input instances of a dataset processed by a neural network. A single perturbation is trained and applied to both images of a sequence or pair in joint adversarial attacks. In the optical flow case this means the same perturbation δ is trained and applied to both images I_0 and I_1. On the other hand, disjoint attacks train a separate perturbation for all images of a sequence or pair. Frame-specific and universal attacks can be combined with joint or disjoint attacks, resulting in a total of four distinct attack types. Perturbations that are applied to multiple input images have to fulfill more constraints, hence the attack strength is expected to decrease in these attack types.

Ranjan et al. [6] developed the first adversarial attack on optical flow networks. The proposed attack is patch-based and aims to achieve a high attack strength to demonstrate the low adversarial robustness of the optical flow neural networks FlowNetC, FlowNet2.0, SpyNet and PWC-Net [44, 46, 47, 48]. To evaluate the adversarial robustness, the authors use the unattacked flow f instead of the ground truth flow f_{gt}. Using f enables attacking unlabeled data instead of requiring ground truth data. The circular perturbation patch is applied to both images of the sequence with the same rotation and scaling. Therefore, the optical flow in the perturbed region becomes the zero flow. The area of the perturbed patches is limited

to 5% of the image area. Ranjan et al. [6] found that all examined network architectures are vulnerable to adversarial attacks. Neural networks that are based on encoder-decoder architectures (FlowNetC and FlowNet2.0) are more vulnerable to this patch-based attack than spatial pyramid based methods (SPyNet, PWCNet)[6]. Interestingly, conventional methods such as variational models are not affected at all. The real world applicability of this attack could be confirmed by printing adversarial patches on paper and placing them in a scene.

A global attack on optical flow was introduced by Schrodi et al. [7]. The authors explore causes for the vulnerability of optical flow networks and relate the aperture problem to the degradation of flow estimation. Patches with repetitive patterns processed by networks with relatively small receptive fields (FlowNetC [44]) cause the same ambiguities of the aperture problem. Large receptive fields in PWC-Net [48] or RAFT [50] see areas of the image unaffected by the patch attack, increasing their adversarial robustness. Adversarial examples for these frame-specific or universal targeted attacks are generated with the iterative method of FGSM (I-FGSM). Because the number of optimisation steps in I-FGSM is fixed to enforce the perturbation constraint, non-optimal perturbations are generated. Therefore, the attack of Schrodi et al. [7] imposes a weaker constraint on the global perturbation.

The perturbation-constrained flow attack (PCFA) by Schmalfuss et al. [80] aims to generate global perturbations which result in strong attacks with various targets. Optimisation is realised with the L-BFGS algorithm and the perturbation is bounded using the L_2-norm. The authors propose AEE, MSE or cosine similarity to quantify the attack strength. PCFA was able to reveal vulnerabilities of multiple state-of-the-art optical flow networks: FlowNet2.0, SPyNet, PWCNet and RAFT [46, 47, 48, 50]. For attacks with a zero-flow target PCFA can create stronger attacks than I-FGSM while bounding perturbations to the same ϵ. In Section 3.3 this method is explained in detail, as PCFA serves as foundation of the global adversarial attack on scene flow.

As pointed out in Section 2.1.7 the stereo matching problem can be solved precisely if all assumptions hold. However, in most scenarios in the real-world these assumptions are violated at least in some parts of the input images. Therefore, neural networks are trained to learn disparity estimation, which especially enhances performance in these regions.

In general stereo neural networks are robust under Gaussian and uniform noise, but not robust against adversarial attacks [9, 10]. Wong et al. [10] attack three state-of-the-art stereo matching neural networks with global non-targeted frame-specific disjoint adversaries. To generate these adversarial examples FGSM, I-FGSM and MI-FGSM are used. MI-FGSM is a modified version of I-FGSM, which incorporates gradient information from previous steps. Adversarial examples generated with small I-FGSM step sizes cause severe effects on disparity estimation. In flat regions or regions with uniform textures small perturbations are sufficient to fool the neural networks. The adversarial attack aims at increasing flatness, which is possible with small perturbations in these regions. Consequently, non-flat regions require larger perturbations.

A global non-targeted universal disjoint attack on stereo networks is proposed by Berger et al. [9]. The attack optimises a single perturbation tile of 64×64 pixels, which is repeatedly added to the input to cover the entire image. The perturbations are trained separately for the left and right input image, but are universal for the entire KITTI dataset [1]. This way perturbations are not influenced by scene bias (KITTI: sky on top, road on bottom). The resulting perturbations are effective and can be transferred to different networks trained on different datasets. These works demonstrate the vulnerability of stereo networks to adversarial attacks, indicating another attack vector for adversarial attacks on scene flow estimation. Augmenting the training data with adversarial examples is an effective defense against these attacks without causing a loss of accuracy [9, 10].

To the best of the author's knowledge no attacks on scene flow networks have been proposed yet. As scene flow requires solutions to optical flow and disparity estimation, attacks on scene flow can be seen as a form of a combined attack. Depending on network inputs and outputs, different perturbations and targets have to be defined. For instance, RAFT-3D takes two input images and two disparity maps as input data [55]. Therefore, perturbations may have to be trained and applied for images and disparities separately. Moreover, in the case of targeted attacks different definitions of zero flow or negative flow have to be specified for scene flow. This work proposes a method to train a global constrained targeted frame-specific disjoint adversarial attack for scene flow in Chapter 4.

3 Applied Techniques in Detail

In this chapter the deep neural networks which are attacked (RAFT-3D and GA-Net) as well as the principle used for attacking (PCFA) are demonstrated. All relevant properties of these methods are stated, such that the global adversarial attack on scene flow presented in Chapter 4 can be followed easily.

3.1 GA-Net: Guided Aggregation Net

GA-Net [34] is a deep neural network to solve the problem of stereo matching. The input of the network is an RGB stereo pair and the output a disparity map (see Figure 3.1). Note that GA-Net does not rely on extrinsic or intrinsic camera calibration as input but is trained on rectified images (e.g. KITTI). Previous state-of-the-art stereo network architectures are based on multiple 3D-convolutional layers. These 3D convolutions are computationally expensive and require a lot of memory. Therefore, Zhang et al. [34] propose to incorporate semi-global matching (SGM) and local methods for cost aggregation in their network (see Section 2.1.6 for SGM). As the standard SGM and local cost aggregation are not differentiable, training of an end-to-end stereo network with these techniques is not possible. The authors propose new differentiable neural network layers which mimic the behaviour of SGM and local cost aggregation. These layers are called semi-global guided aggregation (SGA) and local guided aggregation (LGA) layer.

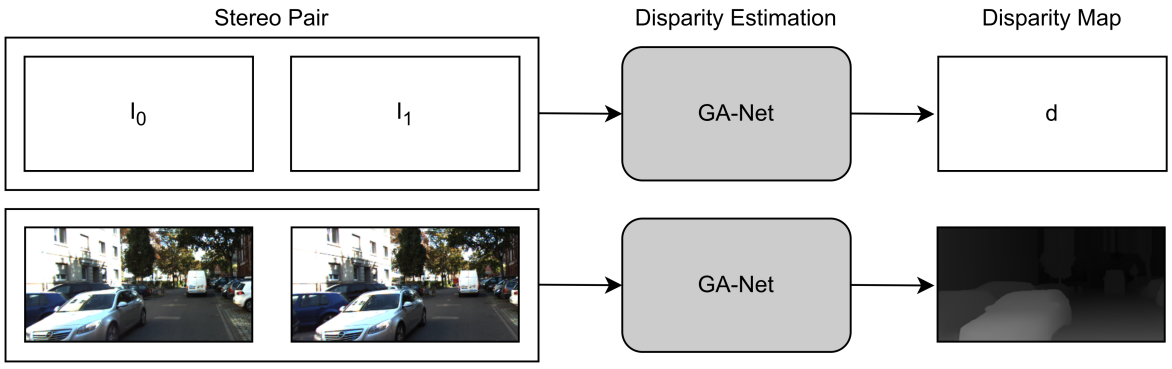


Figure 3.1: GA-Net uses a stereo image pair as input and outputs a disparity map.

3.1.1 Local Guided Aggregation Layer (LGA)

Let $C(p, d)$ be the four-dimensional cost volume $H \times W \times D_{max} \times F$, where H and W are the image dimensions, D_{max} is the maximum displacement disparity and F the length of the feature vector. In deep neural networks, cost filtering of the cost volume is used to refine thin structures and object edges after the downsampling and upsampling of the encoder-decoder architecture. Traditional cost filtering uses a $K \times K$ kernel to filter the cost volume in the neighbourhood N_p. The LGA layer filters the costs of the cost volume correspondingly with a $K \times K \times 3$ kernel, with filter weights for the three disparities d - $1, d, d + 1$. The weights of these filters are learned during training using a subnet consisting of multiple 2D convolutional layers. This subnet uses the left image as input an outputs the filter weights for the LGA layers. In the GA-Net architecture LGA layers are implemented before the softmax layer of the disparity regression.

3.1.2 Semi-Global Guided Aggregation Layer (SGA)

The semi-global guided aggregation layer (SGA) is a differentiable approximation of the semi-global matching algorithm [27]. Semi-global matching improves the disparity estimation, especially in occluded, textureless or reflective regions. Constant penalty parameters P_1 and P_2 are used in semi-global matching (see 2.1.6). The SGM layer learns these penalty parameters during training. Furthermore, the original cost-aggregation along a path in the images is modified. The SGA layers use four cost aggregation path directions: left, right, top and down. The terms are constrained by weights to avoid very large cost values.

3.1.3 Disparity Regression

The disparity regression is used at the end of the GA-Net architecture to infer the disparity map $D(x, y)$ from the cost volume. Each possible disparity value $d' \in \{0, ..., D_{max}\}$ is weighted by its probability. The probability of each disparity for each pixel is calculated with the softmax function σ. The softmax function maps a vector of K real values to a new vector of K real values that sum up to one. Therefore, the new vector can be interpreted as probability distribution of the K elements. For the disparity regression, a slice of the cost volume can be interpreted as a vector with K elements.

Summing up the product of candidate disparity d' and the corresponding probabilities of d' for each pixel p results in the disparity map. Note that the negative cost is used for softmax, as low cost means high probability.

$$D(p) = \sum_{d'=0}^{D_{max}} d' \times \sigma(-C(p, d')) \tag{3.1}$$

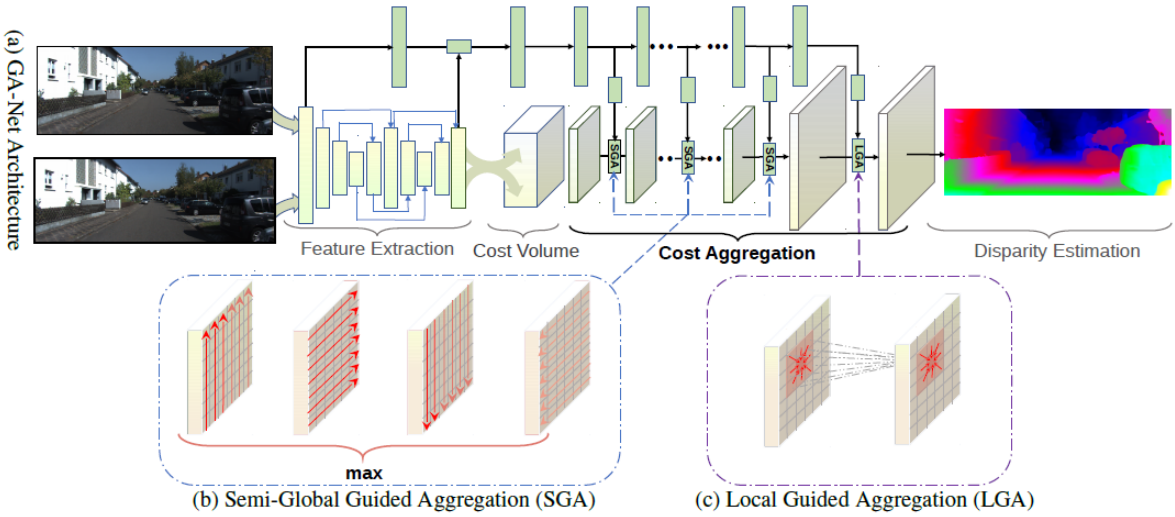


Figure 3.2: GA-Net architecture [34]: Three SGA Layers and one LGA layer are used in the cost aggregation stage of the network. The SGA Layers select the cost of the most expensive path from the four examined directions (b).

3.1.4 Network Architecture

Zhang et al. [34] experiment with different architectures of GA-Net, which consist of combinations of 3D convolutional layers, SGA and LGA layers. These architectures use three SGA layers and one LGA layer for cost-aggregation before the disparity regression as Figure 3.2 illustrates. In this work the architectures GA-Net-11 and GA-Net-15 are used for the adversarial attacks on RAFT-3D. The numbers in the architecture identifier indicate the number of 3D convolutional layers. In comparison to other state-of-the-art methods, GA-Net uses fewer 3D convolutional layers (11 or 15) than GC-Net (19) or PSMNet (25). This results in less parameters, faster inference and training times and higher accuracy due to LGA and SGA layers.

3.1.5 Results

GA-Net currently ranks 56th on the KITTI stereo flow 2015 benchmark with a disparity bad pixel error of 1.88%. The most accurate method submitted to the benchmark is called GA-Net+ADL, which is based on the GA-Net architecture and achieves an error 1.55%. These results show, that GA-Net is a state-of-the-art method for disparity estimation from stereo image pairs.

3.2 RAFT-3D: Scene Flow Estimation

RAFT-3D is a deep neural network architecture for scene flow estimation. It is built upon the key ideas of RAFT, which iteratively updates a 2D motion field to estimate optic flow. The method proposed by Teed and Deng [55] also iteratively updates a motion field T, however for the scene flow problem they extend the concept and use a rigid motion SE(3)-field (see Section 2.1.1). The name recurrent all-pairs field transform 3D (RAFT-3D) refers to the iterative update (recurrent) of an SE(3) motion field using an all-pixel-pairs correlation volume. The SE(3) motion field T has six dimensions, three for translation and three for rotation. An accurate motion field estimates the scene flow in such a way, that T explains the movement of world points between $t = 0$ and $t = 1$.

3.2.1 Inputs and Outputs

The input to the RAFT-3D scene flow processing pipeline illustrated in Figure 3.3 is a stereo pair of RGB images. The disparities d_0 at time $t = 0$ and d_1 at time $t = 1$ are estimated using the stereo network GA-Net-15 (see Section 3.1). The final inputs of RAFT-3D are both left images of the stereo pairs and both disparity maps registered to the left images. The input images are converted from RGB to BGR and normalized using the mean and standard deviation of ImageNet [41]. Disparity values are sampled at $1/8$ resolution and converted to depth using the camera intrinsics. The output of RAFT-3D is the SE3 field T, which induces an optical flow $(u, v)^\top$ and a change in depth. For the first point in time the output disparity d is equal to the input disparity d_0. The change in depth is used to calculate the disparity d' at $t = 1$, which is registered to the same frame as $d_0 = d$ (left image at $t = 0$). The scene flow problem is then estimated by $(u, v, d, d')^\top$, with $\Delta d = d - d'$.

3.2.2 Feature Extraction

RAFT-3D uses two separate feature extraction networks. The first feature extractor uses both left images at $1/8$ resolution with shared weights and outputs a 128-dimensional feature vector. A pretrained ResNet-50 network is used as second feature extraction network (context encoder), which uses $I_{0,0}$ as input. This context encoder retrieves semantic and contextual information about rigid objects in the scene at $1/8$ resolution.

3.2.3 Correlation Pyramid

A 4D correlation volume $C_1 \in \mathbb{R}^{H \times W \times H \times W}$ compares the similarity between all pairs of pixels. It is realised by the dot product of the pixels feature vectors. The last two dimensions of the full resolution correlation volume are pooled with a 2×2 kernel average pooling three times,

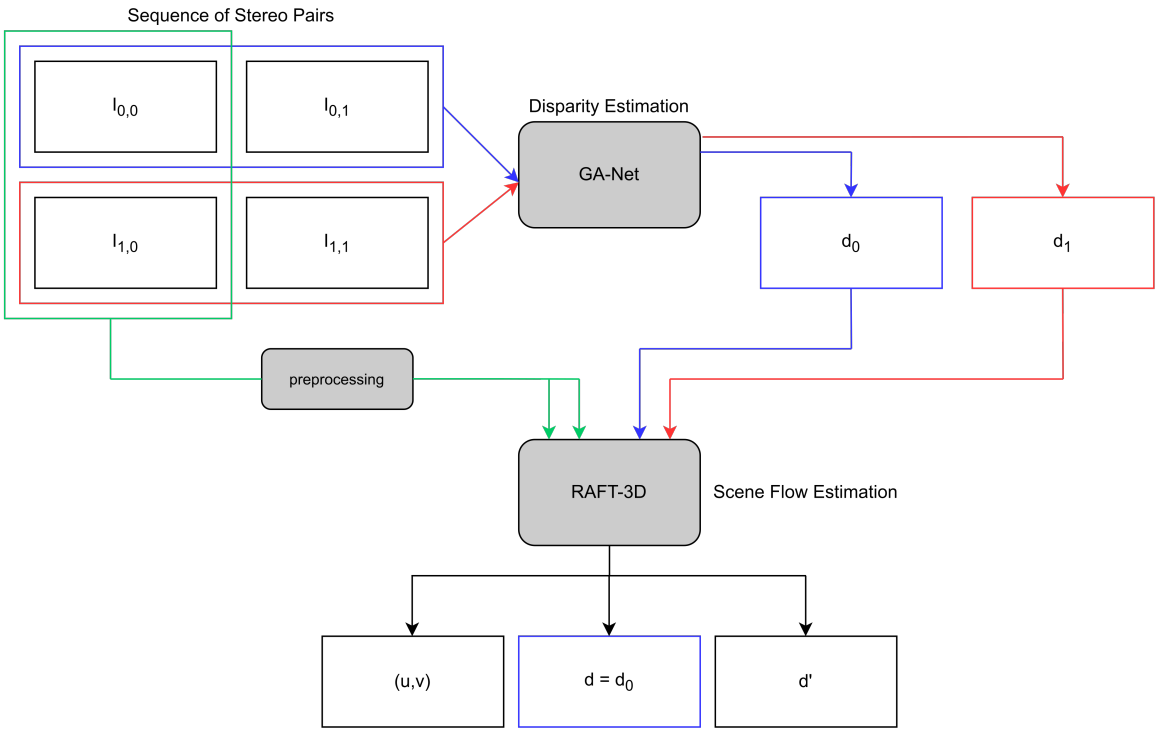


Figure 3.3: Processing pipeline of RAFT-3D: All images of the stereo pairs are used for disparity estimation by GA-Net. The resulting disparities and the preprocessed left images $I_{0,0}, I_{1,0}$ are used as inputs for RAFT-3D for scene flow estimation.

resulting in three additional lower resolution correlation volumes C_2, C_3, C_4. The resulting correlation pyramid is then defined by $C = \{C_1, C_2, C_3, C_4\}$ and indexed by bilinear sampling around the requested pixel.

3.2.4 Rigid Motion Embeddings

The information about rigid objects is a meaningful prior for scene flow estimation. Object detection networks can deliver this additional information to scene flow networks but introduce non-differentiable components. Therefore, Teed and Deng [55] introduce rigid motion embedding vectors, which softly and differentiably group pixels into rigid moving objects. For each pixel a rigid motion embedding vector is created, which encodes the affiliation of this pixel to an object in the scene. Pixels with similar rigid motion embeddings belong to the same object and therefore follow the same SE(3) motion. The rigid motion embeddings are learned unsupervised during RAFT-3D training using features, which include the output from the pretrained ResNet-50 context encoder.

3.2.5 Dense-SE(3) Layer

The projections in RAFT-3D use the pinhole camera model and assume piecewise constant rotation and translation for rigid objects. To transform from homogeneous world coordinates to image coordinates the projective function π in Equation (3.2) is used. Here $d = \frac{1}{Z}$ is the inverse depth, f the focal distance and (c_x, c_y) the principal point of the image plane.

$$\pi(\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}) = \begin{pmatrix} f\frac{X}{Z} + c_x \\ f\frac{Y}{Z} + c_y \\ \frac{1}{Z} \end{pmatrix} = \begin{pmatrix} x \\ y \\ d \end{pmatrix} \tag{3.2}$$

If a dense depth map Z is given, the inverse projection π^{-1} maps from pixel coordinates to 3D world points, with $d = \frac{1}{Z}$:

$$\pi^{-1}(\begin{pmatrix} x \\ y \\ d \end{pmatrix}) = \frac{1}{d}\begin{pmatrix} \frac{x-c_x}{f} \\ \frac{y-c_y}{f} \\ 1 \\ d \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.3}$$

The 3D motion between the stereo frame pair is represented by T, which can be used to construct a function which maps a pixel in $I_{0,0}$ to its corresponding pixel in $I_{1,1}$. The correspondence $(x', y', d')^\top$ for a certain pixel $(x, y, d)^\top$ can be found using the SE(3) motion for this pixel T_p:

$$\pi(T_p \cdot \pi^{-1}(\begin{pmatrix} x \\ y \\ d \end{pmatrix})) = \begin{pmatrix} x' \\ y' \\ d' \end{pmatrix} \tag{3.4}$$

The point is first projected into world space by π^{-1}, then the SE(3) motion T_p can be applied. After application of the motion T_p the point is projected back into image space using π.

The optical flow $(u, v)^\top$ and disparity change Δd can then be calculated using the correspondences:

$$\begin{pmatrix} x' \\ y' \\ d' \end{pmatrix} - \begin{pmatrix} x \\ y \\ d \end{pmatrix} = \begin{pmatrix} u \\ v \\ \Delta d \end{pmatrix} \tag{3.5}$$

The high resolution SE(3) motion field estimated by RAFT-3D is at $1/8$ resolution of the input images. Other encoder-decoder based architectures down- and upsample flow estimates further, which results in loss of details. Optimisation of T is achieved by iterative updates by the Dense-SE(3) layer which uses the update operator. As shown in Equation (3.4), the current estimate

of T can be used to get the 2D correspondences. Accordingly, the recurrent update operator uses the current estimated correspondence to index from the correlation volume. The indexed correlation features are used as input, besides the current flow field and depth residual. The depth residual compares the depth of the current correspondences using the depth map at $t = 1$ and the depth resulting from application of T to the depth map at $t = 0$. The output of the update operator is a a revision map and a rigid motion embedding map, which are used as input for the Dense-SE(3) layer.

The differentiable optimisation Dense-SE(3) layer maps the revision map to a SE(3) field update. It uses a similarity measure of rigid motion embeddings to identify if pixels belong to the same object. The objective function of the Dense-SE(3) layer tries to find the transformation T, which describes the motion of a neighbourhood of pixels, while respecting the affiliation of pixels to rigid objects. The SE(3) field T is consecutively optimised by the update operator and Dense-SE(3) layer twelve times. After the last iteration, T is upsampled to the original image resolution.

3.2.6 Training

The network is trained in an end-to-end manner using the ground truth data f_{gt} for scene flow. As the ground truth data consists of optical flow and disparities (for KITTI), Teed and Deng [55] infer the inverse depth change from the ground truth disparities. To compute the L_1 loss, the optical flow and inverse depth change f_k induced by all k iterative updates of the SE(3)-motion field T_k are compared with the ground truth data. The first iteration T_1 has the lowest loss weight and the weight increases for every consecutive SE(3) field estimate:

$$L = \sum_{k}^{12} 0.9^{12-k} ||f_k - f_{gt}||_1 \tag{3.6}$$

3.2.7 Results

Currently, RAFT-3D ranks eighth in the KITTI scene flow evaluation 2015 benchmark. Three additional methods that use modified versions of RAFT-3D or incorporate the network architecture achieve rank three, four and six. Thus, RAFT-3D is a proven state-of-the art solution for estimating dense and accurate scene flow estimates.

3.3 PCFA: Perturbation-Constrained Flow Attack

The perturbation-constrained flow attack by Schmalfuss et al. [8] generates strong global adversarial examples for the optical flow networks FlowNet 2.0, SPyNet, PWCNet and RAFT

[46, 47, 48, 50]. The attack is optimised for destructibility of flow predictions rather than real world applicability. Given an optical flow network, the attack aims to find perturbations δ_t, δ_{t+1} for both input images which respect three constraints. The first goal is to minimise the proximity $\mathcal{L}$ of the predicted flow f and the target flow f_t:

$$\underset{\delta_t, \delta_{t+1}}{\operatorname{argmin}} \mathcal{L}(f, f_t). \tag{3.7}$$

The perturbation constraint bounds the L_2-norm of the perturbation below a certain bound ϵ:

$$||\delta_t, \delta_{t+1}||_2 \leq \epsilon\sqrt{2IC}. \tag{3.8}$$

To make ϵ independent of the image dimensions the factor IC is used. $I = H \times W$ is the amount of pixels in the input images and C the number of color channels. Because two perturbations are generated, the factor is multiplied by 2. The last constraint ensures, that the perturbed image values stay in the valid color range and can be realised with clipping:

$$I_i + \delta_i \in [0, 1]^{C \times I}, i = t, t + . \tag{3.9}$$

These constraints are used to optimise the perturbation δ by minimising the loss function ϕ:

$$\underset{\delta}{\operatorname{argmin}} \phi(\delta, \mu), \;\; \phi(\delta, \mu) = \mathcal{L}(f, f_t) + \mu \cdot |\max(0, ||\delta_t, \delta_{t+1}||_2 - \epsilon\sqrt{2IC})| \tag{3.10}$$

The first term penalises deviations from the target flow. The second term, weighted by parameter μ linearly penalises perturbations that exceed the perturbation budget $\epsilon\sqrt{2IC}$. For instance, if all pixel perturbations are smaller than ϵ, the penalty term resolves to 0. The minimisation problem in Equation (3.10) is solved by the L-BFGS algorithm [88] and the target flow f_t is set to the zero flow f_0. Schmalfuss et al. [8] identified the mean-squared error (MSE) instead of AEE as an adequate flow proximity function. Compared to the AEE the squared norm of the MSE circumvents undefined derivatives if adversarial and target flow match. The optimisation of perturbations is stopped if the desired proximity to the target flow is achieved or can be bounded by a maximum number of optimisation steps. The authors could also generate effective adversarial examples for other flow targets, such as negative flow or arbitrary flow.

The frame-specific PCFA attack on RAFT [50] with images from the KITTI dataset [1] with a zero target flow achieves an average endpoint error between the zero flow and the attacked flow (attack strength) of 3.76. Finally, Schmalfuss et al. [80] generate joint and universal perturbations for different optical flow networks. The adversarial robustness measure shows, that all optical flow networks can be manipulated by universal perturbations. The AEE between the unattacked flow and the attacked flow using universal perturbations is between 2.25 (SPyNet [47]) and 3.52 (RAFT [50]). The concept of the PCFA attack and its measures for robustness and attack strength are extended to enable attacks on scene flow estimation networks in Chapter 4.

4 A Global Adversarial Attack on Scene Flow

To the best of found knowledge no adversarial attacks on scene flow networks have been proposed in literature yet. In the following chapter a framework is established to enable adversarial attacks on end-to-end scene flow estimation networks. Extensions to existing attack methods on stereo- and optical-flow networks are made to adjust for the new setting. As scene flow estimation is a combination of optical flow and stereo matching, network inputs can be attacked at different stages of the processing pipeline in decoupled approaches. In general the scene flow problem has more inputs than optical flow and stereo matching. Therefore, these multiple inputs as well as intermediate results can be subject to perturbations. To limit perturbation size for adversarial attacks on scene flow the loss functions have to incorporate these different types of perturbations. Furthermore, for targeted attacks new target definitions for scene flow have to be established.

4.1 Targets for Scene Flow

The presented adversarial attacks on stereo flow networks in Chapter 2.4.4 do not define a target. However, the adversarial attacks on optical flow networks by Schrodi et al. [7] and Schmalfuss et al. [80] are targeted. Targeted attacks aim at generating adversarial examples which lead to a certain target output of the attacked network. Proposed target flows include the zero flow or the negative flow. To create a targeted attack for scene flow, the concept of zero flow and negative flow can be extended by including the disparity change. A scene flow is defined by the vector $(u, v, d, d')^\top$. As both disparities d and d' are registered to the same reference frame (e.g. $I_{0,0}$), the disparity change Δd can be calculated by:

$$\Delta d = d - d'. \tag{4.1}$$

Stereo correspondences of objects close to the camera have larger disparities than objects in the background. A negative change in disparity means the disparity increased over time, indicating that the object moved closer to the camera or the depth of the corresponding scene point decreased. Similarly, a positive Δd means the depth of the object increased over time or the object moved further away in Z-direction. A disparity change of zero indicates a static

scene with no movement in Z-direction. The target flow of an adversarial attack on scene flow is defined by:

$$s_t(x,y) = \begin{pmatrix} u_t \\ v_t \\ \Delta d_t \end{pmatrix}. \tag{4.2}$$

Then the zero flow is defined by s_0:

$$s_0(x,y) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{4.3}$$

Consequently, the negative target scene flow can be expressed by $\tilde{s}$:

$$\tilde{s}(x,y) = \begin{pmatrix} -u \\ -v \\ -\Delta d \end{pmatrix}. \tag{4.4}$$

The attack strength metric defined for attacks on optical flow networks can also be applied to the new defined target flow s_t. The disparity change of the attacked flow $(u, v, d, d')^\top$ can be computed, resulting in the attacked scene flow vector $s^* = (u^*, v^*, \Delta d^*)$. To quantify the attack strength standard measures $m \in \{\text{MSE, AEE}\}$ can be applied:

$$\arg\min_{s^*} m(s^*, s_t) = \arg\min_{s^*} m(\begin{pmatrix} u^* \\ v^* \\ \Delta d^* \end{pmatrix}, \begin{pmatrix} u_t \\ v_t \\ \Delta d_t \end{pmatrix}). \tag{4.5}$$

However, it should be noted, that now the three dimensions of the scene flow vector are not composed of the same units. The values for flow (u, v) denote displacement on the image plane in pixels, whereas the Δd component denotes disparity changes. Disparity change is directly related to depth change, however these changes in disparity or depth are not in the same unit as the pixel displacement of the optical flow. Therefore, measures such as MSE or AEE judge the proximity to the target flow differently for flow and disparity change. In practice, it could be shown for real world datasets that values for optical flow and disparity change are in the same order of magnitude (see Table 4.2).

4.2 Perturbations for Scene Flow

The standard inputs to a scene flow problem are two stereo image pairs at time $t = 0$ and $t = 1$, with $i = 0$ representing the left image and $i = 1$ the right image. Then $I_{t,i} \in \{I_{0,0}, I_{0,1}, I_{1,0}, I_{1,1}\}$ describes these four images. A coupled method processes these four input images and includes

the stereo matching problems for both stereo pairs. Moreover, decoupled approaches separate the disparity estimation from the scene flow estimation. For instance, RAFT-3D [55] relies on the disparity estimation of GA-Net [34] for scene flow estimation. This results in a total of six possible inputs which can be perturbed, in contrast to two inputs for adversarial attacks on optical flow or stereo matching:

1. **Images:** Perturbations for all images of both stereo pairs can be trained to attack a scene flow estimation network.
2. **Disparity Maps:** The intermediate results in decoupled approaches for disparity information can be perturbed to perform adversarial attacks.

Depending on the architecture of the scene flow network (coupled or decoupled) and the processed inputs, a subset of these six inputs can be selected for perturbation optimisation. When optimising these perturbations, the different range of values of image and disparity inputs have to be considered for learning rate and perturbation bound selection.

4.2.1 Inequality Constraints on Perturbation Sizes

An efficient adversarial attack for scene flow networks causes a large scene flow error by adding small perturbations to the inputs. Constraints on perturbation size keep the perturbations small. For the scene flow attack the L_2-norm of the perturbations of the perturbed inputs is bounded in the same way as in PCFA. Depending on the selection of attacked inputs, again the different value ranges of input images and input disparities have to be considered. Therefore, additionally to the bound ϵ used by PCFA for images, the bound ϵ_d is introduced to limit perturbations of input disparities. Let $\delta_{i,t}$ be the perturbation for an input image $I_{t,i}$, where $t \in \{0, 1\}$ denotes the chronological order of the stereo pair sequence and $i \in \{0, 1\}$ the position of the image in the stereo pair (left, right). Then the perturbations for attacking all four images with P pixels and C color channels are bounded in the same way as in PCFA:

$$\sum_{i=0}^{1}\sum_{t=0}^{1} ||\delta_{i,t}||_2 \leq \epsilon\sqrt{4PC} \tag{4.6}$$

In theory, arbitrary combinations of subsets of the four input images can be used for adversarial attacks on scene flow. The same concept can be applied to the input disparities. Let θ_0 and θ_1 be the perturbations of the input disparity maps for time $t = 0$ and $t = 1$. The perturbations of both disparity maps of size P_d are then bounded by ϵ_d:

$$\sum_{t=0}^{1} ||\theta_t||_2 \leq \epsilon_d\sqrt{2P_d} \tag{4.7}$$

4.3 Loss Functions for Adversarial Examples

To generate adversarial examples for scene flow networks the perturbations to the image inputs δ and disparity inputs θ are optimised while respecting the inequality constraints. This is achieved by minimisation of a loss function ω in the same way as in PCFA [80]. A subset of the set of network inputs is selected for optimisation of perturbations. The penalty for the perturbation $\tilde{\delta}$ of a single network input of dimensions $P = H \cdot W$ with C color channels and L_2-norm perturbation bound $\tilde{\epsilon}$ can be expressed by the penalty function p:

$$p(\tilde{\delta}, \tilde{\epsilon}) = \max(0, ||\tilde{\delta}||_2 - \tilde{\epsilon}\sqrt{PC}). \tag{4.8}$$

The perturbation constraints for image inputs and disparity map inputs are configured differently, because of the different range of values. Therefore, two weight parameters α and β are introduced, which control the penalty for perturbations of images and disparity maps separately. The penalties of the N image perturbations δ_n and M disparity perturbations θ_m are summed up and multiplied by the corresponding penalty factor. In practice up to four image perturbations are optimised ($N = 4$) and in the decoupled approach both disparity maps are perturbed ($M = 2$). Finally, the sum of the proximity $\mathcal{L}$ between attacked flow s^* and target flow s_t and the penalties of image perturbations and disparity map perturbations result in the loss:

$$\omega = \mathcal{L}(s^*, s_t) + \alpha \cdot \sum_{n=0}^{N} |p(\delta_n, \epsilon)| + \beta \cdot \sum_{m=0}^{M} |p(\theta_m, \epsilon_d)|. \tag{4.9}$$

The target flow s_t is constant, whereas the adversarial flow s^* is the resulting flow after applying the current perturbations δ and θ to the inputs of the attacked network. This loss function enables a multitude of different adversarial attack types on scene flow networks. No disparity maps are required as input for coupled networks. Consequently, no disparity perturbations are required for attacking such a network. In this case $\beta = 0$ disables the penalty for disparity perturbation and the loss function becomes the PCFA loss function with the capability for more than two input images. Furthermore, it is possible to only attack a subset of input images, for instance only images at time $t = 1$ or only from one camera (left, right). Another type of attack only optimises perturbations for the disparity maps, which can be achieved by setting $\alpha = 0$. An unconstrained attack, which only aims at an adversarial flow which resembles the target flow best is possible by setting $\alpha = \beta = 0$.

4.3.1 Optimisation of Perturbations

The procedure to generate adversarial examples for scene flow starts with the selection of a set of attacked inputs and the definition of a desired target flow f_t. In the first step, the original network inputs are used to predict the current scene flow estimation. Because the

scene flow network is end-to-end differentiable an optimiser can use the loss function ω to modify the network inputs, such that the loss is minimised. Again, here the optimiser learning rates for image inputs and disparity inputs have to be configured separately, because of the different range of values. For all attacks in this work the PyTorch implementation of the Adam optimisation algorithm [91] is used. At the start of the optimisation process the proximity to the target flow contributes the most to the loss, while no perturbation penalties are present. Each step of the optimiser modifies the selected network inputs, increasing the perturbation penalties as well as the proximity to s_t. For all following optimiser steps the current perturbed network inputs are used as inputs to predict the updated scene flow estimate. This optimisation process can be stopped if a certain proximity to the target flow has been achieved or a predefined number of optimisation steps were taken.

4.4 Attack Types

One goal of the comparison of the five different attack types is to show which attacked inputs cause high attack strength. Additionally, the effectiveness of the perturbation constraints in place can be checked with the results of an unconstrained attack. The combined adversarial attack on RAFT-3D and GA-Net affects inputs at an earlier stage in the scene flow estimation pipeline. Therefore, this attack type is able to exploit the low adversarial robustness of prior disparity estimation networks.

1. **GSFA: Unconstrained Attack**
   The global scene flow attack (GSFA) on RAFT-3D generates unconstrained perturbations for all RAFT-3D inputs, which are both left images and both disparity maps. Because no perturbation constraints are in place the loss function ω with $\alpha = \beta = 0$ only penalises the proximity to the target flow:

   $$\omega = \text{MSE}(s^*, s_0) \tag{4.10}$$

2. **GSFAC: Constrained Attack**
   Perturbations for both input images and both disparity maps ($n, m \in \{0, 1\}$) are generated in the constrained global scene flow attack (GSFAC). Image perturbations are bounded by ϵ and disparity perturbations by ϵ_d. The penalty factors α and β allow to steer perturbation penalties for input disparities and images separately.

   $$\omega = \text{MSE}(s^*, s_0) + \alpha \cdot \sum_{n=0}^{1} |p(\delta_n, \epsilon)| + \beta \cdot \sum_{m=0}^{1} |p(\theta_m, \epsilon_d)| \tag{4.11}$$

   Once an appropriate ratio between α and β for image and disparity perturbations has been established by experiments, more strict penalty coefficients in GSFAC-strict further limit perturbation size.

3. **GSFAC-I: Perturbation of Input Images**
   For this attack only the images ($n \in \{0, 1\}$) considered by RAFT-3D are attacked. As disparities are untouched, no penalty has to be calculated for disparity map perturbations:

$$\omega = \text{MSE}(s^*, s_0) + \alpha \cdot \sum_{n=0}^{1} |p(\delta_n, \epsilon)| \tag{4.12}$$

4. **GSFAC-D: Perturbation of Disparity Maps**
   The GSFAC-D attack perturbs only both input disparity maps ($m \in \{0, 1\}$) of RAFT-3D. Again, untouched inputs do not have to be considered in the loss function for this attack:

$$\omega = \text{MSE}(s^*, s_0) + \beta \cdot \sum_{m=0}^{1} |p(\theta_m, \epsilon_d)| \tag{4.13}$$

5. **Coupled GSFAC: Simultaneous Attack on RAFT-3D and GA-Net**
   The attack on the coupled scene flow estimation of GA-Net and RAFT-3D generates perturbations for all four images ($n \in \{0, 1, 2, 3\}$) of the sequence of stereo pairs. The perturbations δ_n are optimised for the raw images before preprocessing of RAFT-3D and disparity estimation of GA-Net. Therefore, this attack type can also exploit adversarial weaknesses of the stereo matching network GA-Net. After each optimisation step the disparities d_0 and d_1 are estimated again by GA-Net using the perturbed images. These implicitly attacked disparities are used as input, together with the preprocessed explicitly attacked left images $I_{0,0}$, $I_{1,0}$ for RAFT-3D. No perturbations are optimised for the disparity map inputs, as they are implicitly included by attacking GA-Net:

$$\omega = \text{MSE}(s^*, s_0) + \alpha \cdot \sum_{n=0}^{3} |p(\delta_n, \epsilon)| \tag{4.14}$$

   This attacks optimises perturbations of the images before the preprocessing of RAFT-3D. Therefore, the value range of the RGB images is $[0, 255]$ which has to be considered for learning rate and perturbation bound selection. For the experiments in Section 5.3.4 the GA-Net-11 architecture is used, because a combined computation graph of RAFT-3D and GA-Net-15 exceeds feasible GPU memory requirements. GA-Net-11 uses fewer convolutional layers and therefore requires less memory when differentiating end-to-end.

4.5 Adversarial Attacks on RAFT-3D

The applicability of the global adversarial attack on scene flow is demonstrated on the state-of-the-art network RAFT-3D [55]. As RAFT-3D is a decoupled network it enables the comparison

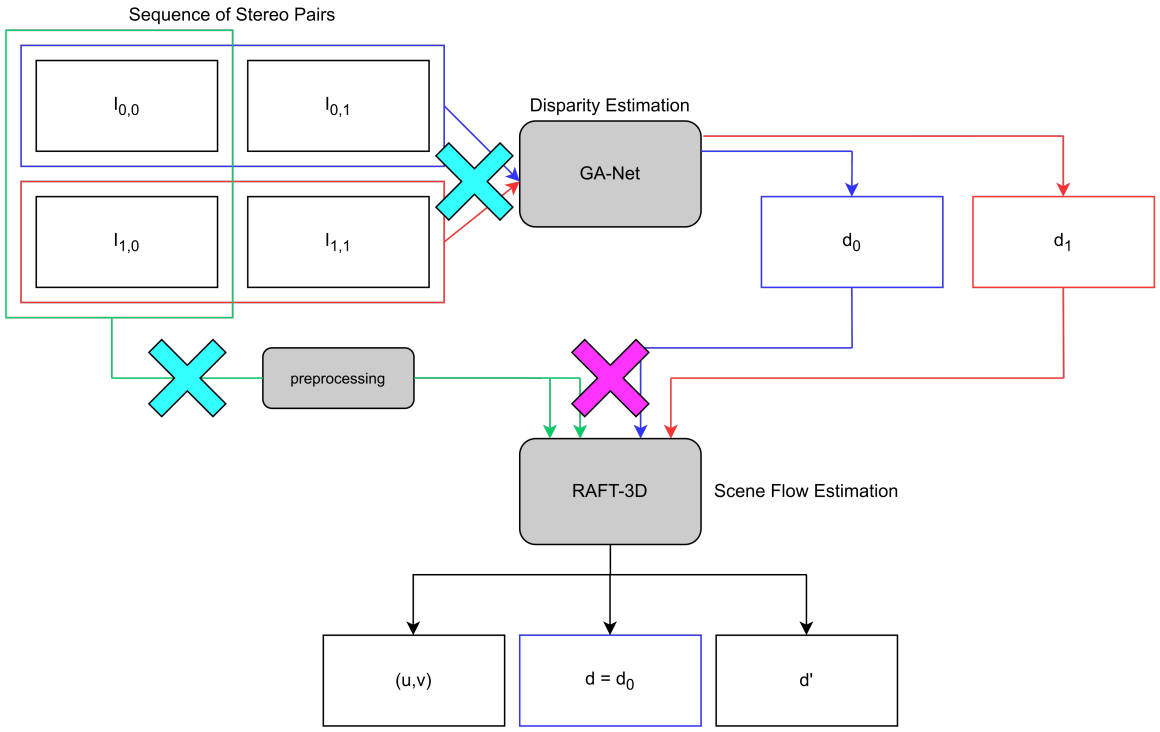


Figure 4.1: Points of attack in the scene flow estimation pipeline: Four attack types (GSFA, GSFAC, GSFAC-I, GSFA-D) on the decoupled scene flow estimation optimise perturbations of the four direct RAFT-3D inputs (cross in magenta). GSFAC-coupled perturbs both stereo pairs before disparity estimation of GA-Net and preprocessing of RAFT-3D (crosses in light blue).

of all proposed attack types shown in Table 4.1. In the following, five types of global scene flow attacks (GSFA) are defined. Four attack types focus on the coupled (preprocessed) inputs of the RAFT-3D scene flow network. The fifth attack type extends the attack to the decoupled disparity estimation by GA-Net [34]. As mentioned in Section 3.2 RAFT-3D uses four inputs, which include both left images $I_{0,0}$, $I_{1,0}$ and both disparity maps d_0, d_1. Furthermore, GA-Net uses both images of a stereo pair for disparity estimation. The input perturbations are optimised before or after the preprocessing step and disparity estimation of RAFT-3D, as shown in Figure 4.1. The zero scene flow s_0 is selected as target flow and MSE is chosen as proximity measure $\mathcal{L}$ for all attacks described in this section. To find adequate values for learning rates and the penalty parameters α and β an experiment based fine-tuning is conducted in Section 5.2. The results of experiments with the five proposed attack types on different datasets are shown in Chapter 5

Attack Type	Networks Attacked		Perturbed Inputs					
	RAFT-3D	GA-Net	$I_{0,0}$	$I_{0,1}$	$I_{1,0}$	$I_{1,1}$	d_0	d_1
GSFA	✓		✓		✓		✓	✓
GSFAC	✓		✓		✓		✓	✓
GSFAC-I	✓		✓		✓			
GSFAC-D	✓						✓	✓
GSFAC coupled	✓	✓	✓	✓	✓	✓		

Table 4.1: Overview of the five attack types. The adversarial attacks optimise perturbations for a subset of all six inputs. The coupled attack extends the adversarial robustness evaluation to the disparity estimation network GA-Net.

	Inputs		Outputs		
	$I_{0,0}, I_{1,0}$	d_0, d_1	u	v	Δd
mean value	-0.42	30.46	4.92	8.79	-3.82

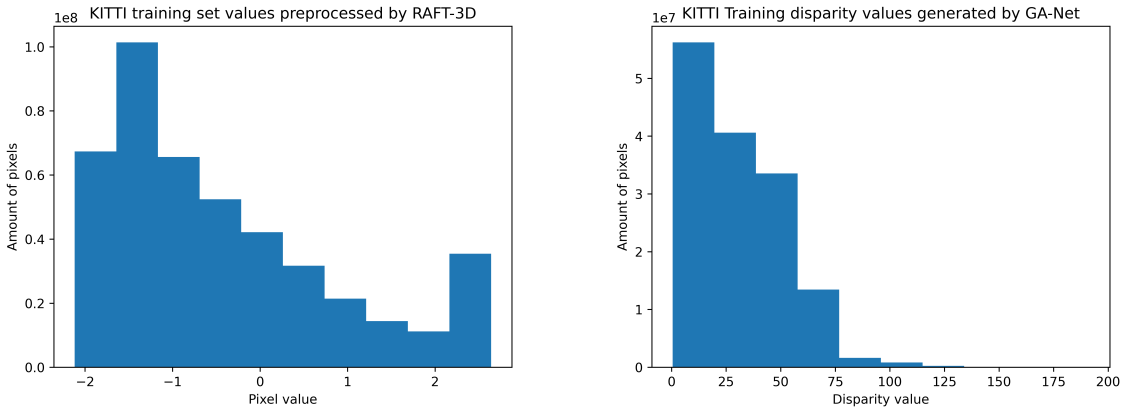
Table 4.2: Comparison of mean values for inputs of and outputs of RAFT-3D on the 200 training images of the KITTI 2015 scene flow benchmark.

4.5.1 Value Ranges

After the preprocessing of the RAFT-3D inputs the value ranges of images and disparity maps are different. The normalisation of images results in RGB pixel values which approach a zero mean, whereas the disparity maps of GA-Net show values from 0 to 150. Analysing all input scenes of the KITTI training datasets results in the histograms shown in Figure 4.2. The normalized input images value range has its mean close to zero. The values of the input disparity maps however are much larger. This has to be considered when selecting the learning rates for the optimiser and perturbation bounds ϵ, ϵ_d. The outputs of RAFT-3D are optical flow (u, v) and disparity change Δd. Both quantities are not in the same unit, however the range of values is in the same order of magnitude, which enables the use of flow proximity measures MSE or AEE to penalise deviations from the target flow. These value distributions are important to select appropriate learning rates and perturbation bounds for images and disparity maps. As disparity values are by an order of magnitude larger than image values, a higher learning rate γ_d and perturbation bound ϵ_d for disparities is selected for the optimiser and loss function.

4.5.2 Disparity Map Perturbations

RAFT-3D relies on the disparity estimation from both stereo pairs by GA-Net. These estimated disparities d_0 and d_1 at times $t = 0$ and $t = 1$ are network inputs for the RAFT-3D model. The disparity map output of GA-Net is upscaled to the full resolution of the input images. However,



(a) Histogram of pixels in preprocessed $I_{0,0}, I_{1,0}$.

(b) Histogram of pixels in d_0, d_1.

Figure 4.2: Comparison of RAFT-3D input value ranges. The preprocessed images $I_{0,0}, I_{1,0}$ show pixel values centered around a mean close to 0. The mean of disparity maps is at 30.46 (see Table 4.2) and by an order of magnitude larger.

RAFT-3D samples the disparity maps at only an $1/8$ of the input resolution. Therefore, when optimising perturbations on full resolution disparity maps, only every eighth pixel is modified by backpropagation, which also keeps the perturbation penalty low. This results in a *dot pattern* in the perturbed disparity maps, as seen in Figure 4.3. Because only every eighth pixel is processed by RAFT-3D and all the other pixels are discarded, it is reasonable to penalise only perturbations of these downsampled disparity maps in the loss function. In the following, all visualisations of disparity maps and their perturbations show the downsampled version used by RAFT-3D.

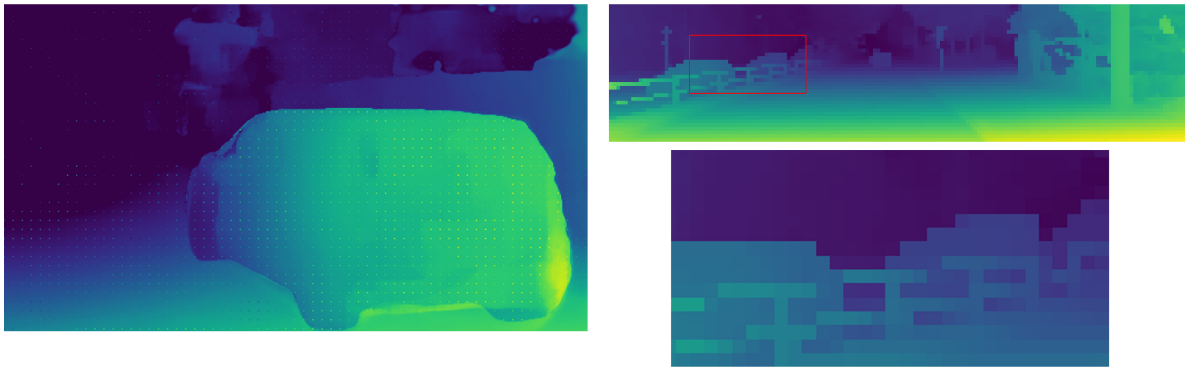


Figure 4.3: Effects of optimising perturbations for full resolution disparity maps: The left frame shows the *dot pattern* in input perturbation d_0, caused by the RAFT-3D downsampling. The contrast of the disparity map is enhanced to make the *dot pattern* visible. On the right side on top a downsampled disparity map at $1/8$ resolution is shown. All pixels in this downsampled disparity map are used for scene flow estimation by RAFT-3D. The bottom right frame shows the region encircled in red. The low resolution of the downsampled disparity maps is visible.

4.6 Implementation Details

All types of the global scene flow attack are implemented in the Python programming language. For the framework a virtual environment with Python version $3.7.9$ is created. The CUDA toolkit version 11.3 enables GPU computing in this environment. All experiments can be run on a single Nvidia RTX A6000 GPU with 48 GB memory. The GSFAC-coupled attack type has the highest memory requirement with up to 42 GB, because the loss computation graph is built across the two neural networks GA-Net and RAFT-3D. In other attack types only RAFT-3D is part of the adversarial attack, which reduces memory requirements to 8 GB.

Deep neural networks are loaded and deployed with the PyTorch $1.12.1$ machine learning framework. The code for RAFT-3D is publicly available at https://github.com/princeton-vl/RAFT-3D [55]. RAFT-3D depends on the lietorch package [92] available at https://github.com/princeton-vl/lietorch, which extends the tensor concept from PyTorch for SE(3) motion fields. In PyTorch tensors are multidimensional matrices of scalar values, whereas lietorch adds support for multidimensional matrices of SE(3) motion elements to PyTorch. The authors provide downloads for pretrained models, including a standard model used for GSFAC-I on the Spring dataset. The fine-tuned RAFT-3D model for KITTI with bilaplacian smoothing is also available for download and used in the other attacks. The source to build GA-Net [34] and multiple pretrained models are available at https://github.com/feihuzhang/GANet.

The GSFAC and GSFAC-coupled procedures are described in pseudocode in Algorithm 4.1 and 4.2. All attacks search for the best set of perturbations found during 400 optimisation steps

(see Section 5.2.1). The current best set of perturbations is saved in the *best* variable, which is initialized with the value ∞. If an optimisation step leads to a lower loss, the improved set of perturbations is saved to the *best* variable. In GSFAC-coupled the disparity estimation of GA-Net is repeated after every perturbation of the pair of stereo images (inside the while loop). In the other attack types the disparity is estimated one-time before the optimisation of perturbations begins. Therefore, GSFAC-coupled requires 7 seconds per optimisation step, whereas the other attack types require less than 1.5 seconds. The runtime to optimise all perturbations of the stereo pairs for the 200 KITTI scenes in the GSFAC-coupled attack is about one week. For other attack types, such as GSFAC, the runtime for the KITTI dataset is about one day. These runtimes only consider sequential computation and can be sped up by parallel processing of KITTI input instances using multiple GPUs.

There is a Python script for GSFAC-coupled and another script for all other attack types. A JSON file which contains all parameters for the scene flow attack can be passed as argument to this script to execute the adversarial attack. In the JSON config file the set of attacked inputs is defined. The config file specifies which dataset is used (KITTI or Spring) and which indices of these datasets are used. Additionally, the learning rates γ and γ_d, perturbation bounds ϵ and ϵ_d and the penalty weights α, β are specified in this file. After all specified dataset indices are processed, the effect of the attack is automatically evaluated and reports metrics regarding scene flow estimation accuracy and perturbation size.

Algorithmus 4.1 GSFAC

procedure GSFAC($I_{0,0}, I_{0,1}, I_{1,0}, I_{1,1}$)
  $d_0 \leftarrow$ GANet15($I_{0,0}, I_{0,1}$) // GA-Net estimate disparity at $t = 0$
  $d_1 \leftarrow$ GANet15($I_{1,0}, I_{1,1}$) // GA-Net estimate disparity at $t = 1$
  $I_{0,0}, I_{1,0} \leftarrow$ preprocess($I_{0,0}, I_{1,0}$) // RAFT-3D preprocessing
  $I'_{0,0}, I'_{1,0}, d'_0, d'_1 \leftarrow I_{0,0}, I_{1,0}, d_0, d_1$ // Remember unperturbed inputs
  $s_t \leftarrow s_0$ // select zero scene flow as target

  $\gamma \leftarrow 0.01$ // Setup optimiser for perturbations
  $\gamma_d \leftarrow 0.05$
  $opt \leftarrow$ Adam($I_{0,0}, I_{1,0}, d_0, d_1, \gamma, \gamma_d$)

  steps $\leftarrow 0$
  best $\leftarrow \infty$
  while steps ≤ 400 **do**
    $s^* \leftarrow$ RAFT3D($I_{0,0}, I_{1,0}, d_0, d_1$) // RAFT-3D estimate adversarial scene flow

    $\delta_{0,0} = I'_{0,0} - I_{0,0}$ // Get perturbations
    $\delta_{1,0} = I'_{1,0} - I_{1,0}$
    $\theta_0 = d'_0 - d_0$
    $\theta_1 = d'_1 - d_1$

    loss $\leftarrow \omega(s^*, s_t, \delta_{0,0}, \delta_{1,0}, \theta_0, \theta_1, \alpha, \epsilon, \beta, \epsilon_d)$
    if loss $<$ best **then**
      best $\leftarrow$ loss
      $\delta^*_{0,0} = \delta_{0,0}$
      $\delta^*_{1,0} = \delta_{1,0}$
      $\theta^*_0 = \theta_0$
      $\theta^*_1 = \theta_1$
    end if

    loss.backward()
    $I_{0,0}, I_{1,0}, d_0, d_1 \leftarrow opt$.step() // Update perturbations
  end while
  return $\delta^*_{0,0}, \delta^*_{1,0}, \theta^*_0, \theta^*_1$
end procedure

Algorithmus 4.2 coupled GSFAC

procedure GSFAC-COUPLED($I_{0,0}, I_{0,1}, I_{1,0}, I_{1,1}$)
  $I'_{0,0}, I'_{1,0}, I'_{1,0}, I'_{1,1} \leftarrow I_{0,0}, I_{0,1}, I_{1,0}, I_{1,1}$ // Remember unperturbed inputs
  $s_t \leftarrow s_0$ // select zero scene flow as target

  $\gamma \leftarrow 0.5$ // Setup optimiser for perturbations
  $opt \leftarrow \mathsf{Adam}(I_{0,0}, I_{0,1}, I_{1,0}, I_{1,1}, \gamma)$

  steps $\leftarrow 0$
  best $\leftarrow \infty$
  while steps ≤ 400 **do**
    $d_0 \leftarrow \mathsf{GANet11}(I_{0,0}, I_{0,1})$ // GA-Net estimate disparity at $t = 0$
    $d_1 \leftarrow \mathsf{GANet11}(I_{1,0}, I_{1,1})$ // GA-Net estimate disparity at $t = 1$
    $s^* \leftarrow \mathsf{RAFT3D}(I_{0,0}, I_{1,0}, d_0, d_1)$ // RAFT-3D estimate adversarial scene flow

    $\delta_{0,0} = I'_{0,0} - I_{0,0}$ // Get perturbations
    $\delta_{0,1} = I'_{0,1} - I_{0,1}$
    $\delta_{1,0} = I'_{1,0} - I_{1,0}$
    $\delta_{1,1} = I'_{1,1} - I_{1,1}$

    loss $\leftarrow \omega(s^*, s_t, \delta_{0,0}, \delta_{0,1}, \delta_{1,0}, \delta_{1,1}, \alpha, \epsilon)$
    if loss $<$ best **then**
      best $\leftarrow$ loss
      $\delta^*_{0,0} = \delta_{0,0}$
      $\delta^*_{0,1} = \delta_{0,1}$
      $\delta^*_{1,0} = \delta_{1,0}$
      $\delta^*_{1,1} = \delta_{1,1}$
    end if

    loss.backward()
    $I_{0,0}, I_{0,1}, I_{1,0}, I_{1,1} \leftarrow opt.\mathsf{step}()$ // Update perturbations
  end while
  return $\delta^*_{0,0}, \delta^*_{0,1}, \delta^*_{1,0}, \delta^*_{1,1}$
end procedure

5 Experiments and Results

In this chapter the proposed global adversarial attacks on scene flow are applied to the state-of-the art scene flow network RAFT-3D [55] using inputs from the real-world KITTI scene flow 2015 benchmark [1]. For the adversarial attack procedure up to six parameter values have to be selected manually. For image and disparity map perturbation generation three parameters each have to be configured: the penalty term weight, the perturbation bound and the learning rate. Therefore, in Section 5.2 experiments with various parameter combinations for penalty term weights and learning rates are conducted to identify reasonable configurations. Once efficient parameters have been identified, experiments with the five different attack types are discussed in the following sections. In addition to the KITTI scene flow 2015 benchmark, a novel synthetic scene flow benchmark created from sequences of the animated short film Spring is used to evaluate the adversarial robustness of RAFT-3D (see Section 5.1). The attack strength of the different attack types as well as the effects on perturbation sizes are compared at the end of this chapter.

5.1 Benchmarks for Flow Problems

There exist various benchmarks to evaluate the performance of methods for flow problems. These benchmark consist of input data and labeled ground truth output data for the specific flow problem. To compare how close the estimated flow or disparity is to the ground truth different measures are used in literature. In the following the relevant measures for this work are presented. The benchmark datasets KITTI and Spring, which are used for testing the adversarial attacks defined in Chapter 4 are introduced. The impact on scene flow estimation accuracy of an adversarial attack can be analysed by comparing the scene flow estimation of a neural network before and after perturbation of inputs.

5.1.1 Error Metrics for Scene Flow

To compare how accurate a predicted flow f of size $H \times W$ matches the ground truth flow f_{gt} the endpoints or angles of the flow vectors can be compared [90]. The angular error AE

is defined as the angle between the predicted flow vector $(u, v)^\top$ and the ground truth flow $(u_{gt}, v_{gt})^\top$ in three-dimensional space:

$$AE(\begin{pmatrix} u \\ v \end{pmatrix}, \begin{pmatrix} u_{gt} \\ v_{gt} \end{pmatrix}) = \arccos(\frac{1 + u \cdot u_{gt} + v \cdot v_{gt}}{\sqrt{1 + u^2 + v^2} \cdot \sqrt{1 + u_{gt}^2 + v_{gt}^2}}). \tag{5.1}$$

The endpoint error EE is calculated by the euclidean distance of the endpoints of the two flow vectors:

$$EE(\begin{pmatrix} u \\ v \end{pmatrix}, \begin{pmatrix} u_{gt} \\ v_{gt} \end{pmatrix}) = \sqrt{(u_{gt} - u)^2 + (v_{gt} - v)^2}. \tag{5.2}$$

These measures both determine the accuracy of a single predicted vector. To evaluate an entire flow field, the average of the selected measure $m \in \{AE, EE\}$ can be computed:

$$\frac{1}{W \cdot H} \sum_{y=0}^{H} \sum_{x=0}^{W} m(f(y, x), f_{gt}(y, x)). \tag{5.3}$$

The averages are called average angular error AAE and average endpoint error AEE.

In practice stereo matching is preceded by rectification, resulting in 1D disparity maps of size $H \times W$. Therefore, the predicted disparity map d and the ground truth d_{gt} are both scalar valued. The average error ω of the prediction d is then quantified by the average absolute error:

$$\omega(d, d_{gt}) = \frac{1}{W \cdot H} \sum_{y=0}^{H} \sum_{x=0}^{W} |d(y, x) - d_{gt}(y, x)| \tag{5.4}$$

5.1.2 KITTI Scene Flow 2015 Benchmark

KITTI Scene Flow 2015 [1] is a benchmark for scene flow estimation methods. The real-world scenes comprise traffic situations captured from the point of view of a passenger car. The car is equipped with a color stereo rig with a baseline of 54 cm. The benchmark consists of 200 training and 200 test scenes with the ground truth data provided by a laser scanner and a GPS system. Each scene consists of two synchronized and rectified stereo pairs. Each stereo pair consists of two RGB color images with known intrinsics, resulting in a total of four images per scene. The resolution of the color images is 0.5 megapixel. The goal of the benchmark is to accurately estimate the scene flow $(u, v, d, d')^\top$.

Menze et al. [1] introduce a new per-pixel metric, which is labeled as *bad pixel error* (BP). A pixel is considered *good* if the disparity or flow estimation endpoint distance is less than 3 pixels to the ground truth. For large displacements the endpoint error is allowed to exceed the

distance of 3 pixels, but has to remain smaller than 5% of the displacement vector magnitude or disparity value:

$$\text{BP}(x,y) = \begin{cases} 1 \text{ if } (EE(f(x,y), f_{gt}(x,y)) > 3 \wedge \frac{EE(f(x,y), f_{gt}(x,y))}{f_{gt}(x,y)} > 0.05) \\ 0 \text{ otherwise} \end{cases} . \tag{5.5}$$

The ratio of bad pixels of an estimated flow vector field f of size $H \times W$ can then be computed by:

$$\overline{\text{BP}}(f, f_{gt}) = \frac{1}{H \cdot W} \sum_{y=0}^{H} \sum_{x=0}^{W} BP(x,y). \tag{5.6}$$

The KITTI ground-truth data includes binary image masks which encode invalid or occluded regions. Therefore, the *bad pixel error* is only evaluated for included pixels for the corresponding benchmark. The KITTI Scene Flow 2015 benchmark distinguishes three categories: foreground and background regions and their combination (all valid pixels). The $D1$ and $D2$-error is the ratio of bad pixels for the disparity estimations d and d'. The flow error Fl is the ratio of bad pixels for the predicted optical flow $f = (u, v)^\top$. The scene flow error is the ratio of scene flow outlier pixels. In scene flow a pixel is considered erroneous if any of the $D1$, $D2$ or Fl metrics report a bad pixel.

5.1.3 Spring Scene Flow Benchmark

The Spring scene flow dataset is a large, high-resolution synthetic benchmark for scene flow, optical flow and stereo. Scene data is extracted from the open-source Blender movie Spring using a virtual stereo camera rig. The image resolution is 1920×1080 and ground truth disparity and optical flow data is available in UHD (3840×2160). The entire dataset consists of 6000 stereo image pairs from 47 scenes. The ground truth data is available for different directions, e.g. forward or backward optical flow.

For experiments in this work the entirety of the 6000 scenes of the dataset is too extensive. Therefore, a subset called *Spring200* is generated. In this subset up to six randomly selected stereo image pairs for each of the 47 scenes are chosen, resulting in 200 stereo image pairs. For these selected scenes the disparity from left to right image of the stereo pair is used as ground truth disparity. In the same way the forward optical flow from the left images is used, to match the input and output setting established by RAFT-3D. For evaluation of scene flow, the disparities and optical flow ground truth data is downsampled to 1920×1080 to match the input stereo image resolution. With these adjustments the same evaluation procedure used for experiments with the KITTI scene flow benchmark is feasible. The experiments with GSFAC-I (see Section 5.3.3) use this subset to evaluate the effectiveness of the global constrained scene flow adversarial attack on input images of RAFT-3D.

5.2 Parameter Refinement

The global perturbation constrained adversarial attack on scene flow uses the loss function ω and an optimiser to find efficient adversarial examples. Because disparity maps and images have a different range of values, two separate penalty terms with separate perturbation bounds ϵ, ϵ_d are used. Consequently, two penalty term weights α, β are used for two reasons. First, in a homogeneous attack the contribution to the total loss of image perturbation penalty term and disparity map perturbation penalty term should be similar. Secondly, two penalty terms and weights enable attacks which allow different perturbation sizes on a specific set of inputs.

The following parameter experiments use the GSFAC attack type, which perturbs the direct RAFT-3D inputs (preprocessed images and disparity maps). A subset of $n = 20$ randomly selected images from KITTI scene flow 2015 training is used for the experiments. The perturbation bounds are fixed to $\epsilon = 0.01$ for images and $\epsilon_d = 0.1$ for disparity maps, as disparity maps show higher values in general (see Table 4.2). A parameter selection which generates adversarial examples with high attack strength and low perturbation size is favored. Therefore, for each parameter setting the KITTI bad pixel error and the perturbation size is analysed. Here, the root-mean-square error (RMSE) between the original and perturbed image is calculated to measure the perturbation size. The proposed GSFAC procedure in Algorithm 4.1 is able to stop optimisation of perturbations after a predefined number of steps. For the following experiments, the best perturbations found in at most 400 steps are selected as final perturbations.

5.2.1 Penalty Weights of GSFAC

To examine the effect of the α, β penalty weights on the corresponding perturbations, the set of parameter values $V = \{0, 10, 20, 50, 100, 500, 1000\}$ with $|V| = 7$ is defined. This results in 49 different combined parameter assignments for $\alpha, \beta \in V$. In addition to the fixed perturbation bounds, the learning rates for images and disparity maps are fixed to $\gamma = \gamma_d = 0.01$ for this step of the parameter refinement. Experiments could show that in 400 optimisation steps sufficiently large perturbations could be generated with these learning rates.

Table 5.1 shows a subset of the results of the experiments with different α, β penalty weights. The results of all 49 experiments are added to the Appendix A.1 of this work. The first row shows the baseline performance of RAFT-3D on the $n = 20$ subset of the KITTI training split. The bad pixel error for scene flow is 1.2%, for optical flow 1% and for disparities d, d' smaller than 0.5%. As the baseline shows no adversarial attack, the perturbation size is zero for all inputs.

The results of the unconstrained GSFA wit $\alpha, \beta = 0$ are shown as another baseline. The GSFA is effective and leads to the highest bad-pixel error percentages, with 96.6% bad pixels for scene flow. As perturbation size is unconstrained this high attack strength comes with the largest perturbation sizes. In general the range of values for disparity maps is higher, therefore the

	Parameters				KITTI Bad-Pixel Error				Perturbation Size RMSE			
Experiment	α	ϵ	β	ϵ_d	Scene Flow	Optical Flow	d	d'	$I_{0,0}$	$I_{1,0}$	d_0	d_1
RAFT-3D	-	-	-	-	1.2%	1.0%	0.4%	0.5%	0	0	0	0
GSFA	0	-	0	-	96.6%	91.8%	81.6%	86.1%	0.23	0.22	6.93	2.70
43	1000	0.01	10	0.1	91.9%	88.2%	67.0%	71.4%	0.12	0.09	4.74	2.05
44	1000	0.01	20	0.1	92.1%	86.5%	67.8%	72.0%	0.11	0.09	4.44	1.91
37	500	0.01	20	0.1	91.8%	86.7%	64.8%	70.7%	0.13	0.10	4.89	2.10
45	1000	0.01	50	0.1	83.4%	78.3%	43.3%	53.2%	0.10	0.08	3.08	1.49
29	100	0.01	10	0.1	**93.4%**	**90.2%**	**69.7%**	**75.6%**	0.16	0.14	5.23	2.21
48	1000	0.01	1000	0.1	21.0%	20.8%	0.5%	4.5%	**0.02**	**0.02**	**0.34**	**0.26**

Table 5.1: Comparison of a subset of the 49 experiments testing different penalty weights α for images and β for disparities. Highest error and lowest perturbation sizes marked in bold.

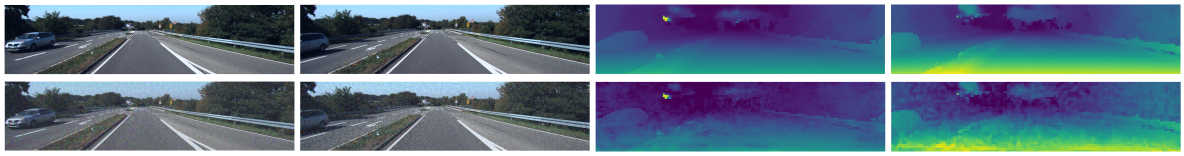
corresponding perturbation size RMSE is also higher than for images. Additionally, as disparity map values are higher, the resulting penalties from perturbations are higher as well. Therefore, to achieve a balanced penalty contribution of all three terms of the loss function ω, the penalty weight β has to be chosen smaller than α. For all experiments perturbation size for inputs of time step $t = 0$, which are $I_{0,0}$ and d_0, are higher than for time step $t = 1$ ($I_{1,0}$ and d_1).

The selection of experiments shown in Table 5.1 is based on effective attacks with low perturbation sizes. The experiments $43, 44, 37, 45$ and 29 all show the same characteristics:

1. High bad-pixel errors for scene flow and optical flow ($> 75\%$).
2. Medium to high bad-pixel errors for disparity outputs d and d' ($> 50\%$). Note that for RAFT-3D the input d_0 is equal to the output d. This means any increase in error is caused by the perturbation alone.
3. Effective limitation of perturbation sizes compared to the unconstrained attack GSFA.
4. High weight α to penalise perturbations of the images.
5. Low weight β for disparity perturbation penalty compared to α.

The experiment 48 has the most strict penalties for perturbations with $\alpha = \beta = 1000$. While this is effective to limit the perturbation RMSE to 0.02 for images and 0.34 for disparity maps, the bad-pixel error for flows is only around 21%. Compared to the other experiments, these strict perturbation penalties lead to an adversarial attack, which is unable to completely destroy scene flow estimation.

In general all the 49 experiments disrupt the accurate scene flow estimation of RAFT-3D (see Figure 5.1). However, these experiments show the obvious trade-off between attack strength and limitations on perturbation size. A high attack strength requires larger perturbations of inputs. In reverse conclusion, strict perturbation penalties lead to small perturbations of inputs



(a) Top row: Original inputs for RAFT-3D ($I_{0,0}, I_{1,0}, d_0, d_1$) from left to right. Bottom row: Perturbed inputs.

(b) Top row: Optical flow (left) and disparity change (right) estimated by RAFT-3D with original inputs. Bottom row: Scene Flow Estimation using the perturbed inputs.

Figure 5.1: Experiment 37: The perturbation penalty terms use the factors $\alpha = 500$ and $\beta = 20$. The perturbations of the disparities are more perceptible than the image perturbations. These perturbations lead to a scene flow estimation close to the target flow s_0.

which result in low attack strength. The results of experiment 43 with $\alpha = 1000$ and $\beta = 10$ proved reasonable in terms of attack strength and perturbation size. Therefore, the $\alpha : \beta$ ratio of $1:100$ is used for the remaining experiments in this work.

5.2.2 Learning Rates for GSFAC

For the experiments the optimisation of perturbations is bounded by the runtime, which is the number of optimisation steps. In general, a carefully selected learning rate enables the machine learning algorithm to find good optima. If the learning rate is too high, local optima can be skipped. An optimisation procedure where the learning rate is too small may approach a minimum of the loss function slowly. The goal of this step of the parameter refinement is to find suitable learning rates for the Adam optimiser. The PyTorch implementation of Adam is able to apply different learning rates to specific subsets of the learned parameters. Therefore, the learning rate γ is used for perturbation of RAFT-3D input images and γ_d for the perturbation of disparity maps. Again, as disparity map values are higher than preprocessed image values, γ_d has to be larger than γ.

The set K of candidate learning rates is defined as

$$\gamma, \gamma_d \in K = \{0.01, 0.05, 0.1, 0.25, 0.5, 1.0\}. \tag{5.7}$$

	Parameters		KITTI Bad-Pixel Error				Perturbation Size RMSE				Best Perturbation	
Experiment	γ	γ_d	Scene Flow	Optical Flow	d	d'	$I_{0,0}$	$I_{1,0}$	d_0	d_1	Steps	Loss
RAFT-3D	-	-	1.2%	1.0%	0.4%	0.5%	0	0	0	0	-	-
GSFA	0.01	0.01	96.6%	91.8%	81.6%	86.1%	0.23	0.22	6.93	2.70	376	11.6
6	0.05	0.01	89.8%	89.5%	1.5%	56.6%	0.14	0.08	**0.79**	**0.35**	286	**25.1**
0	0.01	0.01	90.7%	89.5%	6.1%	60.1%	**0.11**	**0.07**	1.05	0.50	331	34.1
1	0.01	0.05	93.5%	89.3%	69.1%	77.0%	**0.11**	**0.07**	4.66	1.78	275	42.1
7	0.05	0.05	93.0%	89.6%	56.0%	74.6%	0.15	0.09	3.52	1.19	205	43.9
8	0.05	0.1	**96.7%**	**89.8%**	**89.2%**	**91.5%**	0.14	0.08	6.51	2.16	223	55.2
30	1.0	0.1	69.9%	69.7%	7.5%	45.2%	0.71	0.57	1.03	0.36	176	422.3

Table 5.2: Comparison of a subset of the 36 experiments testing different learning rates γ for perturbation of images and γ_d for perturbation of disparity maps. Highest error and lowest perturbation sizes marked in bold.

All possible assignments of these values to the learning rates result in 36 experiments. For all experiments the average number of steps to find the best set of perturbation is 175, which justifies the runtime bound of 400 steps. The perturbation bounds are fixed to $\epsilon = 0.01, \epsilon_d = 0.1$ and the penalty weights set to the values $\alpha = 1000, \beta = 10$ identified in Section 5.2.1.

Table 5.2 shows the results of a subset of the experiments. In the Appendix A.2 the results of all experiments are available. In three of these experiments high learning rates for disparity perturbation ($y_d >= 0.5$) result in *NaN* values contained in the RAFT-3D output optical flow or disparity maps. These experiments cannot be evaluated because of the erroneous outputs and are excluded from the selection. The next to last column in Table 5.2 shows the average number of steps the optimiser takes to generate the best perturbation according to the loss function. Only for the unconstrained GSFA the number approaches the maximum number of 400 optimisation steps. The experiments require less than 400 steps on average to generate the best perturbations. The last column shows the mean loss function output for each of the best set of perturbation for the 20 RAFT-3D input instances.

The selected experiments $0, 1, 6, 7$ and 8 highlight the effects of specific learning rate parameters. The smaller learning rates in K are sufficient to generate effective perturbations in 400 optimisation steps. As expected, small learning rates lead to smaller perturbation sizes, while the trade-off between attack strength and perturbation size persists.

In general low learning rates generate effective adversarial examples. The outcome of experiment 1 with $\gamma = 0.01$ and $\gamma_d = 0.05$ shows a balanced performance regarding attack strength and perturbation size. The bad-pixel errors for all outputs is high, and the perturbation sizes for images is low. The optimisation procedure of experiment 1 finds the best perturbation after an average of 275 steps with a mean loss of 42.1. The perturbation of d in experiment 1 also leads to a higher bad-pixel error of d' than experiments that use a γ_d smaller than 0.05. Therefore, the learning rates $\gamma = 0.01$ for image perturbations and $\gamma = 0.05$ for disparity perturbations are effective and used in further experiments.

The parameter refinement showed, that in general all parameters obey the trade-off between attack strength and perturbation size. Extreme parameter selections lead to low attack strength or high perturbation sizes or cause RAFT-3D to generate erroneous outputs. In conclusion the parameter configuration with $\epsilon = 0.01$, $\gamma = 0.01$, $\alpha = 1000$ for image perturbations and $\epsilon_d = 0.1$, $\gamma_d = 0.05$, $\beta = 10$ for disparity map perturbations proved to be effective. The average contribution of each penalty term to the loss function ω with this parameter selection is balanced. The proximity to target flow term on average contributes 37% to the total loss. The perturbation penalty terms contribute on average 24% (images) and 39% (disparity) to the total loss ω.

5.3 Global Scene Flow Attack on RAFT-3D

The effectiveness of the five proposed global adversarial attack types on scene flow is tested on the KITTI 2015 scene flow benchmark, which consists of 200 images and includes the ground truth scene flow. The bad-pixel error for the scene flow estimation of RAFT-3D on this data is 1.33% (see Table 5.3). The KITTI bad-pixel error metric is non-linear and relatively strict, hence it can reach high percentage values even if the AEE of the estimated flow is only slightly off. Additionally, because of different value ranges the perturbation sizes of images and disparity maps cannot be compared directly. A remedy for both of these problems is the introduction of a linear and value range independent measure of estimation performance. The relative squared error compares the estimation performance of a model to another model which simply uses the mean value of the ground truth data for every estimation [93]. The ratio of the errors of these models is called the relative squared error RSE. An RSE of 0% means the examined model provides an exact estimation of the ground truth data. On the other hand, high values up to 100% indicate an inaccurate model. Let y be the ground truth data with mean $\overline{y}$ and $\tilde{y}$ the predictions of the model:

$$RSE(\tilde{y}, y) = \frac{\sum_{i=1}^{n}(y_i - \tilde{y_i})^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{5.8}$$

The RSE can be used as secondary metric to assess the accuracy of the flow estimation besides the KITTI bad-pixel error. On the KITTI test dataset RAFT-3D achieves RSE values of 0.4% for optical flow prediction and 0.07% and 0.11% for disparity estimations d, d'. To evaluate the perturbation size of adversarial examples, the RSE of the original input and the perturbed input can be calculated. Because the RSE is value range independent this enables a direct comparison of image and disparity map perturbation sizes.

The RSE is an error measure which does not respect characteristics of the human perception. Therefore, the structural similarity index SSIM is added as a second measure to evaluate perturbation sizes. The SSIM is a model based on human perception and used to measure the similarity of two images. The range of SSIM is in the interval $[0, 1]$, where 1 denotes identical images. Zanforlin et al. [94] assigned labels based on human-perception to certain SSIM value

	KITTI Bad-Pixel Error				RSE		
	Scene Flow	Optical Flow	d	d'	Optical Flow	d	d'
RAFT-3D	1.33%	1.19%	0.52%	0.68%	0.41%	0.07%	0.11%
GSFA	89.41%	85.23%	67.69%	76.41%	96.11%	34.5%	32.69%

Table 5.3: Results of scene flow estimation for RAFT-3D and the unconstrained adversarial attack GSFA on the $n = 200$ KITTI scenes.

ranges and found that SSIM values greater than 0.95 indicate imperceptible perturbations. The SSIM metric can also be expressed as percentage value.

To analyse the qualitative effects of perturbations on scene flow prediction, distinct example input scenes of the KITTI benchmark are selected. In general, in scenes where the car is stationary the length of most scene flow vectors is small or zero, thus the actual scene flow is close the target flow s_0. Therefore, the perturbation-induced flow matches the target flow s_0 more easily in stationary scenes. In scenes where the car is driving, the average length of scene flow vectors is higher, which increases the distance to the zero target scene flow. To demonstrate the effects of the global adversarial attacks, figures depicting stationary and driving scenes are selected to highlight effects on scene flow estimation and perturbation size.

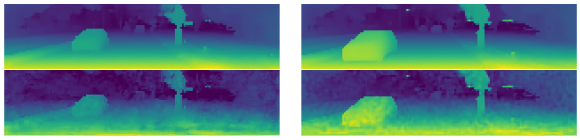
5.3.1 GSFA: Unconstrained Attack

The global scene flow attack without constraints on perturbation size achieves a high attack strength. The adversarial flow approaches the target scene flow s_0 in all the 200 scenes of KITTI training without exceptions. Figure 5.2 shows the input perturbations and the estimated scene flow of RAFT-3D (optical flow and disparity change) for KITTI scene 38. For GSFA the learning rates are set to $\gamma = 0.01$ and $\gamma_d = 0.05$ and without the perturbation penalty terms the loss function becomes:

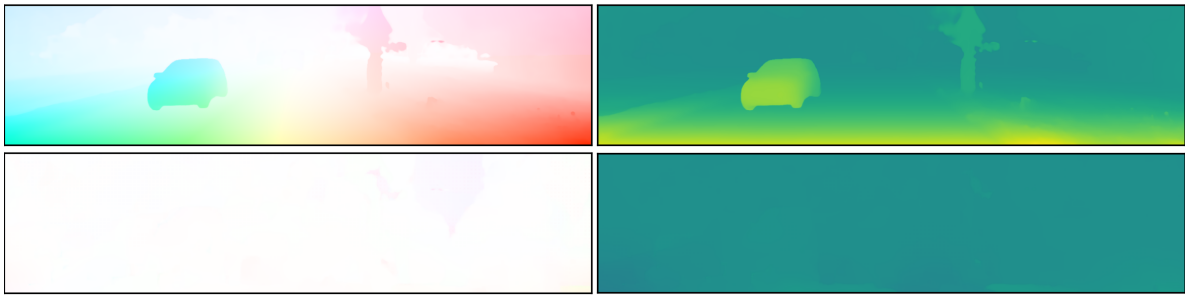
$$\omega = \text{MSE}(s^*, s_0) \tag{5.9}$$

The perturbation size is implicitly bounded by the learning rate and maximum number of 400 optimisation steps. However, the resulting perturbation size is large and the perturbations on both input types are perceptible by human observers (see Figure 5.3). The average KITTI bad-pixel error of RAFT-3D for the 200 training scenes increases from 1.33% to 89.4%, which makes the unconstrained attack the most destructive. This is expected, as there is a trade-off between attack strength and perturbation size. The following GSFAC experiments aim at generating imperceptible perturbations while maintaining a high attack strength.

(a) Input images $I_{0,0}, I_{1,0}$ (top) and their perturbed versions (bottom).

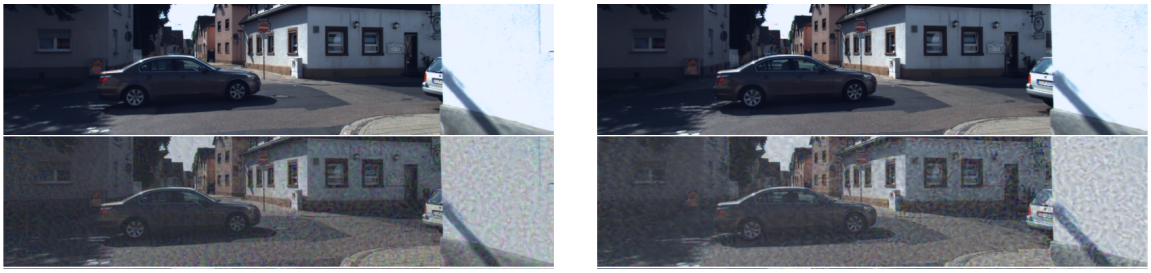


(b) Input disparity maps d_0, d_1 (top) and their perturbed versions (bottom).

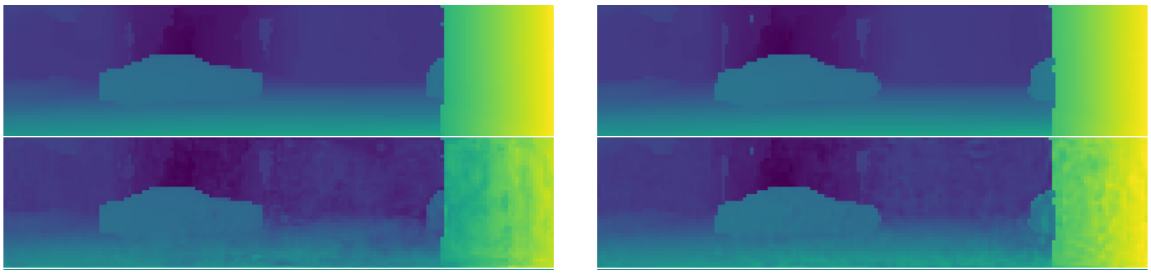


(c) Top: RAFT-3D estimated scene flow. Bottom: the flow and disparity change resulting from the adversarial examples. The target flow s_0 is matched closely.

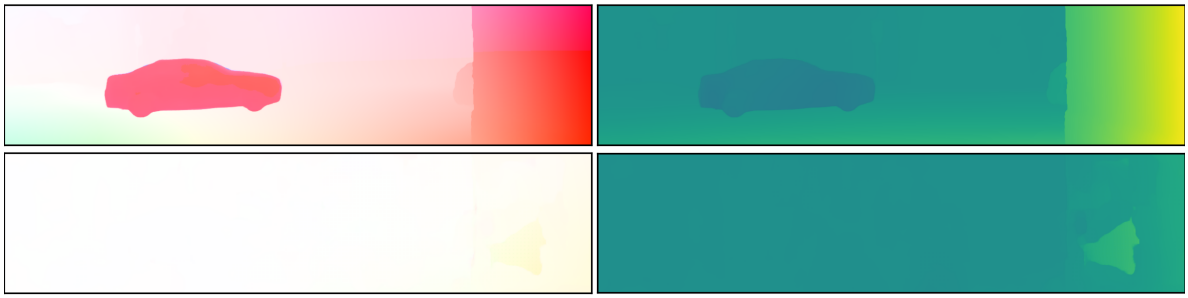
Figure 5.2: GSFA attack on KITTI scene 38.



(a) Input images $I_{0,0}, I_{1,0}$ (top) and their perturbed versions (bottom).



(b) Input disparity maps d_0, d_1 (top) and their perturbed versions (bottom).



(c) Scene flow estimations of RAFT-3D (top) and after input perturbation using GSFA (bottom).

Figure 5.3: GSFA attack on KITTI scene 86.

5.3.2 GSFAC: Perturbation Constrained Attack

The learning rates and penalty weight ratio $1:100$ found in the parameter refinement is used in the experiments with the perturbation constrained scene flow attack GSFAC. Perturbations are generated for both input images $I_{0,0}, I_{1,0}$ and both disparity maps d_0, d_1. The first set of relaxed constraints use the penalty term weights $\alpha = 10^3, \beta = 10$. The learning rates are set to $\gamma = 0.01$ and $\gamma_d = 0.05$. The perturbation bounds $\epsilon = 0.01$ and $\epsilon_d = 0.1$ stay the same as in the parameter refinement. The final loss function of GSFAC with relaxed constraints is shown in Equation (5.10):

$$\omega(\delta_n, \theta_m) = \text{MSE}(s^*, s_0) + 10^3 \cdot \sum_{n=0}^{2} |p(\delta_n, 0.01)| + 10 \cdot \sum_{m=0}^{2} |p(\theta_m, 0.1)|. \tag{5.10}$$

A second experiment *GSFAC-strict* with higher penalty term factors is realised with $\alpha = 10^4, \beta = 10^2$:

$$\omega(\delta_n, \theta_m) = \text{MSE}(s^*, s_0) + 10^4 \cdot \sum_{n=0}^{2} |p(\delta_n, 0.01)| + 10^2 \cdot \sum_{m=0}^{2} |p(\theta_m, 0.1)|. \tag{5.11}$$

Table 5.4 shows the results of both adversarial attacks. The relaxed constraints effectively decrease the perturbation size for the input images $I_{0,0}, I_{1,0}$. The disparity perturbations are not limited effectively by GSFAC, but GSFAC-strict reduces this perturbation size using an increased penalty weight $\beta = 100$. The bad-pixel error and the RSE for all outputs still increase substantially after GSFAC compared to the unattacked RAFT-3D flow estimation. The experiment *GSFAC-strict* also effectively disturbs scene flow estimation. However, the output disparity d with a bad-pixel error of 6.6% is not affected as much as in GSFAC (61%). The increase in bad-pixel error for d is caused by the perturbation alone, as in RAFT-3D the input disparity d_0 is equal to the output d. This means, that the perturbations of GSFAC-strict keep disparity values in the vicinity of the 3 pixel threshold of the KITTI bad pixel error. The RSE values, which compare estimated flow and ground truth flow, confirm the bad-pixel error data.

Figure 5.4 shows the effects of increased penalty weights on perturbations. The perceptibility of the perturbations decreases with higher penalty weights, however the image perturbations remain perceptible for humans. Disparity map perturbations in *GSFAC-strict* are imperceptible and hardly perceptible for GSFAC. The resulting attacked flow in Figure 5.4 is heavily impaired, even by strict perturbation penalty weights. The car depicted in the scene vanishes almost completely from the optical flow and disparity change predicted by RAFT-3D for inputs perturbed with GSFAC or GSFAC-strict.

There exist particular scenes in the KITTI training set, which are more robust against GSFAC and GSFAC-strict (see Figure 5.5 and Figure 5.6). As perturbation constraints become stricter, the global attack becomes a patch-like attack. The noisy pattern in the perturbations δ_0, δ_1 in Figure 5.5 is concentrated on the left image region where the driving van is located. Other

(a) Perturbed $I_{0,0}$ by GSFA.

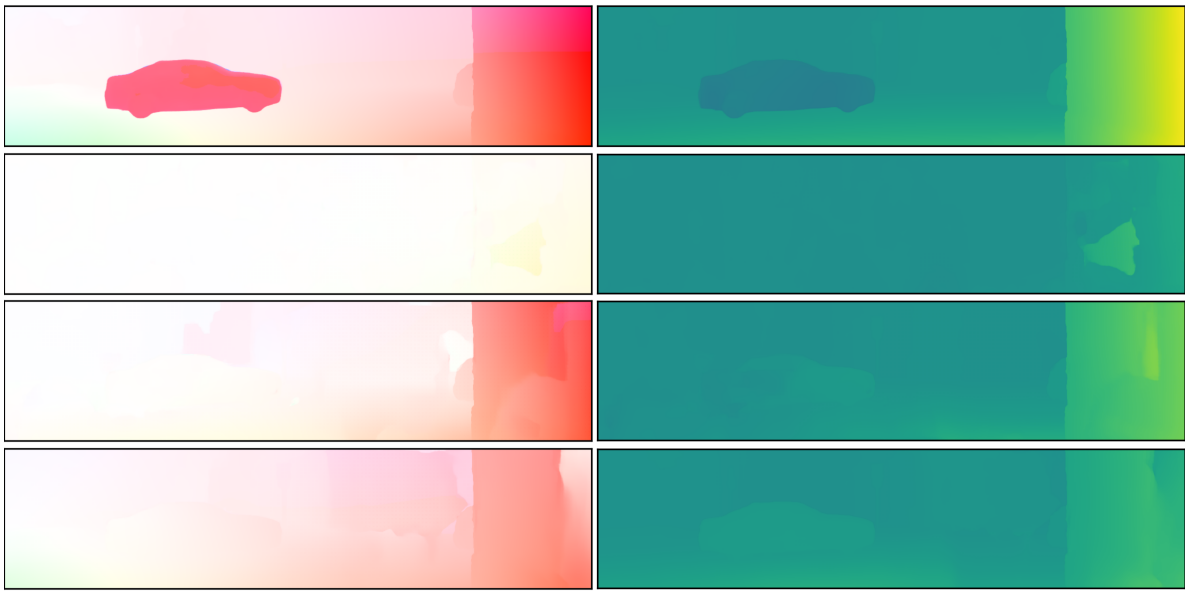
(b) Perturbed d_0 by GSFA.

(c) Perturbed $I_{0,0}$ by GSFAC.

(d) Perturbed d_0 by GSFAC.

(e) Perturbed $I_{0,0}$ by GSFAC-strict.

(f) Perturbed d_0 by GSFAC-strict.



(g) Scene flow estimations ((u, v) left, Δd right) with RAFT-3D and after GSFA, GSFAC, GSFAC-strict (from top to bottom).

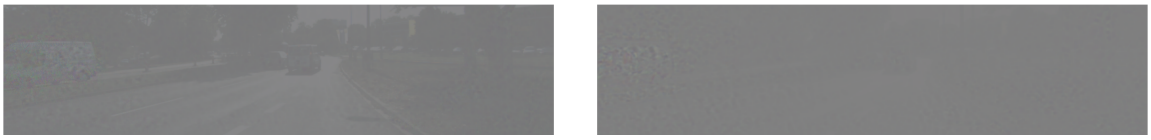
Figure 5.4: Comparing different perturbation constraints for adversarial attacks on RAFT-3D for KITTI scene 86.

	KITTI Bad-Pixel Error				RSE		
	Scene Flow	Optical Flow	d	d'	Optical Flow	d	d'
RAFT-3D	1.33%	1.19%	0.52%	0.68%	0.41%	0.07%	0.11%
GSFAC	84.98%	81.12%	60.95%	68.92%	83.24%	39.32%	35.48%
GSFAC-strict	70.03%	67.38%	6.66%	43.39%	53.62%	0.26%	2.49%

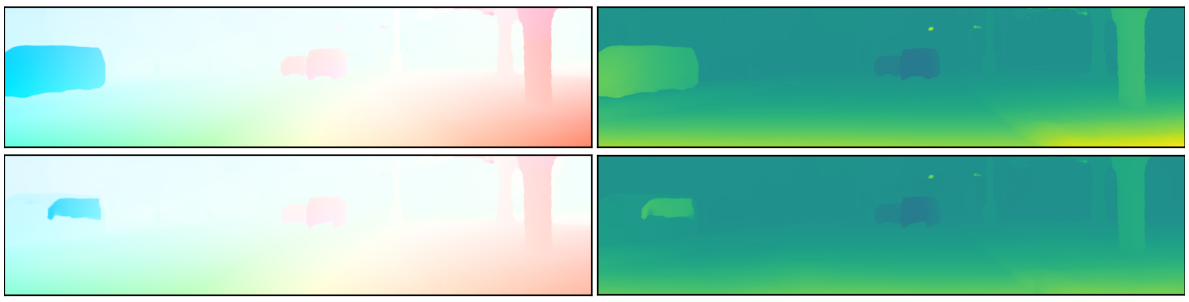
	RSE				SSIM			
	$I_{0,0}$	$I_{1,0}$	d_0	d_1	$I_{0,0}$	$I_{1,0}$	d_0	d_1
GSFA	3.59%	3.43%	3.77%	0.60%	87.62%	89.94%	94.74%	94.74%
GSFAC	0.71%	0.30%	3.53%	0.54%	95.94%	98.13%	95.13%	97.46%
GSFAC-strict	0.16%	0.06%	0.23%	0.05%	98.76%	99.52%	99.57%	99.73%

Table 5.4: Top: Scene flow estimation accuracy of RAFT-3D and the effects of GSFAC, GSFAC-strict. Bottom: Perturbation size RSE and SSIM are limited by GSFAC compared to GSFA.

22



(a) Perturbations δ_0, δ_1 generated by GSFAC-strict. The strict constraints cause patch-like perturbations.



(b) Top: RAFT-3D scene flow. Bottom: Flow after GSFAC-strict attack.

Figure 5.5: The GSFAC perturbations with strict constraints cause only minor differences in the attacked scene flow for specific scenes. The van on the left partially disappears.

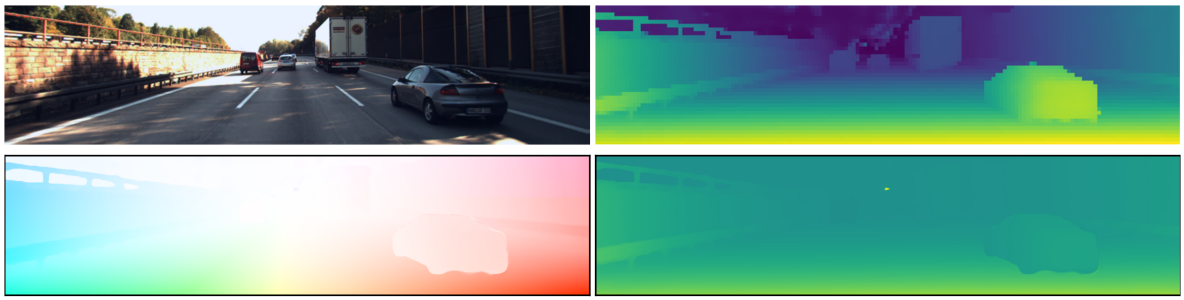
Figure 5.6: RAFT-3D is relatively robust against *GSFAC-strict* on particular KITTI scenes with high-contrast moving objects. Top: $I_{0,0}$ of scene. Bottom: Attacked flow s^*.

regions of the image exhibit a low perturbation size. The resulting attacked flow is affected the most in the regions of the patch-like perturbations. In the other areas of the scene, scene flow estimation is robust against GSFAC-strict. The analysed robust scenes in Figure 5.6 show, that scene flow of objects with high contrast is harder to attack. Therefore, increasing the contrast of input images during preprocessing may increase adversarial robustness of RAFT-3D.

5.3.3 GSFAC-D and GSFAC-I: Perturbation of Input Types

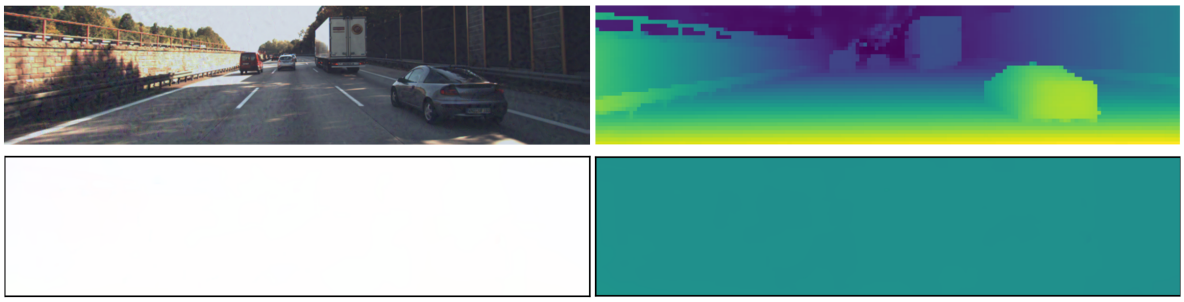
The attacks GSFAC-D and GSFAC-I limit the perturbation of RAFT-3D inputs to images and disparities respectively. In GSFAC-D only perturbations for the disparity maps are optimised, whereas in GSFAC-I the images are perturbed. For these kinds of attacks the loss function uses the same parameters as in GSFAC for the particular input type. For GSFAC-D the parameters are $\gamma_d = 0.05$, $\epsilon_d = 0.1$ and $\beta = 10$. The GSFAC-I uses $\gamma = 0.01$, $\epsilon = 0.01$ and $\alpha = 1000$. The results show, that perturbation of images has severe effects on the scene flow estimation (see Table 5.5). Perturbation of disparity maps alone have weaker effects on the scene flow estimation compared to GSFAC-I or other attack types.

The perturbation size of the GSFAC-I attack on images is even slightly lower than in GSFAC. The estimated scene flow after perturbation of $I_{0,0}, I_{1,0}$ shows a bad-pixel error of 82.85%, slightly lower than GSFAC (84.89%) which adds perturbations to the disparity maps. Without perturbations of the disparity maps, the accurate RAFT-3D estimation of d using GA-Net stays untouched. However, just by perturbation of input images the estimation error of d' is raised from the baseline 0.68% to 41.78% bad-pixel percentage. Comparing the d' estimation bad-pixel error of GSFAC and GFSAC-I shows a decrease from 68.9% to 41.8%, while the RSE decreases from 35.5% to 3%. In this particular case of d' the strictness of the KITTI bad-pixel percentage is apparent. The attack type which focuses on the perturbation of input images only is able to generate scene flows that approach the target zero scene flow, with stronger effects on the optical flow component of the scene flow.



(a) Scene flow estimation of RAFT-3D on KITTI scene 196.

(b) GSFAC-D: Large perturbation on d_0 (top right) with medium effects on scene flow estimation.



(c) GSFAC-I: Perturbation on $I_{0,0}$ (top left).

Figure 5.7: Comparison of effects of GSFAC-D and GSFAC-I on scene flow estimation of RAFT-3D.

GSFAC-D generates two large perturbations for d_0 and d_1 (see Figure 5.7(b)). The perturbation size magnitude of GSFAC-D is comparable to the unconstrained GSFA, with 3.01% RSE for d_0 and 0.93% for d_1. The large perturbations cause an increased bad-pixel error and RSE of the scene flow estimated by RAFT-3D. However, for GSFAC-D the scene flow and optical flow bad-pixel percentage only reach 10% to 12% which is small compared to other attack types.

	KITTI Bad-Pixel Error				RSE		
	Scene Flow	Optical Flow	d	d'	Optical Flow	d	d'
RAFT-3D	1.33%	1.19%	0.52%	0.68%	0.41%	0.07%	0.11%
GSFAC-I	82.85%	82.81%	0.52%	41.78%	88.95%	0.07%	3.08%
GSFAC-D	11.86%	10.03%	10.90%	11.27%	2.36%	27.40%	21.69%

Table 5.5: Effects on scene flow estimation of RAFT-3D using perturbations generated by GSFAC-I and GSFAC-D.

GSFAC-I on Spring dataset

The experiments of GSFAC-I on the KITTI dataset discovered, that perturbation of input images is sufficient to perform efficient adversarial attacks on the scene flow network RAFT-3D. To test the transferability of the scene flow attacks to other datasets, GSFAC-I is applied to a subset of scenes of the synthetic Spring dataset. The subset consists of 20 scenes, which contain different scene flow types with a stationary or moving camera. The scenes in Spring have a frame rate of 60 FPS compared to 10 FPS in the KITTI dataset. Therefore, the average length of the scene flow vectors is smaller and large regions of the scene already match the target flow s_0. The unattacked flow being close to the target flow eases the optimisation of perturbations in the same way the KITTI experiments for stationary scenes showed. The AEE between the estimated unattacked scene flow and target scene flow s_0 for the Spring dataset is 5.45, compared to 7.46 in KITTI. In general the scene flow estimation accuracy of RAFT-3D compared to the ground truth scene flow on Spring is low: The bad-pixel error metric from KITTI for the disparities d, d' reports 13.7% and 14% (KITTI 0.52% and 0.68%). For scene flow the percentage of bad pixels is 58.9% compared to 1.33% for KITTI, which confirms the strictness of the bad-pixel ratio measure.

The previous attacks used the RAFT-3D model, which was fine-tuned on KITTI and uses bilaplacian smoothing [55]. However, the experiments showed that the KITTI fine-tuned model performs worse on Spring than the standard RAFT-3D model. In the GSFAC-I on Spring the standard model is used and the same attack parameters as in the attack on KITTI inputs are used ($\gamma = 0.01$, $\epsilon = 0.01$ and $\alpha = 10^3$). The best set of perturbations $\{\delta_0, \delta_1\}$ found during the 400 optimisation steps is used as final perturbation.

Table 5.6 shows the results of the attack, with lower perturbation sizes than in all other attack types on KITTI. The RSE values of the perturbations of $I_{0,0}, I_{1,0}$ are 0.06% and 0.03%, while the SSIM values are above 99%. Analysing the perturbed images shown in figures 5.8 and 5.9, reveals hardly perceptible perturbations in particular scenes and imperceptible perturbations in most scenes. The AEE proximity to the target flow decreased effectively from 5.45 to 0.16, which is also lower than in all attacks featuring the KITTI dataset.

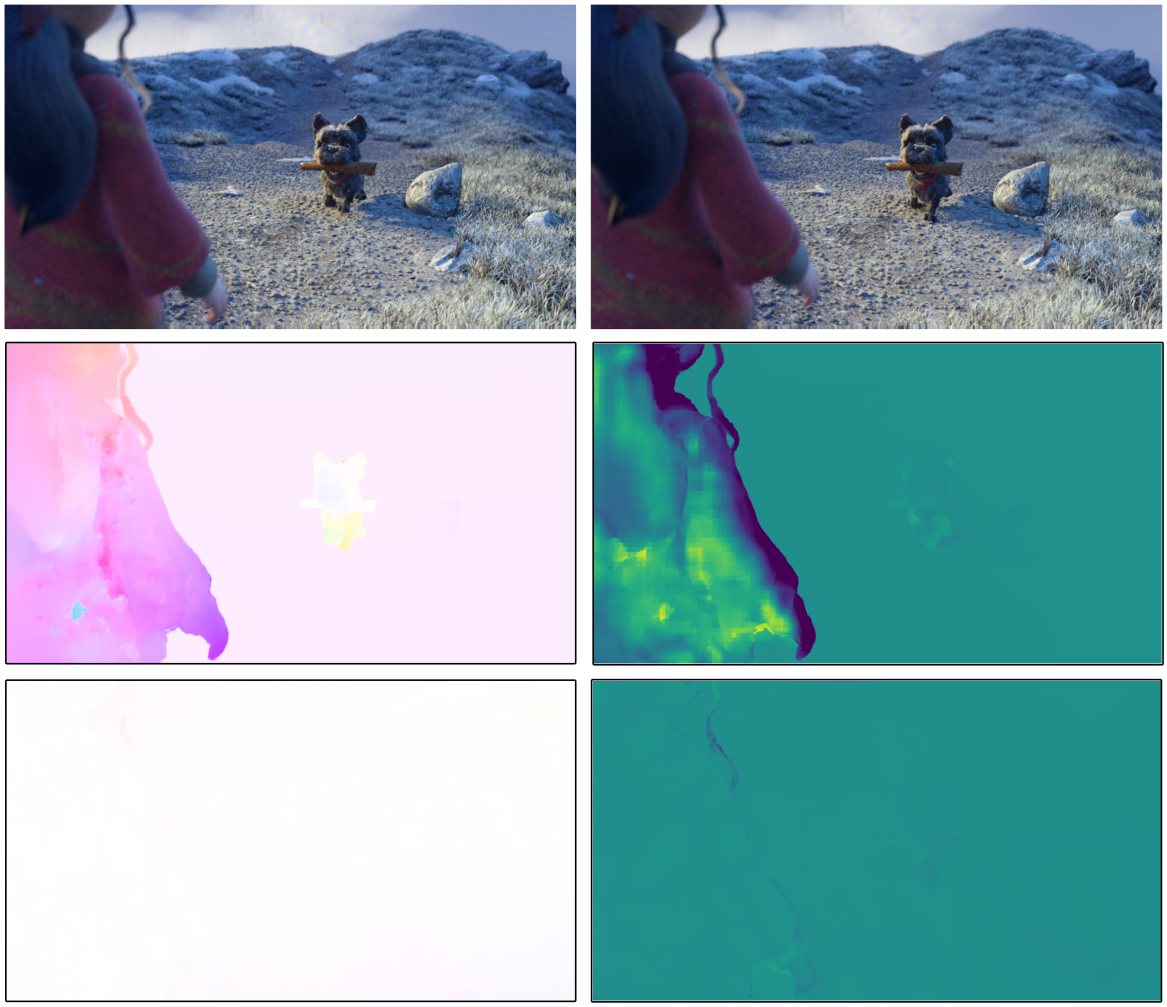
	Perturbations				Scene Flow Estimation	
	RSE		SSIM		AEE	RSE
	$I_{0,0}$	$I_{0,1}$	$I_{0,0}$	$I_{0,1}$	s^* to s_0	s^* to s_{gt}
RAFT-3D Spring	-	-	-	-	5.45	83.1%
GSFAC-I Spring	0.06%	0.03%	99.37%	99.72%	0.16	100.4%

Table 5.6: Effects on GSFAC-I on Spring dataset. Average of measures for $n = 20$ scenes of the Spring dataset.

(a) Spring scene 18. **(b)** Spring scene 56.

Figure 5.8: GSFAC-I on Spring dataset. Perturbed input images on top. RAFT-3D estimated optical flow and disparity change in the second row. The bottom row shows the scene flow estimation of RAFT-3D using the GSFAC-I perturbed images from the top row.

(a) GSFAC-I image perturbations of Spring scene 140 (contrast enhanced to improve visibility).



(b) Perturbed images on top. The second row shows the scene flow estimation by RAFT-3D using unperturbed inputs (optical flow left, disparity change right). The bottom row shows the resulting scene flow after adding the GSFAC-I perturbations from **(a)** to the input images.

Figure 5.9: GSFAC-I on Spring scene 140.

5.3.4 GSFAC-coupled: Adversarial Attack including GA-Net

The coupled constrained global scene flow adversarial attack (GSFAC-coupled) optimises four perturbations, one for each image of the two stereo pairs (see Figure 5.10). The perturbed images are then used as inputs for the disparity estimation of GA-Net and at the RAFT-3D preprocessing stage. Therefore, the perturbations have effects on two networks. Without an accurate disparity estimation of GA-Net no meaningful scene flow estimation is feasible. Additionally, the other attack types have shown that RAFT-3D is vulnerable to image perturbations.

This combined attack also considers the adversarial robustness of GA-Net. For this attack the GA-Net-11 architecture is used, because an end-to-end computation graph of RAFT-3D and GA-Net-15 exceeds feasible GPU memory limitations. To account for the reduction of 3D-convolutional layers in GA-Net, a new baseline scene flow estimation of RAFT-3D with disparities estimated by GA-Net-11 is created. As expected, the results are worse as the disparity priors are not as accurate. The bad-pixel percentage error for scene flow of the new baseline increases from 1.33% to 4.12% (RSE: from 0.41% to 0.55%) compared to the use of GA-Net-15. Because the perturbations are optimised before preprocessing and disparity estimation, the value range of the RGB images is in $[0, 255]$.

Similarly to the other attack types the minimum loss perturbation set $\{\delta_0, \delta_1, \delta_2, \delta_3\}$ during 400 optimisation steps is chosen. The learning rate is set to $\gamma = 0.5$ because of the higher pixel value range. Perturbations are penalised if pixel value perturbations exceed 1 and the penalty weight term is set to $\alpha = 1$. This results in the loss function in Equation (5.12):

$$\omega(\delta_0, \delta_1, \delta_2, \delta_3) = \text{MSE}(s^*, s_0) + \sum_{n=0}^{3} |p(\delta_n, 1)| \tag{5.12}$$

The results of GSFAC-coupled reveal a low adversarial robustness of GA-Net. The attack efficiently destroys the input disparity maps d_0, d_1 of RAFT-3D for all scenes. GSFAC-coupled aims at setting the disparity value for all pixels in d_0 close to zero and introduces artefacts in d_1 (see Figure 5.11). After the attack even for human observers it is hard to find correspondences between the two disparity maps.

The scene flow bad-pixel error percentage for GSFAC-coupled for the 200 training scenes is the highest among all constrained attack types (87.8%). Additionally, the size of the four input image perturbations is similar to GSFAC-strict. The average perturbation size of the left stereo pair images is higher, as these are also directly used as RAFT-3D inputs after preprocessing. The left image perturbation size RSE is 0.24% for $I_{0,0}$ and 0.15% for $I_{1,0}$, compared to 0.04% and 0.01% for the right images. Recalling the subjective classification of the SSIM by Zanforlin et al. [94], the perturbations of GSFAC-coupled are imperceptible by humans. The GSFAC-coupled attack effectively limits the perturbation size to generate imperceptible perturbations, which cause high attack strength and uncover the low adversarial robustness of GA-Net disparity

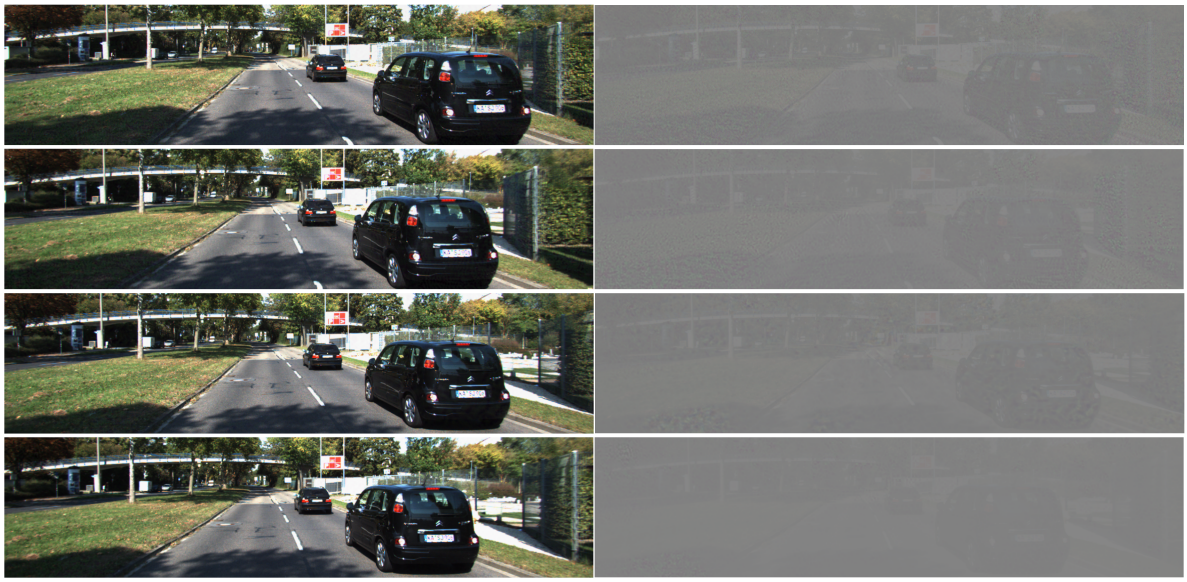


Figure 5.10: GSFAC-coupled perturbed images (left) and perturbations δ_i (right).

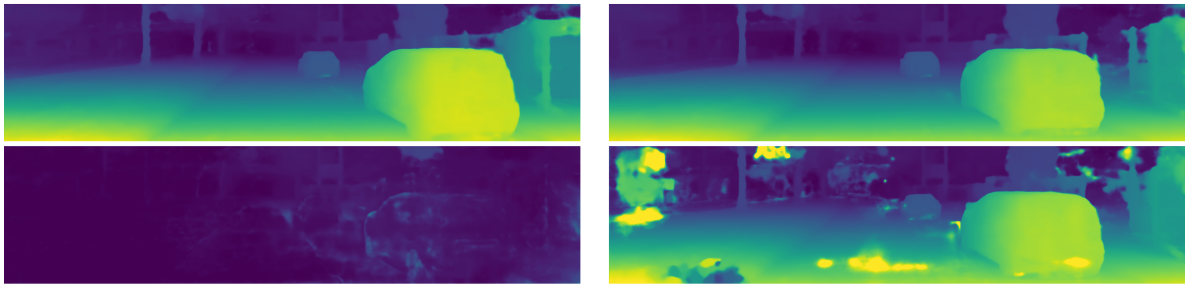


Figure 5.11: Unattacked disparity maps estimated by GA-Net before (top) and after GSFAC-coupled (bottom).

estimations. In Figure 5.12 the adversarial scene flow of KITTI scene 20 approaches the target flow s_0. The average endpoint error of GSFAC-coupled between target flow s_0 and the adversarial flow for all 200 KITTI scenes is 0.38. Based on the fact that the AEE between the baseline RAFT-3D flow and s_0 is 7.5, the attack effectively generates the target flow. RAFT-3D cannot recover from the inaccurate attacked disparity estimations of GA-Net.

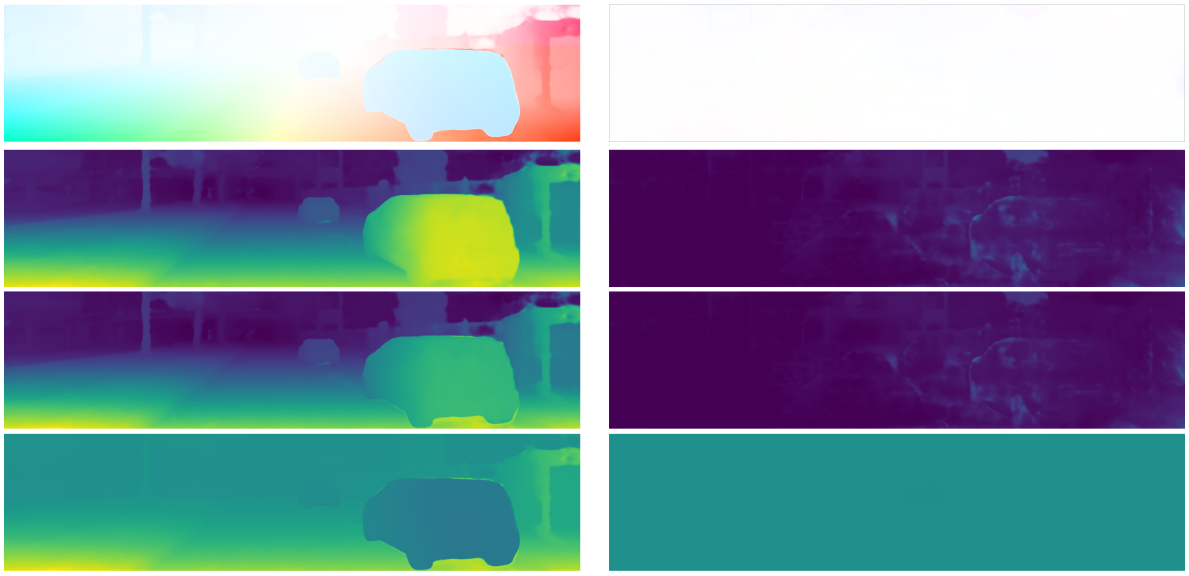


Figure 5.12: Left: RAFT-3D scene flow estimation $(u, v), d, d', \Delta d$ from top to bottom. Right: Adversarial scene flow s^* after GSFAC-coupled.

5.3.5 Comparison of Attack Types

The results of the six different experiments in Table 5.7 show that RAFT-3D and GA-Net are vulnerable to adversarial attacks. GSFAC-coupled generates hardly perceptible or imperceptible perturbations for all scenes, which result in scene flow estimations that are close to the target flow s_0. For most scenes GSFAC-strict is able to do so as well. However, there are some specific scenes where GSFAC-strict is unable to create strong attacks. The realisation of perturbation constraints by penalty terms in the loss function is effective in all constrained attacks, with average SSIM values greater 95% for all perturbed inputs. If no constraints or relaxed constraints are used, the perturbations are perceptible for the majority of scenes. In general, the GSFAC-I attack showed that perturbing only the input images is sufficient to generate an effective adversarial example. Perturbations of the input disparity maps by GSFAC-D are not effective on their own.

Because RAFT-3D uses d_0 estimated by GA-Net directly as output disparity d, the estimation error of d for all attacks except GSFAC-coupled is caused by the perturbation θ_0. For instance, the perturbations of d_0 of GSFAC cause an increase of bad-pixel error percentage from 0.52% to 60.95%. This means for more than half of the disparity map pixels the disparity value is changed by the perturbation in such a way, that the pixel disparity value exceeds the absolute error threshold of 3 from the KITTI bad-pixel metric. The increased penalty weight for disparity perturbations in GSFAC-strict limits this effect of the perturbation to a 6.66% bad-pixel percentage, while still affecting the output disparity d'. Nevertheless, GSFAC-I showed that perturbations of disparity maps is not necessary to effectively disturb the estimation of d'. To

	KITTI Bad-Pixel Error				RSE			AEE
	Scene Flow	Optical Flow	d	d'	Optical Flow	d	d'	s^* to s_0
RAFT-3D GA-Net-15	1.33%	1.19%	0.52%	0.68%	0.41%	0.07%	0.11%	7.47
GSFA	89.41%	85.23%	67.69%	76.41%	96.11%	34.5%	32.69%	0.28
GSFAC	84.98%	81.12%	60.95%	68.92%	83.24%	39.32%	35.48%	1.27
GSFAC-strict	70.03%	67.38%	6.66%	43.39%	53.62%	0.26%	2.49%	3.25
GSFAC-D	11.86%	10.03%	10.90%	11.27%	2.36%	27.40%	21.69%	7.07
GSFAC-I	82.85%	82.81%	0.52%	41.78%	88.95%	0.07%	3.08%	0.88
RAFT-3D GA-Net-11	4.12%	3.01%	1.90%	2.70%	0.55%	0.23%	0.28%	7.50
GSFAC-coupled	87.81%	83.44%	84.40%	84.65%	96.72%	92.86%	97.14%	0.38

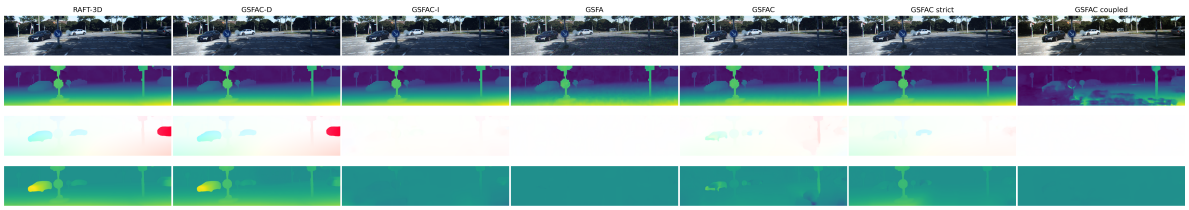
Table 5.7: Effects of different adversarial attack types on RAFT-3D scene flow estimation accuracy.

	RSE						SSIM					
	$I_{0,0}$	$I_{0,1}$	$I_{1,0}$	$I_{1,1}$	d_0	d_1	$I_{0,0}$	$I_{0,1}$	$I_{1,0}$	$I_{1,1}$	d_0	d_1
GSFA	3.59%	3.43%	-	-	3.77%	0.77%	87.62%	89.94%	-	-	94.74%	97.01%
GSFAC	0.71%	0.30%	-	-	3.53%	0.54%	95.94%	98.13%	-	-	95.13%	97.46%
GSFAC-strict	**0.16%**	0.06%	-	-	**0.23%**	**0.05%**	**98.76%**	99.52%	-	-	**99.57%**	**99.73%**
GSFAC-D	-	-	-	-	3.01%	0.93%	-	-	-	-	97.57%	97.80%
GSFAC-I	0.68%	0.28%	-	-	-	-	96.17%	98.33%	-	-	-	-
GSFAC-coupled	0.24%	**0.04%**	**0.15%**	**0.01%**	-	-	97.99%	**99.60%**	**98.58%**	**99.86%**	-	-

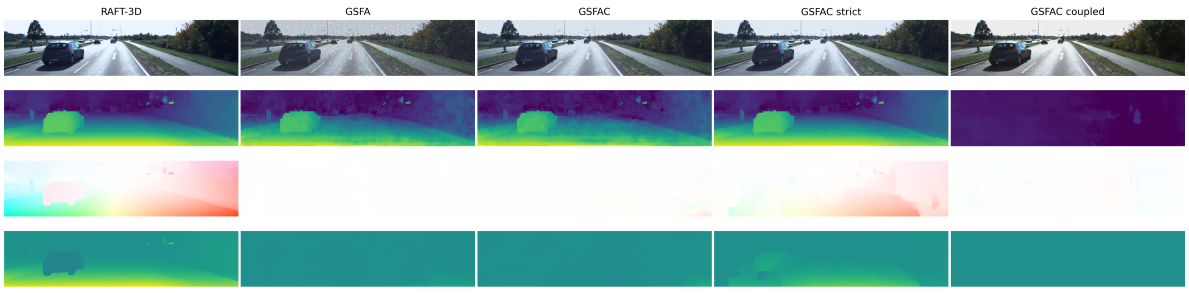
Table 5.8: Comparison of perturbation sizes of RAFT-3D inputs for different scene flow attack types. Smallest perturbations are marked in bold.

create effective attacks with minimal perturbations GSFAC-I or GSFAC-coupled are suited well, as the comparison of perturbation sizes in Table 5.8 shows.

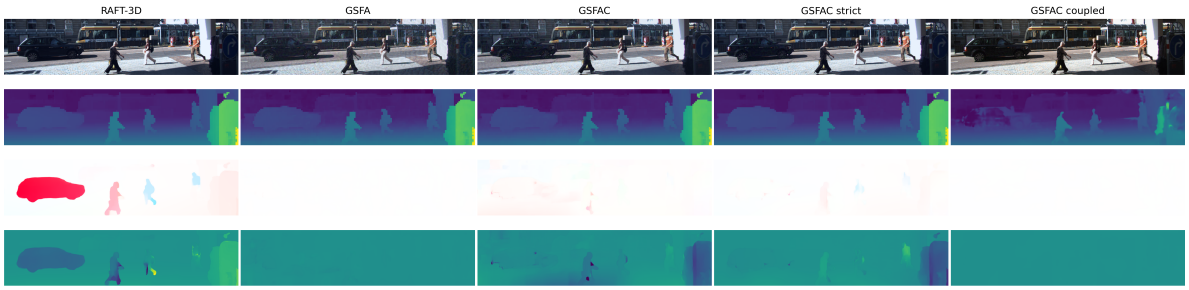
All attacks reduce the proximity of adversarial flow to target flow (see Table 5.7). The strict constraints cause a reduction of AEE from 7.47 to 3.25. Among the constrained attacks GSFAC, GSFAC-I and GSFAC-coupled show the best proximity to s_0. Figure 5.13 display the effects of the different attacks on the same scene. In general the trade-off between limitations of perturbation size and attack strength can be seen in these examples. In GSFAC-strict for non-stationary scenes the basic structure of the scene flow is not completely erased (see Figure 5.13**(b)**). However, GSFAC-coupled delivers the highest attack strength while keeping perturbations in a hardly perceptible range. Therefore, GSFAC-coupled is deemed as the most effective attack.



(a) KITTI scene 11 with GSFAC-D, GSFAC-I (second and third column).



(b) KITTI scene 78: Moving camera causes a dense scene flow field with large displacement vector magnitude.



(c) KITTI scene 169: A stationary camera causes an unattacked sparse flow field, where only moving objects mismatch the target scene flow s_0.

Figure 5.13: Comparison of attack types from left to right: RAFT-3D baseline, (GSFAC-D, GSFAC-I), GSFA, GSFAC, GSFAC-strict, GSFAC-coupled. (Perturbed) inputs and outputs of RAFT-3D from top to bottom: $I_{0,0}, d_0, (u, v), \Delta d$.

6 Conclusion and Outlook

6.1 Outlook

The research topic adversarial attacks on flow neural networks is relatively new and so far only attacks on optical flow and stereo networks have been proposed. This work introduced a method to generate effective adversarial examples for scene flow estimation networks. However, the method was only tested on the RAFT-3D network. Further promising research directions include the application of GSFA or other scene flow adversarial attacks on different scene flow estimation networks. Currently, multiple networks based on the RAFT-3D architecture rank in the top 10 of the KITTI 2015 scene flow benchmark [1]. Applying the scene flow adversarial attack on scene flow networks, which are not based on RAFT architecture (e.g. CamLiFlow) would enable a comparison of adversarial robustness. Additionally, the effects on adversarial robustness of extensions or modifications of other RAFT-3D based networks could be examined. The results of these comparisons may reveal insights about possible defense mechanisms. Previous works could show, that adversarial examples optimised for one network are also effective on another network to some extent [83]. Therefore, the transferability of the perturbations generated by GSFA and its variations could be tested on other scene flow networks.

Perturbations of images is sufficient for the proposed scene flow attacks GSFAC-I or GSFAC-coupled. A possible cause for this vulnerability could be the context encoder of RAFT-3D. The context encoder uses a pretrained ResNet50 to extract context information about rigid objects of the scene. A hypothesis for the vulnerability of RAFT-3D to image perturbations is the destruction of the rigidity property of objects. Image perturbations may cause the context encoder to split a perturbed object into multiple smaller rigid objects or prohibit object detection. Excluding the image perturbations only for the context encoder part of RAFT-3D could be a way to test this hypothesis. Analysing the rigid object embeddings of RAFT-3D for the perturbed inputs is another way to check if this hypothesis holds.

The experiments in this work optimise frame-specific disjoint perturbations for input images and disparity maps. Schmalfuss et al. [8] generated universal and joint perturbations to attack optical flow networks. Further experiments with GSFA could optimise different combinations of universal, frame-specific and joint or disjoint perturbations. As frame-specific disjoint perturbations impose the least constraints it is expected that RAFT-3D is less vulnerable against the other attack types.

This work focuses on theoretical aspects of adversarial robustness of scene flow networks. The proposed attack method is of theoretical nature and may not be applicable in the real-world. Brown et al. [78] showed that the adversarial patch attack could be transferred to real-world scenarios using printed stickers. For scene flow adversarial attacks the applicability in the real-world is another potential research direction.

Finally, defensive mechanisms against adversarial attacks on scene flow would be an interesting research topic. The GSFA perturbations added to the input images show very intricate patterns, which could be partially erased by simple Gaussian smoothing during the RAFT-3D preprocessing step. Kurakin et al. [89] and Goodfellow et al. [75] could show, that including adversarial examples into the training set of a neural network increases the adversarial robustness. The adversarial examples generated with GSFA and its variations can be used for adversarial retraining of RAFT-3D or other scene flow networks to examine the effects on robustness. Effects of smoothing and other defense mechanisms from literature on the adversarial robustness of RAFT-3D and other scene flow networks is subject to further research.

6.2 Conclusion

Research findings from Ranjan et al. [6], Schrodi et al. [7] and Schmalfuss et al. [8] revealed the vulnerability of optical flow estimation neural networks to adversarial attacks. For stereo matching networks effective adversarial attacks were proposed by Berger et al. [9] and Wong et al. [10]. In this work the concept of adversarial attacks on flow networks is extended to the scene flow problem, which combines disparity and optical flow estimation. The framework developed in this work is based on the PCFA attack [80] and introduces new targets and loss functions for the scene flow problem setting. With the targeted global scene flow attack (GSFA) and its variations coupled and decoupled scene flow estimation networks can be attacked. Input perturbations are generated by the Adam optimisation algorithm, which minimises a loss function which respects the proximity to the target flow and the perturbation size. The perturbations can be limited to arbitrary combinations of different types of inputs or intermediate disparity estimations. Additionally, perturbation sizes are effectively limited by penalty terms in the loss functions.

The analysis of extensive experiments with different combinations of parameters identified suitable parameter settings for GSFA. Experiments covering all variations of GSFA on the state-of-the-art scene flow estimation network RAFT-3D [55] using input data from 200 scenes of the KITTI 2015 scene flow benchmark [1] are conducted. All attack variations except GSFAC-D show a high attack strength. The GSFAC-I method, which perturbs two input images of RAFT-3D, could show that perturbations of disparity maps are not neccessary to create an effective attack. The effectiveness of GSFA is also shown for the synthetic scene flow benchmark Spring using scenes from an animated movie.

While no constraints on perturbations (GSFA) or relaxed constraints in GSFAC produce adversarial examples with a large perturbation size, strict constraints on perturbation sizes by GSFAC-strict lead to imperceptible but effective perturbations. A second point of attack is realised with GSFAC-coupled, where the prior disparity estimation network GA-Net used by RAFT-3D is also included in the adversarial attack. GSFAC-coupled exploits vulnerabilities of RAFT-3D and GA-Net and creates hardly perceptible perturbations on the consecutive pairs of stereo images. The experiments showed that attacks on coupled and decoupled scene flow estimation networks are feasible, as they exhibit similar vulnerabilities as optical flow and disparity estimation networks.

Besides flow accuracy, future evaluations of scene flow networks should consider reporting adversarial robustness metrics. The global scene flow attack presented in this work is primarily of theoretical nature. Further implications of GSFA for scene flow networks used in real-world scenarios is a future research question.

A Appendix

A.1 Penalty Weights Refinement Experiments

Experiment Index	Parameters				KITTI Bad-Pixel Error				Perturbation Size RMSE			
	α	ϵ	β	ϵ_d	Scene Flow	Optical Flow	d	d'	$I_{0,0}$	$I_{1,0}$	d_0	d_1
RAFT-3D	-	-	-	-	1.2%	1.0%	0.4%	0.5%	0	0	0	0
0	0	0.01	0	0.1	96.6%	91.8%	81.6%	86.1%	0.23	0.22	6.93	2.70
1	0	0.01	10	0.1	94.8%	91.7%	75.0%	80.0%	0.20	0.19	6.09	2.42
2	0	0.01	20	0.1	94.3%	89.7%	70.8%	77.4%	0.19	0.18	5.18	2.28
3	0	0.01	50	0.1	85.4%	79.4%	47.0%	53.5%	0.12	0.12	2.95	1.44
4	0	0.01	100	0.1	74.9%	70.7%	29.0%	37.3%	0.09	0.09	2.02	1.06
5	0	0.01	500	0.1	42.3%	41.6%	1.1%	7.9%	0.04	0.04	0.64	0.44
6	0	0.01	1000	0.1	20.0%	19.9%	0.5%	2.9%	0.02	0.02	0.29	0.24
7	10	0.01	0	0.1	97.3%	91.7%	81.4%	85.6%	0.21	0.19	7.33	2.75
8	10	0.01	10	0.1	94.8%	91.4%	73.6%	78.2%	0.19	0.18	5.60	2.36
9	10	0.01	20	0.1	93.5%	90.5%	68.0%	73.5%	0.17	0.16	4.70	2.07
10	10	0.01	50	0.1	84.7%	80.5%	46.2%	53.8%	0.13	0.12	3.15	1.49
12	10	0.01	500	0.1	47.9%	46.1%	6.2%	16.7%	0.05	0.05	0.89	0.56
13	10	0.01	1000	0.1	20.8%	20.7%	0.5%	3.1%	0.02	0.02	0.31	0.26
14	20	0.01	0	0.1	96.5%	91.7%	82.7%	87.7%	0.20	0.17	6.90	2.74
15	20	0.01	10	0.1	94.3%	91.0%	71.5%	77.9%	0.18	0.17	5.54	2.27
16	20	0.01	20	0.1	93.0%	89.0%	67.7%	73.6%	0.17	0.16	5.09	2.12
17	20	0.01	50	0.1	84.9%	78.6%	46.8%	54.2%	0.11	0.11	2.77	1.37
18	20	0.01	100	0.1	74.2%	69.8%	28.5%	37.5%	0.09	0.09	2.07	1.10
19	20	0.01	500	0.1	43.8%	42.3%	5.8%	12.5%	0.04	0.04	0.77	0.50
20	20	0.01	1000	0.1	20.9%	20.8%	0.5%	4.4%	0.02	0.02	0.32	0.26
21	50	0.01	0	0.1	96.4%	91.6%	83.6%	87.8%	0.19	0.15	7.68	2.87
22	50	0.01	10	0.1	93.6%	90.2%	71.7%	78.1%	0.17	0.15	5.63	2.31
23	50	0.01	20	0.1	93.0%	89.0%	65.3%	71.8%	0.17	0.15	4.98	2.14
24	50	0.01	50	0.1	80.5%	76.2%	36.6%	46.4%	0.11	0.11	2.78	1.34
25	50	0.01	100	0.1	71.6%	68.5%	22.4%	34.0%	0.08	0.08	1.89	1.00
26	50	0.01	500	0.1	46.7%	45.1%	5.3%	15.5%	0.05	0.05	0.81	0.53
27	50	0.01	1000	0.1	21.2%	21.0%	0.5%	4.5%	0.02	0.02	0.34	0.26
28	100	0.01	0	0.1	96.7%	91.6%	87.1%	89.8%	0.17	0.12	7.74	2.88
29	100	0.01	10	0.1	93.4%	90.2%	69.7%	75.6%	0.16	0.14	5.23	2.21
30	100	0.01	20	0.1	93.0%	88.9%	68.1%	74.9%	0.16	0.14	4.88	2.07
31	100	0.01	50	0.1	83.1%	79.5%	40.4%	49.4%	0.12	0.11	2.95	1.44
32	100	0.01	100	0.1	70.6%	67.3%	20.5%	32.1%	0.08	0.08	1.58	0.92
33	100	0.01	500	0.1	44.5%	42.9%	5.4%	11.6%	0.04	0.04	0.77	0.50
34	100	0.01	1000	0.1	19.9%	19.8%	0.5%	2.8%	0.02	0.02	0.29	0.24
35	500	0.01	0	0.1	96.5%	91.5%	89.4%	90.9%	0.11	0.07	8.67	3.20
36	500	0.01	10	0.1	93.8%	89.3%	73.5%	77.3%	0.13	0.10	5.25	2.22
37	500	0.01	20	0.1	91.8%	86.7%	64.8%	70.7%	0.13	0.10	4.89	2.10
38	500	0.01	50	0.1	84.0%	76.7%	43.4%	52.5%	0.10	0.09	2.93	1.39
39	500	0.01	100	0.1	74.5%	68.8%	29.3%	37.4%	0.08	0.07	1.84	1.01
40	500	0.01	500	0.1	38.4%	38.1%	0.6%	6.2%	0.03	0.03	0.52	0.40
41	500	0.01	1000	0.1	21.5%	21.4%	0.5%	4.3%	0.02	0.02	0.33	0.26
42	1000	0.01	0	0.1	95.8%	91.1%	90.5%	91.4%	0.10	0.06	9.51	3.42
43	1000	0.01	10	0.1	91.9%	88.2%	67.0%	71.4%	0.12	0.09	4.74	2.05
44	1000	0.01	20	0.1	92.1%	86.5%	67.8%	72.0%	0.11	0.09	4.44	1.91
45	1000	0.01	50	0.1	83.4%	78.3%	43.3%	53.2%	0.10	0.08	3.08	1.49
46	1000	0.01	100	0.1	74.2%	70.0%	31.5%	37.7%	0.08	0.07	2.03	1.09
47	1000	0.01	500	0.1	42.9%	41.8%	1.6%	9.7%	0.04	0.04	0.66	0.46
48	1000	0.01	1000	0.1	21.0%	20.8%	0.5%	4.5%	0.02	0.02	0.34	0.26

Table A.1: Results of the penalty weights refinement experiments.

A.2 Learning Rates Refinement Experiments

	Parameters		KITTI Bad-Pixel Error				Perturbation Size RMSE				Best Loss	
Experiment Index	y	y_d	Scene Flow	Optical Flow	d	d'	$I_{0,0}$	$I_{1,0}$	d_0	d_1	Step	Loss
RAFT-3D	-	-	1.2%	1.0%	0.4%	0.5%	-	-	-	-	-	-
0	0.01	0.01	90.7%	89.5%	6.1%	60.1%	0.11	0.07	1.05	0.50	330.6	34.1
1	0.01	0.05	93.5%	89.3%	69.1%	77.0%	0.11	0.07	4.66	1.78	275.2	42.1
2	0.01	0.1	93.1%	90.2%	75.4%	78.9%	0.09	0.06	8.24	2.88	243.5	68.4
3	0.01	0.25	95.8%	89.7%	88.6%	90.3%	0.09	0.06	14.08	5.21	157.1	146.4
4	0.01	0.5	97.1%	89.0%	95.1%	94.2%	0.09	0.06	17.98	7.20	86.4	245.7
5	0.01	1	97.0%	86.9%	95.9%	95.2%	0.08	0.06	21.13	9.50	41.0	335.9
6	0.05	0.01	89.8%	89.5%	1.5%	56.6%	0.14	0.08	0.79	0.35	285.5	25.1
7	0.05	0.05	93.0%	89.6%	56.0%	74.6%	0.15	0.09	3.52	1.19	205.1	43.9
8	0.05	0.1	96.7%	89.8%	89.2%	91.5%	0.14	0.08	6.51	2.16	223.0	55.2
9	0.05	0.25	95.8%	88.2%	84.6%	87.0%	0.16	0.10	8.98	3.27	102.2	98.8
10	0.05	0.5	95.6%	89.0%	89.3%	89.2%	0.15	0.09	13.66	5.24	78.2	147.8
11	0.05	1	97.2%	88.7%	95.9%	94.6%	0.18	0.12	15.87	7.42	33.3	228.4
12	0.1	0.01	88.0%	87.5%	2.6%	53.3%	0.21	0.13	0.85	0.34	241.0	72.7
13	0.1	0.05	93.4%	88.7%	67.0%	80.8%	0.21	0.13	3.81	1.19	216.2	73.4
14	0.1	0.1	92.6%	89.4%	76.4%	81.2%	0.18	0.11	7.14	1.99	210.0	79.1
15	0.1	0.25	95.2%	88.4%	87.0%	88.6%	0.18	0.11	9.38	3.10	112.2	102.7
17	0.1	1	92.4%	84.4%	90.5%	89.4%	0.23	0.14	14.04	6.78	35.7	222.1
18	0.25	0.01	81.9%	80.9%	5.4%	55.1%	0.34	0.22	1.01	0.36	255.4	150.3
19	0.25	0.05	94.1%	86.8%	78.1%	84.6%	0.26	0.17	4.21	1.27	246.2	113.9
20	0.25	0.1	90.6%	84.1%	84.1%	86.0%	0.24	0.15	8.70	2.19	267.1	118.2
21	0.25	0.25	90.6%	84.9%	88.4%	87.3%	0.20	0.13	13.19	3.56	167.1	150.0
22	0.25	0.5	86.6%	79.1%	84.6%	84.0%	0.25	0.16	13.76	4.71	82.5	191.6
23	0.25	1	77.5%	70.6%	75.7%	74.9%	0.25	0.17	17.85	7.77	55.0	277.8
24	0.5	0.01	75.9%	75.5%	1.3%	43.8%	0.46	0.33	0.70	0.27	142.9	270.8
25	0.5	0.05	84.0%	77.2%	70.4%	75.5%	0.42	0.29	4.61	1.31	227.8	227.6
26	0.5	0.1	84.9%	79.3%	81.3%	80.2%	0.35	0.25	10.08	2.33	264.5	196.0
27	0.5	0.25	86.6%	79.6%	84.8%	84.1%	0.28	0.21	14.78	3.79	184.3	194.6
29	0.5	1	69.8%	65.5%	67.9%	68.0%	0.40	0.33	16.91	7.04	46.8	312.0
30	1	0.01	69.9%	69.7%	7.5%	45.2%	0.71	0.57	1.03	0.36	176.1	422.3
31	1	0.05	79.0%	73.4%	72.1%	70.9%	0.60	0.49	5.46	1.47	221.4	368.3
32	1	0.1	74.7%	70.7%	71.0%	71.1%	0.60	0.52	10.98	2.72	250.8	308.4
33	1	0.25	76.6%	70.3%	74.1%	74.2%	0.52	0.48	16.81	4.38	206.7	253.6
34	1	0.5	72.0%	65.8%	71.3%	70.2%	0.56	0.52	17.87	5.61	110.7	277.2

Table A.2: Results of the learning rates refinement experiments.

Bibliography

[1] M. Menze, C. Heipke, and A. Geiger, "Joint 3d estimation of vehicles and scene flow," in *ISPRS Workshop on Image Sequence Analysis (ISA)*. (Cited on pages 7, 18, 19, 22, 28, 29, 37, 46, 61, 62, 85 and 86)

[2] D. Stoyanov, "Stereoscopic scene flow for robotic assisted minimally invasive surgery," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2012, pp. 479–486. (Cited on page 7)

[3] H. Saito, N. Inamoto, and S. Iwase, "Sports scene analysis and visualization from multiple-view video," in *2004 IEEE International Conference on Multimedia and Expo (ICME)(IEEE Cat. No. 04TH8763)*, vol. 2. IEEE, 2004, pp. 1395–1398. (Cited on page 7)

[4] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," vol. 2, pp. 722–729, 1999. (Cited on pages 7 and 27)

[5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013. (Cited on pages 7, 30, 31, 32 and 33)

[6] A. Ranjan, J. Janai, A. Geiger, and M. J. Black, "Attacking optical flow," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2404–2413. (Cited on pages 7, 31, 33, 35, 36 and 86)

[7] S. Schrodi, T. Saikia, and T. Brox, "Towards understanding adversarial robustness of optical flow networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8916–8924. (Cited on pages 7, 33, 36, 47 and 86)

[8] J. Schmalfuss, P. Scholze, and A. Bruhn, "A perturbation constrained adversarial attack for evaluating the robustness of optical flow," *arXiv preprint arXiv:2203.13214*, 2022. (Cited on pages 7, 33, 34, 35, 45, 46, 85 and 86)

[9] Z. Berger, P. Agrawal, T. Y. Liu, S. Soatto, and A. Wong, "Stereoscopic universal perturbations across different architectures and datasets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15 180–15 190. (Cited on pages 8, 33, 36, 37 and 86)

[10] A. Wong, M. Mundhra, and S. Soatto, "Stereopagnosia: Fooling stereo networks with adversarial perturbations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 4, 2021, pp. 2879–2888. (Cited on pages 8, 18, 33, 36, 37 and 86)

[11] S. Georgieva, R. Peeters, H. Kolster, J. T. Todd, and G. A. Orban, "The processing of three-dimensional shape from disparity in the human brain," *Journal of Neuroscience*, vol. 29, no. 3, pp. 727–742, 2009. (Cited on page 11)

[12] K. Ikeuchi, Ed., *Computer Vision: A Reference Guide*, ser. Springer Reference. New York: Springer, 2014. (Cited on page 12)

[13] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003. (Cited on pages 13, 14 and 16)

[14] J. Cederberg, *A Course in Modern Geometries*, ser. Undergraduate Texts in Mathematics. Springer New York, 2013. [Online]. Available: https://books.google.de/books?id=jvfTBwAAQBAJ (Cited on page 13)

[15] I. Agricola, T. Friedrich, and P. Spain, *Elementary Geometry*, ser. Student mathematical library. American Mathematical Society, 2008. [Online]. Available: https://books.google.gl/books?id=LLXxBwAAQBAJ (Cited on page 13)

[16] Y. Furukawa, C. Hernández *et al.*, "Multi-view stereo: A tutorial," *Foundations and Trends® in Computer Graphics and Vision*, vol. 9, no. 1-2, pp. 1–148, 2015. (Cited on page 14)

[17] H. Kim, S.-j. Yang, and K. Sohn, "3d reconstruction of stereo images for interaction between real and virtual worlds," 11 2003, pp. 169– 176. (Cited on page 15)

[18] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artificial intelligence*, vol. 78, no. 1-2, pp. 87–119, 1995. (Cited on page 16)

[19] H. Hirschmüller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8. (Cited on page 16)

[20] K. Zhang, J. Lu, G. Lafruit, R. Lauwereins, and L. Van Gool, "Robust stereo matching with fast normalized cross-correlation over shape-adaptive regions," in *2009 16th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2009, pp. 2357–2360. (Cited on page 17)

[21] B. J. Super and W. N. Klarquist, "Patch-based stereo in a general binocular viewing geometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 247–253, 1997. (Cited on page 17)

[22] Y. S. Heo, K. M. Lee, and S. U. Lee, "Robust stereo matching using adaptive normalized cross-correlation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 33, no. 4, pp. 807–822, 2010. (Cited on page 17)

[23] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof, "Pushing the limits of stereo using variational stereo estimation," in *2012 IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 401–407. (Cited on page 17)

[24] L. Alvarez, R. Deriche, J. Sanchez, and J. Weickert, "Dense disparity map estimation respecting image discontinuities: A pde and scale-space based approach," *Journal of Visual Communication and Image Representation*, vol. 13, no. 1-2, pp. 3–21, 2002. (Cited on page 17)

[25] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof, "Pushing the limits of stereo using variational stereo estimation," in *2012 IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 401–407. (Cited on page 17)

[26] R. Ben-Ari and N. Sochen, "Variational stereo vision with sharp discontinuities and occlusion handling," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–7. (Cited on page 17)

[27] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 328–341, 2007. (Cited on pages 17, 18 and 40)

[28] K. Zhou, X. Meng, and B. Cheng, "Review of stereo matching algorithms based on deep learning," *Computational intelligence and neuroscience*, vol. 2020, 2020. (Cited on page 18)

[29] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625. (Cited on pages 18 and 28)

[30] L. Mehl, "Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo." (Cited on pages 18 and 28)

[31] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," *CoRR*, vol. abs/1512.02134, 2015. [Online]. Available: http://arxiv.org/abs/1512.02134 (Cited on pages 19 and 28)

[32] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *Proceedings of the IEEE/CVF international conference on computer vision workshops*, 2019, pp. 0–0. (Cited on page 19)

[33] Z. Shen, Y. Dai, X. Song, Z. Rao, D. Zhou, and L. Zhang, "Pcw-net: Pyramid combination and warping cost volume for stereo matching," in *European Conference on Computer Vision*. Springer, 2022, pp. 280–297. (Cited on page 19)

[34] F. Zhang, V. Prisacariu, R. Yang, and P. H. Torr, "Ga-net: Guided aggregation net for end-to-end stereo matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 185–194. (Cited on pages 19, 22, 30, 39, 41, 49, 53 and 56)

[35] X. Cheng, Y. Zhong, M. Harandi, Y. Dai, X. Chang, H. Li, T. Drummond, and Z. Ge, "Hierarchical neural architecture search for deep stereo matching," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 158–22 169, 2020. (Cited on pages 19 and 29)

[36] B. D. Lucas, T. Kanade *et al.*, *An iterative image registration technique with an application to stereo vision*. Vancouver, 1981, vol. 81. (Cited on page 20)

[37] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981. (Cited on pages 20, 21 and 25)

[38] C. Schnörr, "Determining optical flow for irregular domains by minimizing quadratic functionals of a certain class," *International Journal of Computer Vision*, vol. 6, no. 1, pp. 25–38, 1991. (Cited on page 21)

[39] F. Bartolini and A. Piva, "Enhancement of the horn and schunck optic flow algorithm by means of median filters," in *Proceedings of 13th International Conference on Digital Signal Processing*, vol. 2, 1997, pp. 503–506 vol.2. (Cited on page 21)

[40] N. Bauer, P. Pathirana, and P. Hodgson, "Robust optical flow with combined lucas-kanade/horn-schunck and automatic neighborhood selection," in *2006 International Conference on Information and Automation*, 2006, pp. 378–383. (Cited on page 21)

[41] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017. (Cited on pages 21, 30 and 42)

[42] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1915–1929, 2012. (Cited on page 21)

[43] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587. (Cited on page 21)

[44] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in

Proceedings of the IEEE international conference on computer vision, 2015, pp. 2758–2766. (Cited on pages 21, 23, 28, 35 and 36)

[45] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241. (Cited on pages 21 and 29)

[46] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470. (Cited on pages 21, 28, 35, 36 and 46)

[47] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4161–4170. (Cited on pages 21, 35, 36 and 46)

[48] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943. (Cited on pages 22, 35, 36 and 46)

[49] F. Zhang, O. J. Woodford, V. A. Prisacariu, and P. H. Torr, "Separable flow: Learning motion cost volumes for optical flow estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 807–10 817. (Cited on page 22)

[50] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *European conference on computer vision*. Springer, pp. 402–419. (Cited on pages 22, 29, 30, 36 and 46)

[51] H.-P. Huang, C. Herrmann, J. Hur, E. Lu, K. Sargent, A. Stone, M.-H. Yang, and D. Sun, "Self-supervised autoflow," 2022. [Online]. Available: https://arxiv.org/abs/2212.01762 (Cited on page 22)

[52] Z. Zhang, P. Ji, N. Bansal, C. Cai, Q. Yan, X. Xu, and Y. Xu, "Clip-flow: Contrastive learning by {s} emi-supervised iterative pseudo {l} abeling for optical flow estimation," *arXiv preprint arXiv:2210.14383*, 2022. (Cited on page 22)

[53] J. Jeong, J. M. Lin, F. Porikli, and N. Kwak, "Imposing consistency for optical flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3181–3191. (Cited on page 22)

[54] A. Jahedi, L. Mehl, M. Rivinius, and A. Bruhn, "Multi-scale raft: Combining hierarchical concepts for learning-based optical flow estimation," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2022, pp. 1236–1240. (Cited on page 22)

[55] Z. Teed and J. Deng, "Raft-3d: Scene flow using rigid-motion embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8375–8384. (Cited on pages 22, 29, 30, 37, 42, 43, 45, 49, 52, 56, 61, 77 and 86)

[56] Z. Ren, J. Yan, B. Ni, B. Liu, X. Yang, and H. Zha, "Unsupervised deep learning for optical flow estimation," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. (Cited on page 22)

[57] Y. Zhu, Z. Lan, S. Newsam, and A. G. Hauptmann, "Guided optical flow learning," *arXiv preprint arXiv:1702.02295*, 2017. (Cited on page 22)

[58] C. Bailer, B. Taetz, and D. Stricker, "Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4015–4023. (Cited on page 22)

[59] P. Liu, M. Lyu, I. King, and J. Xu, "Selflow: Self-supervised learning of optical flow," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4571–4580. (Cited on pages 22 and 23)

[60] A. Wedel and D. Cremers, *Stereo scene flow for 3D motion analysis*. Springer Science & Business Media, 2011. (Cited on pages 23, 24, 25, 26 and 27)

[61] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision*. Springer, 2004, pp. 25–36. (Cited on pages 26 and 27)

[62] C. Vogel, K. Schindler, and S. Roth, "3d scene flow estimation with a rigid motion prior," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 1291–1298. (Cited on page 27)

[63] F. Huguet and F. Devernay, "A variational method for scene flow estimation from stereo sequences," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–7. (Cited on page 27)

[64] M. Jaimez, M. Souiai, J. Gonzalez-Jimenez, and D. Cremers, "A primal-dual framework for real-time dense rgb-d scene flow," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 98–104. (Cited on page 27)

[65] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Rigid scene flow for 3d lidar scans," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1765–1770. (Cited on page 27)

[66] A. K. Ushani, R. W. Wolcott, J. M. Walls, and R. M. Eustice, "A learning approach for real-time temporal scene flow estimation from lidar data," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5666–5673. (Cited on page 27)

[67] E. Ilg, T. Saikia, M. Keuper, and T. Brox, "Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 614–630. (Cited on page 28)

[68] G. Yang and D. Ramanan, "Learning to segment rigid motions from two frames," in *CVPR*, 2021. (Cited on page 28)

[69] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019. (Cited on page 29)

[70] L. Sommer, P. Schröppel, and T. Brox, "Sf2se3: Clustering scene flow into se (3)-motions via proposal and selection," in *DAGM German Conference on Pattern Recognition*. Springer, 2022, pp. 215–229. (Cited on page 29)

[71] L. Mehl, A. Jahedi, J. Schmalfuss, and A. Bruhn, "M-fuse: Multi-frame fusion for scene flow estimation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 2020–2029. (Cited on page 29)

[72] X. Liu, C. R. Qi, and L. J. Guibas, "Flownet3d: Learning scene flow in 3d point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 529–537. (Cited on page 29)

[73] A. Behl, D. Paschalidou, S. Donné, and A. Geiger, "Pointflownet: Learning representations for rigid motion estimation from point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7962–7971. (Cited on page 29)

[74] H. Liu, T. Lu, Y. Xu, J. Liu, W. Li, and L. Chen, "Camliflow: Bidirectional camera-lidar fusion for joint optical flow and scene flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5791–5801. (Cited on page 29)

[75] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014. (Cited on pages 30, 33 and 86)

[76] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial intelligence safety and security*. Chapman and Hall/CRC, 2018, pp. 99–112. (Cited on pages 30 and 33)

[77] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013. (Cited on page 30)

[78] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," *arXiv preprint arXiv:1712.09665*, 2017. (Cited on pages 30, 31 and 86)

[79] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *International conference on machine learning*. PMLR, 2018, pp. 284–293. (Cited on page 30)

[80] J. Schmalfuss, L. Mehl, and A. Bruhn, "Attacking motion estimation with adversarial snow," *arXiv preprint arXiv:2210.11242*, 2022. (Cited on pages 30, 36, 46, 47, 50 and 86)

[81] S. Aziznejad and M. Unser, "Deep spline networks with control of lipschitz regularity," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3242–3246. (Cited on page 31)

[82] D. Karmon, D. Zoran, and Y. Goldberg, "Lavan: Localized and visible adversarial noise," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2507–2515. (Cited on page 31)

[83] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*. Ieee, 2017, pp. 39–57. (Cited on pages 32 and 85)

[84] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017. (Cited on page 32)

[85] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy *et al.*, "Technical report on the cleverhans v2. 1.0 adversarial examples library," *arXiv preprint arXiv:1610.00768*, 2016. (Cited on page 32)

[86] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019. (Cited on page 32)

[87] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018. (Cited on pages 32 and 33)

[88] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989. (Cited on pages 32 and 46)

[89] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016. (Cited on pages 33 and 86)

[90] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International journal of computer vision*, vol. 92, pp. 1–31, 2011. (Cited on pages 34 and 61)

[91] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. (Cited on page 51)

[92] Z. Teed and J. Deng, "Tangent space backpropagation for 3d transformation groups," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. (Cited on page 56)

[93] A. Botchkarev, "Evaluating performance of regression machine learning models using multiple error metrics in azure machine learning studio," *Available at SSRN 3177507*, 2018. (Cited on page 68)

[94] M. Zanforlin, D. Munaretto, A. Zanella, and M. Zorzi, "Ssim-based video admission control and resource allocation algorithms," in *2014 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. IEEE, 2014, pp. 656–661. (Cited on pages 68 and 80)

All links were last followed on March 01, 2023.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

---

place, date, signature