



Höhere Technische Bundeslehranstalt
und Bundesfachschule
im Hermann Fuchs Bundesschulzentrum

Autonomous Car Mapping and Tracking

Diploma Documentation

School autonomous focus on Mobile Computing and Software Engineering

Performed in school year 2019/2020 by:

Alexander Voglsperger (AV), 5AHELS

Simon Moharitsch (SM), 5AHELS

Advisors:

Dipl. Ing. Müller Gerhard

February 4, 2020

Thema:

Autonomous Car Mapping and Tracking

Subtopics and Editor:

Implementing SLAMS and DeepTAM, Image Pre-Processing

Alexander Voglspurger, 5AHELS

Advisors: Dipl. Ing. Müller Gerhard

Implementing DeepTAM, Gathering Trainingdata

Simon Moharitsch, 5AHELS

Advisors: Dipl. Ing. Müller Gerhard

Projectpartner:

Designation: Johannes Kepler University - Artificial Intelligence Lab

Address: Altenberger Straße 69

ZIP, location: 4040 Linz, Austria

Contact person: Dr. Nessler Bernhard

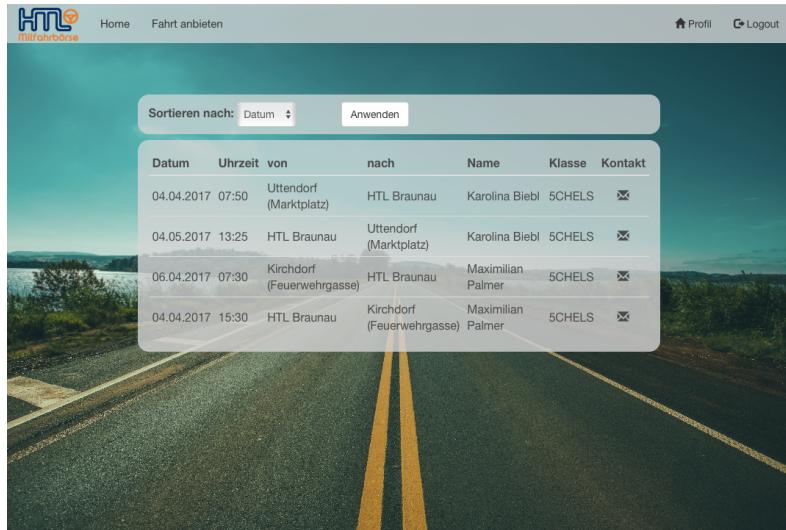
Phone: +43 (0)732 2468 4539

E-Mail: nessler@ml.jku.at

DIPLOMA DOCUMENTATION

Author	Alexander Voglsperger, Simon Moharitsch
Vintage Schoolyear	5AHELS 2019/2020
Topic of the diploma documentation	Autonomous Car Mapping and Tracking
Cooperation- partner	Johannes Kepler University - Artificial Intelligence Lab
Taskdefinition	A camera delivers a sequence of 2D pictures of the environment in front of a car. Only sing these pictures the programm should generate a 3D map. Since the pictures don't contain any depth information a SLAM (Simultaneous Localization and Mapping) should be applied.
Realization	As a foundation ROS was used because it is freely available and has an active community supporting the project. The ORB SLAM and LSD SLAM have already been implemented in ROS as nodes and can be used with changing a few things to get it working. DeepTAM is fairly new and hasn't been implemented into ROS yet.
Outcome	<p>Loreum ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Loreum ipsum dolor sit amet. Loreum ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Loreum ipsum dolor sit amet.</p>

**Illustrative graph, Landing page of HTL-carpooling:
photo
(incl. explanation)**



**Accessibility of
diploma thesis**

HTL Braunschweig archive, or
[https://diplomarbeiten.berufsbildendeschulen.
at/](https://diplomarbeiten.berufsbildendeschulen.at/)

Approval (date / signature)

Examiner

Head of College / Department

Statement

I declare in lieu of oath that I have written this diploma thesis independently and without outside help, have not used sources and aids other than those stated directly and have made the sources used verbatim and in terms of content taken as such recognizable.

Braunau/Inn, 04.02.2020

Alexander Voglsperger

Location, Date

Author

Signature

Braunau/Inn, 04.02.2020

Simon Moharitsch

Location, Date

Author

Signature

Contents

Abstract	ix
Summary	x
1 SLAM^{AV}	1
1.1 What is SLAM?	1
1.2 Application	1
1.3 History	1
1.4 Existing Methods	2
2 Robot Operating System^{AV}	3
2.1 What is the Robot Operating System?	3
2.2 Design	3
2.2.1 Topics	4
2.2.2 Nodes	4
2.3 Licenses and OS	4
2.4 Tools	4
2.4.1 Rosbag	4
2.4.2 RQt	4
2.4.3 CatKin	5
2.4.4 Rviz	5
2.4.5 Roslaunch	6
3 Artificial Neural NetworksSM	7
3.1 What is a Artificial Neural Networks?	7
3.2 Areas of Application	7
3.3 Components of an ANN	7
3.3.1 Neurons	7
3.3.2 Connection and weights	7
3.3.3 Propagation function and activation function	8
3.3.4 Bias	8
3.4 Organization	8
3.4.1 Feed Forward ANN	8
3.4.2 CNN	9
3.4.3 Encoder-Decoder-Based Architecture	9
4 ORB-SLAM2^{AV}	10
4.1 What is ORB-SLAM?	10
4.2 How does the ORB-SLAM work?	10
4.2.1 Extracting Keypoints	10
4.2.2 Loop-closing and Bundle Adjustments	11

4.2.3	Localization	11
4.2.4	Input/Output	11
5	LSD-SLAM^{AV}	12
5.1	What is LSD-SLAM?	12
5.2	Difference Feature-Based and Direct	12
5.3	How does the LSD-SLAM work?	13
5.3.1	Components that make up the LSD-SLAM	13
5.3.2	Depth Map Estimation	13
5.3.3	Map optimization	14
5.3.4	Input/Output	14
6	DeepTAMSM	15
6.1	What is DeepTAM?	15
6.2	Tracking	15
6.2.1	Network Architecture	15
6.3	Mapping	16
6.3.1	Network Architecture	16
6.3.2	Training	16
7	Workflow^{AV/SM}	17
7.1	Used Hardware ^{AV}	17
7.2	Used Software ^{AV}	17
7.2.1	Raspberry Pi	17
7.2.2	PC	17
7.3	Setup ^{AV}	18
7.3.1	Streaming video from Pi to PC	18
8	INFO: Gliederung und InhaltSM	20
8.1	Gliederung	20
8.2	Beispiel Gliederung	20
8.3	Inhalt	21
9	INFO: Zitieren, Abbildungen, Quelltext^{AV}	23
9.1	Abbildungen	23
9.2	Zitate, Quellen, Fußnoten	24
9.3	Listings, Code	25
9.3.1	Beispiele	25
10	Fazit und Persönliche Erfahrungen	27
10.1	Fazit	27
10.2	Persönliche Erfahrungen	27
A	\LaTeX^{AV}	28
A.1	Die Vorlage	28
A.2	Programme	28
A.3	Bitmap Fonts	28
A.4	LaTeX Quelltext	29
A.5	Gerüst	29

A.6 Formatierungen	30
A.7 Überschriften	31
A.8 Autoren SM	31
A.9 Bilder einfügen	31
A.10 Querverweise	32
A.11 Aufzählungen	32
A.12 Mathematische Formeln	33
A.13 Programm Quelltext	33
A.14 Code im Text	34
A.15 Seitenümbreche	34
A.16 Quellen und Literatur	35
A.17 Links	35
A.18 Fußnoten	36
A.19 Tabellen	36
A.20 Mehrspaltiger Text	36
Glossary	37
Abbildungsverzeichnis	39
Quelltextverzeichnis	40
Authors	42

Abstract

Im Vorwort teilt der Bearbeiter dem Leser wichtige Tatsachen mit, die Erklärungen zu seiner Arbeit beinhalten – z.B. die Motivation für die Bearbeitung des Themas oder besondere Schwierigkeiten bei der Bearbeitung und/oder Materialbeschaffung.

Hier können auch Mitteilungen persönlicher Natur enthalten sein – z.B. Dank an Institutionen/Personen für die geleistete Unterstützung.

Summary

Die *Zusammenfassung* oder auch *Kurzfassung* soll den Inhalt der Diplomarbeit auf maximal einer halben Seite zusammenfassen.

Dieses Dokument dient als Vorlage und Beschreibung für die Dokumentation der Diplomarbeit. Es werden Hinweise zur Erstellung einer guten Dokumentation gegeben. Dies betrifft welchen Inhalt die Arbeit haben soll genauso wie welche Regeln eingehalten werden müssen und mit welchen technischen Mitteln das Dokument erstellt werden kann.

Beim Inhalt dieser Arbeit wurden alle grundlegenden Qualitätsregeln eingehalten und kann daher als Musterlösung gesehen werden. Zum Erstellen wurde das Textsatzsystem L^AT_EX verwendet. Es ist vorgesehen, dass der L^AT_EXQuelltext dieses Dokuments als Ausgangspunkt für die eigene Dokumentation verwendet wird.

Dieses Dokument sollte unbedingt aufmerksam gelesen werden ehe mit der eigenen Arbeit begonnen wird.

1 SLAM^{AV}

1.1 What is SLAM?

SLAM is an acronym and stands for **S**imultaneous **L**ocalisation **A**nd **M**apping. “*SLAM is concerned with the problem of building a map of an unknown environment by a mobile robot while at the same time navigating the environment using the map.*” [1]

This problem is thus a chicken-and-egg problem because neither the map or location are known, and have to be estimated at the same time. Cameras, ultrasonic sensors and laser radar (lidar) sensors are most commonly used for fetching the 2D and 3D data of the robot’s surroundings [2].

There are several algorithms out there, which try to solve this problem using algorithms and some even deep learning. Mostly they achieve an approximate map, which is done in a reasonable time span. Many popular SLAM-algorithms use methods that include *particle filters*, *extended Kalman filter* and *Covariance intersection*[1] [3].

1.2 Application

The biggest selling point for using SLAM implementations is pretty simple. Many places where autonomous robots may be required don’t have good enough maps that are up-to-date , if the exist at all or it might be in an environment where positioning for instance GPS can’t be used properly [4]. If slams weren’t be used then someone would have to go to the place and make a map. This would delay the mission and add to the costs.

With a robot, that is capable of using a SLAM method to detect and locate itself in the unknown surroundings this wouldn’t be an issue. The robot could go in, generate a map that updates itself and use it to navigate around.

Existing approaches that are used are in self-driving cars, unmanned aerial vehicles, autonomous underwater vehicles, planetary rovers and newer domestic robots.

1.3 History

The decisive work in SLAMs was done in the research by R.C. Smith and P. Cheeseman who worked on the representation and estimation of spatial uncertainty in 1986. Another major work in this area was done by a research group with the head being *Hough F. Durrant-Whyte*. Durrant-Wyte and his group showed that answer to SLAMs lies in the nearly infinite amount of data that can be used. This lead to the motivation of finding algorithms which are trackable and approximate in a time realistic manner.

Sebastian Thrun was playing.

1.4 Existing Methods

There exists a big variety of SLAM methods, that try to achieve the same goal using different approaches [5]. Most known or popular are the following:

- EKF SLAM

Utilizes the extended Kalman filter. The algorithm uses the likely-hood for data association. It was the go-to SLAM from 1990 to the early 2000s until Fast SLAM was introduced [6].

- Fast SLAM

Works recursively so it scales logarithmically to the scale of the landmark. It can handle much bigger landmarks than the EKF-SLAM ever could without requiring as much computing power [6].

- ORB-SLAM2

It's a real-time SLAM library for Monocular, Stereo and RGB-D cameras. It can detect loops and relocate the camera in real-time. It uses camera trajectory and sparse 3D reconstruction to get information out of the image sequence [7].

- DVO-SLAM

Implements a *dense visual SLAM* system for RGB-D cameras. It's based on *Dense Visual Odometry* and was extended to include frame-to-key matching with loop closure to older key-frames [8].

- RGB-D SLAM

Utilizes the depth information of a RGB-D camera, e.g., Microsoft Kinect or Intel Real-Sense Cameras [9].

- LSD-SLAM

It's a direct monocular SLAM. It tracks the *direct image alignment* and estimates geometry in form of *semi-dense depth map* instead of relying on keypoints [10].

2 Robot Operating System^{AV}

2.1 What is the Robot Operating System?

The Robot Operating System, which is also known as ROS is a flexible framework for writing software that gets utilized on robots. It was founded by Willow Garage in 2012 and gets primarily maintained by the Open Source Robot Foundation (OSRF) [11]. In Europe the project gets coordinated by the Fraunhofer IPA in form of the *ROS Industrial Consortium Europe*. ROS is a middle-ware which is not a operating system but provides services that manage hardware abstraction, low-level device control, message-passing between processes and package management. “*It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.*” [12]

2.2 Design

The processes of ROS are represented in nodes which are in a graph structure. Everything gets managed by a single process called *ROS Master*, to whom all other nodes register on startup. But instead of sending all of the messages over the master, the master sets up a peer-to-peer connection between the nodes. This decentralized architecture is helpful as many robots consists of many computer hardware which is connected via a network and are likely to transfer big messages [13].

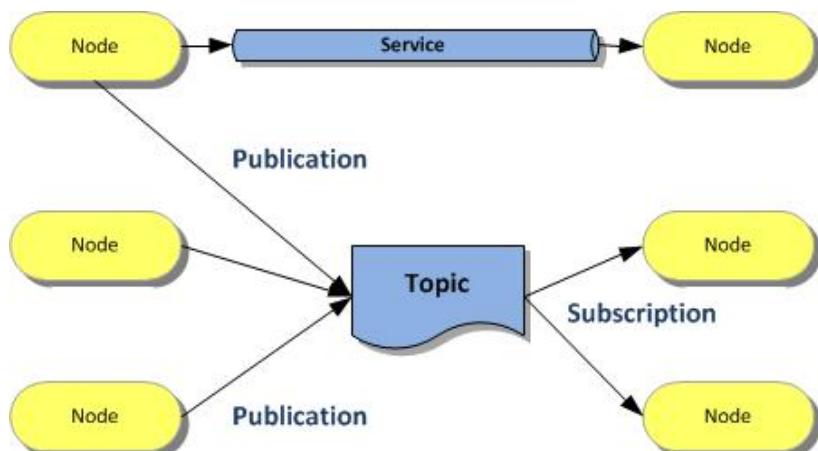


Figure 2.1: ROS structure
Source: <https://tinyurl.com/tkf2smq>

2.2.1 Topics

It is based on a topic system, where a topic acts like a bus over which nodes send and receive messages. Each topic must be unique in its name, which is usually set by the developer. The process of publishing and subscribing is handled anonymously so that no node knows which nodes are sending and receiving messages on a certain topic.

2.2.2 Nodes

A node, which represent a single running process, can provide data using a matching topic and publish it to the system, where theoretically every other node can subscribe to it, to get the data.

2.3 Licenses and OS

The language-independent tools and the main client libraries have been released under the BSD license and as such they are open source software for commercial and research use. The majority of 3rd party packages are released under several other open-source licenses.

The ROS libraries are geared toward a UNIX-System which is mainly due to their dependence on a large collection of open source software and libraries. For example *Ubuntu* is in the list of supported operating systems, while others like *Fedora*, *MacOS* and *Windows* are “experimental” and are mainly supported by the community [14].

2.4 Tools

One of the core functionalities that ROS provides are the tools which allow the developers to visualize 2D and 3D data, record data, easily navigating ROS packages, creating complex scripts that configure and setup processes. Thanks to this tools it simplifies and provides solution for common robotic development.

2.4.1 Rosbag

Rosbag is a tool that can be used over the command line to record, playback and store ROS message data. The data gets stored in a file called bag, where it records the messages as they come in. It's possible to play these bag files. By doing this the recorded messages get published into the system, as they were live. It's very handy if you need data for later development or to use a bunch of different scenarios for testing

2.4.2 RQt

RQt provides a graphical overview of the ROS computation graph. It shows the nodes and how they are connected to each other. It also shows if a node is even subscribing to a topic or publishes something. Other than that it can be used to subscribe to different topics and show them directly in RQt.

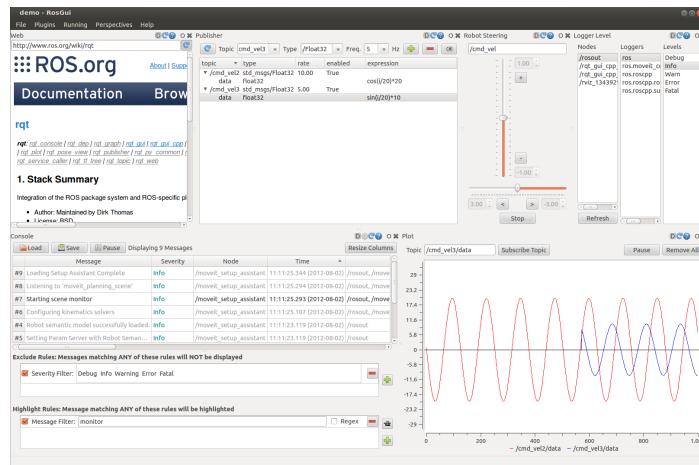


Figure 2.2: RQt interface

Source: <https://wiki.ros.org/RQt>

2.4.3 CatKin

Catkin is the newer ROS build system, which compiles the files in the source folder. It is based on CMake and is cross-platform and language-independent as most other ROS tools.

2.4.4 Rviz

A visualizer for three-dimensional data where robots, environments and sensor information can be visualized. It is highly customisable with display many types of visualisation and plugin support.

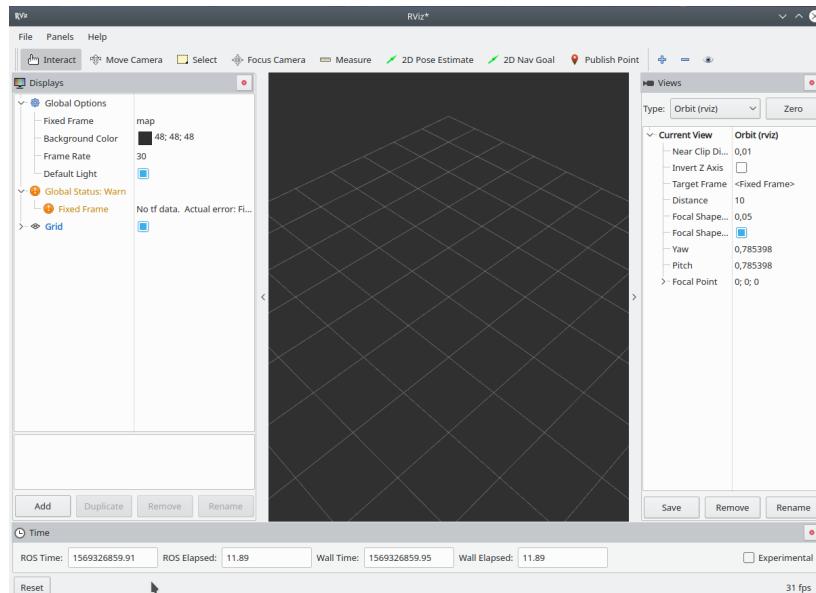


Figure 2.3: Rviz interface

2.4.5 Roslaunch

Roslaunch is a tool for launching multiple ROS nodes and setting parameters on startup. It can be used to launch nodes locally or remotely on a server. The configuration for a start script is written in a launch file using XML. In these files it's easy to make a automated startup and configuration process to be executed with one command. It's possible to execute launch files in other launch files to chain them together.

3 Artificial Neural NetworksSM

3.1 What is a Artificial Neural Networks?

Artificial Neural Networks(ANN) are inspired by biological neural networks that constitute animal brains. Important to notice is that they are not faithful models of biologic neural or cognitive phenomena. In fact most of these models are more closely related to mathematical and/or statistical models(For Example: clustering algorithms). Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules.

3.2 Areas of Application

ANN are viable computational models for a wide variety of problems, including pattern classification, speech synthesis and recognition, adaptive interfaces between human and complex physical system, function approximation, associative memory, clustering, forecasting and prediction, combinatorial optimization, nonlinear system modeling, and control [15]

3.3 Components of an ANN

Simplified a ANN consists of three main components(neurons, connection and the weight associated with them) the propagation function and a bias. In the following topics I will give you a short summary what these components are and afterward I will explain how they work together.

3.3.1 Neurons

Neurons are elementary units in an ANN. A neuron gets one ore more inputs and depending on the value of the inputs the output is set. A neuron can get its inputs from other neurons or, if its at the beginning, from the source of the data that needs to be processed. Depending on the Type of ANN they are placed in different structures. In most cases the output of a Neuron is a number between 0 and 1.

3.3.2 Connection and weights

These Neurons are Connected. Which neurons are connected with others depends on the structure. The weights characterize how important a connection between neurons is.

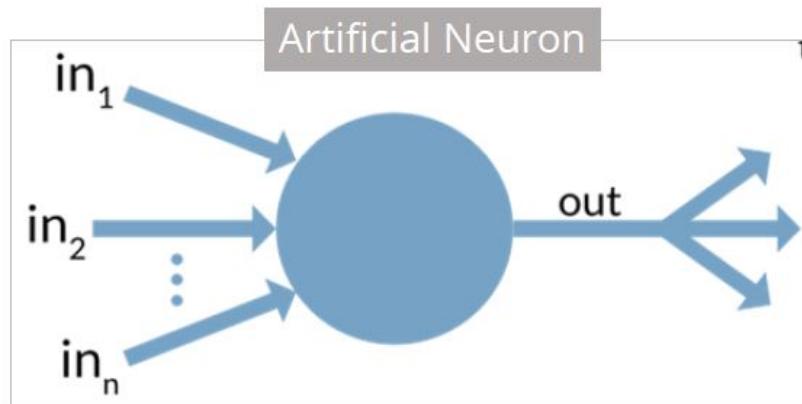


Figure 3.1: General view of Neurons
 Source: <https://tinyurl.com/yyfthk7c>

For example:

A neuron, we call it base for this example, has two neurons connected to it as inputs. The weight of the connection of the first neuron has a bigger weight than the second connection. That means the output of the base depends more on the input of the first neuron

3.3.3 Propagation function and activation function

This is a function which takes the Inputs of a neuron, the weight of these connections and the bias and adds them up. The resulting value is processed by the activation function which sets the output. One of the most common activation function is the sigmoid function because it is not a step function which means the output doesn't change instantaneously. That's important for the training algorithm.

3.3.4 Bias

The bias is a Neuron which has no Inputs. A bias is used to shift the decision boundary to the left or right.

3.4 Organization

A artificial neural network can be organized in many different ways.

3.4.1 Feed Forward ANN

The following picture demonstrates a feed forward ANN. There are a variable number of hidden layers depending on the purpose of the neural network. Nothing in the hidden layer is visible.

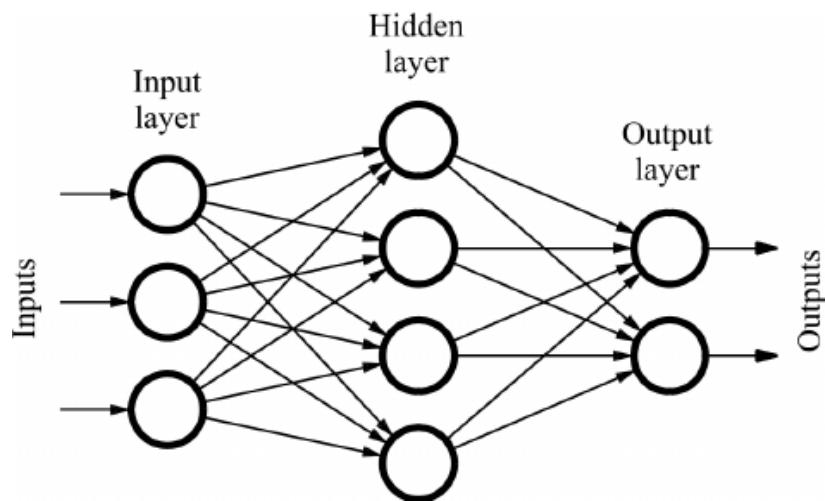


Figure 3.2: Feed forward network
Source: <https://tinyurl.com/y3yaqboq>

3.4.2 CNN

3.4.3 Encoder-Decoder-Based Architecture

4 ORB-SLAM2^{AV}

4.1 What is ORB-SLAM?

ORB-SLAM2 is a versatile, real-time SLAM implementation which uses Mono-, Stereo- and RGB-D cameras. It's designed to generate a 3D map from prominent points in the picture and keypoints. It features loop closing, re-localization and a reusable map [16]. It works in a wide variety of use cases. The SLAM can be used on a small hand-held camera or drones up to self driving cars. ORB-SLAM2 is based on ORB-SLAM and was inter alia developed by Raúl Mur-Artal who already worked on ORB-SLAM.



Figure 4.1: ORB-SLAM Example image

Source: <https://tinyurl.com/ruvnj39>

4.2 How does the ORB-SLAM work?

4.2.1 Extracting Keypoints

The SLAM uses a feature-based method. This means that it extracts features on prominent keypoints throughout the image input. These feature information is then distributed to all operations which handle them independent from the camera type. [16]

This is how finding these keypoints works on different camera types:

- Stereo Image
For a stereo camera setup the keypoints get extracted for both images separately and then the left keypoints are searched on the right image. Then the found points get compared to the original ones that were found on the right side
- RGB-D Image
On a RGB-D camera keypoints get extracted using prominent keypoints and then

calculating the approximate position using the depth information from the information from the depth sensor.

- Monocular Image

On a Monocular image the approximate position gets triangulated by using multiple images. The Disadvantage is that they don't provide a scale information and only do rotational and translational movement estimations.

4.2.2 Loop-closing and Bundle Adjustments

Loop-closing and bundle adjustments are performed in two steps. First the loop-closing will happen when the system detects overlapping environments where the system changes scaling to reconnect certain parts as scale drifting will occur on monocular cameras.

Second step is the bundle adjustment, which gets executed after a successful loop-closing, where the system tries to optimize all keypoints and keyframes using the Levenberg-Marquardt method (alternative to Gauss-Newton method).[17] Also the camera orientation and position will be optimized to compensate errors in tracking. All the bundle adjustment is done in a separate thread since this is a heavier task.

When finished, the updated and optimized keyframes and keypoints get merged into the original keyframes and keypoints [16].

4.2.3 Localization

When an area has been mapped well in the past the *Localization Mode* can be turned on which deactivates the Local Mapping and the Loop Closing thread and thus saving computing power. Locating is done by continuously comparing the previous points with the current points of the image. This works when an area is unmapped but drifting might add up. Matching the current points with the one on the map will ensure that it is drift-free [16].

4.2.4 Input/Output

Input Data: rectified Monochrome/Color images

Output Data: Rough 3D map with pixel-points and image with current prominent keypoints

5 LSD-SLAM^{AV}

5.1 What is LSD-SLAM?

LSD-SLAM stands for Large-Scale Direct Monocular SLAM and is a fairly new real-time monocular SLAM that is fully direct-driven instead of relying on keypoint/keyfeature [10]. The algorithm works on the image intensity for tracking and mapping at the same time. This method allows building large-scale maps on normal processors that are consistent when comparing it to current state-of-the-art algorithms.

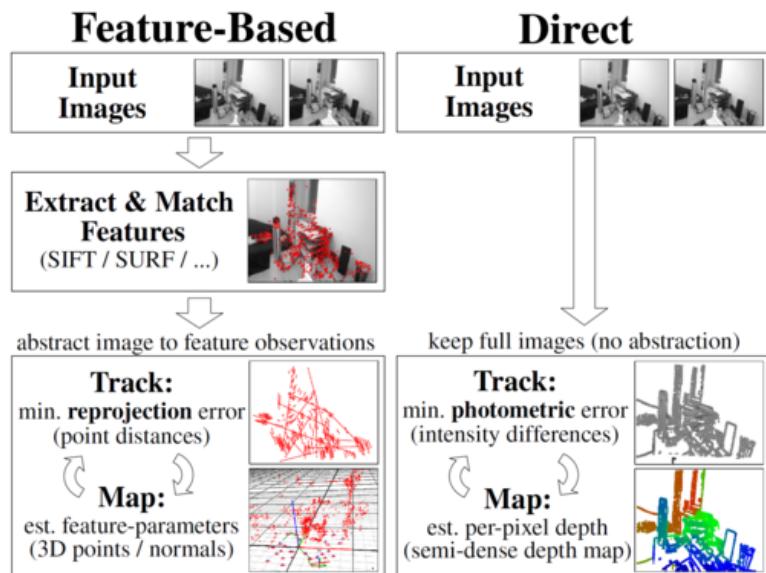


Figure 5.1: Feature-Based and Direct Difference

Source: <https://tinyurl.com/ycmjhb9d>

5.2 Difference Feature-Based and Direct

- Feature-based

A feature-based SLAM (e.g. ORB-SLAM 4) looks for distinctive points in the image and then uses only these keypoints to process the information. When there are only a few prominent keypoints (e.g indoor, tunnels) the result will not be that accurate.

- Direct

Direct based SLAMs use all information that's provided in the image. This does not

only include distinctive points but also edges and sometimes surfaces and thus makes it possible to create a more accurate and denser 3D map [10].

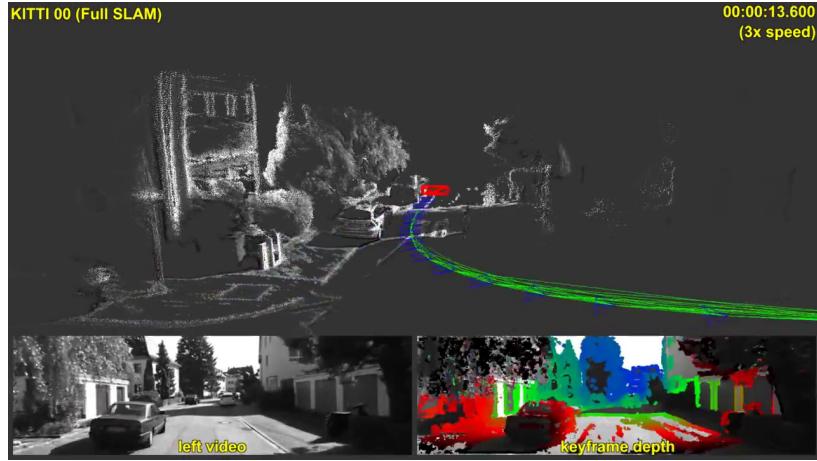


Figure 5.2: LSD-SLAM Example image
Source: <https://tinyurl.com/qkuamyy>

5.3 How does the LSD-SLAM work?

5.3.1 Components that make up the LSD-SLAM

- Tracker

The *Tracking* component uses the current frame in relation to the last frame to continuously track the camera movement.

- Depth Map Estimation

Depth map estimation is done using tracked frames to refine or replace the current frame. Depth information is calculated by filtering over a small per-pixel baseline. If the camera has moved too far or the image has changed too much a new keyframe is initialized [10].

- Map Optimization

When a keyframe gets replaced as a tracking reference, the refinement process stops and it gets included in the 3D depth map. *Map optimization* then starts working to detect loop-closures or scale-drifts. This is done by a similarity transformation to frames that were taken nearby.

5.3.2 Depth Map Estimation

New keyframes are created when there where no frames before or the camera has moved/rotated so far that the set threshold has been exceeded. When this happens the new latest frame is chosen to become the new keyframe and the keypoints from the previous keyframe get projected onto the new one. The process is followed by scaling to fit the needs of the Direct Image Alignment. When that is done the keyframe replaces the previous ones and gets used to track the subsequent frames.

Not every frame results in a new keyframe. Frames that don't make it to a new keyframe are used to improve and refine the current keyframe. The refinement is done by using a small stereo comparison for regions in an image where the expected use for advancement is higher. The result of this comparison then gets merged into the existing 3D point cloud to add potentially new information or refining existing pixels [10].

5.3.3 Map optimization

Scaling, rotation and movement isn't always perfect, which results in drift. Even if the drift effect is little it adds up, which might result in some very off map [10]. The Pose-Graph-Optimization is a optimization algorithm which aims to fix these drifts with pretty good results. The advantages of the pose-graph-optimization are that it's fast and vulnerable for poor initialization estimates [18].

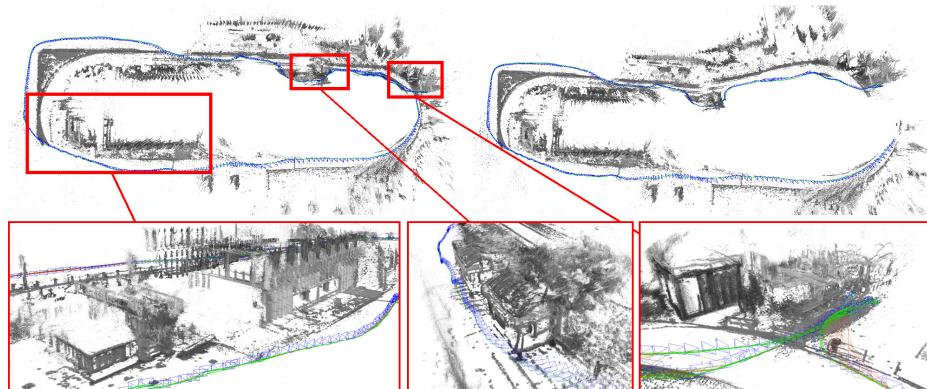


Figure 5.3: LSD-SLAM Pose-Graph-Optimization (left after loop-closure, right before loop-closure)

Source: <https://tinyurl.com/qkuamyy>

5.3.4 Input/Output

Input Data: rectified monocular image, camera info

Output Data: image with probability colored points, 3D point cloud

6 DeepTAMSM

6.1 What is DeepTAM?

DeepTAM provides a keyframe-based dense camera tracking and depth map estimation system that is entirely learned. The idea of DeepTAM is based on DTAM [19]. The generic idea is: drift-free camera tracking via a dense depth map towards a keyframe and aggregation of depth over time. But the way to implement this concept is different. In the DeepTAM deep networks are used for tracking and mapping. These networks learn only from data. It also processes more than two images for the 6 DOF egomotion and depth estimation. With that it can avoid the drift of the use of keyframes and as more keyframes come in it can refine the depth map.

6.2 Tracking

The main objective is to estimate a 4×4 transformation matrix T . This matrix maps a point in the keyframe coordinate system to the coordinate system of the current camera frame. DeepTAM uses an more efficient way. It generates a virtual keyframe and tries to predict the increment instead of trying to estimate T . For more details see [20].

6.2.1 Network Architecture

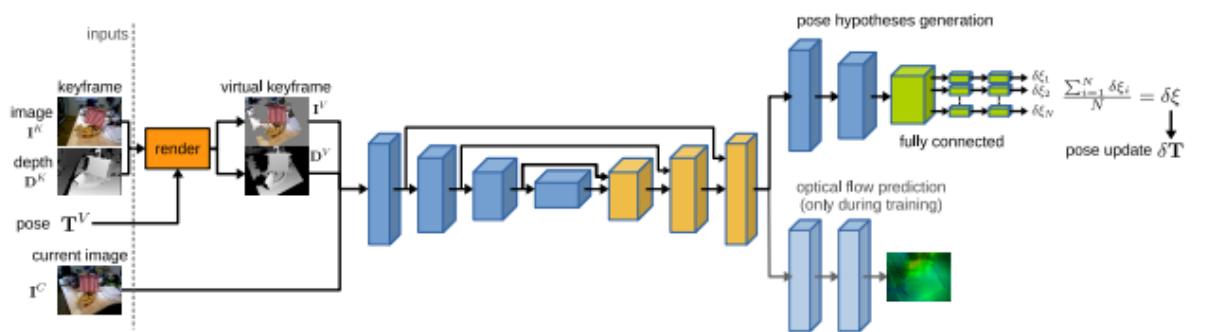


Figure 6.1: Schematic of DeepTAM

Source: <http://lmb.informatik.uni-freiburg.de/Publications/2019/ZUB19a>

To estimate the 6 DOF pose between a keyframe and an image the encoder-decoder-based architecture is used. To estimate camera motion you have to relate the keyframe to the current image. Because of that DeepTAM uses optical flow as an supportive task. With

this optical flow the network is ensured to take advantage of the relationship between both frames. It uses two network branches for predicting the pose. One is the optical flow prediction and the other is the pose hypotheses generation. This improves the accuracy for the pose prediction.

6.3 Mapping

DeepTAM computes a set of depth maps every keyframe. For good quality depth maps, information will be accumulated in a cost volume. From this cost volume the depth map will be extracted by means of a convolutional neural network.

6.3.1 Network Architecture

6.3.2 Training

In about 8 days in total the training of the mapping network can be accomplished on a NVIDIA GTX 1080Ti.

optical flow

7 Workflow^{AV}/_{SM}

7.1 Used Hardware^{AV}

For video capturing a *Raspberry Pi 3B+* with a *Raspberry Pi Camera V2* or a *Raspberry Pi Wide Angle Lense Camera* are used. The Raspberry Pi sends the video feed to a separate more power full PC over WiFi using a Python3 script.

For processing a *Lenovo ThinkStation S20* or a *Lenovo W550s* are used depending on the amount of processing power is required. For more intense work a server access at the Johannes Kepler University was supplied to work on their system.

As the work is based around implementing it on the Audi Autonomous Driving Cup (AADC) car a remote controlled model car was borrowed for a few weeks.

7.2 Used Software^{AV}

7.2.1 Raspberry Pi

The Raspberry Pi is running Raspbian Buster since it is well optimized for the mini computer and only required to be able to execute a Python script to send the raw video feed over http to the the processing device.

7.2.2 PC

The ThinkStation and the laptop are running Kubuntu 18.04, which is basically Ubuntu but has a GUI that's a more like Windows and is supported until May 2023.

The ThinkStation has a eight core Intel Xeon CPU, a GTX 1660TI and 12GB of RAM inside. The Laptop has a four core Intel i7 and 8GB of RAM built in.

ADTF

At first Ubuntu 16.04 with Automotive Data and Time-Triggered Framework (ADTF) was used since it's the recommended environment by the AADC car manufacturer DigitalWerk. There were many compatibility ans stability issues and it is very difficult to get into the whole system as it's not very beginner friendly. Students at the JKU said that they spent at least four weeks to know how it somewhat worked. After trying to get the basics of ADTF working it was clear that switching to ROS might be better.

ROS

Running ROS Melodic on Kubuntu 18.04 was pretty straight forward. The instructions on the ROS website are very clear and can be directly copied without issues. The principle of

the workspace is also easy to understand. In the source folder the modules get put in and when compiling the modules automatically generates a setup file to use them.

7.3 Setup^{AV}

As the PC and laptop are not the best idea to run around with, a Raspberry Pi is used instead to stream the video over WiFi to the PC/laptop which are connected to the router over LAN. This makes the camera setup very portable as the pi, camera and powerbank are packed together and don't have much weight. The PC can sit somewhere where and just processing the received video signal.

7.3.1 Streaming video from Pi to PC

Enabling Camera

To use a camera on a Raspberry Pi the interface needs to be enabled first. This can be done in the built-in tool called *raspi-config*. In this tool under the subsection called *Interfacing Options* there is a option with the name *Camera*. When this is done the camera can be used after a restart.

Python Script

In the code snipped 7.1 at first a *piCamera* instance with the name *cam* is created. As parameters the resolution gets set to *1280x720* pixels and the framerate is set to *30* frames per second (FPS). If needed the image can be rotated, e.g the camera is mounted upside down. When starting the camera a output and format are expected. For the output a separate class is used which sets how and when a new frame can be published and for the format the *mjpeg* video codec is chosen, as a pack for getting mjpeg-streams already exists in ROS and it's not power hungry when running it on the Raspberry Pi.

After the camera “recording” has started successfully the server is started to make the stream accessible to other devices. The server runs until the user closes the script using *CTRL + C*. After closing the server the *finally* block gets called, where the camera “recording” is stopped so that other programs can use the camera again.

```

1 #Only resolution and framerate in Constructorparameters
2 with picamera.PiCamera(resolution='1280x720', framerate=30) as cam:
3     output = streamingOutput()
4     # Rotation if needed
5     cam.rotation = 180
6     #start stream
7     cam.start_recording(output, format='mjpeg')
8
9     try:
10         #IP, Port
11         hostAndPort = ('',5000) # '' as IP automatically get's ip
12         server = streamingServer(hostAndPort, streamingHandler)
13         server.serve_forever() # Serves as long as script is running
14     finally:
15         cam.stop_recording() # Stops stream to make camera accessable from other
                           # apps again

```

Listing 7.1: Main Function of CamStream Feed

The streamingHandler that is shown in snipped 7.2 handles the actions that are taken when client connects to the RaspberryPi. At the beginning it checks if the client is requesting the `/stream.mjpg` file. If the client is not requesting that specific file a *404 Not Found* Error is returned. But if the correct file is requested at first a *200 OK* code. In addition to the status code headers are send, which tell the client to not use cache. After sending the HTTP OK to the client a permanent loop is startet which always waits until a new image from the camera is ready and then sends it to the client as an jpeg image. The loop ensures that the always client receives the latest image and so creates a video. Should the client drop the connection an exception is raised which causes the loop to stop and end the handler for that specific client until the client connects again.

```

1 class streamingHandler(server.BaseHTTPRequestHandler):
2     def do_GET(self):
3         # Check if user is requesting stream
4         if self.path == '/stream.mjpg':
5             # yes --> Send Stream
6             # send header for beginning streaming
7             self.send_response(200) # OK
8             self.send_header('Age', 0)
9             self.send_header('Cache-Control', 'no-cache, private')
10            self.send_header('Pragma', 'no-cache')
11            self.send_header('Content-Type', 'multipart/x-mixed-replace,
12                           boundary=FRAME')
12            self.end_headers()
13
14            # sending Stream
15            try:
16                while True:
17                    with output.condition:
18                        output.condition.wait() # Wait for new image
19                        frame = output.frame
20                        # Header for sending image
21                        self.wfile.write(b'--FRAME\r\n')
22                        self.send_header('Content-Type', 'image/jpeg')
23                        self.send_header('Content-Length', len(frame))
24                        self.end_headers()
25                        self.wfile.write(frame)
26                        self.wfile.write(b'\r\n') # End transmission of frame
27            except Exception as ex:
28                print('Client removed ' + self.client_address + ' : ' + str(ex))
29        else:
30            # User requested something else --> Send 404 Page
31            self.send_error(404)
32            self.end_headers()
```

Listing 7.2: StreamingHandler of CamStream

8 INFO: Gliederung und InhaltSM

8.1 Gliederung

Die vorhergehenden Kapitel sind Muss-Bestandteile der Diplomarbeit. Ab hier kann die Gliederung (Aufteilung in Kapitel) frei gewählt werden.

Das Dokument soll durch Kapitel und Unterkapitel übersichtlich gegliedert sein. Jedes Kapitel bekommt eine Überschrift mit Nummerierung (1.1, 2.2.1, ...). Mehr als 3 Überschriftebenen (1.1.1) sollten vermieden werden. Zusätzlich muss bei jedem Hauptkapitel oder Kapitel mit einer Fußnote vermerkt werden, wer diesen Abschnitt erstellt hat.

Umfangreichere Arbeitsergebnisse wie Schaltpläne, Messprotokolle, Datenblätter und das Projekttagebuch kommen in einen Anhang am Ende des Dokuments.

In einem Literaturverzeichnis sind alle verwendeten Quellen und Zitate zu sammeln.

Wird die Diplomarbeit von **mehreren Personen** gemeinsam erstellt muss erkenntlich gemacht werden wer für welches Kapitel verantwortlich war.

8.2 Beispiel Gliederung

Hilfestellung für eine mögliche Benennung und Gliederung der Hauptkapitel:

Problemanalyse und Spezifikation

Erläuterung des Was: Aufgabenstellung ganz detailliert (=Pflichtenheft). Hier wird erläutert, was zu machen war. Das Wie ist hier normalerweise fehl am Platz.

Entwurf

Erläuterung des Wie: Technologie mit Begründung, bzw. Abwägen der Vor- und Nachteile. Lösungswege, Algorithmen.

Implementierung

Zeigt genau die Umsetzung des Entwurfs anhand wesentlicher Quelltext-Ausschnitte.

Test und Inbetriebnahme

Erkläre wie getestet wurde und was notwendig ist um das Produkt von Null weg zu installieren.

Bedienungsanleitung

Erklärt dem Benutzer die wichtigsten Schritte bei der Bedienung des Systems. Eventuell ist eine eigenes Bedienerhandbuch für spezielle Benutzergruppen (z.B. Administratoren) notwendig.

Fazit, Schlussfolgerungen

Hier werden die Projektergebnisse zusammengefasst. Was ist gelungen was nicht. Welche Erkenntnisse wurden gewonnen.

Persönliche Erfahrungen

Hier (und nur hier) darf subjektiv aus der Ich Perspektive über das Projekt philosophiert werden.

8.3 Inhalt

Das Ziel ist die eigene Arbeit anderen (technisch versierten, aber projektfremden) Personen nachvollziehbar zu machen. Ein Mitschüler der ein gutes Informatik Fachwissen hat soll den Text verstehen können.

Die Dokumentation eines Informatik-Projekts soll **Programmquelltext** enthalten! Dieser soll gut dokumentiert und lesbar sein. Nur wenig zusätzlicher Text soll notwendig sein um das Programm zu verstehen. Wählt nur jene Programmteile aus, die wirklich interessant sind, nehmt nicht jene Dinge die man in jedem Buch nachlesen kann.

Werden zur Implementierung Libraries, Frameworks, Methoden, etc. verwendet die wesentlich über den normalen Unterrichtsstoff hinausgehen, so sollten diese kurz erklärt werden. Es ist aber nicht notwendig und nicht zielführend ganze Tutorials zu erstellen. Nach einer allgemeinen Übersicht, die zu einem grundsätzlichen Verständnis verhelfen soll, genügt es auf entsprechende Quellen zu verweisen.

Der **Umfang** der Arbeit ist nicht wesentlich, wichtig ist die Qualität des Inhalts. Ca. 40 Seiten pro Person sind eine gute Richtlinie.

Der Inhalt soll aus der **eigenen Feder** stammen. Kopieren fremder Quellen (auch wenn diese ins Deutsche übersetzt werden müssen) ist auf ein Minimum zu beschränken. Wenn kopiert wird dann ist immer genau anzugeben von wo. Siehe auch Kapitel 9.

Von Dir verfasste Texte sind Deine Visitenkarte. Bemühe Dich alles so gut zu machen wie Du nur kannst.

- Strebe nach **Perfektion** — auch was die Rechtschreibung und Grammatik angeht.
- Gib keinen Text aus der Hand mit dem Du nicht 100% zufrieden bist.
- Lösche unfertige Textstellen ehe Du den Text weitergibst.
- Suche Dir jemanden zum Korrekturlesen, in eigenen Texten übersieht man gerne Fehler die einem Anderen sofort auffallen.

Keine Erklärungen aus der **Ich-Perspektive** abgeben (Ausnahme: Fazit am Ende).

Vermeide den Text so zu schreiben wie man spricht, bei Text gelten etwas andere Regeln als bei einer Präsentation. Verzichte auf das übernehmen mundartlicher Ausdrucksweisen.

Überlege Dir beim Schreiben einer Textstelle ob der Leser das notwendige Hintergrundwissen hat um zu verstehen was Du ausdrücken willst. Hast Du schon vorher erklärt was man an dieser Stelle wissen sollte? Versteht man von was die Rede ist? Beachte, dass Du top in

das Thema eingearbeitet bist. Was Dir völlig klar erscheint ist dem Leser vielleicht nur ein spanisches Dorf.

Sei aber auch nicht zu weitschweifig. Ein guter Text ist kein langer Text sondern ein Text an dem man beim besten Gewissen nichts mehr wegkürzen kann. Schreibe zuerst etwas weitschweifiger und kürze dann radikal. Sei nicht zimperlich, wenn Dir eine Stelle nicht gefällt, lösche diese und fange von vorne an.

9 INFO: Zitieren, Abbildungen, Quelltext^{AV}

In diesem Kapitel sind Beispiele angeführt, wie Abbildungen, Zitate und Quelltext zu verwenden sind.

Die Zitate, Abbildungen und Listings werden automatisch in das Quellen- und Abbildungsverzeichnis übernommen. Das Literatur-, Abbildungs- und Listingsverzeichnis sind am Ende der Arbeit zu finden.

9.1 Abbildungen

Abbildung sind mit einer Abbildungsnummer und einer Unterschrift zu versehen die kurz die Abbildung beschreibt. Abbildungen gehören zum umgebenden Text und müssen dort erwähnt werden.

Beispiel: In Abbildung 9.1 wird das Logo der HTL Braunau dargestellt.

Wird auf eine Abbildung referenziert die sich weiter von der aktuellen Textstelle entfernt befindet so kann auch die Seitennummer hinzugefügt werden.

Beispiel: Siehe Abbildung 9.1 auf Seite 23.



Figure 9.1: Logo der HTL Braunau.

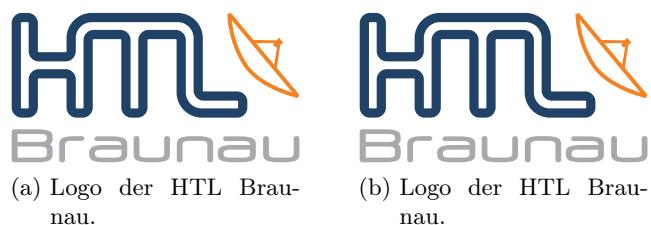


Figure 9.2: Zwei Logos der HTL Braunau in einer Abbildung zusammengefasst.



Figure 9.3: Dreimal das Logo der HTL Braunaau.

9.2 Zitate, Quellen, Fußnoten

Es muss dem Leser möglich sein alle dargestellten Informationen selbst zu überprüfen. Daraus gilt der wissenschaftliche Grundsatz: Wer Informationen/Erkenntnisse verwendet die nicht von einem selbst stammen muss dies eindeutig kennzeichnen. Das gilt für Bücher, Zeitschriftenartikel aber auch für alle Internetquellen.

Dem Leser Hinweise darüber zu geben wo genauere/weitere Informationen zu einem Thema zu finden sind ist ein weiterer Grund für Zitieren.

Es ist erlaubt fremde Quellen zu zitieren solange dies eindeutig erkennbar ist und sich die zitierten Stellen auf einen kurzen Absatz mit wenigen Zeilen beschränken. Auch wenn nicht wortwörtlich zitiert wird ist die Quelle der Information anzugeben.

Wörtliche Zitate werden durch Anführungszeichen begonnen und beendet sowie kursiv geschrieben:
“ Wissenschaftliches Plagiat: Man kann sich zwar mit fremden Federn schmücken, aber man kann nicht mit ihnen fliegen.” [?]

Längere Zitate werden durch Einrücken vom normalen Text abgesetzt:

“We tend to think of navigating a website as clicking from page-to-page via some kind of global navigation that’s always visible. When it comes to a single page, we often think scrolling is the one and only way to move from one end to the next.”[?]

Die eckigen Klammern mit Nummer kennzeichnen die Quelle. Am Ende des Dokuments werden alle Quellen im Literaturverzeichnis aufgelistet. [?].

Das Verwenden fremden Gedankenguts ohne die Quelle anzugeben ist ein Plagiat (geistiger Diebstahl, [?]) und unter Umständen sogar eine Urheberrechtsverletzung.

Verweise in das Literaturverzeichnis sind nicht auf Zitate beschränkt sondern können auch eingesetzt werden um darauf hinzuweisen wo zu einem Thema mehr Informationen erhältlich sind.

Es ist zunehmend eine Kurzzitierweise in Fußnoten üblich: Nachname des Verfassers, Kurztitel, Seitenangabe.¹

¹ John Doe, “Verwendung von Fußnoten”, Seite 12.

9.3 Listings, Code

Quelltext ist ein wesentlicher Bestandteil einer Informatik Diplomarbeit. Listings werden wie Abbildungen nummeriert und mit einer Unterschrift versehen. Ebenfalls müssen sie im Text referenziert werden.

9.3.1 Beispiele

Listing 9.1 auf Seite 25 zeigt ein Hallo Welt Programm.

```

1 #include <stdio.h>
2 #include <conio.h>
3 #include <stdlib.h>
4
5 void main()
6 {
7     printf ("Hello!");
8     getch ();
9
10    system ("cls"); // Kommentar
11
12    printf ("Hello World!");
13    getch ();
14 }
```

Listing 9.1: Mein erstes C-Programm.

Das PHP Programm 9.2 dient zum ermitteln aller Dateien eines Unterverzeichnisses.

```

1 <?php
2 if ($handle = opendir(realpath(__DIR__ . '/../../userimages'))) {
3
4     while (false !== ($file = readdir($handle))) {
5         echo "$file//";
6     }
7
8     closedir($handle);
9 }
10 ?>
```

Listing 9.2: Alle Dateinamen ausgeben.

```

1 // Calculates the mouse position.
2 function getMousePos(canvas, evt) {
3     var rect = canvas.getBoundingClientRect();
4     return {
5         x: evt.clientX - rect.left,
6         y: evt.clientY - rect.top
7     };
8 }
```

Listing 9.3: Mausposition ermitteln.

```

1 /* Zugriff auf ein HTML Tag */
2 body {
3     background-color: blue;
4 }
```

```

5  /* Zugriff auf eine Klasse */
6  .class {
7      font-weight: bold;
8  }
9
10 /* Zugriff auf eine ID */
11 #id {
12     color: red;
13 }
14
15 /* Zugriff auf eine Pseudoklasse */
16 #id:nth-child(@number*3-2) {
17     color: blue;
18 }
19

```

Listing 9.4: Kurzer CSS–Quelltext.

```

1 <html>
2   <head>
3     <title>Test HTML Einbindung</title>
4     <style type="text/css">
5       #content {
6         width: 500px;
7         margin: 0 auto;
8       }
9     </style>
10    </head>
11    <body>
12      <div id="content">
13        <!-- Content goes here -->
14      </content>
15    </body>
16 </html>

```

Listing 9.5: HTML–Quelltext

```

1 var gIntervalId = window.setInterval ( "checkPos()", 5000 );
2
3 function success(position)
4 {
5   var s = document.querySelector('#status');
6
7   if (s.className == 'success')
8   {
9     // Kommentar
10    return;
11  }
12}

```

Listing 9.6: JavaScript–Quelltext

10 Fazit und Persönliche Erfahrungen

10.1 Fazit

Zusammenfassung der Projektergebnisse. Besondere Erkenntnisse. Beurteilung des Lösungswegs. Eventuelle Alternativen und möglicher Erweiterungen.

10.2 Persönliche Erfahrungen

Hier (und nur hier) darf aus der Ich-Perspektive geschrieben werden.

A $\text{\LaTeX}^{\text{AV}}$

Das vorliegende Dokument wurde in \LaTeX erstellt. \LaTeX (gesprochen Latech) ist ein Textsatzsystem das speziell für umfangreiche und komplexe wissenschaftliche, technische und mathematische Dokumente entwickelt wurde. Man schreibt Quelltext wie bei einem Programm und übersetzt diesen Quelltext in ein PDF Dokument. Siehe [?].

A.1 Die Vorlage

Die \LaTeX Quelltexte dieses Dokuments sind gedacht um als Vorlage für die eigenen Diplomarbeit verwendet zu werden. Dazu muss der Inhalt durch die eigene Arbeit ersetzt werden. `Vorlage_DA.tex` ist das zentrale Haupt-Dokument, in dieses werden die einzelnen Kapitel inkludiert. Die Dateien für die eingefügten Kapitel finden sich im Unterordner `chapters`.

Die Vorlage kann von GitHub geladen werden: <https://github.com/matejkaf/latex-da-vorlage>

A.2 Programme

Programme (Editor + PDF Compiler) für \LaTeX :

Für Mac: MacTeX <https://tug.org/mactex/>

Für Windows: MiKTeX <http://miktex.org>

Zusätzlich auch die neueste Adobe Reader Version installieren!

Online: Mit dem Service "Overleaf" (<https://www.overleaf.com/>) gab es positive Erfahrungen. Damit können die Dokument online erstellt und gemeinsam verwendet werden.

A.3 Bitmap Fonts

Bei MiKTeX unter Windows kann es ein Font Problem geben. Falls die Schrift nicht scharf ist — PDF so vergrößern dass ein Buchstabe gut 10 cm groß ist — dann sieht man die Pixel, siehe Abbildung A.1. In diesem Fall wird ein sogenannter Bitmap-Font verwendet. Besser ist ein Vektor-Font, dieser lässt sich beliebig ohne Qualitätsverlust vergrößern.

Lösung: Mit "MiKTeX Package Manager" das Package `cm-super` installieren.



The image shows two versions of the word "shown". The top version is rendered in a bitmap font, where each character is a distinct, pixelated shape. The bottom version is rendered in a vector font, where the characters are smooth, continuous black outlines.

Figure A.1: Oben Bitmap-, unten Vektor-Font

A.4 LaTeX Quelltext

Bei LaTeX wird der Text, dessen Gliederung in die Formatierung in puren Textfiles mit der Endung .tex beschrieben. Die Zeichenkodierung der Files muss UTF-8 sein.

Der "LaTeX Compiler" (Programm mit dem Namen pdflatex) übersetzt diese Files in ein PDF Dokument.

A.5 Gerüst

Die Grundstruktur eines L^AT_EX Dokuments:

```
\documentclass[a4paper,10pt,final,oneside]{scrartcl}

\usepackage{anyfontsize}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[ngerman]{babel}
\usepackage[pdftex]{hyperref}
\usepackage{url}

\begin{document}

Ab hier kommt der Inhalt

\end{document}
```

Hinweis: Durch die Vorlage ist diese Grund-Struktur bereits vorgegeben.

A.6 Formatierungen

Mehrere Leerzeichen werden ignoriert
ein einfacher Zeilenumbruch gilt als Leerzeichen.

Eine leere Zeile kennzeichnet einen neuen Absatz.

Mehrere Leerzeichen werden ignoriert
ein einfacher Zeilenumbruch gilt als Leerzeichen.
Eine leere Zeile kennzeichnet einen neuen Absatz.

Diverse Formatierungen:

TypeWriter	<code>\texttt{...} o. {\ttfamily ...}</code>
Fett	<code>\textbf{...} o. {\bfseries ...}</code>
<i>Italic</i>	<code>\textit{...} o. {\itshape ...}</code>
<i>Slanted</i>	<code>\textsl{...} o. {\slshape ...}</code>
KAPITÄLCHEN	<code>\textsc{...} o. {\scshape ...}</code>
Normal	<code>\textmd{...} o. {\mdseries ...}</code>
<i>Emph</i>	<code>\emph{...} o. {\em ...}</code>
Sans Serif	<code>\textsf{...} o. {\sffamily ...}</code>
<u>Unterstrichen</u>	<code>\underline{...}</code>
Größen	<code>\tiny \scriptsize \footnotesize \small \normalsize \large \Large \LARGE \huge \Huge</code>
Zentriert	<code>\begin{center}... \end{center}</code>
Gesch. Leerz.	<code>\sim</code>
Zeilenumbruch	<code>\backslash oder \newline</code>
Absatzumbruch	<code>\par oder leere Zeile.</code>

A.7 Überschriften

```
\chapter{Hauptkapitel}
\section{Kapitel}
\subsection{Unterkapitel}
\subsubsection{Unterunterkapitel}
```

Die Kapitelnummerierung und das Inhaltsverzeichnis werden automatisch erstellt.

A.8 AutorenSM

Für eine Diplomarbeit mit mehreren Autoren muss nachvollziehbar sein wer für welchen Teil der Urheber ist. Dies ist durch **Namenskürzel** (2-stellig) in den Überschriften darzustellen (siehe in diesem Dokument).

Eine Person soll immer für ein komplettes Hauptkapitel (`chapter`) oder Kapitel (`section`) verantwortlich sein. Eine Aufteilung auf Ebene der Unterkapitel (`subsection`) oder Unterunterkapitel (`subsubsection`) sollte vermieden werden.

Die Namenskürzel werden in `Vorlage_DA.tex` definiert:

```
% Initialen der Autoren
\def\authorInitialsA{MM} % Max Mustermann
\def\authorInitialsB{FF} % Frieda Fröhlich
\def\authorInitialsC{FE} % Fritz Einstein
\def\authorInitialsD{WA} % Weiterer Autor
```

Für das Einfügen der Namenskürzel ins Dokument dienen in weiterer Folge die Befehle `\authorA`, `\authorB`, `\authorC`, `\authorD`.

Beispiel — erzeugt die Überschrift dieses Kapitels:

```
\section{Autoren\authorB}
```

A.9 Bilder einfügen

Formate: pdf, jpg und png. Dateien im Verzeichnis `media/images` ablegen.

```
In Abbildung \ref{fig:htl01} sieht man das Logo der HTL Braunau.
\begin{figure}[H]
\centering
\includegraphics[width=0.3\textwidth]{./media/images/htl_c_cmyk_rein.pdf}
\caption{Logo der HTL Braunau.}
\label{fig:htl01}
\end{figure}
```

In Abbildung A.2 sieht man das Logo der HTL Braunau.



Figure A.2: Logo der HTL Braunau.

Hinweis: Dateipfade mit "/" bilden!

Siehe http://en.wikibooks.org/wiki/LaTeX/Importing_Graphics

Der Befehl `\label` gibt der Abbildung einen eindeutigen Namen. Durch `\ref` wird dieser Name referenziert, d.h. es wird die automatisch generierte Abbildungsnummer eingefügt (dazu muss das L^AT_EX Dokument 2-mal erstellt werden!)

Siehe 9.1, Seite 23 für Abbildungen die mehrere Bilder enthalten.

A.10 Querverweise

Mit Hilfe von Querverweisen verweist man auf andere Stellen im Dokument. Z.B. in der Form: "Siehe 9.1" — es wird die Kapitelnummer bzw. Abbildungsnummer angegeben.

Es kann auf Abbildungen und auf Überschriften verwiesen werden. Diese erhalten zuerst mit `\label` einen Namen.

```
\section{Kapitelname} \label{ref:meinebezeichnung}
```

Möchte man auf diese Elemente verweisen gibt man den Namen im `\ref` Befehl an.

```
Eine genaue Beschreibung dieses Themas ist  
in Kapitel \ref{ref:meinebezeichnung} zu finden.
```

Ein Doppelpunkt als Teil des Namens ist erlaubt. Auf diese Weise können zum Beispiel Namen von Abbildungen, Überschriften und Listings unterschieden werden (`fig:/ref:/code:`).

L^AT_EX macht aus Querverweisen automatisch PDF Links.

A.11 Aufzählungen

```
\begin{itemize}
\item Eins
\item Zwei
\item Drei
\end{itemize}
```

- Eins
- Zwei
- Drei

```
\begin{enumerate}
\item Eins
\item Zwei
\item Drei
\end{enumerate}
```

1. Eins
2. Zwei
3. Drei

A.12 Mathematische Formeln

Abgesetzte Formel:

```
\begin{equation*}
\frac{1+x}{1-x} \cdot \sqrt[3]{2} \cdot \binom{n}{k}
\end{equation*}
```

Abgesetzte Formel:

$$\frac{1+x}{1-x} \cdot \sqrt[3]{2} \cdot \binom{n}{k}$$

Formel im Textfluss:

```
$\frac{1+x}{1-x}, \sqrt[3]{2}, \binom{n}{k}$
```

Formel im Textfluß: $\frac{1+x}{1-x}, \sqrt[3]{2}, \binom{n}{k}$

A.13 Programm Quelltext

Die Umgebung `lstlisting` übernimmt das Formatieren von Programmquelltext.

Listing `\ref{code:complex}` zeigt die Implementierung eines besonders komplexen Algorithmus.

```
\begin{lstlisting}[
language=java,
caption={Komplizierter Quelltext.},
label=code:complex
]
while(x>0) {
    x--;
    // bla bla
}
\end{lstlisting}
```

Listing A.1 zeigt die Implementierung eines besonders komplexen Algorithmus.

```
1 while(x>0) {
2     x--;
3     // bla bla
```

4 }

Listing A.1: Komplizierter Quelltext.

Siehe auch https://en.wikibooks.org/wiki/LaTeX/Source_Code_Listings

A.14 Code im Text

Mit dem Befehl `lstinline` können kurze Programmfragmente direkt in den Textfluss integriert werden.

Das sieht dann so aus:

```
\lstinline[language=java]{labels.add("") + i}.
```

Allzu lange sollten diese Programmausschnitte aber nicht sein.

Mit dem Befehl `lstinline` können kurze Programmfragmente direkt in den Textfluß integriert werden. Das sieht dann so aus: `labels.add("") + i`. Allzu lange sollten diese Programmausschnitte aber nicht sein.

A.15 Seitenübrüche

Ein Seitenumbruch kann mit `\pagebreak` erzwungen werden.

Soll etwas als ganzes auf der Seite stehen und nicht umgebrochen werden, so kann dieser Teil in eine `minipage` Umgebung eingeschlossen werden.

```
\begin{minipage}{\linewidth}
In diesem Teil findet kein Seitenumbruch statt.
\end{minipage}
```

Häufig wird dies bei Programmlistings nötig sein:

Das Listing `\ref{code:codenopagebreak}` ist in eine `minipage` eingebunden. Dies wirkt sich so aus, dass dieses Listing nur als ganzes auf der Seite steht, sollte es nicht mehr Platz haben wird es komplett auf die folgende Seite gesetzt und es entsteht ein Leerraum auf der vorhergehenden Seite.

```
\begin{minipage}{\linewidth}
\begin{lstlisting}[
    language=java,
    caption={Java Quelltext.},
    label=code:codeexample1
]
public void prepend(Node n) {
    n.next=start;
}
```

```

        start=n;
}
\end{lstlisting}
\end{minipage}
```

Das Listing A.2 ist in eine minipage eingebunden. Dies wirkt sich so aus, dass dieses Listing nur als ganzes auf der Seite steht, sollte es nicht mehr Platz haben wird es komplett auf die folgende Seite gesetzt und es entsteht ein Leerraum auf der vorhergehenden Seite.

```

1 public void prepend(Node n) {
2     n.next=start;
3     start=n;
4 }
```

Listing A.2: Java Quelltext.

A.16 Quellen und Literatur

Alle Quellen befinden sich im Dokument `chapters/Post-01-literatur.tex`. Eine Quelle erhält mit `\bibitem{bib:name}` einen Namen.

Im Text wird durch den Befehl `\cite` zitiert. Bsp.:

Siehe `\cite{bib:latexintro}`.

Siehe [?].

A.17 Links

Links können in der Form einer URL eingefügt werden. Dies kann (in kleinerem Umfang) statt Einträgen im Literaturverzeichnis verwendet werden.

`\url{http://www.orf.at}`

`http://www.orf.at`

Lange URL's sehen nicht besonders gut aus. Etwa `https://www.youtube.com/watch?v=_vQaOvPsLko&list=PL6gx4Cwl9DGBsvRxJJ0zG4r4k_zLKrnxl&index=49`.

In diesem Fall bietet es sich an solche URL's mit einem Kurz-URL-Dienst (z.B. tinyurl.com) zu verkürzen. Obige URL in verkürzter Form: `http://tinyurl.com/hqtzqan`

A.18 Fußnoten

```
Eine Fußnote kann man einfach%
\footnote{Hier zum Beispiel}
irgendwo in den Text einfuegen.
```

Eine Fußnote kann man einfach¹ irgendwo in den Text einfügen, diese wird an das untere Ende der Seite gesetzt.

A.19 Tabellen

Mit der Umgebung `tabular` bzw. erweitert: `tabularx`.

Siehe <https://en.wikibooks.org/wiki/LaTeX/Tables>

A.20 Mehrspaltiger Text

Um einen zweispaltigen Text zu erzeugen kann die `multicols` Umgebung verwendet werden

```
\begin{multicols}{2}
Text in 2 Spalten
\end{multicols}
```

Hier wurde `multicols` verwendet um die Abbildung platzsparend neben dem Text zu platzieren. Siehe Logo der HTL Braunaau in Abbildung A.3.



Figure A.3: Logo auf der rechten Seite.

Um einen Spaltenenumbruch an einer bestimmten Stelle zu erzwingen:

```
\columnbreak
```

¹Hier zum Beispiel

Glossary

AADC Audi Autonomous Driving Cup. 17

ADTF Automotive Data and Time-Triggered Framework. 17

FPS Frames per Second. 18

lidar Laser radar. 1

optical flow A mathematical expression. 16

OSRF Open Source Robotics Foundation. 3

RGB-D camera Camera which captures an image that contains depth information besides the normal image. 2

ROS Robot Operating System. 3–6, 18, 40

SLAM Simultaneous Localization and Mapping. 1, 2, 10, 12

Bibliography

- [1] S. Riisgaard and M. R. Blas, “Slam for Dummies.” <https://tinyurl.com/y32jtecm>.
- [2] S. Prabhu, “Introduction to slam (simultaneous localisation and mapping).” ARreverie, 2019. <https://tinyurl.com/y5an9jq9>.
- [3] T. Bailey and H. Durrant-Whyte, “Simultaneous localization and mapping (slam): part ii,” *IEEE Robotics Automation Magazine*, vol. 13, pp. 108–117, Sep. 2006.
- [4] S. Martin, “What is simultaneous localization and mapping?.” Techapeek, 2019. <https://tinyurl.com/y5yaqv5u>.
- [5] C. Stachniss, U. Frese, and G. Grisetti, “OpenSLAM Website.” Openslam, 2019. <https://openslam-org.github.io/>.
- [6] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, “Fastslam: A factored solution to the simultaneous localization and mapping problem,” in *In Proceedings of the AAAI National Conference on Artificial Intelligence*, pp. 593–598, AAAI, 2002.
- [7] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, “Orb-slam: A versatile and accurate monocular slam system,” *IEEE Transactions on Robotics*, vol. 31, pp. 1147–1163, Oct 2015.
- [8] F. Steinbruecker, J. Sturm, and D. Cremers, “Real-time visual odometry from dense rgbd images,” in *Workshop on Live Dense Reconstruction with Moving Cameras at the Intl. Conf. on Computer Vision (ICCV)*, 2011.
- [9] F. Endrees, J. Hess, and N. Engelhard, “Rbg-d slam.” ROS Wiki, 2019. <https://tinyurl.com/yynyqwg2>.
- [10] J. Engel, T. Schops, and D. Cremers, “Lsd-slam: Large-scale direct monocular slam,” in *European Conference on Computer Vision (ECCV)*, September 2014. <https://vision.in.tum.de/research/vslam/lsdslam?redirect=1>.
- [11] O. Foundation, “Osr foundation homepage.” Website, 2019. <https://www.osrfoundation.org/>.
- [12] ROS, “About ros.” ROS, 2019. <https://www.ros.org/about-ros/>.
- [13] C. Robotics, “Ros 101: Intro to the robot operating system.” Robohub, 2014. <https://tinyurl.com/y4q6paak>.
- [14] ROS, “Is ros for me?,” 2020. <https://www.ros.org/is-ros-for-me/>.
- [15] M. H. Hassoun, *Fundamentals of Artifical Neural Networks*. The MIT Press, 1995.

- [16] R. Mur-Artal and J. D. Tardós, “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,” *IEEE Transactions on Robotics*, vol. 33, pp. 1255–1262, Oct 2017.
- [17] E. Weisstein, “Levenberg-marquardt method,” 2020.
<http://mathworld.wolfram.com/Levenberg-MarquardtMethod.html>.
- [18] E. Olson, J. Leonard, and S. Teller, “Fast iterative optimization of pose graphs with poor initial estimates,” pp. 2262–2269, 2006.
- [19] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, “Dtam: Dense tracking and mapping in real-time,” in *2011 International Conference on Computer Vision*, pp. 2320–2327, Nov 2011.
- [20] H. Zhou, B. Ummenhofer, and T. Brox, “Deeptam: Deep tracking and mapping with convolutional neural networks,” *International Journal of Computer Vision*, 2019.

List of Figures

2.1	ROS structure Source: https://tinyurl.com/tkf2smq	3
2.2	RQt interface Source: https://wiki.ros.org/RQt	5
2.3	Rviz interface	5
3.1	General view of Neurons Source: https://tinyurl.com/yyfthk7c	8
3.2	Feed forward network Source: https://tinyurl.com/y3yaqboq	9
4.1	ORB-SLAM Example image Source: https://tinyurl.com/ruvnj39 . .	10
5.1	Feature-Based and Direct Difference Source: https://tinyurl.com/ycmjhb9d	12
5.2	LSD-SLAM Example image Source: https://tinyurl.com/qkuamyy	13
5.3	LSD-SLAM Pose-Graph-Optimization (left after loop-closure, right before loop-closure) Source: https://tinyurl.com/qkuamyy	14
6.1	Schematic of DeepTAM Source: http://lmb.informatik.uni-freiburg.de/Publications/2019/ZUB19a	15
9.1	Logo der HTL Braunau.	23
9.2	Zwei Logos der HTL Braunau	23
9.3	Drei Logos der HTL Braunau	24
A.1	Oben Bitmap-, unten Vektor-Font	29
A.2	Logo der HTL Braunau.	32
A.3	Logo auf der rechten Seite.	36

Listings

7.1	Main Function of CamStream Feed	18
7.2	StreamingHandler of CamStream	19
9.1	Mein erstes C-Programm.	25
9.2	Alle Dateinamen ausgeben.	25
9.3	Mausposition ermitteln.	25
9.4	Kurzer CSS-Quelltext.	25
9.5	HTML-Quelltext	26
9.6	JavaScript-Quelltext	26
A.1	Komplizierter Quelltext.	33
A.2	Java Quelltext.	35

Authors

Alexander Voglsperger

Birthday, Place of birth: 25.03.2001, Ried im Innkreis
School education: Volksschule Aurolzmünster
Informatik Hauptschule Aurolzmünster
HTL Braunau
Internship: Team7 Natürlich Wohnen GmbH, 4 Weeks, IT
Krankenhaus Ried im Innkreis, 4 Weeks, IT
Johannes Kepler University - AI Lab, 4 Weeks,
Mapping and Tracking on self-driving car
Address: Forchtenau 196
4971, Aurolzmünster
Österreich
E-Mail: alexander.voglsperger@gmail.com



Simon Moharitsch

Birthday, Place of birth: 01.01.1970, Braunau am Inn
School education: Volksschule
Hauptschule
HTL
Internship: Firmenname, Zeit, Tätigkeit
Address: Strasse Nummer
PLZ, Ort
Österreich
E-Mail: max@mustermann.com

