# Acknowledgments

# Contents

# 1. Abstract

# 2. Introduction

Parking lots, public transport, bicycle lanes — we are guided by a verity of different **line markings** everyday in our lives. Nevertheless this domain and companies involved in this sector lack two aspects crucial to our today's society — digitalization and automatization.

To solve this issue and given the existing but outdated tools in the marking industry, the unhealthy working conditions, and the worker shortage in the market, we, *Mark-Robotics*, strive to develop the **first autonomous ground line marking robot for airports**.

This *Interdisciplinary Project in an Application Subject* (IDP) will therefore shed some light in the, until now, quite poorly explored domain of airport taxiway ground line markings in respect to mobile robot navigation. We will discuss how accurate camera-based line detection can be used to navigate a robot precisely and see on a pre-prototype how line detection can be implemented with the help of *machine learning* and a *convolutional neural network* (CNN). This technology lays the ground work for the development of an autonomous line marking robot for airports. In particular *line detection*, or more precisely detecting partially erased lines, is crucial, as 90% of the line marking work on airports is repainting of already existing lines, which must be done up to twice a year.

## 2.1. Problem Statement

Toxic paints, high planning effort, required accuracy under time pressure, and predominantly night shifts make line marking highly complicated and unattractive. At airports, 90% of the work is repainting of already existing lines, which must be done up to twice a year. To improve on this, and in order to digitize the marking industry a first step towards the development of an autonomous line marking robot should be made with this *Interdisciplinary Project in an Application Subject* (IDP).

While developing such a line marking robot, accurate navigation is a really challenging task. For this reason an approach for **camera-based line detection** will be discussed, which is needed to navigate the line marking robot precisely along lines that require to be repainted.

## 2.2. Motivation

Accurate line detection for (partially erased) ground line markings, is a crucial element towards the development of an autonomous line marking robot for airports. This *Interdisciplinary Project in an Application Subject* (IDP) is therefore easily motivated by the fact that any founding during this project might influence and contribute towards the further development of the robot. Especially, the accurate detection of partially erased ground line markings is a largely untouched domain, as highlighted in subsequent Chapter 3, uplifting additional research in this area.

## 2.3. Requirements

As for the requirements of this *Interdisciplinary Project in an Application Subject* (IDP), and more specifically, the goal to develop a pre-prototype for accurate line detection, we can differentiate them into two main parts. While there are *hard* constraints that should be solved, there are also *soft* constraints, which are nice to have, but optional. The following table 1 visualizes all requirements in a structured manner.

| **Mandatory Requirements** | **Optional Requirements** |
|---|---|
| • Reliable **detection** of **ground line markings** | • Detection of *partially erased* ground line markings |
| • **Detection** is based on a computer vision approach | • Accurate detection using a machine learning model |
| • Video sample taken from an urban context (e.g. street line markings) | • Video sample taken from an airport context (e.g. taxiway line markings) |
| • Command-line execution of testing script | • GUI execution for testing |

**Table 1** Mandatory and optional requirements for the IDP Project.

## 2.4. Mark-Robotics

*Mark-Robotics* is a Munich based pre-seed start-up, thriving to develop the first **autonomous ground line marking robot** for airports. Founded on a first prototype, developed during the intensive two weeks TUM course Think.Make.Start. (TMS), the start-up project evolved. The company is run currently established by the three founders *Johannes Frey*, *Johannes Schaaf* and *Johannes Münichsdorfer*.

# 3. Related Work

In the domain of *line detection* countless research has been conducted. Even though work surrounding in the area of *line detection* can be generally considered related work in respect to this *Interdisciplinary Project in an Application Subject* (IDP), we restrict the scope to work specifically targeting ground line markings in the following. This in fact, inherently in-cooperates the entire research in the field of *autonomous vehicles*.

# 4. Approach

## 4.1. Overview

## 4.2. Training Pipeline

### 4.2.1. Image Capturing

### 4.2.2. Image Labeling

Automatic Pre-Labeling

Manual Labeling

### 4.2.3. Erasing Sensitive Information

### 4.2.4. Preparing Images and Annotations for Training

### 4.2.5. Training using Mask R-CNN

## 4.3. Results

# 5. Evaluation

# 6. Future Work

# 7. Conclusion

# 8. Example Chapter: Research and Comparison

## 8.1. Example Section: Comparison and Decision

As all three methods described above have strengths and weaknesses in their specific domain, it is clear that every method suits best for a concrete use case. In this section, I am going to compare Method A, B, and C with respect to the possibilities, difficulties, and restrictions in each one respectively. Table 2 shows this comparison in a structured way. As option A, the accessibility service works best for use cases where we need to track touch events and read out screen information, option B, which is about building a VPN service, is the best one in all cases where we need to track network usage. On the other hand, if we need to gather text or voice input from the user, method C, i.e. the InputMethod service, satisfies our needs. Considering what we want to achieve, or more closely what we actually need to track or recognize, is especially important in this domain. Though we could for sure implement all three methods simultaneously and ask our user to activate all of them, the chance of acceptance will be probably quite rare. Since tracking always involves handling a lot of sensitive private data, we should only use those parts that are really needed. Additionally, those cases are also the only ones, that will be, to some point, accepted by users.

| **Tracking** | **Method A**<br>Accessibility service | **Method B**<br>VPN service | **Method C**<br>InputMethod (IME) service |
|---|---|---|---|
| Overview | Implementing an own accessibility service inside your application in order to retrieve the users window content as well as his or her touch events. This option furthermore enables the developer to take actions for the user. The service is running constantly in the background and needs to be manually activated by the user starting it for the first time. | Building an custom VPN service into your application to capture and handle the network traffic on the device. This option enables to control the entire network traffic, accordingly every sent and received IP packet, to as well as from the phone. Furthermore the service allows to configure own DNS servers and to include only specific applications inside the VPN. Has to be manually activated on first usage. | Implementing an own InputMethod service, i.e. a custom keyboard, allows to receive all input to the device by the user. This includes text and voice input as well as hardware key-events. We can therefore see what the user is typing and react to it accordingly. An InputMethod service has to be manually activated. |
| Possibilities | + Retrieve the entire window content across third party applications as view hierarchy. This enables a developer to see whatever the user is currently viewing.<br>+ Recognize all touch events.<br>+ Take actions for the user (e.g. pressing a button, view, navigating to the home screen, etc.). | + Retrieve the entire network traffic with all contacted servers, etc..<br>+ Capture domain names with a custom DNS server.<br>+ Notice every time the device is contacting external servers.<br>+ *Per-App* VPN option, to include only specific applications inside the service. | + Receive the entire input to the device as well as all hardware key-events.<br>+ Potential benefit for the user if the IME service implements a great keyboard. |
| Difficulties | — — Hard to verify if the assumed context is actually correct. This means that e.g. the user might currently not visit a website if we detect an URL somewhere, but instead just views a message with an URL inside.<br>— — The user has to manually enable every accessibility service inside Android settings application in a complicated procedure.<br>— — — Uncertain future of Google accepting apps which incorrectly use the accessibility service | — Difficult to verify if the assumed context is correct. This means that e.g. the user might currently not visit a website if we detect a connection to a specific *WWW* URL, but instead an app just downloads new images in the background. → Easier with *Per-App* VPN.<br>— — Must handle every IP packet manually.<br>— The user has to manually accept the VPN service on first usage, with a dialog inside the requesting application. | — — Hard to verify if the assumed context is really correct. This means that e.g. the user might currently not visit a website if we detect an URL as input, but instead just writes a message to someone.<br>— Developing a great keyboard is really expensive as users expect a lot.<br>— The user has to manually enable the new IME service inside Android settings application. |
| Restrictions | !!! Should not be used for tracking purposes! Google declares that an accessibility service should be only used for helping disabled or people in challenging environments. | ! No restrictions specified, but since we handle sensitive private data this should be done very carefully! | ! No restrictions specified, but since we handle sensitive private data this should be done very carefully! |

**Table 2** Comparison of Method A to C in respect to tracking on Android devices.

[...]

Considering this new knowledge from the previous paragraph I decided to move the focus for the tracking detection to partner websites only, instead of also including partner applications. Hence, this work concentrates on tracking detection of partner websites only. Since most of the online shopping on Android devices mainly takes place inside a browser application, please see figure 1, e.g. Chrome [1], Firefox [2] or Samsung Internet Browser [3], the loss of partner apps (applications) does not have a great impact anyway. To detect those websites we need to compare all three methods again more closely. An opening of a website normally involves a text input by the user, a network connection to a server on which the website is stored and finally a view that is displaying the content. Therefore, all three methods would work. With C we can detect the *URL* input of the user through his keyboard and with B we can easily the detect the network connection. With method A, we can later gather information from the screen and, with that, detect the view and the content inside it, where the website is displayed. Since there is quite a low error rate detecting network traffic, both other domains have at least one big disadvantage, which leads to a higher error rate. Gathering *URL* information with method A or C does not necessarily mean that the user is currently actually visiting a website, but could be also viewing e.g. an *URL* inside a text message that was sent to him, or is entering text for a text message. It is hard to tell at which point the user is viewing the website since we only want to notify him in those cases. With method B we can almost entirely eliminate this problem since a network connection to a specific website in a browser application is normally only established if the user is effectively opening it. With this in mind, it is clear to see that option B, building a VPN service, satisfies our needs best. Including other advantages, like an easy and straightforward setup process as well as the ability to include only certain applications into our VPN service, qualifies it as an excellent choice. In the following chapter 4 I am going to describe my approach to use a VPN service for the presented scenario in much detail.

[...]

[1] https://play.google.com/store/apps/details?id=com.android.chrome
[2] https://play.google.com/store/apps/details?id=org.mozilla.firefox
[3] https://play.google.com/store/apps/details?id=com.sec.android.app.sbrowser

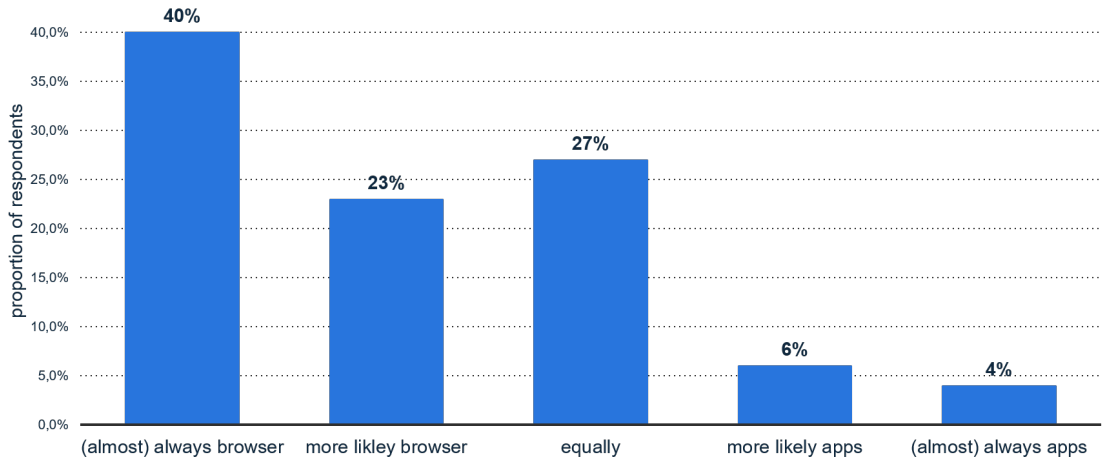


**Figure 1** Comparison graph: "Considering online shopping on mobile devices, are you more likely to use a browser applications or custom shop applications?" [1, The survey question as well as the captions of the graph are translated to english in a contextual manner.]

# 9. Appendixes

# List of Figures

# List of Tables

# 10. Bibliography

[1] Statista. (11.07.18) Nutzen Sie beim mobilen Onlineshopping eher den Browser oder eine shopeigene App? [Graph]. [Online]. Available: https://de-statista-com.eaccess.ub.tum.de/prognosen/879501/umfrage-zur-nutzung-eines-browsers-shopeigener-app-beim-onlineshopping