

Semesterarbeit, Abteilung Informatik

OSM Crosswalk Detection

Hochschule für Technik Rapperswil

Herbstsemester 2015

18. Dezember 2015

<i>Autoren:</i>	Bühler Severin & Kurath Samuel
<i>Betreuer:</i>	Prof. Keller Stefan
<i>Arbeitsperiode:</i>	16.09.2015 - 18.12.2015
<i>Arbeitsumfang:</i>	240 Stunden, 8 ECTS pro Student
<i>Link:</i>	https://github.com/geometalab/OSM-Crosswalk-Detection

Inhaltsverzeichnis

0.1	Ausgangslage	4
0.2	Ergebnisse	4
0.3	Ausblick	5
1	Technischer Bericht	6
1.1	Literaturrecherche	7
1.1.1	Suchquellen	7
1.1.2	Auswertung	7
1.1.3	Extraction of Road Markings from Aerial Images	7
1.1.4	Segmentation of Occluded Sidewalks in Satellite Images	8
1.1.5	Fazit	9
1.2	Evaluation Crowdsourcing-System	10
1.2.1	Kandidaten	10
1.2.2	MapRoulette	10
1.2.3	To-Fix	11
1.2.4	Evaluationsmatrix	11
2	Software Dokumentation	13
3	Projektmanagement	14
3.1	Entwicklungsumgebung und Infrastruktur	14
3.1.1	IDE (Integrated Development Environment)	14
3.1.2	SCM (Source Control Management)	14
3.1.3	Projektmanagement Tool	14

Abstract

Zebrastreifen sind ein essentieller Bestandteil der Fussgängernavigation, diese sind jedoch nur spärlich erfasst, was zu nicht optimalen Routen führt. Um dem entgegen zu wirken, befasst sich dieses Projekt mit der automatischen Erkennung von Zebrastreifen auf Orthofotos (Satellitenbildern). Dabei entstand eine Applikation, die auf den Orthofotos den Strassen folgt, diese in kleine Bilder unterteilt und mit Hilfe eines Deep learnig Ansatzes entscheidet, ob es sich um ein Zebrastreifen handelt oder nicht. Das führte zu einer Erkennungsrate von über 85% und könnte in Zukunft den Behörden bei der Erfassung der Daten (derzeit noch händisch) unterstützen. Weiter ist es möglich diese Lösung auszubauen und auf andere Objekte anzuwenden.

Management Summary

0.1 Ausgangslage

Das Erfassen von Zebrastreifen geschieht heutzutage noch händisch durch die jeweiligen Behörden. Dieses Projekt befasst sich damit, diesem noch manuellen Vorgang einen automatisierten Aspekt zu verleihen. Dabei wird auf Informationen zu Strassenverläufen und Orthofotos (Satellitenbilder) zurückgegriffen.

0.2 Ergebnisse

Es soll eine Applikation entstehen, die mit dem Input von Strassen und Orthofotos Zebrastreifen erkennt und als Output die jeweiligen Koordinaten liefert.

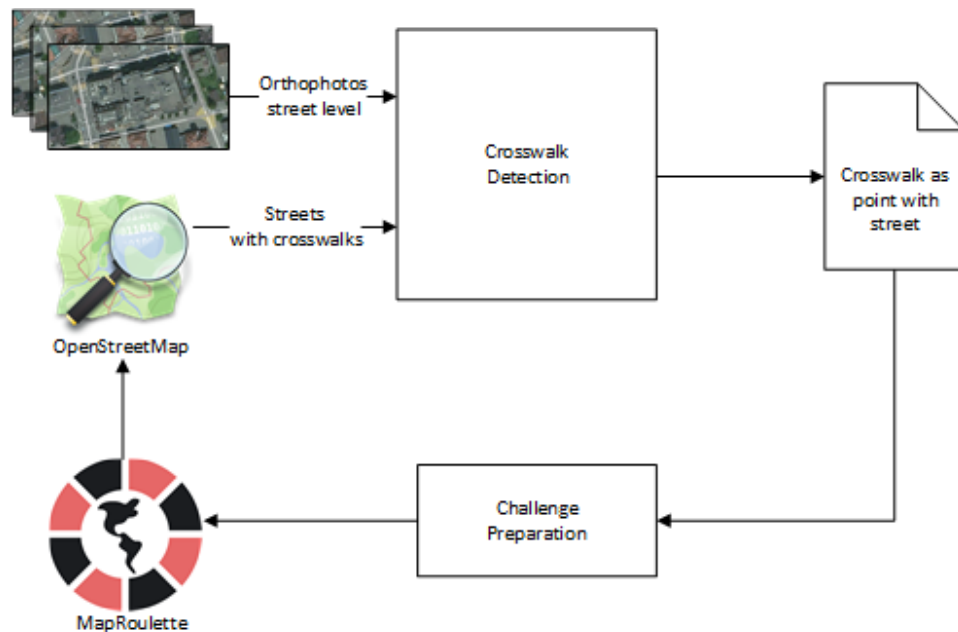


Abbildung 1: Überblick

0.3 Ausblick

Das Projekt bietet viele Ausbaumöglichkeiten und kann nicht nur auf Zebrastreifen angewendet werden. Es ist auch denkbar auf den Strassen nach Markierungen zu suchen, wie Stop oder Bus etc.

Kapitel 1

Technischer Bericht

Stand der Technik

Um abzuklären, ob es schon Arbeiten gab, die ein ähnliches Problem lösen, nahmen wir uns im Rahmen der Semesterarbeit Zeit für eine Literaturrecherche. Dabei gingen wir auf die HSR Bibliothek und deren Mitarbeiter zu.

1.1 Literaturrecherche

1.1.1 Suchquellen

Folgende Quellen wurden uns empfohlen, um Recherchen in diesem Umfeld durchzuführen:

- <http://recherche.nebis.ch/>
- <http://ieeexplore.ieee.org/>
- <http://scholar.google.ch/>

1.1.2 Auswertung

Bei der Recherche stiessen wir auf verschiedenen Projekte, die sich mit der Problematik des Erkennens von Fussgängerstreifen auseinander setzen. Leider sind diese Arbeiten eher im Bereich der Bilderkennung für die Steuerung von autonom fahrenden Autos/Robotern angesiedelt. Arbeiten die treffender sind, werden im Anschluss angeführt.

1.1.3 Extraction of Road Markings from Aerial Images

Yuichi Ishino und Hitoshi Saji (Japan, 2008)

<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4655024>

An der Universität Shizuoka in Japan gab es vor einigen Jahren eine Arbeit zur Erkennung von Fussgängerstreifen und Mittellinien (Traffic Lane Lines) auf Orthofotos (Aerial images).

Ihr Algorithmus befolgt dabei folgende Strategie: Der Algorithmus geht den Strassen entlang und richtet die Bilder aus, dass die Fussgängerstreifen immer vertikal zur Achse laufen. Danach wird eine sogenannte Binarization durchgeführt. Es setzt alle Pixel unter einem Schwellwert auf 0 (weiss) und alle Pixel darüber auf 1 (schwarz). Es wurden zwei Schwellwerte zurvor berechnet, einmal für sonnige und einmal für schattige Bilder. Mit der Annahme, dass die Strasse schwarz/grau und der Fussgängerstreifen leuchtend weiss sind, sieht man nun ein gleichmässiges Muster in der Helligkeitsverteilung des Bildes. Ein Fouriertransformation würde eine saubere Frequenz liefern.

Die Arbeit von Ishino und Saji geht von einigen Grundannahmen und Voraussetzungen aus, die die Erkennung sehr erleichtern:

- Die Fussgängerstreifen sind immer gerade und werden durch keine Inseln unterbrochen.
- Die Auflösung der Bilder ist genug gross, um das Streifenmuster ohne Probleme zu erkennen.
- Der Fussgängerstreifen ist immer deutlich heller als die Strasse selbst.
- Der Streifen werden durch keine Hindernisse wie Bäume, Autos verdeckt oder beeinflusst.
- Die Bilder wurde zuvor in die Kategorien schattig und sonnig eingeteilt worden. Auf ihnen wird mit verschiedenen Treshholds gearbeitet.
- Die Strassen müssen die Fussgängerstreifen immer vertikal schneiden.

Schlussfolgerung

Die Arbeit der Univerisität von Shizuoka verfolgte einen ähnlichen Ansatz, den wir mit der Fouriertransformation in Betracht ziehen. Leider gehen die Dokumentverfasser von einigen Grundannahmen aus, die sich nicht mit der unseren Arbeit decken. Man kann fast schon von Laborbedingungen sprechen. Doch gibt es einigen Techniken, die sich auch für unsere Arbeit verwenden lassen. Diese sind unten aufgeführt.

1.1.4 Segmentation of Occluded Sidewalks in Satellite Images

Turgay Senlet und Ahmed Elgammal, The State University of New Jersey, USA (2012)

<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6460256>

Das Projekt von Turgay Selent und Ahmed Elgammal setzte sich mit der Erkennung von primär Gehwege (sidewalks) und Fussgängerstreifen auf Satellitenbildern auseinander.

Dabei waren die Hauptprobleme, dass viel Gehweg von Bäumen oder Schatten verdeckt werden. Um diesem Problem Herr zu werden, benutzten sie einen Farbklassifizierer. Um Fussgängerstreifen zu klassifizieren stellten sie eine Sammlung an Frequenzen in allen möglichen Winkeln zusammen.

Leider wird im Artikel zu dieser Arbeit nicht weiter in die Erkennungsmethoden eingegangen.

1.1.5 Fazit

Aus allen Arbeiten konnten wir doch einige Techniken finden, die uns die Erkennung erleichtern könnten. Diese sind hier aufgelistet:

- Binarization image
- Median Filter (für Verbesserung der Bildqualität von ungenauen Bildern)

1.2 Evaluation Crowdsourcing-System

1.2.1 Kandidaten

- MapRoulette¹
- To-Fix²

1.2.2 MapRoulette

MapRoulette verwendet für ihre Challenges und Tasks ein einfaches JSON Format. Der erstellt werden Challenges mittels POST und mit PUT können diese upgedatet werden.

Beispiel Challenge

Erstellen: POST /api/admin/challenge/<slug>

Updaten: PUT /api/admin/challenge/<slug>

Challenge JSON:

```
{
  "title": "Repair Motorways",
  "description": "Repair all motorways",
  "blurb": "The idea is to repair all motorways",
  "help": "Repair the ways where it is broken on the map",
  "instruction": "Look at the map for broken pieces.",
  "active": true,
  "difficulty": 2
}
```

Beispiel Task

Erstellen: POST /api/admin/challenge/<slug>/task/<task_identfier>

Updaten: PUT /api/admin/challenge/<slug>/task/<task_identfier>

Challenge JSON:

¹<http://maproulette.org/>

²<http://osmlab.github.io/to-fix/#/task/tigerdelta>

```

{
  "instruction" : "This is a hard task!",
  "geometries" : {
    "type": "FeatureCollection",
    "features": [
      { "type": "Feature",
        "geometry":
          { "type": "Point",
            "coordinates": [-41.4710170873565, 31.235521774136]
          },
        "properties": {"osmid": 12345}
      }
    ]
  }
}

```

1.2.3 To-Fix

To-Fix verwendet für ihre Task ein CSV Format, welches direkt über das grafische Benutzerinterface publiziert werden kann.

Beispiel CSV

```

object_type,object_id,st_astext
way,51446110,POINT(-94.4176451 43.3273692)
way,187403368,POINT(32.9369086 2.1997495)
way,220866128,POINT(-68.5 49.647521)
way,223982938,POINT(18.4823301 59.6732909)
way,109819283,POINT(-83.1888421 40.0485764)

```

1.2.4 Evaluationsmatrix

Um die beiden Kandidaten zu vergleich haben wir eine Evaluationsmatrix erstellt, dabei haben wir diverse für uns relevante Kriterien erarbeiten und diesen jeweils auf einer Skala von 1 bis 10 gewichtet. In einem zweiten Schritt haben wir den Kandidaten für die jeweiligen Kriterien Punkte vergeben.

Kriterium	Gewicht	Maproulette	Resultat	To-Fix	Resultat
Challenge ist leicht erstellbar	5	6	30	7	35
Challenge ist leicht publizierbar	7	8	56	8	56
Anbieter ist relevant bei der Community	8	8	64	4	32
Dokumentation	7	5	35	5	35
Kontaktperson	5	5	25	6	30
Total	32	32	210	30	188

Abbildung 1.1: Evaluationsmatrix

Entscheid 1. Crowdsourcing-System

Beide Kandidaten haben Vor- und Nachteile, wie aus der Evaluationsmatrix ersichtlich ist. Für uns ist das wichtigste Kriterium, wie relevant der Anbieter bei der Community ist, was sich dann auch im Resultat stark ausgewirkt hat. Da MapRoulette Challenges gerne abgearbeitet werden, tendieren wir für diesen Kandidaten.

Kapitel 2

Software Dokumentation

Kapitel 3

Projektmanagement

3.1 Entwicklungsumgebung und Infrastruktur

3.1.1 IDE (Integrated Development Environment)

Entscheid 2. PyCharm

Wieso:

Beiden Projektmitgliedern ist JetBrains IntelliJ bekannt und PyCharm ist im Umgang nahe zu identisch. Für Studenten sind die Entwicklungsumgebungen kostenlos verfügbar.

3.1.2 SCM (Source Control Management)

Entscheid 3. GitHub

Wieso:

Der Umgang mit Git ist beiden Projektmitgliedern bestens bekannt. GitHub ist ohne Unkosten von überall verfügbar. Das Geometalab der HSR publiziert über diesen Weg diverse Projekte.

3.1.3 Projektmanagement Tool

Entscheid 4. Jira

Wieso:

Jira ist den Projektmitgliedern schon aus dem SE2-Projekt bekannt und hat sich sehr bewährt. Das Dashboard ist übersichtlich gestaltet, es ermöglicht eine Übersicht über die aktuellen Tasks auf einen Blick. Alle Mitglieder haben zu jederzeit

Zugriff auf die Plattform, dies erhöht die Transparenz Weiter bietet Jira diverse Reports um Auswertungen über das Projekt zu fahren.

Abbildungsverzeichnis

1	Überblick	4
1.1	Evaluationsmatrix	12

Liste der Entscheidungen

1.1 Crowdsourcing-System	12
3.2 PyCharm	14
3.3 GitHub	14
3.4 Jira	14