FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Diplomarbeitspräsentation

Using Mobile Edge Computing Technologies for Real-Time Cornering Assistance

Masterstudium:
Software Engineering & Internet Computing

Matthias Karan

Technische Universität Wien
Institut für Information Systems Engineering
Arbeitsbereich: Distributed Systems
Betreuer: Priv.-Doz. Dr. Hong-Linh Truong

Problem Statement

We introduce a novel system that assists drivers in real-time while cornering. The system is designed to be deployable to cloud and edge-computing infrastructures.

Goals:
- warn drivers ahead of curves in real-time
- recommend safe speeds to enter a curve
- use novel edge- and cloud-computing architectures and algorithms

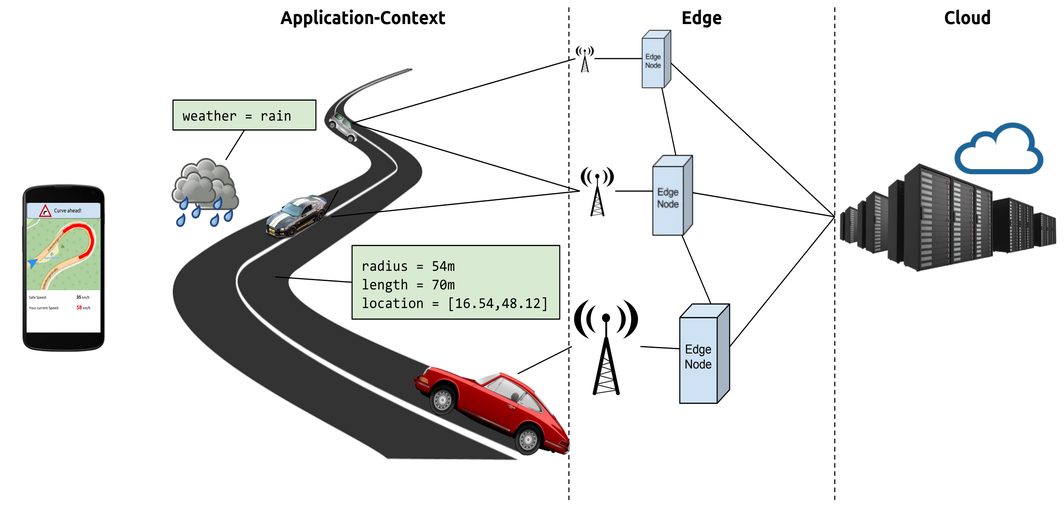

Fig.1: Overview of the context and infrastructure of the system
Icons used from http://bbcpersian7.com, http://www.pngpix.com/download/black- ford- shelby- gt- h- top- view- car- png- image, https://www.wpclipart.com/, http://clipart-library.com/cl part/1007672.htm and warszawianka from http://www.Freestockphotos.biz

Research Questions & Approach

- **data aspects**: frequency of GPS? what external data to combine?
- how to separate computational **tasks** across the infrastructure?
- **algorithms**: how to detect curves and recommend speeds?
- what is the **performance**, **data quality**, **cost** of the system?

=> determine frequencies, fuse data sources, identify desired output
=> design architecture to support tasks and MEC infrastructure
=> design new algorithms and implement a prototype
=> evaluate the prototype

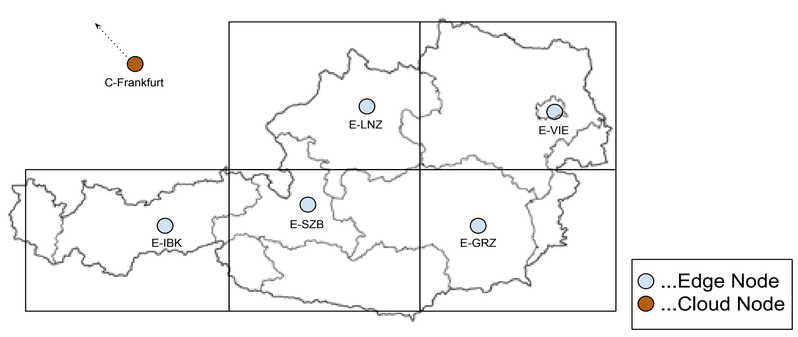

Fig.2: Illustration of how a future edge-cloud-infrastructure could look like in Austria

System & Algorithm Design

Microservice-Architecture:
supports virtualized MEC infrastructure, separates tasks and concerns and enables scalability.

Application-Specific Services:
to perform recommendation and detection of curves.

External Services:
to retrieve weather and map data.

Fig.3: Design of our proposed cornering-assistance-system

Algorithms:
- **curve-detection:** detect curves and their properties (radius, length, etc.) using map data.
- **recommend:** calculate a safe speed to enter a curve using calculated properties.
- **predict-upcoming-curve:** find the next upcoming curve to a driver's current location.
- **load-balance:** choose appropriate cloud or edge nodes to handle client requests.

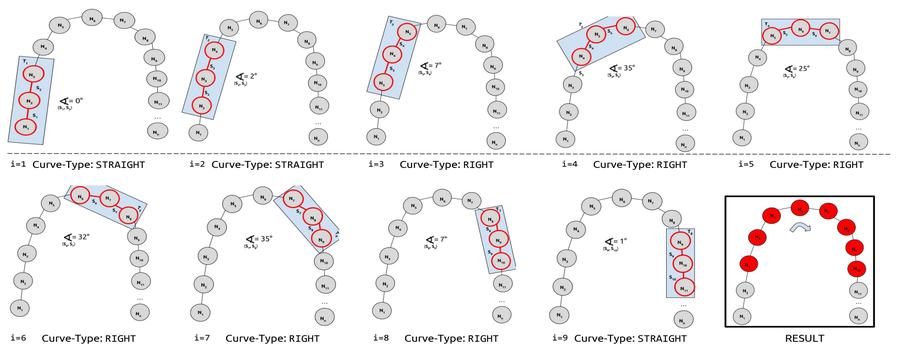

Fig.4: Illustration of the steps of our proposed Curve Detection Algorithm

Experiments & Results

Performance
- 1500-2000 drivers can use our prototype concurrently
- with load balancing we can scale to many thousands drivers
- ~0.5 seconds response time

Data Quality
- 86% curve detection rate
- 76% detected live before approached by simulated drivers
- highly fault tolerant to GPS errors and even full outages

Estimated Costs
Theoretical Scenario:
serve all drivers of Austria = 3 million drivers concurrently

- cloud-model: ~20.000$ per month
- cloud/edge-model: ~27.000$ per month

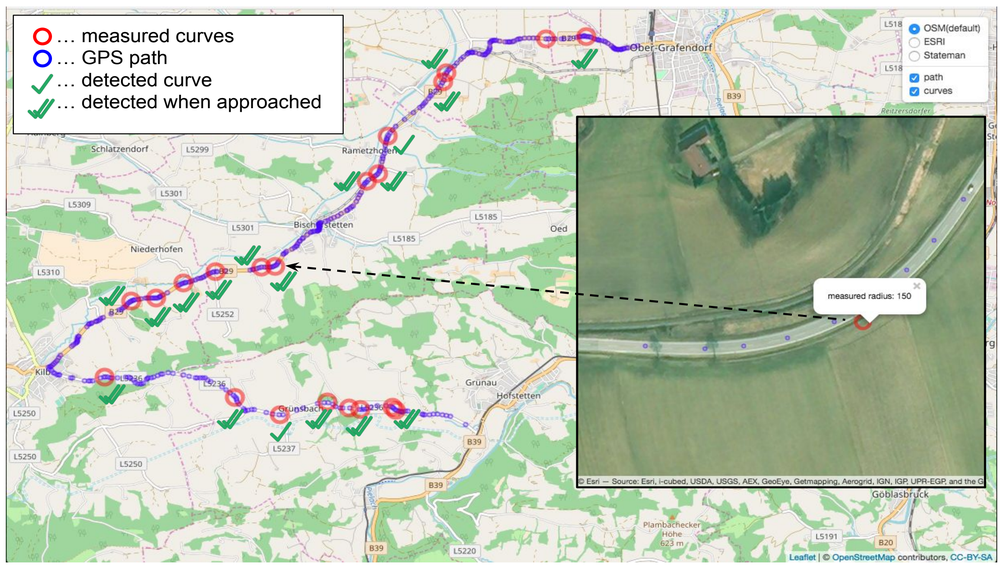

Fig.5: Test results on our predefined test-track for the evaluation of the prototype

Prototype
available at: https://github.com/rdsea/EdgeCorneringAssistance

Kontakt: matthias.karan@gmail.com