



Track Mates

SBB: Indoor-Routing für Alle

Technische Informationen für die Jury



Inhaltsverzeichnis

1	Zugänge	3
1.1	Aktueller Stand des Sourcecodes	3
1.2	Deployment	3
2	Ausgangslage	3
3	Lösungsansatz	3
4	Implementation	4
5	Technischer Aufbau	5
5.1	Bausteinsicht	5
5.2	Verteilungssicht	5
5.3	Technologien und Frameworks	6
6	Abgrenzung / Offene Punkte	6
7	Literatur	7
8	Anhang	8
8.1	Azure Resource Group	8
8.2	Abgespeicherte Pfade	8

1 Zugänge

1.1 Sourcecode

- <https://github.com/search?q=topic%3Atrackmate+org%3Abaernhaeck&type=repositories> [Organisation & Repository Overview]
- <https://github.com/baernhaeck/trackmate-backend> [Coordinator & Persistence]
- <https://github.com/baernhaeck/trackmate-finder> [PWA for Navigation]
- <https://github.com/baernhaeck/trackmate-recorder> [PWA for feeding data]
- <https://github.com/baernhaeck/trackmate-embedding> [Service Images to Feature Vector]
- <https://github.com/baernhaeck/trackmate-instructions> [Service Textual/Audio Way Description]
- <https://github.com/baernhaeck/trackmate-object-detection> [Service Obstacle Detection]
- <https://github.com/baernhaeck/trackmate-misc> [Documentation, Presentation, Misc]

1.2 Deployment

- <https://black-water-00f820303.5.azurestaticapps.net> [Finder PWA]
- <https://agreeable-mushroom-0ed68b003.5.azurestaticapps.net> [Recorder PWA]
- <https://trackmate-backend-bscxaycdesb5gkeg.westeurope-01.azurewebsites.net/swagger> [Backend Swagger Docs]
- <https://trackmate-embedding-cbfje4ebcfgsfaay.westeurope-01.azurewebsites.net/docs> [Embedding Swagger Docs]
- <https://trackmate-instruction-fxa6e4g4e8hcaabp.westeurope-01.azurewebsites.net/docs> [Instruction Swagger Docs]
- <https://trackmate-object-detection-dagne6gdada8ekgg.westeurope-01.azurewebsites.net/docs> [Object Detection Swagger Docs]
- <http://trackmate-db--zin1cgvl.lemonstone-52a4c370.northeurope.azurecontainerapps.io:7474/> [Neo4J UI]
- <http://trackmate-db--zin1cgvl.lemonstone-52a4c370.northeurope.azurecontainerapps.io:7687/> [Neo4J bolt endpoint]

2 Ausgangslage

Die SBB will ihre Kunden auf der gesamten Reisekette gut informieren. Dazu gehört sowohl die Bereitstellung von Fahrplaninformationen als auch das Indoor-Routing für die Wegbeschreibung in den Bahnhöfen. Diesbezüglich werden Reisenden in der SBB Mobile App auf einer Karte die Fusswege vom und zum Bahnhof sowie Umsteigewege angezeigt. Personen mit einer Sehbehinderung hilft diese Darstellung jedoch nichts. Daher ist es wichtig, eine zusätzliche barrierefreie Lösung anzubieten.

3 Lösungsansatz

Die grundsätzliche Herausforderung beim Indoor-Routing für Menschen mit Sehbehinderung besteht in der fortlaufenden Positionsbestimmung. Denn auch wenn die initiale Position bekannt ist und dadurch eine Wegbeschreibung vorgelesen werden kann, können Wegpunkte nicht erkannt werden. Für unsere Lösung dient die Fussgänger Navigation von Google Maps als Vorbild. Die Google Maps App auf dem Smartphone gibt auditive und haptische Hinweise, an bestimmten Wegpunkten und weiss jeweils, wo sich eine Person befindet. Das Problem beim Indoor-Routing ist, dass es keine GPS-Positionsbestimmung und dass drei Dimensionen existieren (z.B. mehrere Stockwerke). Dadurch müssen andere Möglichkeiten zur Positionsbestimmung geschaffen werden. Es gibt verschiedenste Ansätze jedoch benötigen viele davon zusätzliche Infrastruktur [1]. Damit Bahnhöfe nicht mit zusätzlicher Infrastruktur ausgerüstet werden müssen haben wir uns dafür entschieden Sensoren von Smartphones zu verwenden. Dabei verwenden wir für «TrackMate» eine Kombination von Smartphone Sensoren zur probabilistischen Bestimmung der Position verwendet, wobei die Kamera der Hauptinput ist.

4 Implementation

Das System «TrackMate» besteht aus zwei Teilen. Der erste Teil «Recorder» beschäftigt sich mit der Datenerfassung, während der zweite Teil «Finder» die Indoor-Navigation übernimmt.

Der Recorder zeichnet mittels einer Smartphone Kamera verschiedene Wege durch einen Bahnhof auf. Während dem Abgehen des Weges wird ein Graph aufgebaut. Dabei ist ein Weg für sich allein jeweils ein Graph. Die fortlaufend aufgenommenen Bilder werden als Vektor «embedding» repräsentiert und pro Bild wird ein Knoten angelegt. Die Kante wird zum jeweils vorher aufgenommenen Bild bzw. Knoten hergestellt. Zudem kommen Informationen zu Distanzen auf die Kanten, welche vom Smartphone aber auch im Postprocessing hinzugefügt werden können. Im Postprocessing werden die gleichen bzw. ähnlichen Knoten zusammengeführt und so entsteht potenziell ein einziger Graph, auf dem navigiert werden kann.

Der Finder wird aus der SBB-App heraus aufgerufen und kennt daher Ankunftsort und Zielort. Für Auskunft und Zielort wird die jeweilige Node im Graph gefunden und der kürzeste Pfad wird bestimmt. Die App gibt dem Benutzer den Hinweis, wo der nächste Wegpunkt (repräsentiert durch einen bestimmten markierten Knoten) zu finden ist. Dies wird mit einem Text-To-Speech das er hingehen muss. Während dem Abgehen des Wegs wird für laufend mittels visuellen Abgleiches und unter Einbezug der Beschleunigung und Richtung die Position im Graphen bestimmt. Zusätzlich wird laufend eine Obstacle Detection durchgeführt und der Benutzer wird visuell und haptisch gewarnt, wenn eine Kollision droht.

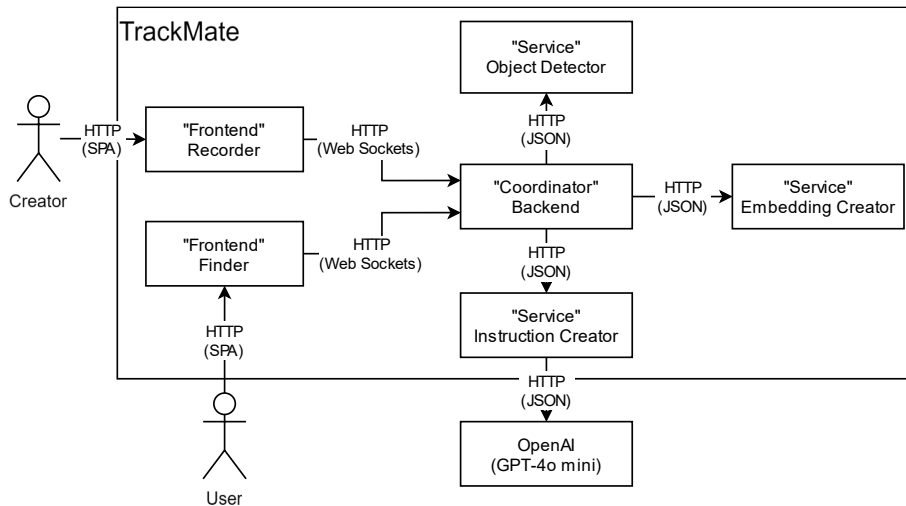
In der Schweiz beträgt die Netzabdeckung mit 4G über 99% [2] über alle Telekomanbieter hinweg war 2023 mit einem Upload Speed über das 4G Netz von min. 10Mbit/s [3] zu rechnen. Die Anwendung «TrackMate» ist Bandbreiten optimiert und benötigt nur 21 Kbit/s.

Die Lösung ist auf Azure gehostet und alle Komponenten sind einsatzbereit. Was nicht implementiert wurde ist angedacht und die bereits implementierten Einzelteile bestätigen, dass alles wie geplant implementiert werden kann.

5 Technischer Aufbau

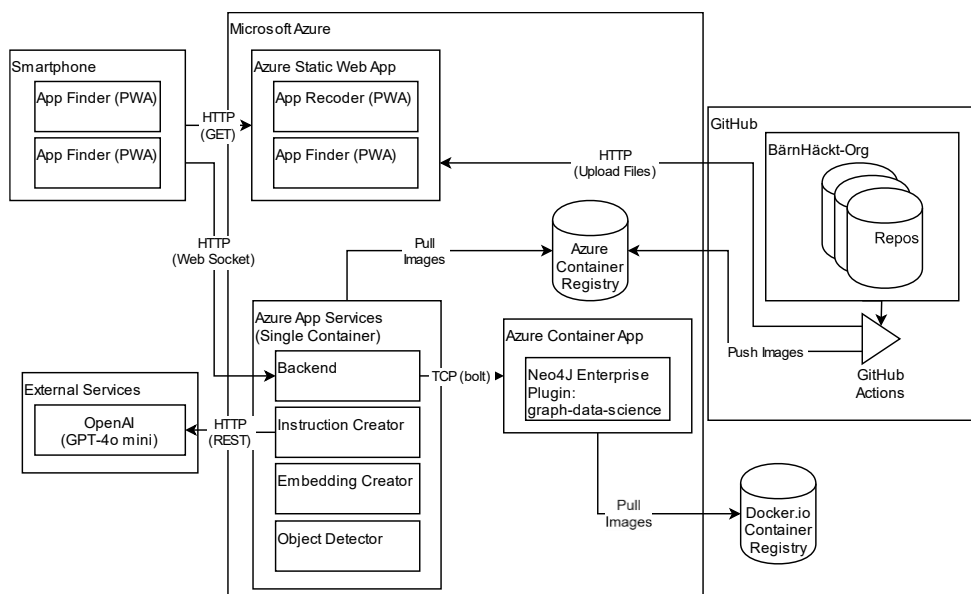
5.1 Bausteinsicht

Die Lösung bietet zwei PWA's als Frontend, welche mittels Websockets mit dem Backend kommunizieren. Das Backend speichert die Daten und benutzt andere Services, welche KI-Dienste zur Verfügungstellung.



5.2 Verteilungssicht

Die Lösung wird zur Laufzeit auf Microsoft Azure gehostet. Dabei kommen die Azure Services App Services für Single Container, Container Instance, Static Web App zum Einsatz. Als schneller Cache wird ein File Share über einen Azure Storage Account zur Verfügung gestellt. Zur Entwicklungszeit wird die Lösung auf GitHub gehostet, CI/CD ist mittels GitHub Actions implementiert und alle Teile werden kontinuierlich ohne Downtime deployt. Die Docker Images werden auf der GitHub Container Registry abgelegt. Der Anhang 8.1 Azure Resource Group zeigt die Implementierung auf Azure.



5.3 Technologien und Frameworks

- Frontend: Vue.js
- Backend: .NET 8, Docker
- Persistenz: Neo4j mit graph-data-science Plugin, Docker
- WebSocket Protocol: SignalR
- ML Services: Python v3.12, Docker
 - Embedding: Resnet Image Classification Mode. Trainiert auf Imagenet. TrackMate verwendet die Representation vom zweit letzten Layer (vor der Klassifikation), um die beste Repräsentation aller Features zu erhalten.
 - Instructions: GPT-4o mini für Text generierung und Wisper für Text-To-Speech (Open AI)
 - Objekt Detection: YOLO v10 ein Modell für Real-Time Object Detection

6 Abgrenzung / Offene Punkte

Grundsätzlich haben wir uns bei der Implementierung der Lösung auf die wichtigsten Elemente der «User Journey» konzentriert. Gleichzeitig wollten wir verifizieren, dass die kritischen Elemente wie gewünscht funktionieren. Dabei haben wir folgende Dinge nur konzeptuell angedacht und noch nicht implementiert.

- Probabilistische Kalkulation: Es wird lediglich das Embedding verwendet, um einen passenden Knoten zu finden, keine weiteren Sensordaten.
- Merging von Knoten: Es gibt kein automatisches Post-Processing, welches gleiche Knoten erkennt und zusammenführt, um einen zusammenhängenden Graphen zu bilden.
- Manuelles Nachbereiten: Es gibt kein UI und Prozess zur manuellen nachbereitung, so das Daten präzisiert werden können.
- Persistenz für Bilder: Nur die Embeddings werden behalten, ohne die Quellbilder. Die Bilder könnten einfach temporär gespeichert werden und für das manuelle Postprocessing zu ermöglichen.
- Distanz auf Kanten: Es gibt keine Informationen auf den Kanten.
- Labeling von Knoten: Knoten werden nicht speziell gekennzeichnet (start/end & turning point)
- Graph Optimierungen: Es gibt keine Optimierungen, welche die Graph Traversierung und Suche in der Realität schnell genug machen würde.
- Mapping auf Karte: Die Karten der SBB sind nicht auf den Graphen gemappt.
- Performance Objekterkennung: Die Objekterkennung verwendet nur CPU und keine GPU, dadurch ist sie zu langsam (aktuell ~2s, es wäre jedoch ohne Probleme ~10ms möglich)

7 Literatur

[1] Chen, R., Chen, L. (2021). Smartphone-Based Indoor Positioning Technologies. In: Shi, W., Goodchild, M.F., Batty, M., Kwan, MP., Zhang, A. (eds) Urban Informatics. The Urban Book Series. Springer, Singapore. https://doi.org/10.1007/978-981-15-8983-6_26











[2] <https://www.comcom.admin.ch/comcom/de/home/dokumentation/zahlen-und-fakten/mobilfunkmarkt/mobilfunkabdeckung.html> (24.08.2024)

[3] <https://www.opensignal.com/reports/2023/05/switzerland/mobile-network-experience> (24.08.2024)

8 Anhang

8.1 Azure Resource Group

Die Azure Resource Group "trackmate".

<input type="checkbox"/> Name ↑	Type ↑
▼ App Service	
<input type="checkbox"/>  trackmate-backend	App Service
<input type="checkbox"/>  trackmate-embedding	App Service
<input type="checkbox"/>  trackmate-finder	App Service
<input type="checkbox"/>  trackmate-instruction	App Service
<input type="checkbox"/>  trackmate-object-detection	App Service
▼ Container App	
<input type="checkbox"/>  trackmate-db	Container App
▼ Container registry	
<input type="checkbox"/>  trackmate	Container registry
▼ Static Web App	
<input type="checkbox"/>  trackmate-finder	Static Web App
<input type="checkbox"/>  trackmate-recorder	Static Web App
▼ Virtual network	
<input type="checkbox"/>  trackmate	Virtual network

8.2 Abgespeicherte Pfade

Gespeicherte Nodes in Neo4J für verschiedene Pfade an der BFH.

Query: MATCH (n) RETURN n LIMIT 250

