MultimodalRouting - Multimodale Routing Engine für Rollstuhlfahrer und Fahrradlogistiker

Schlussbericht

Zuwendungsempfänger:

Lötzsch, Tilsch & Sultova GbR

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS19S01 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

Die Community um OpenStreetMap und Wheelmap pflegt eine umfangreiche Sammlung freier Daten über Barrierefreiheit, die für Rollstuhlfahrer und Lastenräder sehr nützlich sind. Wir wollen das Potential dieser Daten nutzen, um eine für die Bedürfnisse dieser Nutzer optimierte Navigationsanwendungen zu entwickeln.

Primäre Motivation war, körperlich eingeschränkten Menschen bei selbstständiger Verkehrsteilnahme an unbekannten Orten zu unterstützen. Von Anfang an wurde erkannt, dass die im Projekt entwickelten Routinglösungen auch auf viele andere Einsatzzwecke übertragbar sind. In einer bestehenden Kooperation wurde seit längerem nach ähnliche Werkzeugen gesucht, um alternative Innenstadtlogistikkonzepte durch geeignete Software zu unterstützen.

Projektziel war die Entwicklung einer Routingengine, die Navigation für kombinierte Nutzung von nichtmotorisiertem Individualverkehr und öffentlichen Verkehrsmitteln unterstützt und dabei vorhandene Informationen zu akuten Hindernissen (z.B. nicht funktionierende Fahrstühle, Ausfälle/Umleitungen im ÖPNV) berücksichtigt. Besonderen Fokus sollte auf die Verwendbarkeit individualisierbarer Routingprofile gelegt werden.

Wichtigste Meilensteine waren die Untersuchung frei verfügbarer Datenquellen mit einem Fokus auf barrierearmes Routing, die Implementierung einer Routingengine für aggregierte dynamische Daten, sowie der Entwurf eines Frontends, dass verfügbare Daten verständlich visualisiert.

Technische Herausforderung

Suchalgorithmen sind wissenschaftlich ausführlich untersucht. Zur informierten Routensuche in euklidischen Räumen werden meist Varianten des vollständigen und optimalen A*-Algorithmus verwendet. Er hat im Verhältnis zur Anzahl der Knoten eine quadratische worst-case Laufzeit.

Es existiert eine Vielfalt freier Routingengines, mittels derer auf OpenStreetMap-Daten navigiert werden kann. Diese sind üblicherweise auf Performance optimiert, damit auch für lange Strecken effizient optimale Lösungen gefunden werden können. Um das zu erreichen, wird sich auf eine beschränkte Anzahl verfügbarer Routingprofile festgelegt, für die jeweils eine Zuordnung von Streckenabschnitt zu Routingkosten (z.B. Zeitaufwand) statisch vorberechnet wird.

Um während des Routings auf dynamische Größen (z.B. Wahrscheinlichkeit dass ein Fahrstuhl nicht funktioniert, sich ändernde individuelle Nutzerpräferenzen oder aus überwundenen Strecken erlernte Heuristiken) reagieren zu können, muss ein flexiblerer Ansatz verwendet werden. Zu unseren Aufgaben gehörte es daher, entweder eine bestehende Routingengine für die Nutzung mit dynamischen Routingkosten anzupassen oder eine neue Routingengine zu entwickeln.

Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Wir wollen körperlich eingeschränkten Personen mit Hilfe von Public Interest Tech in Form von durch Crowdsourcing gesammelten offenen Daten und einer ehrenamtlich gepflegten Free/Libre Open Source App helfen. Nach unserer Auffassung sollten solche Werkzeuge zur Daseinsvorsorge zählen und halten die Prototypfinanzierung nach dem Paradigma „Public Money, Public Code“ daher für angemessen.

Neben der konkreten App, bietet auch das entwickelte Routingframework ein großes Potential als Plattform weiterer Anwendungen, die Menschen mit besonderen Anforderungen unterstützen können.

Wir wurden darauf aufmerksam gemacht, dass Eltern mit Kinderwagen und Senioren mit Rollatoren ähnliche Anforderungen haben. Es gibt bereits

Überlegungen anderer Entwickler, auf unserer Routingengine aufbauend eine Navigationsanwendung für Blinde zu entwickeln.

Im Kontext von verkehrsmittelübergreifenden Verkehrskonzepten und alternativen Antriebstechnologien, kann multimodales Routing einen Beitrag zum Klimaschutz leisten.

Spannende Anwendungen könnten sich ergeben aus Kooperation mit Nahverkehrsanbietern (Park and Ride), Anbietern von Leihfahrrädern und E-Scootern, sowie im Bereich E-Mobilität (Einbeziehung von Ladezeiten in die Routenplanung). Es existieren zahlreiche Möglichkeiten, wie das Framework für stadtplanerische Fragestellungen ein hilfreiches Werkzeug darstellen kann.

*Ausführliche D*arstellung der Ergebnisse

Zusammenfassung

Es wurde ein Framework implementiert, mittels dessen Routinganwendungen für Spezialanforderungen unter Verwendung von aggregierten Daten aus unterschiedlichen Quellen gebaut werden können.

Im einfachsten Fall müssen Entwickler nur die anwendungsspezifischen Routingprofile mittels Kostenfunktion selbst implementieren. Dafür kann wahlweise Java oder Javascript verwendet werden.

Das Framework stellt alle anderen notwendigen Komponenten bereit. Dazu zählen Routingengine, Anbindung von Datenquellen, UI sowohl für die Darstellung von Routen in der App, als auch zusätzliche Kartenoverlays zur Darstellung von Debuginformationen. Auf diese Weise können Datenqualität für den benötigten Zweck und Eigenschaften der entwickelten Kostenfunktion visualisiert werden.

Das Backend ist vollständig in Clojure(script) implementiert und kann dadurch sowohl für die JRE als auch nach Javascript compiliert werden. Mit dem Framework gebaute Routinganwendungen können wahlweise als Client-/Server-Architektur oder vollständig im Client laufen. Unsere Build-Pipeline kann die Anwendungen mittels Cordova als native Android-, iOS- oder Electron-Apps bauen.

Auf Basis des Frameworks wurde eine Navigationsanwendung für Menschen mit körperlichen Einschränkungen entwickelt, die sich derzeit in einer frühen Testphase befindet. Eine Integration in eine Logistikanwendung für Fahrradkuriere ist geplant.

Einzelne Funktionen (z.B. Darstellung von Isochronen) können auch als separate Karte in Webseiten eingebunden werden.

Anforderungsanalyse, allgemeine Recherche

~2 Mannwochen

Eine ausführliche Anforderungsanalyse, die gemeinsam mit Vertretern der Zielgruppe durchgeführt wurde, half uns die Bedürfnisse unserer Nutzer besser zu verstehen und zu priorisieren. Besonders wertvoll war es, von Personen mit sehr unterschiedlichen Einschränkungen zu erleben, welche Hindernisse für sie relevant sind, zu sehen bei welchen Alltagssituationen bestehende Lösungen für diese Anwender nicht funktionieren und erklärt zu bekommen, welche Informationen als besonders hilfreich erachtet werden.

Es setzte sich die Erkenntnis durch, wie vielfältig die benötigten Routingprofile sind und dass auch für einen einzelnen Nutzer je nach Situation unterschiedliche Routingprofile benötigt werden (z.B. abhängig davon, ob er mit einer Begleitperson, einem Zusatzmotor oder einer anderen eingeschränkten Person unterwegs ist).

Das Bedürfniss relevante Informationen detailiert und dennoch übersichtlich visualisiert zu bekommen bestätigte sich. Alternative Darstellungsoptionen wurden gemeinsam erörtert.

Untersuchung vorhandener OSM Routingengines

~1,5 Mannwochen

Als vielversprechenste Kandidaten für eine Anpassung ergaben sich Graphhopper, Openrouteservice und BRouter.

Graphhopper zeichnet sich insbesondere durch gut lesbaren Code, umfangreiche Dokumentation, reichhaltige Funktionalität, effiziente Datenstrukturen und eine stetige Weiterentwicklung aus.

Die Einarbeitung in bestehende Routingengines gab hilfreiche Anregungen für die spätere eigene Implementierung.

Untersuchung der Qualität existierender Daten

~1 Mannwoche

Es wurde in verschiedenen Regionen untersucht, ob die verfügbaren Daten für die von uns geplanten Lösungsansätze ausreichen. Dafür wurden sowohl Datensätze konkreter relevanter Orte im Detail betrachtet, als auch qualitative Vergleiche verschiedener Regionen berechnet. Es stellte sich heraus, dass Dresden aufgrund der guten Datenlage ein vielversprechender Ort für Tests ist.

Frontendentwicklung

~5 Mannwochen

Besondere Relevanz für unsere Anwendung hat die übersichtliche Informationsdarstellung auf Karten. Das UI-Coaching half uns eine geeignete Darstellung zu finden, die neben der Beurteilung von Barrierefreiheit auch Auskunft über die Konfidenz der Daten liefert.

Ein entwickeltes Plugin für React-Leaflet, dass beliebige Visualisierungen auf Leaflet- und Mapbox-Karten mittels D3 ermöglicht, kann auch für viele andere Projekte im Bereich Data Science hilfreich sein.

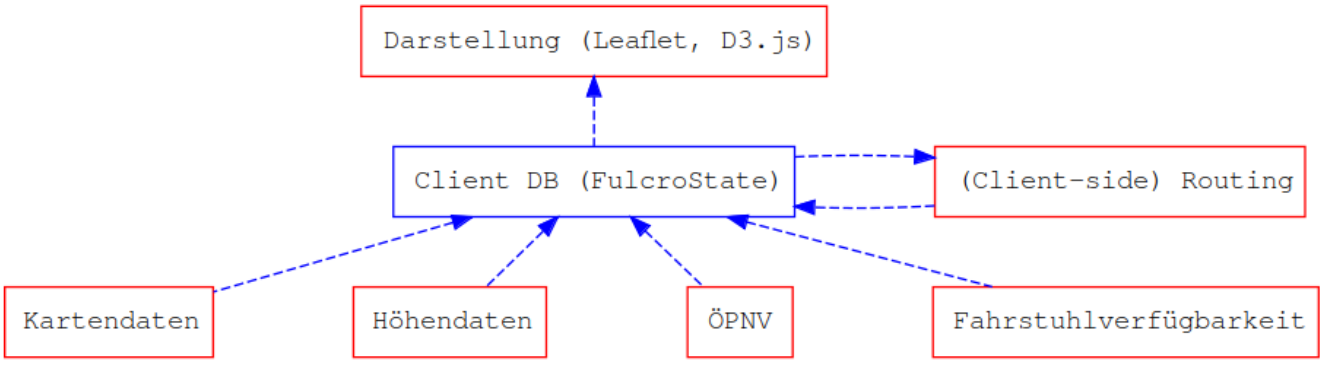
Durch Verwendung von Framework7 kann unsere Webanwendung auf den gängigen Plattformen mit nativem Look and Feel genutzt werden.

Framework

~7 Mannwochen

Wir haben uns von vornherein bemüht, möglichst große Teile der Anwendung für ähnliche Projekte wiederverwendbar zu designen.

Der Kern unserer Architektur besteht aus einer Datenbank, die Geofeatures strukturell ähnlich zum JOSM-Dateiformat speichert. Diese Datenbank kann mit Geodaten verschiedenster Quellen befüllt werden. Die von uns implementierte Routingengine kann verwendet werden, um auf den vereinheitlichten Geofeatures Routen zu berechnen. Routen können selbst wiederum als Pfad oder Relation in der Datenbank abgelegt werden. Das Frontend stellt Methoden bereit, um beliebige Geofeatures darzustellen.

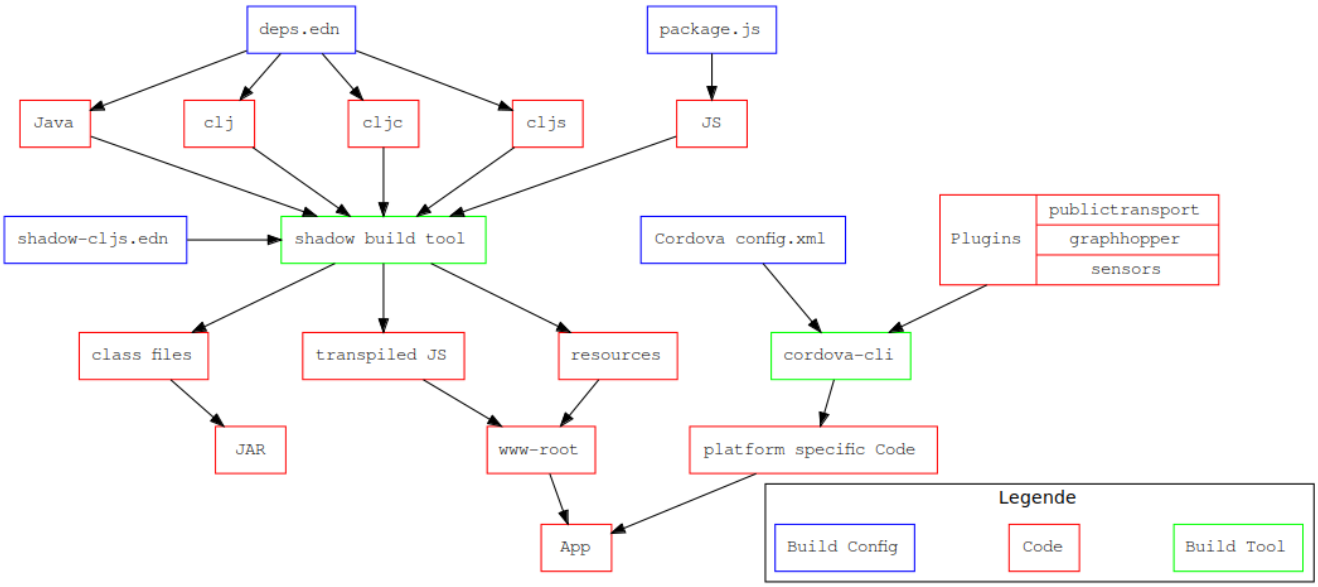


Auf Basis dieser Architektur ist eine effiziente Entwicklung möglich, da für neue Routingprofile geschriebene Kostenfunktionen und hinzugefügte Datensätze beliebig miteinander kombiniert und für Debugging-Zwecke ohne zusätzlichen Aufwand visualisiert werden können.

Das Frontend läuft im Browser. Der Rest des Frameworks ist aufgrund der Implementierung in Clojure(Script) sowohl nach Javascript als auch für die JVM compilierbar.

Anwendungsentwicklung unter Nutzung des Frameworks besteht primär daraus, Kostenfunktionen und optional zusätzliche Anbindungen für Datenquellen wahlweise in Javascript oder Java zu implementieren. Wenn auch dafür Clojure(Script) verwendet wird, behalten Prototyp-Entwicker die volle Flexibilität, welche Teile der Anwendung später im Front- oder Backend laufen.

Das Framework stellt eine Build- und CI-Pipeline bereit, um neben reinen Webanwendungen mittels Cordova auch native Apps für Android und iOS zu bauen.



Um plattformunabhänig, energiesparend und zeitlich hochauflösend den aktuellen Gerätestandort zu ermitteln, nutzen wir das Cordova Background Geolocation Plugin.

Anbindung von Datenquellen

~4 Mannwochen

Für die Nutzung mit unserem Routingframework wurden bereits eine Vielzahl an Datenquellen angebunden. Zum Zweck der Rollstuhlnavigation sind das Kartendaten, Höhendaten, ÖPNV-Fahrplandaten, und Informationen über die aktuelle Funktionalität von Fahrstühlen.

Eine Anbindung an Graphhopper-Routenabfragen kann genutzt werden, wenn für ein in Graphhopper existierendes Routingprofil (z.B. PKW) performante (überregionale) Abfragen gestellt werden sollen.

Während der Entwicklung ist die Nutzung von Overpass-Abfragen als Kartenquelle sehr hilfreich.

Routingengine

~3 Mannwochen

Die entwickelte Routingengine verwendet das einheitliche Geofeature-Format des Frameworks und funktioniert unabhängig von der Herkunft der Daten.

Die Engine expandiert Geofeature (Punkte, Linien, Flächen, Relationen) in einen Routinggraph im Format der Clojure-Bibliothek Loom. Eine vom Anwendungsentwickler übergebene Kostenfunktion, die das Routingprofil definiert, wird verwendet um jeder Kante im Routinggraph ein Gewicht zuzuweisen.

Zur Berechnung der kürzesten Pfade können sowohl die Graphenbibliothek Loom, als auch die zu ihr kompatible Bibliothek Ubergraph verwendet werden. Dafür existieren jeweils Implementierungen von Dijkstra und A*.

Mehrere Loom-Graphen lassen sich problemlos miteinander mergen. Dass erlaubt es die Expansion und Kostenberechnung statischer Daten einmalig, die von dynamischen Daten jedoch zusätzlich so oft wie nötig auszuführen.

Routingergebnisse können wieder in das einheitliche Geofeature-Format gerechnet und die Eingangsdaten für jeden Knoten referenziert werden, indem eine zur Expansion inverse Funktion verwendet wird.

Beta-Tests

~3,5 Mannwochen

Um erste Ergebnisse zu evaluieren und zu verbessern, wurde implementiert, dass von Testnutzern im Alltag zurückgelegte Strecken aufgezeichnet und mit den von uns berechneten Routenvorschlägen verglichen werden können.

Da uns während dieser Projektphase vor allen iOS-Nutzer zur Verfügung standen, wurde der zusätzlich notwendige Aufwand getrieben, um die App auch für iOS bereitzustellen.

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Framework

Das entwickelte Routingframework ist vielseitig einsetzbar. Es ermöglicht mit verhältnissmäßig geringem Aufwand Routinganwendungen für Spezialanforderungen zu bauen.

Es stellt Anbindungen verschiedener Datenquellen und deren Transformation in ein einheitliches Geofeature-Format, Methoden um daraus Routinggraphen zu generieren, sowie darauf arbeitende Routingalgorithmen und Visualisierungen der Ergebnisse bereit. Die Buildpipeline kann genutzt werden, um Webanwendungen, native Android-, iOS- oder Electron-Apps zu bauen. Die Anwendung kann wahlweise vollständig Clientseitig oder als Client-/Serverlösung gebaut werden.

Zum Bau einer Routinganwendung mit nutzerspezifischen Profilen, muss unter Verwendung unseres Frameworks bestenfalls nur die Kostenfunktion implementiert werden.

Bei Bedarf können neben den bereits angebundenen Datenquellen weitere Quellen implementiert werden.

Derzeit können OpenStreetMap-Karten im JOSM-Format (*.osm), deren Json-Entsprechungen (OSM JSON), sowie im Protocolbuffer-Binärformat (*.pbf) genutzt werden. Ebenso existiert die Möglichkeit Daten per Overpass API und aus PostGIS-Datenbanken im Imposm3-Format abzufragen oder MapboxVectorTiles (*.mvt) zu nutzen.

Höhendaten können als TerrainTiles und Fahrpläne aus GTFS-Quellen eingebunden werden. Echtzeitdaten, ob Fahrstühle funktionieren, können über die APIs von accessibility.cloud und der Deutschen Bahn verwendet werden.

Angedacht, jedoch noch nicht implementiert ist die Nutzung von Daten aus dem Open-Data-Portal der Stadt Dresden zu Bordsteinabsenkungen und Behindertenparkplätzen.

Neben unseren Anwendungen für Rollstuhlfahrer und Fahrradlogistiker gibt es weitere Anfragen zur Nutzung des Frameworks, die wir gerne unterstützen. Es gibt konkrete Pläne für eine Navigationsanwendung für Blinde, die auf unserem Framework basierend eine textuelle und auditive Nutzerschnittstelle bereitstellen will. Ebenfalls in Überlegung ist die Nutzung für einen barrierefreien Campusnavigator.

Noch keine konkreten Pläne, jedoch erste Ideen gibt es in Kooperation mit einer Logistikfirma, die Navigation auf Vorfeldern zur Abfertigung von Luftfracht, sowie zur Optimierung der Anlieferung auf Werksgeländen benötigt.

Bisher hat die Arbeit in dem Projekt vor allem die beteiligenten Entwickler in ihrer persönlichen, und fachlichen Weiterentwicklung vorangebracht und mit dem Framework ein großes Potential an Folgeprojekten ermöglicht.

Wir planen das Framework gründlich aufzuräumen, einzelne Funktionalitäten in separate Bibliotheken zu unterteilen und für andere Entwicker verständlich zu dokumentieren.

Routinganwendung für körperlich eingeschränkte Menschen

Unsere Evaluation hat ergeben, dass bereits für sämtliche aus unserer Sicht relevanten Weg- und Hindernissattribute Tags zur Kartographierung in der OpenStreetMap definiert wurden. Leider gibt es teilweise verschiedene Varianten, die berücksichtigt werden müssen.

Die Datenlage ist in verschiedenen Städten, abhängig von der lokalen OSM-Community, sehr verschieden.

Unsere Routinganwendung zeigt den Mehrwert einiger Kartendaten, die von anderen Navigationsanwendungen bisher noch nicht berücksichtigt wurden. Wir untersuchen aktiv, wie durch ehrenamtliche Mapper und durch OpenData-Initiativen ein effizienter Beitrag für körperlich eingeschränkte Menscher erzielt werden kann. Unsere App und die mit ihr verbundene Öffentlichkeitsarbeit soll am Beispiel von Städten mit guter Datenlage motivieren, auch in anderen Regionen schrittweise die Erfassung relevanter Daten zu verbessern.

In Städten, bei denen pro Bus-/Straßenbahnlinie jeweils Fahrzeuge mit einheitlicher Einstiegshöhe verwendet werden, kann ein sehr großer Mehrwert für körperlich eingeschränkte Menschen erzielt werden, indem Haltepunkte danach klassifiziert werden, wie diese nutzbar sind. Dafür sind die 3 Varianten: barrierefrei (wheelchair=yes), bei Verwendung einer Rampe (wheelchair=limited) oder nicht barrierefrei (wheelchair=no) vorgesehen. Aus unserer Sicht ist das der einfachste Weg, wie lokale OSM-Communities einen ersten effizienten Beitrag leisten können.

Für am zweitwichtigsten halten wir das Mappen von Fahrstühlen, Rampen, Stufen/Treppen, Pollern, Schranken und ähnlichen Hindernissen.

Ebenfalls wertvoll ist das Verzeichnen von Straßenübergänge über stark befahrene Straßen mit dem Tag wheelchair, sowie gegebenen Falls die Kennzeichnug der Existenz einer Ampel.

Mit größerem Aufwand verbunden ist das konsequente Mappen von Bürgersteigen (jeweils beide Fahrbahnseiten separat) mit den Attributen für Belag, Glattheit, Breite und Steigung (oder vereinfacht mit dem wheelchair-Attribut).

Nach unseren ersten Erfahrungen, kann eine hilfreiche Rollstuhl-Navigationsanwendung auf Basis von Heuristiken auch dann implementiert werden, wenn nicht jede einzelne Bordsteinabsenkung bekannt ist. Ein nützlicher

Mehrwert wird bereits durch das Mappen unerwartet unzureichender Absenkungen erzielt. Selbstverständlich ist jede verzeichnete Bordsteinabsenkung ein Mehrwert, insbesondere in Kreuzungsbereichen und an anderen wichtigen Übergängen.

Viele OSM-Routingengines können nur auf OSM-Elementen vom Type Linie, jedoch nicht auf solchen vom Typ Punkt routen. So sind sie beispielsweise unfähig, als Punkt gespeicherten Ampeln, Haltestellen oder Fahrstühlen Wegkosten zuzuweisen. Unsere Engine erlaubt bei der Expansion des Routinggraphs aus OSM-Elementen, egal ob Punkt, Linien oder Relation, jeweils einen oder mehrere Graphkanten zu erzeugen. So können bei Ampeln und Haltestellen separat Kosten für erwartete Wartezeit und Überwindung des räumlichen Hindernisses berechnet werden. Bei Fahrstühlen wird eine zusätzliche Graphkante für die erwarteten Kosten bei Defekt hinzugefügt. Diese kann aus Ausfallwahrscheinlichkeit und in diesem Fall nötigem Alternativweg abgeschätzt werden.

Aufgrund der regional sehr unterschiedlichen Datenlage, liefern traditionelle Routingprofile, die Wegen entsprechend ihrer Attribute konstante Streckenkosten zuweisen, nur mäßige Ergebnisse. Insbesondere ist es bei großer Varianz schwierig, die Kosten für verschiedene Wegtypen und der Kombination ihrer Attribute richtig zu gewichten. Wir begegnen der Herausforderung, indem wir eine zweidimensionale Metrik, bestehend aus geschätztem (Zeit-)Aufwand und geschätzter Konfidenz, verwenden. Die Berechnung der finalen Wegkosten im Routinggraph nutzt eine Normalisierung in Abhängigkeit der regionalen Datenlage sowie von Nutzerpräferenzen.

Künftig sollen auch vom Gerät des Nutzers lokal gesamelte Daten genutzt werden, die Aufschluss darüber geben, wieviel Zeit für das Zurücklegen ähnlicher Wege benötigt wurde.

Ein wichtiges Alleinstellungsmerkmal unserer Engine ist der Umgang mit Hindernissen, bei denen wir uns nicht sicher sind, ob der Nutzer sie überwinden kann und möchte. Unser Ziel ist es, dem Nutzer eine selbstbestimmte, informierte Entscheidung zu ermöglichen. Deshalb wollem wir dem Nutzer sowohl die Möglichkeit geben, die als optimal berechnete vollständig barrierefreie Route zu wählen, als auch Abkürzungen die wahrscheinlich mit Aufwand überwindbare Hindernisse beinhalten. Unsere Lösung zeigt daher die als sinnvoll berechneten Alternativen an, visualisiert Hindernisse als solche klar erkenntlich und ermöglicht alle bekannten Informationen über das Hindernis einzusehen.

Künftig wollen wir aus dem vom Nutzer getroffenen Entscheidungen lernen, um die Vorschläge individuell verbessern zu können. Ein neues Nutzerprofil soll zu

Beginn einer sehr konservativen Konfiguration entsprechen, also nach Möglichkeit jeder Form von Hindernissen ausweichen. Wenn der Nutzer sich für Alternativrouten entscheidet und deren Hindernisse überwindet, sollen ähnliche Hindernisse schrittweise im Nutzerprofil freigeschaltet werden. Während der Überwindung von Hindernissen wird die benötigte Zeit gemessen und künftig für die Abschätzung der benötigten Streckenkosten verwendet.

Wir erhoffen uns damit, individuellen Bedürfnissen und Vorlieben entgegen zu kommen, ohne dass jeder neue Nutzer umfangreiche Einstellungen vornehmen muss.

Eine spannende Fragestellung für künftige Experimente ist, wie zuverlässig aus GPX-Tracks, die um Daten von Beschleunigungssensoren angereichert wurden, auf Attribute von Wegen geschlossen werden kann. Dies könnte eine Lösung zur Datenerhebung ermöglichen, bei denen unsere App nutzende Rollstuhlfahrer gebeten werden, die automatisch ermittelten Streckeneigenschaften (z.B. Vorhandensein einer Bordsteinabsenkung) zu bestätigen.

Während der Entwicklung wurde uns von künftigen Nutzern ein großes Interesse gespiegelt. Es besteht ein deutlicher Wunsch nach Autonomie im Alltag und das Bedürfniss insbesondere an unbekannten Orten die Effizienz und Planungssicherheit zum pünktlichen Erreichen von Zielen zu erhöhen.

Wir stehen mit Rollstuhl-Communities in drei verschiedenen Städten in Kontakt, die uns beim Testen unterstützen.

Die im Coaching als Grundlage von nutzerzentrierter Entwicklung angeregten gemeinsamen Tests konnten aufgrund des COVID-19-Lockdowns seit Mitte März nicht wie ursprünglich geplant durchgeführt werden. Der Test- und Entwicklungszyklus soll aber zeitnah wieder aufgenommen werden.

Nach erfolgreichen Tests soll die App bei F-Droid und Google Play veröffentlicht werden. Ob auch eine Verbreitung über den Apple App Store stattfinden wird, ist unsicher. Bisher sind die umfangreichen Bemühungen ein Apple Developer Certificate zu erwerben gescheitert.

Es ist beabsichtigt, die App ehrenamtlich weiter zu entwickeln. Hilfreich wäre es, die App schlank zu halten und möglichst viel der Funktionalität mit anderen Anwendungen im Framework zu teilen.

Anwendung für Fahrradkuriere

Das entwickelte Routingframework soll genutzt werden, um eine auf die Bedürfnisse von Fahrradkurieren abgestimmte Routenberechnung in eine bestehende Logistikanwendung zu integrieren.

Geeignete Routingprofile werden sowohl für Rennradfahrer, als auch Lastenradfahrer benötigt.

Die Routenberechnung kann vom Fahrer zur Navigation, vom Dispatcher für die Einsatz-Planung und vom System zur Abschätzung Entfernungs- und Zeitabhängiger Auftragspreise verwendet werden.

Eine Firmenspezifische Routinganwendung kann es ermöglichen, Verkehrsinformationen, Informationen über Streckensperrungen, sowie Hinweise zu Abhol- und Zustellorten zwischen verschiedenen Fahrern und Dispatchern auszutauschen.

Eine einheitliche Fahrerapp kann sowohl für die Navigation, zum melden der aktuellen Position an den Dispatcher und wartende Kunden, zur Übermittlung neuer Aufträge an den Fahrer, als auch zur allgemeinen Kommunikation zwischen Fahren und Dispatcher verwendet werden.

Einbettung des Plugins auf Webseiten

Manche Funktionen des Frameworks sollen auch als mit minimalem Aufwand in Webseiten einbindbare Karten zur Verfügung gestellt werden.

Beispielsweise kann die Berechnug von Isochronen genutzt werden, um auf einer Veranstaltungswebsite den Nutzern Orte von Interesse (z.B. Hotels, Restaurants, Haltestellen) anzuzeigen, die dem Veranstaltungsort nicht nur räumlich in der Nähe sind, sondern von diesem auch bequem erreicht werden können.

Vernetzung und Weiterbildung

Neben der fachlichen Arbeit profitierten wir besonders durch den fachlichen Austausch innerhalb des Teams und mit der Community.

Sehr wertvoll für uns waren die im Rahmen der Förderung angebotenen Coachings.

Auf der Konferenz "State of the Map" haben wir uns mit Vertretern der Open-Street-Map Community und Wissenschaftlern aus dem Bereich der Geo-Informations-Systeme vernetzt. Beim Accessibility Club Summit 2019 bildeten wir uns zu barrierefreier Softwareentwicklung weiter.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

In der Beta-Test-Phase am Ende des Förderungszeitraumes standen uns vor allem Nutzer von Geräten mit iOS zur Verfügung. Daher bemühten wir uns iOS-Builds unserer App zur Verfügung zu stellen. Leider funktioniert dies nur unter Verwendung von Xcode, welches wiederum nur für MacOS existiert. Wir schaften uns daher einen Mac zum bauen und ein iPad für eigene Tests auf der Plattform an. Wir können unsere App nun auch für iOS compilieren, per USB-Kabel installieren und identisch zur Android-App nutzen.

Um Apps auf entfernten iOS-Geräten (insbesondere für Testnutzer in anderen Städten wichtig) installieren zu können, muss die App mit einem Apple Developer Certificate signiert werden. Wir haben daher auch ein solches Zertifikat bei Apple käuflich erworben. Leider wurde das Zertifikat nach kurzer Zeit ohne Begründung von Apple widerrufen. Nach sehr vielen Telefonaten mit dem Apple-Support, bei denen wir immer wieder weitergeleitet wurden, uns jedoch von keiner Abteilung geholfen werden konnte, entschieden wir uns für einen komplett neuen Versuch. Obwohl wir unter dem Namen eines anderen Teammitgliedes, mit anderer Telefonnummer und Mailadresse auf anderer Hardware eine neue Apple ID angelegt, als Apple Developer registriert und über eine andere Kreditkarte das Developer Certificate bezahlt haben, wurde das Zertifikat ebenfalls gesperrt. In Folge dessen trafen wir die Entscheidung, uns zunächst auf Android zu beschränken..

Da iOS eine weit verbreitete Plattform ist und von vielen Rollstuhlfahrern genutzt wird, sollten wir mittelfristig mit Unterstützung erfahrener iOS-Entwickler einen weiteren Versuch unternehmen, um die Anwendung auch im Apple App Store zu veröffentlichen.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Der vollständige Quellcode steht auf der Online-Kollaborations- und Codehosting-Plattform Github zur Verfügung.

https://github.com/multimodalrouting

Die Rollstuhlnavigationsapp wurde bei Testnutzern installiert.

Im nächsten Schritt ist eine Veröffentlichung bei F-Droid und Google Play geplant. Die Verbreitung über den Apple App Store kann nicht garantiert werden.

Eine technische Dokumentation des Routingframeworks ist auf der Projektwebsite geplant.

https://multimodalrouting.github.io

Gelante Vorträge mussten aufgrund von COVID-19 abgesagt werden. Es ist jedoch angedacht, diese bei künftigen Konferenzen nachzuholen.

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Die im Projektantrag beschriebene Anwendung wurde unter Einhaltung des Kosten- und Zeitplans als Prototyp umgesetzt.

Änderungen der geplanten Routingarchitektur

Ursprünglich war ein Backend geplant, das aus zwei separaten Komponenten besteht:

1. Zur Berechnung von Routen auf statischen OSM-Daten sollte eine bestehende Engine ausgewählt und angepasst werden (hinzufügen von parametrisierbaren Rollstuhl-Profilen).

2. Die Berechnung von Routen auf dem Kombination von Ergebnissen der ersten Komponente (statische OSM-Daten) und zusätzlichen dynamischen Daten (z.B. Informationen ob Fahrstühle aktuell funktionieren) sollte per Eigenentwicklung gelöst werden.

Grund für diese Architekturüberlegung war es, dass die unterschiedlichen Herausforderungen, bei den beiden zu lösenden Teilaufgaben tendenziell verschiedenen hätten gelöst werden müssen.

Die Laufzeitkomplexität von Routing auf statischen Daten kann um einen spürbaren linearen Faktor gesenkt werden, wenn die Wegkosten aller Streckenabschnitte einmalig vorberechnet wird. Auf Performance optimierte Engines verwenden dafür effiziente low-level Datenstrukturen.

Auf dynamischen Daten ist diese Optimierung nicht möglich.

Die Erkenntnis, das für Routing innerhalb einer Stadt die Anzahl der Kanten (Wege) klein genug sind, um auch ohne die Optimierung hinreichend schnell zu sein, hat uns umdenken lassen.

Wir führen das Routing auf den statischen OSM-Daten und der zusätzlichen dynamischen Daten gemeinsam in der von uns entwickelten Engine durch. Dies spart nicht nur eine große Abhängigkeit, sondern macht die Lösung auch deutlich

Plattformunabhängiger. Da unsere Engine auch nach Javascript compiliert, ist die Anwendung vollständig im Browser lauffähig.

In allen innerstädtischen Tests wurden die Routen in unter einer Sekunde berechnet.

Wenn nötig (Routen über größere Entfernungen), kann auch bei der jetzigen Architektur hierarchisches Routing verwendet werden.

Entwicklung nativer Apps

Ursprünglich war eine Webanwendung vorgesehen, die später auch als Progressive Web App verwendet werden kann.

Dank der Expertiese eines Teammitgliedes, wurde dann aber nicht nur eine Mobile First Strategy angewendet, sondern gleich Hybride Web Apps per Cordova entwickelt.

Nutzung von Machine Learning Methoden

Da unsere AI-Expertin aufgrund eines kurz vor Beginn des Förderungszeitraums begonnenen neuen Jobs keine Kapazitäten für uns zu entwickeln hatte, wurde sie ausschließlich beratend tätig. Die experimentelle Nutzung von Machine Learning Methoden zur Ermittlung von Parametern des Routingprofils wurde daher verschoben und zunächst durch einfachere Heuristiken ersetzt.

Statt dessen wurden mehr Ressourcen in das Tooling des Frameworks, insbesondere die Entwicklung einer Build- und CI-Pipeline investiert.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Die Graphhopper Routing Engine wurde in der Zwischenzeit um Funktionen erweitert, die uns neue Möglichkeiten bieten.

So werden seit September 2019 (v0.13) Vektor-Tiles ausgeliefert. Diese können von uns als eine Datenquelle verwendet werden.

https://www.graphhopper.com/blog/2019/09/18/graphhopper-routing-engine-0-13-released/

Im Austausch mit der OSM-Community wurden wir auf das Projekt StreetComplete aufmerksam gemacht. Über von uns definierte Regeln kann damit eine bestehende Community gebeten werden, Datensätze zu vervollständigen, die für besseres Rollstuhlrouting hilfreich sind.

https://github.com/westnordost/StreetComplete/