ROZŠÍŘENÁ REALITA V MOBILECH PRO LOKÁLNÍ VIZUÁLNÍ NAVIGACI PRO ANDROID

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

DUŠAN BEZDĚK

BRNO 2013
ROZŠÍŘENÁ REALITA V MOBILECH PRO LOKÁLNÍ VIZUÁLNÍ NAVIGACI PRO ANDROID
AUGMENTED REALITY FOR MOBILE DEVICES FOR LOCAL VISUAL NAVIGATION FOR ANDROID

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
DUŠAN BEZDĚK

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. VÍTĚZSLAV BERAN, Ph.D.

BRNO 2013
**Abstrakt**

Tato bakalářská práce se zabývá návrhem a implementací aplikace, která využívá technik rozšířené reality a slouží jako navigace pro chytré telefony s operačním systémem Android. Navigace je realizována pomocí rozšířené reality, kdy je do obrazu reálného světa vykreslena cesta směřující k požadované adrese. Tento program využívá pro získání mapových dat rozhraní Google directions API a informaci o poloze zařízení poskytuji GPS souřadnice. Dokument seznamuje čtenáře s detailním návrhem a realizací aplikace a popisuje vybrané problémy při řešení implementace. Závěr je věnován procesu testování a shrnutí výsledků práce.

**Abstract**

This bachelor’s thesis deals with a proposal and implementation of application, that uses augmented reality and serves as a navigation for smartphones with the operating system Android. The navigation uses augmented reality for viewing virtual path into the canvas of the real world. This path is heading to the target destination. Map datas are provided by Google directions API and the location of device is fixed by GPS. This thesis describes a proposal and the realization of the application and writes about the solution of important problems in its implementation. The end of the thesis mentions the testing process and the final chapter evaluates the final results.

**Klíčová slova**

Rozšířená realita, Android, chytrý telefon, Google directions API, Google maps API, OpenGL ES, orientační sensory, kompas, GPS.

**Keywords**

Augmented reality, Android, smartphone, Google directions API, Google maps API, OpenGL ES, orientation sensors, compass, GPS.

**Citace**

Bezděk Dušan: Rozšířená realita v mobilech pro lokální vizuální navigaci pro Android, bakalářská práce, Brno, FIT VUT v Brně, 2013
Rozšířená realita v mobilech pro lokální vizuální navigaci pro Android

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vítězslava Berana, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dušan Bezděk
15. května 2013

Poděkování

Chtěl bych poděkovat vedoucímu této práce panu Ing. Vítězslavu Beranovi, Ph.D. za vedení práce, odbornou pomoc a poskytnutí spousty cenných rad. Děkuji také svým rodičům za podporu během studia a své přítelkyni za velkou dávku trpělivosti.
# Obsah

<table>
<thead>
<tr>
<th>Kapitola</th>
<th>Název</th>
<th>Stránka</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Úvod</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Teorie</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>Android</td>
<td>3</td>
</tr>
<tr>
<td>2.2</td>
<td>Rozšířená realita</td>
<td>5</td>
</tr>
<tr>
<td>2.3</td>
<td>AR aplikace pro Android</td>
<td>7</td>
</tr>
<tr>
<td>2.4</td>
<td>3D projekce a OpenGL ES</td>
<td>9</td>
</tr>
<tr>
<td>2.5</td>
<td>Google maps Android API</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>Návrh řešení</td>
<td>13</td>
</tr>
<tr>
<td>3.1</td>
<td>Rozbor cíle</td>
<td>13</td>
</tr>
<tr>
<td>3.2</td>
<td>Rozdělení na části</td>
<td>14</td>
</tr>
<tr>
<td>3.3</td>
<td>Uživatelské rozhraní</td>
<td>17</td>
</tr>
<tr>
<td>4</td>
<td>Implementace</td>
<td>19</td>
</tr>
<tr>
<td>4.1</td>
<td>Struktura aplikace</td>
<td>19</td>
</tr>
<tr>
<td>4.2</td>
<td>Implementace bloků</td>
<td>20</td>
</tr>
<tr>
<td>4.3</td>
<td>Získání mapových dat</td>
<td>22</td>
</tr>
<tr>
<td>4.4</td>
<td>Výpočet geolokace</td>
<td>23</td>
</tr>
<tr>
<td>4.5</td>
<td>Orientace zařízení</td>
<td>24</td>
</tr>
<tr>
<td>4.6</td>
<td>Vykreslení cesty</td>
<td>25</td>
</tr>
<tr>
<td>4.7</td>
<td>Vývoj aplikace</td>
<td>28</td>
</tr>
<tr>
<td>4.8</td>
<td>Testování</td>
<td>29</td>
</tr>
<tr>
<td>5</td>
<td>Závěr</td>
<td>32</td>
</tr>
</tbody>
</table>
1 Úvod

Mezi neodmyslitelné fenomény dnešní doby patří bezpochyby mobilní telefony. Tyto zařízení se díky své funkčnosti a velikosti staly našimi každodenními pomocníky, bez kterých bychom se jen stěží obešli. Dnes už si člověk pod pojmem mobilní telefon nepředstaví pouze volání a zasílání zpráv, ale také možnost připojení k internetu, focení, přehrávání hudby a spoustu různých aplikací, které usnadňují běžný život. Tyto vlastnosti jsou výsadou především chytrých telefonů, které disponují operačním systémem umožňujícím vytvářet široké spektrum aplikací a funkcí. Jednou ze zajímavých částí tohoto spektra může být i oblast aplikací, věnující se rozšířené realitě.


Cílem této bakalářské práce je vytvořit aplikaci pro mobilní zařízení s operačním systémem Android, která využívá rozšířenou realitu pro navigaci ve městě. Tato aplikace vykresluje do scény snímaného kamerou navigační informace, tedy směr cesty k dosažení cílové destinace. Pro dosažení této funkčnosti musí aplikace umět propojit obraz z kamery a obraz mapy okolí, přičemž pro orientaci mobilního zařízení je využito jeho geolokačních informací. Informace o poloze získá aplikace pomocí GPS souřadnic a mapová data poskytuje rozhraní Google maps. Navigace je určena do měst a má za úkol zobrazit cestu chodcům.

Tato práce obsahuje dále kapitoly teorie, návrhu, implementace a závěru. Čtenář se v tomto dokumentu může obeznámit s potřebnou teorií k vytvoření této aplikace. Teorie se opírá o fakta spojená s operačním systémem Android, vysvětlení pojmu rozšířené reality, prvky typické pro aplikace rozšířené reality a stručné shrnutí OpenGL ES a mapového rozhraní Google maps. Poté dokument přechází do části návrhu řešení, který popisuje detailní specifikaci a formulaci cíle, rozdělení řešení na podčásti a návrh uživatelského rozhraní. O tuto kapitolu se poté opírá sekce implementace, ve které dochází k realizaci programu a popsání řešení stěžejních problémů vedoucí ke správné funkčnosti aplikace. Součástí vývojového cyklu každé aplikace je také proces testování, který tuto sekci uzávírá. Závěrečnou kapitolu tvoří shrnutí výsledků práce, její hodnocení a možnosti rozšíření práce do budoucna.
2 Teorie


2.1 Android

Android je rozsáhlá open source platforma [1], která vznikla především pro mobilní zařízení (chytré telefony, PDA, tablety, navigace, atp.). Zahrnuje v sobě operační systém, který je založen na jádru Linuxu, middleware, uživatelské rozhraní a aplikace. Je vyvíjen konsorciem Open Handset Alliance, jehož cílem je progresivní rozvoj mobilních technologií, které mají nižší náklady na vývoj a distribuci, a zároveň přináší inovativní uživatelsky přívětivé prostředí. Při vývoji jsou brány v úvahu omezení, kterými disponují klasické mobilní zařízení jako např. výdrž baterie, menší výkonnost a málo dostupné paměti.


1 Viz http://developer.android.com/index.html
CPU. Vrstva Application framework poskytuje různé funkce operačního systému pro vývojáře, kteří jej využívají v jejich aplikacích. Ve vrstvě applications naleznete aplikace pro Android zařízení a aplikace, které můžeme stáhnout a nainstalovat z Android marketu. Jakákoliv aplikace vytvořená vývojářem je umístěna zde.

Samotná platforma Android dává k dispozici nejen operační systém s uživatelským prostředím pro uživatele, ale i kompletní řešení nasazení operačního systému pro mobilní operátory a výrobce zařízení a pro vývojáře aplikací poskytuje efektivní nástroje pro jejich vývoj – Software Development Kit (SDK)².


Aplikace vyvíjené pro prostředí Android má své charakteristické rysy [3]. Těmito rysy jsou myšleny stavební bloky každé aplikace. Každá komponenta disponuje určitými vlastnostmi a ne každá komponenta může být použita jako vstupní bod programu, některé jsou závislé na jiných a každá má svou specifickou roli. Existují čtyři různé typy těchto komponent. Každá část slouží k odlisným účelům a má odlisný životní cyklus, který definuje jak je komponenta vytvářena a jak je komponenta zničena. Těmito komponenty jsou:

³ Viz http://www.eclipse.org/
Aktivity (activities)

Služby (services)
Služba je komponenta, která běží v pozadí a vykonává délko trvající operace pro vzdálené procesy. Tato služba neposkytuje uživatelské rozhraní. Příkladem této komponenty může být hrani muziky na pozadí, přičemž uživatel má spuštěnou jinou aplikaci. Obvykle je spuštěna aktivitou, která se připojí a komunikuje s ní. Služba je implementována děděním z třídy Service.

Poskytovatel obsahu (Content provider)
Tato komponenta se stará o sdílená data. Tato data mohou být uložena v souborovém systému, v databázi, na webu nebo na jakémkoliv dosažitelném úložišti. Ostatní aplikace se mohou dotazovat na tyto data, případně je upravovat. Komponentu je možné také použít pro data, která jsou k dispozici jen dané aplikaci a nejsou sdílená. Poskytovatel obsahu je realizován děděním od třídy ContentProvider a musí implementovat určité metody k provádění transakcí.

Přijímač vysílání (Broadcast receiver)


2.2 Rozšířená realita
Rozšířená realita (AR – augmented reality) je technologie [4], která umožňuje vidět a komunikovat s virtuálními objekty obsaženými v reálném světě. To umožňuje uživateli vnímat svět spojením reálného obrazu světa s virtuálními objekty. V souvislosti s aplikací pro mobilní zařízení to vyznačuje především překrytí, doplnění či zobrazení dalších informací do obrazu reálného světa. Vedle pojmu augmented reality se objevuje také označení virtual reality (viz obr. 2.2), není to však totéž co augmented reality, protože virtual reality nepoužívá obraz reality snímaný kamerou, ale je používán.

v animacích, v již nahraných videích, filmech anebo nahrazuje reálný svět světem simulačním. Oba tyto pojmy lze stotožnit pojmem mixed reality, jenž obsahuje oba tyto systémy.

Hlavní rysy rozšířené reality:
- kombinace reálného a skutečného světa
- interaktivita v reálném čase
- registrace ve 3D prostoru

![Reality-Virtuality (RV) Continuum](image)

Obrázek 2.2: Reality virtuality continua [5].


![Příklad využití rozšířené reality](image)

Obrázek 2.3: Příklad využití rozšířené reality [6].

V současnosti existuje spousta aplikací používající rozšířenou realitu, např. hry, prohlížeče, navigační aplikace atd. (viz obr. 2.3). Tyto aplikace obvykle potřebují ke své činnosti akcelerometr a GPS pro zjištění lokality a natočení zařízení. Kromě těchto aplikací se nalézá rozšířená realita i v jiných oblastech. V medicíně poskytuje chirurgovi skryté informace, jako např. tep srdce, krevní
tlak, může být také využita pro prozkoumání pacienta zevnitř v kombinaci s rentgenem, což pomůže lékaři zjistit problém. V armádě při poskytování dat o bojišti, objekty mohou být označeny indikátory pro potenčionální nebezpečí, virtuální mapy atp. Pro navigaci, informace mohou být zobrazeny na čelním skle automobilu informující o směru k cíli, terénu, počasí, dopravní informace. Může být také použita pro překlad, kdy se snímá text a výsledkem je přeložený text do požadovaného jazyka atd. Využití pro rozšířenou realitu je opravdu mnoho.

Existuje více typů aplikací rozšířené reality. Zde je uvedeno dělení podle rozpoznání objektů v obraze [7]:

**Systémy s technologií klient-server**
Zázemím těchto aplikací je užítí vzdáleného rozpoznávání a lokálního sledování polohy. Rozpoznávání je prováděno formou visual tracking (rozpoznávání podle snímaného obrazu), pro který existují různé algoritmy. Rozpoznání obrazu je prováděno na klientovi, přičemž serverová strana poskytuje repositář pro lokálně přiřazené obrazy. Jedná se především o programy, které se snaží určit určit lokaci zařízení bez geolokačních informací (GPS), nebo je využívá, ale přesnou polohu určí pomocí rozpoznání obrazu.

**Technologie rozpoznání obrazu**

**Rozpoznání podle lokace zařízení**
V současnosti nejvíce rozšířený typ AR mobilních aplikací, je čistě založen na GPS a orientačních senorech. Každý bod zájmu je spojen v databázi s geografickými informacemi. Proces rozpoznání se jednoduše provádí zjištěním, kde jsou jednotlivé prvky databáze situovány a kterým směrem směřuje kamera zařízení. Vypočítá se projekce těchto prvků na displeji podle aktuální pozice a orientace zařízení.

### 2.3 AR aplikace pro Android
**Kamera**
Android framework obsahuje podporu pro různé druhy kamer a jejich vlastnosti dostupných na zařízeních, umožňující snímání obrázků a videí v aplikaci. Aplikace rozšířené reality nemá smysl vytvářet pro zařízení, která nedisponují kamерou.

**Orientační sensory**
Většina zařízení obsahující operační systém Android mají zabudované sensory, které měří pohyb, orientaci a různé přírodní podmínky. Tyto sensory jsou schopny poskytnout nezpracovaná data s velkou přesností a jsou užitečné při pohybu nebo polohování zařízení. Android podporuje tři kategorie sensorů:
- Pohybové sensory – tyto sensory měří pohybové síly a rotační síly pomocí tří os. Tato kategorie obsahuje akcelerometry, gravitační sensory, gyroskopy a rotační vektorové sensory.
- Sensory prostředí – tyto sensory poskytují údaje o různých okolních parametrech, jako teplota vzduchu a tlak, osvětlení, vlhkost. Do této kategorie spadají barometry a teploměry.
- Poziční sensory – sensory, které učují informace o fyzické pozici zařízení. Sem patří orientační sensory a magnetometry.

Orientační sensory jsou kombinací sensorů magnetického pole a akcelerometrického sensoru. S daty z těchto dvou sensorů a s trochou trigonometrie je možné získat náklon, otočení a směr (azimut) zařízení. Veškerá trigonometrie je v Androidu počítána automaticky. Jednotlivé osy orientačních sensorů jsou znázorněny na obrázku (viz obr. 2.4).
Akcelerometr

Akcelerometr měří zrychlení zařízení pomocí tří směrových os (viz obr. 2.5).

Osa $X$ – v normálním zařízení s běžným akcelerometrem zobrazuje osa boční zrychlení. To znamená zleva doprava a zprava doleva. Hodnota je pozitivní, pokud se pohybuje se zařízením směrem doprava a negativní pokud naopak. Např. zařízení má obrazovku směrem vzhůru a je s ním pohybováno směrem doprava, osa vygeneruje kladnou hodnotu.

Osa $Y$ – osa y zobrazuje stejný pohyb jako osa $x$, ale měří délku posunu. Positivní hodnota je naměřena, pokud je zařízení drženo ve stejné poloze, jako při pohybu osou $x$, ale je v pohybu směrem jeho vrcholu a negativní při pohybu opačným směrem.

Osa $Z$ – tato osa znázorňuje vzestupný a sestupný pohyb, přičemž pozitivní hodnoty jsou pro pohyb směrem vzhůru a opačné směrem dolů. Pokud je zařízení v klidu, čtená hodnota bude $-9.8 \text{ m/s}^2$ kvůli působení gravitace. Ve výpočtech se tento fakt nesmí opomenout.

Global positioning system (GPS)

Global position system (GPS) je systém určení polohy, který může poskytnout přesnou polohu prostřednictvím satelitů. GPS je satelitní navigační systém, který je dostupný pro jakékoli zařízení s GPS přijímačem, přestože bude používán pouze pro vojenské účely. Přijímač musí komunikovat s nejméně čtyřmi satelity, které mu posílají informace o aktuální poloze s použitím speciálních algoritmů. Výsledná poloha je však kvůli zpoždění komunikace neúplně přesná. Informace o poloze obsahují souřadnice zeměpisné délky, zeměpisné šířky a nadmořské výšky.

### 2.4 3D projekce a OpenGL ES

3D projekce je způsob mapování trojrozměrného světa na dvojrozměrnou obrazovku [9]. Tato metoda je dnes používána pro všechna média, která zobrazují do 2D obrazu reálný svět. Existují dva základní typy projekcí, a to projekce paralelní a projekce perspektivní (viz obr. 2.6 vlevo). Paralelní projekce ignoruje efekt, při kterém se vzdálené předměty zobrazí menší než předměty, které jsou bližší. Tato projekce neodpovídá zobrazení světa pomocí lidského oka a obvykle se používá při zobrazení profilu, detailů nebo přesnosti trojrozměrných objektů. Popisovaná práce tedy využívá projekci perspektivní, která simuluje zobrazení světa pro člověka. Tato projekce je však náročnější na výpočet a neobejde se bez matice pohledu kamery a transformační matice.

Na obrázku 2.6 je zobrazena transformační matice objektu. Jak z něj jeví, matice má tři nezávislé místa pro posun (oranžové X,Y,Z) a ostatní operace jsou spojeny do červené oblasti. Každá rotace (žluté X,Y,Z) ovlivňuje 4 pole a každé měřítko (fialové X,Y,Z) zastupuje pole jediné.

Obrázek 2.6: Paralelní a perspektivní projekce (vlevo)[10] a transformační matice (vpravo)[10].

Poslední zmíněný proces, a to zpracování transformační matice s maticí pohledu kamery a následné vytvoření 3D projekce, je prováděn knihovnou OpenGL ES [12]. Knihovna obsahuje funkce pro tyto operace a vytváří tím základ pro kvalitní zpracování grafické stránky aplikace. Android poskytuje také řadu standardních nástrojů pro vytváření atraktivního funkčního grafického uživatelského rozhraní, ale pro kontrolu, co vykresluje aplikace na obrazovku nebo pro proniknutí do trojrozměrné grafiky, je nutné použít tento nástroj. OpenGL ES API, který poskytuje Android framework nabízí soubor nástrojů pro vykreslování a zobrazování animované grafiky. Obecně může OpenGL ES sloužit pro 2D a 3D grafiku vestavěných systémů včetně konzoli, chytrých telefonů, různých zařízení a automobilů. Je to odlehčená verze knihovny OpenGL, nejrozšířenější platformy grafického API a poskytuje:
- Průmyslový standard – kdokoliv může stahovat a vytvořit aplikaci využívající OpenGL ES. S širokou vývojářskou podporou je to multiplatformní vestavěný grafický standard. Tento standard umožňuje vývojářům se více soustředit na základní problém a méně na detaily platformy.
- Malá spotřeba – OpenGL ES je navržen pro přizpůsobení se různým nárokům na aplikaci s minimálním zatížením systému a minimem uložených dat.
- Souvislý přechod ze softwarového do hardwarového renderování – ačkoliv OpenGL ES specifikace definuje konkrétní grafické řetězové zpracování, samostatné volání funkcí může být provedeno na speciálním hardwaru, běžící jako softwarová rutina na CPU, nebo může být implementováno oběma těmito způsoby společně. Tato vlastnost umožňuje aplikacím souvislý přechod k použití OpenGL ES hardware akceleraci v zařízeních s vyšším napájením.
- Růst a vývoj – umožňuje nové hardwarové inovace dostupné skrze API prostřednictvím OpenGL rozšiřujících mechanismů.
- Jednoduchý k použití – založen na OpenGL, který je strukturován intuitivním návrhem a logickými příkazy.
- Dobrá dokumentace – existuje spousta dokumentů, knih a ukázkových příkladů.

2.5 Google maps Android API


![Obrázek 2.7: Klasické zobrazení Google map](image)

11
Pokud je na mapu zaostřeno, umí zachytit stisky kláves a dotykové gesta k přejižďení a přibližování map, včetně řízení dotazů na přidané části map. Aplikace mohou také mapy překreslovat. Google API doplňky poskytují knihovnu map k vývoji, sestavení a k provozu aplikací zaměřených na zobrazování map v Android SDK s přístupem do Google maps dat. V současnosti je vydána nová verze v2, která nabízí rozšíření první verze. Mapy užívají vektorové zobrazení, tedy reprezentace dat je méně náročná a mapy se zobrazují rychleji, je vylepšeno cachování, které zamezí zobrazení bílých částí map. Mapy jsou navíc 3D a pohybem výhledu uživatele se může mapa zobrazit v perspektivě.


⁶ Viz http://developers.google.com/maps/documentation/directions/
3 Návrh řešení

V této práci je vytvářena aplikace pro mobilní telefony, které pracují s operačním systémem Android a pro lokální navigaci využívají dat poskytnutých službou Google maps. Aby aplikace v tomto zařízení fungovala, musí mobilní telefon obsahovat kameru pro snímání reálného světa, obrazovku pro vykreslení výsledné scény spojením reálného a virtuálního světa, orientační senzory, akcelerometr a GPS přijímač. Je také nutné zajistit připojení k síti pro získání dat ze serveru (v tomto případě stažení mapových dat), popř. je mít již připravené v paměti. Dnes těmito vlastnostmi disponují obvykle všechny chytré telefony.

V této kapitole se čtenář dočte o detailním rozboru zadání práce, o formulaci cíle, co má program umět, k čemu je určen a pro koho je určen. Je zde nastíněna základní struktura aplikací sloužících k zobrazení navigace v rozšířené realitě a její přizpůsobení k vytvoření této práce. Kapitola dále vysvětluje návrh toku dat v aplikaci a návrh uživatelského rozhraní. Na základě tohoto návrhu je provedena implementace aplikace.

3.1 Rozbor cíle

Pro detailní návrh řešení této práce je nejprve důležité se seznámit s jejím cílem a zadáním. Cílem projektu je vytvořit aplikaci rozšířené reality, která do obrazu snímaného kamерou vykreslí cestu směřující k cílové destinaci (adrese). Uživatel zadá současnou adresu a adresu cíle, ke které se chce dostat, a aplikace zobrazí do kamery trasu, která směřuje k zadanému cíli (viz obr. 3.1). Aby aplikace dosáhla této funkčnosti, musí umět propojit obraz z kamery a zobrazení cesty do jednoho snímku. Pro správný směr a orientaci cesty jsou použita data z orientačních sensorů a akcelerometru. Mapová data poskytují Google maps a informace o poloze zařízení získá aplikace pomocí GPS souřadnic. Celá tato práce je určena pro zařízení s operačním systémem Android.

Obrázek 3.1: Cíl práce. Uživatel se chce přemístit na adresu (vlevo), zadá cílovou destinaci a mapový server poskytne data (uprostřed). Aplikace vykreslí cestu do kamery (vpravo).

Kromě vykreslení cesty do obrazu kamery poskytuje aplikace uživateli také informace o délce zadané cesty, vzdálenosti k cíli, již uražené vzdálenosti a celkového času potřebného pro zvládnutí trasy při pěší chůzi. Pro lepší orientaci je startovní a cílový bod cesty zvýrazněn barevnými šípkami. Aplikace umožňuje uživateli také přepnout do režimu klasického zobrazení mapy, které známe z prostředí Google maps. Uživatel je také schopen měnit režimy zobrazení cesty do kamery. Tato aplikace je určena především pro orientaci ve městech, nicméně může být použita i pro navigaci do auta. Aplikace nemá za úkol přesně reflektovat cestu v obrazu reálného světa, jejím cílem je
poskytnout přibližnou navigaci o směru cesty. Aplikace se sice snaží o co nejpřesnější vykreslení a věrohodnost vykreslení virtuální stezky do reálného světa, ale absolutní přesnost není možná kvůli nemožnosti získat přesnou lokalizaci a polohu zařízení. Dosažení této vlastnosti brání nepřesnost GPS navigace.

Aplikace má za úkol posloužit všem, kteří potřebují zobrazit cestu k cílové adrese. Častým důvodem k použití programu může být právě neznalost cílové adresy, kam se uživatel potřebuje přemístit. Využití programu se hodí také do situací, kdy se uživatel dostane do míst (města), kde to skutečně nezná a potřebuje pomoci. Stejně tak je aplikaci možné použít pouze pro snadnější orientaci, kterým směrem se dostanu do určité části města. S tímto problémem se může setkat široké spektrum uživatelů různého věku, jenž si osvojí práci se svým mobilním zařízením. Navíc je grafické rozhraní intuitivní a jednoduché, nemělo by tedy učinit uživatelům problém.

3.2 Rozdělení na části

Jádrem této aplikace jsou funkční bloky, bez kterých se žádná location based AR aplikace neobejde. Mezi tyto bloky se řadí části, které obslouží hardwarové zařízení telefonu. Jedná se o blok kamery, který poskytuje obraz snímaného reálného světa, blok pro získání GPS souřadnic a blok pro zpracování a poskytnutí dat o orientaci zařízení, který využívá orientačních a akcelerometrických sensorů. Tato část programu snímá a počítá směr, rotaci a naklonění zařízení. Pro získání mapových dat je nutné do aplikace vnořit také blok pro komunikaci s mapovým serverem, který poskytne aplikaci data o zadané cestě. Tato část programu je implementována jako internetový klient. Poslední nezbytnou součástí aplikace je vyhodnocovací blok (renderer), jenž na základě informací z akcelerometru, dat o zadané cestě a údajů o současné pozici vytvoří virtuální obraz. Tento virtuální obraz je spojen s reálným obrazem z kamery a vzniká tak obraz virtuální reality, v tomto případě obraz cesty vykreslený do kamery.

Obrázek 3.2: Tok dat aplikace rozšířené reality s geolokací a orientačními sensory.


### 3.2.1 Získání cesty

Tok dat je zobrazen na obrázku 3.2. Z tohoto schématu lze vyčíst, že tok dat začíná u zadání cesty. Uživatel zadá koncovou a počáteční adresu virtuální cesty, přičemž bude mít na výběr. Buď se rozhodne pro zadání počáteční adresy, nebo zadá jako počáteční adresu současnou pozici. Při této volbě je nutné poskytnout souřadnice okamžité pozice z bloku GPS. Tento blok bude nutné implementovat s grafickým uživatelským rozhraním, které bude požadovat po uživateli patřičná data.

Tok dat se přesouvá do bloku, který má za úkol získat ze serveru mapová data o zadané cestě (na obrázku blok klient). Před zasláním žádosti o tato data je nutné adresy patřičně upravit. Protože tato aplikace bude využívat služeb Google directions API, musí si klient vytvořit http žádost, která odpovídá parametřům této služby. Navíc je nutné transformovat adresy na GPS souřadnice a zvolit režim a dodatečná nastavení pro poskytnutí odpovědi. Server zasílá odpověď, která je uložena ve formátu JSON nebo XML, proto musí tento blok obsahovat také část, která tuto odpověď analyzuje a vypíše z ní potřebné údaje. Pro tuto aplikaci je nezbytné získat z odpovědi GPS souřadnice úseků cesty, informace o délce cesty a dobu trvání cesty. Po uložení těchto údajů o cestě se tok dat může přesunout do bloku, který se stará o vykreslování virtuální cesty k cíli.

Pro každou nově vytvořenou cestu se data ze serveru stahují pouze jednou. Proto je nutné vytvořit objekt, který je vytvořen a inicializován po zpracování odpovědi ze serveru a obsahuje data o jedné konkrétní cestě, které se v průběhu jejího vykreslování nemění. Pokud dojde k zadání nové cesty, je poslán dotaz na server a objekt obsahující původní cestu je nahrazen objektem nesoucím informace o nové cestě. Třída objektu musí navíc obsahovat metody, které slouží k vykreslování tras (např. výpočet vzdáleností mezi jednotlivými úseky cesty, směr úseků, nalezení nejbližší souřadnice cesty k současné poloze atd.). Při samotném vykreslování již bloky pro vytvoření cesty a komunikaci se serverem v činnosti nejsou.

### 3.2.2 Orientace a určení pozice

Ještě před tím, než je možné trasu vykreslit, musí mít vyhodnocovací blok (na obrázku renderovací blok) k dispozici také data o současné pozici zařízení a o natočení, naklonění a směru kamery zařízení. Aplikace musí reagovat na změnu orientace a změnu lokace kamery zařízení. Proto jsou následující bloky schématu, na rozdíl od části získání cesty, v neustálé činnosti při vykreslování cesty. Informace o okamžité zeměpisné pozici zařízení poskytuje blok GPS. Tento blok má za úkol informovat o každé změně pozice posláním údajů o zeměpisné délce a zeměpisně šířce vyhodnocovacímu bloku. Stejnou funkci plní blok orientačních sensorů s tím rozdílem, že poskytuje při každé změně orientace přístroje data z akcelerometru a senzoru magnetického pole. Tyto data musí projít filtrem, který vyhlašuje zašuměné hodnoty.

Nyní má renderovací blok k dispozici data, která vypovídají o orientaci zařízení, stále však nejsou vhodné pro zobrazení virtuálního objektu. Z těchto dat se musí vypočítat transformační matice a musí být přepočítána do vhodného souřadného systému. O tuto operaci se postarájí funkce z Android aplikačního frameworku. Rotační matice je posléze použita pro nastavení zorného úhlu kamery. Tímto způsobem se uživatel rozhodne ve světě, do kterého je umístěna cesta k cíli, odpovídající umístění v reálném světě. Po polohu cesty je počítána z aktuálních zeměpisných souřadnic vzhledem k umístění cesty v reálném světě. K vykreslování virtuální cesty je využita knihovna.
OpenGL ES. Z toho vyplývá, že tento blok musí také implementovat náležitosti související s vykreslováním objektů v této knihovně, a to obrazovku, do které se vykresluje, a renderer, který do této obrazovky vykresluje objekty.

3.2.3 Kamera a spojení vrstev

Pro dosažení rozšířené reality je nutné získat obraz kamery a umístit jej na pozadí výsledného obrazu. K poskytnutí obrazu z kamery je určen blok kamery. Tento blok je navázán na hardwarovou kameru v zařízení, která snímá reálný svět v reálném čase a poskytuje bloku tyto snímky. Tato část struktury aplikace se snímky nepracuje, avšak přeposílá je dále do bloku, který spoji obraz kamery a virtuálního objektu.


3.2.4 Rozdělení podle grafického rozhraní

3.3 Uživatelské rozhraní

Jak již bylo zmíněno, aplikaci lze rozdělit na části, z nichž každá disponuje odlišným grafickým uživatelským rozhraním. Uživatelské rozhraní každé části je přizpůsobeno účelu daného oddílu a jeho návrhu tomuto účelu odpovídá. Při návrhu tohoto uživatelského rozhraní byla snaha o vytvoření jednoduchého a intuitivního ovládání.

Obrázek 3.3: Režim kamery s navigačními tlačítky a informačními ikonami (po stisku tlačítka menu je zobrazena navigační lišta).

Po spuštění aplikace je uživateli zobrazen obraz z kamery společně se dvěmi tlačítky (viz obr. 3.3). Stiskem tlačítka info se uživateli ukáže informace o zadané cestě v dialogovém okně. Po spuštění tlačítka menu se zobrazí navigační lišta aplikace. V tomto menu je možné se odkázat na vytvoření cesty, zobrazení cesty v klasickém režimu Google maps, nastavení zobrazení cesty, informace o programu a ukončení programu. Kromě popsáných tlačítek se zde vyskytují navíc ikony, a to ukazatel příjmu GPS signálu a ukazatel připojení zařízení k internetové síti.

Režim vytvoření cesty poskytuje uživateli formulář, do kterého vloží informace o zobrazované trase (viz obr. 3.4 vlevo). Je nutné vyplnit cílovou adresu a počáteční adresu. V případě, že chce uživatel použít jako startovní bod současnou pozici, zaškrtně volbu současné pozice. Navíc je k dispozici panel, který informuje o barvě vykreslené cesty. Po stisknutí panelu se zobrazí v samostatné liště barvy k výběru. Kromě těchto údajů je možné rozkliknout podrobnější nastavení, v němž se zadá režim zobrazení cesty, profil vykreslované cesty, jas obrazu a průhlednost vykreslené cesty. Tento režim nastavení je doplněn o navigační tlačítka, které umožňují cestu vytvořit, jít zpět a rozkliknout podrobnější nastavení. Pro lepší informaci o tom, zda je možné vytvořit cestu ze současné pozice zařízení a zda je mobil připojen k internetu, slouží ikony v rohu obrazovky.

Mód nastavení je podobný módu vytvoření cesty (viz obr. 3.4 vpravo). Lze jej však spustit pouze pokud je cesta již vytvořena a slouží k vizuálnímu upravení vykreslované cesty. Nabídka tohoto nastavení je totožná s nabídkou pokročilého nastavení v režimu vytvoření cesty, jediným rozdílem je tlačítko smazat cestu, díky kterému se přestane vykreslovat cesta, a také nepřítomnost ikon v rohu zobrazení režimu, které při upravení nastavení nejsou zapotřebí.
Dalším možným režimem je volba zobrazení cesty v prostředí Google maps (viz obr. 3.5). Po spuštění tohoto módu je uživateli zobrazena klasická 2D mapa okolí, v němž je vyznačena současná lokace zařízení a cesta směřující k cíli. Počáteční a koncový bod trasy je navíc speciálně označen.

V tomto režimu má uživatel možnost sledovat a posouvat mapu okolí. Tato mapa je obohacena o tlačítko současné lokace, která soustředí mapu do aktuální pozice a tlačítko zobrazení mapy, které mění měřítko vykreslené mapy.

Režim informace o programu zobrazí dialogové okno, ve kterém jsou vepsány veškeré informace o programu. Tyto informace doplňuje návod k navigaci v aplikaci a popis použití aplikace.

Posledním a nejdůležitějším režimem, který jsem už částečně zmínil, je režim kamery. Tento režim je zobrazen již po spuštění aplikace, čeká však na to, až uživatel vytvoří cestu. Právě v tomto režimu se zobrazí cesta odpovídající umístění v reálném světě společně s obrazem z kamery, což vytváří obraz rozšířené reality. V popředí popsaného obrazu jsou umístěna dvě tlačítka, tlačítko menu a tlačítko info, která jsou již popsána v předchozím odstavci. Tato dvě tlačítka doplňují ikony, které signalizují připojení k internetu a mapování GPS pozice.
4 Implementace


4.1 Struktura aplikace

Protože je aplikace určena pro operační systém Android, musí program obsahovat komponenty, které jsou pro tuto platformu charakteristické. Komponentami jsou myšleny aktivita, služba, poskytovatel obsahu nebo zprávy komunikující mezi těmito komponenty, nazývající se intenty. Struktura aplikace se opírá o návrh rozvržení aktivit, které jsou základními komponenty pro tvorbu aplikace. Každá aktivita je reprezentována samostatnou obrazovkou s uživatelským rozhraním. Z tohoto pohledu lze rozdělit aplikaci podle návrhu části s odlišným grafickým uživatelským rozhraním, který byl lehce nastíněn v kapitole návrhu.

Aplikace je sestavena ze čtyř aktivit, které tvoří základ aplikace (viz obr. 4.1). Vstupním a výstupním bodem programu je aktivita MainActivity. Tato aktivita inicializuje počáteční stav programu a vytváří další instance tříd, potřebné k běhu aplikace. Jsou zde postupně vytvořeny instance sensorů, rendereru, obrazovky a nastavení pro správné překrytí a zobrazení vrstev obrazu. Obsahuje také metody obsluhující spuštění a ukončení běhu aplikace, kde jsou volány metody pro alokaci nebo uvolnění zdrojů služeb (sensory, kamera, vrstva OpenGL). Této aktivitě je přidružen grafický režim, který spojuje vrstvy obrazu z kamery, vykreslované cesty a navigační tlačítka. Při zobrazení rozšířené reality je spuštěna právě tato aktivita. Aktivita je také zodpovědná za spouštění ostatních aktivit aplikace. K tomuto účelu vytváří pro každou další aktivitu intent (zprávu), která aktivuje příslušný režim. Každý další aktivitě náleží návratový kód a při návratu do hlavní aktivity je přijat intent s tímto kódem, který identifikuje, ze kterého režimu se vrácí kontext a podle toho zpracovává data přidružená k této zprávě. Poslední věcí, kterou v této části najdeme, jsou dialogová okna, která se objevují při určitých událostech.

Pokud uživatel v navigačním menu stiskne volbu pro zadání cesty, je pomocí intentu spuštěna aktivita SetPath. Tato aktivita má za úkol zpracovat vyplněný formulář od uživatele a poskytnout jej.
hlavní aktivitě. Formulář obsahuje položky potřebné pro vytvoření virtuální cesty, grafický režim této aktivity tedy odpovídá režimu zadání cesty z kapitoly návrhu uživatelského rozhraní. Aktivita se také stará o ověření, zda zadané adresy cesty skutečně existují. Pokud ano, vyplní aktivita intent daty zadanými uživatelem a pošle jej s příslušným návratovým kódem hlavní aktivitě. Hlavní aktivita se poté postará o vytvoření zadané cesty.

Další aktivitou, kterou v aplikaci najdeme, je aktivita Settings. Aktivita je opět spuštěna z hlavní aktivity a slouží k upravení nastavení zobrazené cesty. Tato komponenta poskytuje uživatelské rozhraní režimu nastavení. Aktivita pouze zpracuje změny zadané uživatelem a pošle jej pomocí zprávy hlavní aktivitě a ta tyto změny provede. Kromě toho umožňuje komponenta také smazat zobrazenou cestu.

Poslední aktivitou programu je aktivita Maps. Tato aktivita má za úkol zobrazit uživateli mapu, kterou poskytuje rozhraní Google maps. Pro zobrazení cesty v této mapě je nutné, aby hlavní aktivity při spuštění komponenty map přidala intentu informace o požadované cestě. Aktivita map tuto zprávu poté zpracuje a zobrazí cestu do mapy. Při návratu z této aktivity opět dochází k zaslání zprávy hlavní aktivitě, tentokrát však intent neobsahuje žádná data a aktivita map je vypnuta.

Obrázek 4.1: Diagram aktivit aplikace.

4.2 Implementace bloků

Struktura aplikace se opírá o návrh řešení z předchozí kapitoly. Veškeré funkční bloky potřebné k fungování aplikace jsou reprezentovány třídami, které budou následně detailně popsány (viz obr. 4.2). Všechny bloky, které poskytují data z hardwarových částí zařízení, jsou implementovány jako komponenty služby. Stejně tak i propojení vrstev do výsledného obrazu může být implementováno jako služba, ale protože je tato činnost provedena v hlavní aktivitě, nelze ji připojit ke službě.

Základní a nezbytnou součástí každé aplikace rozšířené reality je obraz z kamery. Tento obraz je poskytován třídou CameraView. Tato třída je rozšiřuje třídu SurfaceView, která obsluhuje instanci obrazovky a podle potřeby tuto instanci vytváří, uvolňuje nebo upravuje rozměry obrazu podle velikosti obrazovky zařízení. Protože se do obrazovky musí vykreslit data z kamery, implementuje třída rozhraní SurfaceHolder.Callback, které je zodpovědné za předání obrazu z kamery zařízení do aplikace. Po získání dat z kamery je schopna třída vykreslit tato data právě pomocí již zmíněného dědění z třídy SurfaceView. Pro potřeby aplikace je nutné zajistit, aby při ukončení, případného odsunutí aplikace na pozadí, byla schopna aplikace uvolnit zdroj kamery. Tato funkčnost je implementována pomocí dvou metod, z nichž jedna je spouštěna vždy při startu aplikace.
a připojuje se ke zdroji kamery, a druhá, která je vždy spuštěna při ukončení (odstranění na pozadí) aplikace a tento zdroj kamery správně uvolní.

Další důležitou součástí aplikace jsou bloky, které obsluhují orientační a akcelerometrické sensory. Tento blok je implementován ve třídě `Sensors`, který sdílí také získání lokace zařízení pomocí GPS souřadnic. Pro práci s těmito senzory je nutné implementovat rozhraní `SensorEventListener`. Toto rozhraní umožňuje společně s vytvořením objektu `SensorManager` sledovat a poskytovat hodnoty vypovídající o naklonění, směru a natočení zařízení. Pro vyhlašení zašuměných dat je použit jednoduchý filtr. Tento filtr počítá nové hodnoty pomocí poslední předešlé hodnoty a přičtem k ní rozdíl nové a předešlé hodnoty vynásobené alfa konstantou (viz kapitola orientace zařízení). Pro příjem GPS signálu musí třída implementovat také rozhraní `LocationListener`, které ihned posílá informace o změně lokace, respektive o změně zeměpisné výšky a zeměpisné šířky. Stejně tak jako u třídy kamery je i zde potřeba uvolnit senzory při ukončení programu.

Pro uložení cesty, její získání a nastavování je vytvořena třída `Path`. Tato třída obsahuje všechny důležité informace o vykreslované cestě a těmito jsou především kolekce GPS souřadnic cesty a výpočty vzdálenosti a směru mezi jednotlivými souřadnicemi. Pro vytvoření instance této třídy je nutné zadat konstruktoru počáteční a koncovou adresu cesty. Objekt po vytvoření zavolá třídu `MapClient`, která je implementována jako klient, vytvoří dotaz url se zadanými adresami a zasílá požadavek na mapový server. Tento dotaz je obsluhován třídou `AsyncTask`, která poskytuje práci s vláknení. Po přijetí odpovědi ze serveru se zpracování dat začne a v druhém vlákně je vytvořen dialog indikující zpracovávání informací. Detailnějšímu způsobu získání mapových dat je určena následující kapitola. Po přijetí odpovědi ze serveru je odpověď zpracována a informace o získané cestě jsou vytvořeny a poté využity informace potřebné pro fungování aplikace. Těmito informacemi jsou především GPS souřadnice cesty, celková vzdálenost cesty a doba trvání cesty. Uvnitř této třídy jsou pak počítány vzdálenosti mezi nejbližšími souřadnicemi cesty a úhel, kterým směrem je tato cesta vedena. Tyto data jsou poté použity v renderování k, kde je pomocí nich vykresluje
cesta. Mimo to obsahuje metody, které slouží k optimalizovanému vykreslování cesty. Jsou to metody pro výpočet vzdálenosti od současné pozice k souřadnicí cesty, nalezení nejbližšího bodu cesty k současné pozici, popř. počítá vzdálenost, která zbývá k cíli. Veškeré tyto metody jsou použity rendererem pro rychlejší zobrazení cesty.

Rendrovacímu bloku odpovídá třída \textit{GLRenderer}. Tato třída obsahuje metody, které jsou nutné pro vykreslování objektů pomocí OpenGL ES. Mezi tyto metody patří metoda, která inicializuje počáteční stav a nastavuje volby pro vykreslování, metoda pro poskytnutí rozměrů plátňa (obrazovky) do něhož se vykresluje a konečně metoda pro malování objektů. Ještě před tím, než se pustí renderer do malování, musí si vytvořit instance grafických elementů. Základními prvky, které tvoří obraz cesty, jsou kruh a čtverec. Při vykreslování je použito také elementů šipky, která označuje startovní a cílový bod. Každý z těchto tří elementů má svou vlastní třídu (\textit{GLCircle}, \textit{GLSquare}, \textit{GLArrow}), které obsahují informace o vykreslení těchto objektů pomocí OpenGL ES. Samotný způsob vykreslení cesty je popsán v kapitole vykreslení cesty.

Spojení vrstev obrazu z kamery, obrazu z rendereru a navigačních tlačítek je provedeno v hlavní a vstupní části aplikace, tedy ve třídě \textit{MainActivity}, která je zároveň základní aktivitou. Jsou zde postupně vytvořeny instance sensorů, rendereru, kamery a nastavení pro správné překrytí a zobrazení vrstev obrazu. Nejprve je předána obrazovce vrstva rendereru, která obsahuje samotný render a vykreslovací plátno. Toto plátno je nastaveno jako průsvitné, aby byl pod ním vidět obraz z kamery. Až poté je přidána vrstva kamery, jež tvoří pozadí výsledného obrazu. Nakonec je vykreslena vrstva, která obsahuje navigační tlačítka. Ostatní vlastnosti a činnosti třídy byly již zmíněny v textu kapitoly struktury aplikace.

### 4.3 Získání mapových dat

Pro chod aplikace a vykreslení cesty je klíčovým prvkem získání informací o této cestě. K tomuto účelu je použito Google directions API, které poskytuje data popisující délku, dobu trvání k překonání této vzdálenosti, GPS souřadnice koncových bodů cesty a především GPS lokace míst, jenž tvoří tuto cestu. V této kapitole je sepsán postup, kterým se tyto údaje o vykreslované trase získávají.

Google directions API běží na serveru, přičemž data z této služby je možné získat pomocí http dotazu. Proto je nutné, aby zařízení bylo v době získání těchto dat připojeno k internetu. Pokud připojeno není, po zadání cesty k vykreslení vypíše aplikaci uživateli zprávu o tom, že je nutné připojit zařízení k síti. Když jsou adresy koncových částí cesty zadány a zařízení je připojeno k síti, je nutné, aby se vytvořilo patřičné url pro získání odpovědi. Obecné url má následující formu:

\[
\text{http://maps.googleapis.com/maps/api/directions/output?parameters,}
\]

kde output definuje formát, v němž je odpověď poskytnuta. V tomto programu bude zpracovávaným formátem JSON. Pro získání požadovaných informací o cestě pro tuto aplikaci je nutné připojit další parametry dotazu. Těmito parametry jsou především cílová a koncová adresa cesty. Tyto parametry jsou označeny v dotazu řetězcem \emph{origin} pro počáteční adresu a \emph{destination} pro koncovou adresu. Protože vyžaduje API pro adresy hodnoty zeměpisných souřadnic, je nutné adresy převést do příslušných souřadnic. O to se stará třída \textit{Geocoder}, která je obsažena v rozhraní Android API. Tento
objekt implementuje metody, které slouží k převodu adresy ulice na zeměpisné souřadnice a naopak. Posledním povinným parametrem dotazu je parametr sensor, který vypovídá o využití lokačních sensorů pro dotaz. V tomto případě nabývá parametr hodnotu false. K těmto povinným parametřům jsou přidruženy další dva volitelné parametry, kterými jsou parametr mode, definující režim přepravy, a parametr region, který identifikuje oblast. Jelikož aplikace je určena pro chodce, hodnota parametru režimu je walking a oblast je definována hodnotou cs. Výsledný dotaz pro adresy Božetěchova 3 Brno a Kollárova 6 Brno vypadá následovně:


Tyto souřadnice je před použitím nutné rozkódovat. Dekódování probíhá pomocí funkce decodePoly7. Získané souřadnice jsou uloženy jako seznam ve třídě Path, která se stará o uložení dat cesty a o výpočtu sloužící k zobrazení cesty rendererem.

4.4 Výpočet geolokace

Po získání dat z mapového serveru jsou informace o cestě uloženy v objektu Path ve formátu seznamu GPS souřadnic. Samotná znalost souřadnic k vykreslení cesty však nestačí. Je nutné provést výpočet vzdálenosti mezi po sobě jdoucími souřadnicemi a směr, který je reprezentován úhlem, který svolí přímka spojující tyto body s přímku procházející počátečním bodem a směřující k severu. Na základě těchto výpočtů je pak schopen renderer vykreslit požadovanou cestu, která odpovídá skutečnosti. Stejně tak je nutné tento výpočet provést pro každý bod cesty s bodem aktuální lokace zařízení, aby byla cesta umístěna na správné místo odpovídající skutečnosti.

7 Viz http://jeffreysambells.com/2010/05/27/decoding-polylines-from-google-maps-direction-api-with-java
K výpočtu vzdálennosti mezi dvěmi zeměpisnými souřadnicemi je možno využít modifikaci Haversinova vzorce (viz rovnice 4.1) [15]. Tento vzorec počítá s kulovitým tvarem Země, který má dostatečnou přesnost výpočtu pro běžné účely a navíc počítá dostatečně přesně krátké vzdálenosti, což nebyla vlastnost vzorce bez modifikace.

\[
d = a \cos(\sin(\phi_1) \sin(\phi_2) + \cos(\phi_1) \cos(\phi_2) \cos(\Delta \lambda)) * R
\]

(4.1)

\[
\theta = a \tan(2(\sin(\Delta \lambda) \cos(\phi_2) \cos(\phi_1) \sin(\phi_2) - \sin(\phi_1) \cos(\phi_2) \cos(\Delta \lambda)))
\]

(4.2)

Hodnota \( \phi \) určuje zeměpisnou délku, \( \lambda \) zeměpisnou šířku a \( R \) je poloměr Země (6,371 km). Zeměpisné souřadnice musí být v radiánech k dosažení správného výpočtu. K výpočtu azimutu z počáteční souřadnice ke koncové slouží vzorec 4.2.

V aplikaci provádí tento výpočet funkce \textit{distanceBetween}\(^8\), kterou poskytuje rozhraní Android API. Funkce po předání souřadnic počátečního a cílového bodu navrací v poli přibližnou hodnotu vzdálenosti a azimutu z počáteční lokace. Tento výpočet je prováděn pro každé dvě sousedící souřadnice cesty a poté je předán požadavku na vykreslení cesty. Stejný výpočet je důležitý provést také pro každou souřadnicí s aktuální lokací zařízení. Na základě tohoto výpočtu je pak možné umístit cestu do okolního světa na odpovídající pozici. Po vykreslení cesty je volána funkce navíc pro zobrazení navigačních šipek v počátečním a koncovém bodě.

### 4.5 Orientace zařízení

Pro vytvoření aplikace zobrazující rozšířenou realitu je důležitým aspektem výpočet orientace zařízení v reálném prostředí. K tomu, aby bylo možno tuto orientaci zjistit, je nutné, aby zařízení mělo zabudováno akcelerometrické a magnetické senory. Z těchto senorů získává aplikace data o orientaci zařízení, které po zpracování slouží k vykreslení cesty do správné pozice.


\[
output[i] = output[i] + \alpha * (input[i] - output[i])
\]

(4.3)

Hodnota \( \alpha \) představuje konstantu redukce šumu, která může být v intervalu: \((0<\alpha<1)\). Platí, že čím menší tato hodnota je, tím dochází k větší redukci. V aplikaci má konstanta hodnotu 0.15, která dostatečně snižuje hodnotu šumu.

\(^8\) Viz http://developer.android.com/reference/android/location/Location.html
Po přefiltrování těchto hodnot dochází k výpočtu rotační matice. Výpočet obstarává metoda `getRotationMatrix`, kterou obsahuje třída `SensorManager` umístěná v knihovně Android. Tato metoda transformuje souřadný systém zařízení na souřadný systém světa, přičemž jako parametry ji jsou předány hodnoty z akcelerometrického a magnetického sensoru. Rotační matice má následující podobu [17]:

\[
\begin{bmatrix}
M_0 & M_1 & M_2 & 0 \\
M_4 & M_5 & M_6 & 0 \\
M_8 & M_9 & M_{10} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Souřadný systém je nyní následovný. Osa \(x\) je tečná k zemi v bodě, kde se zařízení aktuálně nachází a směřuje na východ. Osa \(y\) je tečná k zemi v bodě, kde se zařízení aktuálně nachází a směřuje na sever. Osa \(z\) směřuje k nebi a je kolmá k zemi. Tuto matici je možné použít v OpenGL. Ale protože OpenGL využívá jiný souřadný systém, je nutné ji přepočítat do souřadného systému OpenGL. K tomu je použita knihovní funkce `remapCoordinateSystem`. Pro landscape režim, ve kterém tato aplikace běží, je nutné přemapovat souřadnicí zařízení \(x\) na světovou souřadnici \(y\) a souřadnicí zařízení \(y\) na \(–x\) světovou souřadnici. Nyní je tato rotační matice načtena funkcí `glLoadMatrixf` pro nastavení pohledu kamery ve scéně OpenGL.

4.6 Vykreslení cesty

K vykreslení cesty do obrazu kamery slouží rozhraní OpenGL ES. Toto rozhraní obsahuje podmnožinu grafického aplikačního rozhraní poskytnutého rozhraním OpenGL, které je určeno pro mobilní zařízení. V současnosti existuje verze OpenGL ES 2.0, nicméně v této práci je využita starší verze 1.1.

Pro použití rozhraní v aplikaci je nutné implementovat kontejner pro grafické vykreslování OpenGL a renderer, který vykresluje objekty do obrazu. Obraz, do kterého se vykresluje je vytvořen pomocí třídy `GLSurfaceView`, kterou poskytuje balík `android.opengl`. Tato třída umožňuje vytvořit plátno, jenž je zobrazeno uživateli, a podporuje vykreslení objektů v tomto plátně pomocí rendereru. Navíc běží na samostatném vláknu. Tomuto obrazu je nutné pro vyobrazení objektů přidružit renderer. Ten je implementován rozhraním `GLSurfaceView.Renderer`, kde je nutné vytvořit metodu pro inicializaci počátečního stavu a nastavení voleb pro vykreslení, metodu pro poskytnutí rozměrů plátna (obrazovky) do něhož se vykresluje a konečně metoda pro malování objektů. Třída tvořící renderer je pojmenována `GLRenderer`.

Cesta vykreslovaná do obrazu je tvořena grafickými elementy, kterými jsou čtverec, kruh a šipka. Pro každý z těchto elementů je vytvořena třída, která definuje, jak se mají příslušné tvary vykreslit. Nejprve jsou definovány vrcholy objektu, které spojují dva nebo více hran. Vrcholy jsou definovány jako pole, které se uloží do byte bufferu, díky němuž je dosaženo rychlejší vykreslování. Vykreslení plochy probíhá pomocí trojúhelníků. Trojúhelník je základní vykreslovací element, který

---


Algoritmus vykreslování cesty probíhá následujícím způsobem. Renderer načte rotační matici, která umožní nastavení pohledu kamery ve scéně. Směry os odpovídají nastavení z předchozí kapitoly, tzn. osa x je tečná k zemi v bodě, kde se zařízení aktuálně nachází a směřuje na východ, osa y je tečná k zemi v bodě, kde se zařízení aktuálně nachází a směřuje na sever a osa z směřuje k nebi a je k zemi kolmá. Třída Path zjistí, která souřadnice cesty leží nejblíže současné pozici zařízení a poskytnou rendereru údaje o směru a vzdálenosti k této souřadnici (viz předchozí kapitola). Renderer si uloží matici současné pozice na zásobník pro pozdější návrat k této pozici a přesouvá se k nejbližšímu bodu cesty. Přechod k pozici probíhá pomocí funkcí OpenGL ES, a to glRotatef a glTranslatef. Scéna se natočí pomocí funkce glRotatef o úhel, který svírá přímka směřující od současného bodu k cílovému bodu s osou y a poté dojde k přesunu v prostoru pomocí funkce glTranslatef do cílové pozice. Jedna jednotka v OpenGL odpovídá jednomu metru ve skutečnosti. Následuje rotace scény nazpět (zápornou hodnotu předchozího úhlu), aby osy souřadnic směřovaly opět na východ a sever. Vykreslovací scéna je navíc posunuta o metr pod objekt kamery, což má za následek vykreslení cesty, která je umístěna pod rovinu kamery a vytváří tak dojem pěšiny, po níž se bude uživatel procházet.

![Obrázek 4.3: Vykreslení úseku cesty.](image-url)
Nyní je scéna v pozici nejbližšího bodu cesty a odtud začíná vykreslování. Nejprve je v tomto bodě vykreslen pod rovinu kamery kruh (viz obr. 4.3a). Tento kruh je vykreslen u každé lokace, která je uložena v seznamu souřadnic. V tomto bodě se budou spojovat přímky vedoucí k nejbližším lokacím cesty a tento kruh je bude propojovat. Odtud se bude trasa vykreslovat dvěma směry. Prvním směrem bude vedena cesta k cíli a druhým směrem povede trasa ke startovní pozici. Uživatel tedy bude mít v každé pozici trasy zobrazenou jak cestu vedoucí k cíli, tak i cestu, kterou již urazil. Po vykreslení kruhu v určité lokaci je nutné vykreslit přímkou vedoucí k další souřadnici cesty. Nejprve se nastaví kamera do pozice směřující k další souřadnici. Opět se jedná o operaci GLRotatef, které je dodán úhel vedoucí k požadované lokaci. Tentokrát se ale scéna posune pomocí GLTranslatef pouze do půli cesty mezi těmito souřadnicemi. V tomto místě se použije funkce GLScalef pro změnu rozměrů čtverce, jehož strana je vynásobena délkou vzdálenosti mezi dvěma souřadnicemi. Vzniká tak obdelník, který je vykreslen z bodu první souřadnice do bodu druhé souřadnice a znázorňuje tak úsek této části cesty (viz obr. 4.3c). Po jeho vykreslení se scéna posune do druhé souřadnice, vykreslí se zde kruh (viz obr. 4.3c) pro spojení cest na určitou trase, kdy osa x ukazuje na východ. Tímto principem dochází k vykreslení všech úseků cesty až do cílové souřadnice (viz obr. 4.3d) a stejně tak je vykreslována i cesta k počáteční souřadnici. Nakonec se načte matice ze zásobníku, která byla uložená při začátku vykreslování, čímž se obraz kamery vrátí na současnou pozici zařízení a uživatelé je vykreslena cesta odpovídající skutečnosti.

Při vykreslování cesty pod úroveň kamery se po určité vzdálenosti dostane cesta na úroveň horizontu a dále není viditelná. Pro urychlení vykreslení, obzvláště u delších tratí, není potřeba vzdálenější cesty vykreslovat. Proto poskytuje třída Path také metodu, která vypočítá, zda je vykreslovaná cesta za horizontem. Pokud ano, k zobrazení této části cesty nedojde a vykreslování cesty končí. S tímto problémem souvisí také fakt, týkající se vzdálenosti horizontu. Při měřítku zobrazované cesty, kdy metr ve skutečnosti odpovídá jednotce vzdálenosti v OpenGL, dojde k dosažení horizontu zhruba po 150 metrech. To může být pro uživatele krátká vzdálenost zobrazené cesty a proto je možné v nastavení použít další dvě měřítky. Měřítko 1:2, kde 1 metr ve skutečnosti odpovídá jednotce vzdálenosti v OpenGL a měřítko 1:3, kde 3 metry reálného světa odpovídají jedné jednotce v programu. Při měřítku 1:2 je vykreslena cesta přibližně do vzdálenosti 300 metrů a při posledním měřítku až do 450 metrů. U těchto upravených měřítek většina však nastává jiný problém, a to ten, že délka vykreslené cesty přesně neodpovídá reálné vzdálenosti. Tento problém se projevuje tak, že virtuální cesta zatáčí dříve, než je skutečná křížovatka. Proto je vytočeno měřítko 1:2, které se snaží najít kompromis mezi oběmi krajními hodnotami. Záleží tedy pouze na uživateli, kterému měřítku dá přednost.
Při zobrazení cesty blížící se k horizontu nastává také komplikace související s plochostí vykreslované cesty. Vykreslovaná cesta je plocha, která je rovnoběžná s rovinou kamery a nemá žádný třetí rozměr. Z toho důvodu je špatně (popř. vůbec) viditelná cesta, která směřuje k horizontu a zatáčí kolmo k přímce, propojující kameru s horizontem. Řešením tohoto problému je vykreslení cesty s profitem x (viz obr. 4.4). To znamená, že profil vykreslované cesty je kříž, jehož dvě přímky jsou k sobě kolmé. Tato vlastnost je relizována tak, že při vykreslování obdělníku, spojujícího dvě zeměpisné souřadnice je scéna přetočena o 90 stupňů a k původnímu obdělníku je přikreslen nový, který je k němu kolmý. Díky této vlastnosti je možně problémové cesty vidět lépe a opět je možné tento rys povolit, případně zakázat v nastavení aplikace.

Pro lepší orientaci uživatele při zdolávání cesty jsou vykresleny navíc dva grafické prvky, které odkazují na počáteční a koncovou souřadnici. Těmito prvky jsou barevné šipky. Červená šipka označuje směr k cíli trasy a naopak modrá šipka informuje o směru, odkud uživatel cestu započal (viz obr. 4.5). Tyto šipky jsou umístěny těsně nad úroveň již zobrazené cesty a směřují k příslušnému bodu cesty. Velikost tohoto prvku navíc ovlivňuje vzdálenost označujících koncových souřadnic od současné zeměpisné lokace. Pokud je tato souřadnice dále než 150 metrů, šipka má konstantní rozměry. Při vzdálenosti menší než tato hranice, rozměry šipky rostou. Pokud je cesta příliš vzdálená od současné lokace zařízení, cestu vidět nelze, ale šipky označující koncové body jsou vykresleny vždy.

Obrázek 4.5: Zobrazení cesty s šipkami označující koncové body (červená šipka označuje cílovou destinaci, modrá počáteční lokaci).

### 4.7 Vývoj aplikace

Tato práce byla vytvářena ve vývojovém prostředí Eclipse, které je oficiálním prostředím k vývoji aplikací pod Android. Toto prostředí nabízí plugin Android Development Tools (ADT), který poskytuje silnou základnu a spoustu rozšíření pro vývoj Android aplikací. Pro implementaci byl použit software development kit (SDK) pro Android. Veškerý kód je tedy sepsán v programovacím jazyce Java, popř. rozložení vizuální stránky aktivit a manifest aplikace je sepsán v souborech formátu XML.

Aplikace je při vývoji průběžně testována a na základě těchto výsledků jsou provedeny úpravy v implementaci. Testování programu probíhá přímo na mobilním zařízení, které je připojeno k počítači pomocí USB kabelu. Pro testování mobilních aplikací poskytuje Android SDK emulátor (virtuální mobilní zařízení, které běží na počítači), který slouží k vývoji a ladění programu na počítači bez použití fyzického zařízení. Toto zařízení je však pro testování popisovaného programu nevhodné, protože neumí simulovat lokaci GPS souřadnic, které je pro funkci aplikace nezbytné. Stejně tak je obtížné v tomto virtuálním zařízení simulovat data z orientačních sensorů a kamery. Proto je aplikace testována přímo na mobilním zařízení.

**4.8 Testování**

Testování aplikace je možné rozdělit na dvě části. Testování, které je možné uskutečnit bez příjmu skutečné GPS lokace, a testování, které je nutné provést s touto lokací. Toto rozdělení odráží skutečnost, že některé funkce aplikace nejsou bez reálného příjmu GPS signálu možné (je sice možné simulovat změnu GPS souřadnic, nelze však zjistit, zda vykreslení cesty do obrazu odpovídá poloze cesty ve skutečnosti). Touto funkcí je myšleno zobrazení cesty do obrazu kamery přesně na místo, kam patří.

**4.8.1 Průběžné testování**

První díl testování se věnuje ověření funkčnosti činností aplikace, které je možné testovat bez příjmu GPS signálu. Jedná se především o ověření správné funkčnosti samostatných částí aplikace. Těmito činnostmi jsou zobrazení obrazu kamery, přepínání mezi okny, uživatelské rozhraní, použití Google map, stažení mapových dat nebo ošetření různých stavů. Tato testování jsou z velké části prováděna průběžně při implementaci.


\(^{11}\) Viz http://developer.android.com/about/versions/android-2.3-highlights.html
zařízení. Mimo to byly definovány testovací adresy, které zamezovaly zdržování při vyplnění polí adres. Aktivita také umožňovala nahráni offline cesty a zvolit měřítko vykreslené cesty.

Při testování uživatelského rozhraní bylo důležité testovat také možné vstupy zadané uživatelem. V aplikaci se vyskytuje pouze jedno místo, které dává prostor uživateli zadat libovolná data, a proto se test vstupů soustředil především na aktivitu zadání cesty. Aktivita umožňuje uživateli zadat počáteční a koncovou adresu cesty. Protože se adresy zpracovávají pomocí třídy Geocoder z Android frameworku, umožňuje tato třída překlad adres celého světa. Stejně tak serverové rozhraní map poskytuje celosvětové využití. Při zadání a poskytování cesty se může objevit problém s přílišnou délkou cesty, kterou není možné zpracovat. Proto má aplikace omezení, které zpracuje a zobrazí cestu pouze v případě, že není delší než 100km. Tato vzdálenost je dostatečná k překonání jakékoliv trasy v rámci města. Existuje zde navíc další podmínka, která zamezuje zobrazit cestu, jejíž nejbližší bod k uživateli je dále než 10km. Tyto podmínky jsou pro účel aplikace plně dodateční. Pokud bude později nezbytné tyto podmínky měnit, je to možné, ale musí se dodatečně otestovat.

4.8.2 Testování v terénu

Protože cílem aplikace je zobrazit rozšířenou realitu závisící na GPS poloze zařízení, je nutné získat skutečnou pozici zařízení a odpovídající obraz kamery v této lokaci. Tyto požadavky není možné získat jiným způsobem, než vyzkoušet aplikaci v terénu. Tento proces testování může být proveden až po otestování samostatných bloků aplikace a ověřuje, zda všechny propojené bloky aplikace vytváří požadovaný výsledek, jímž je obraz rozšířené reality.

Toto testování probíhalo způsobem, kdy jsem se ocitl na libovolném místě v ulicích Brna, a mým cílem bylo dojít na zadanou adresu, která se nachází rovněž v Brně. Při překonávání cesty se pohlíželo na vlastnosti, jako jsou vykreslení cesty do odpovídající cesty v reálném prostředí, vypočítání již ušlé a zbývající délky trasy nebo přesnost přijímaných GPS informací. Při testování se ukázalo, že klíčovým bodem fungování aplikace je právě příjem GPS signálu. Při výborné přesnosti (přesnost určení polohy je kolem 4m) poskytnuté souřadnice je cesta vykreslena do obrazu kamery celkem věrohodně a navíc dochází k přesnému pohybu kamery po trase. Při průměrné přesnosti (naměřená přesnost kolem 24m) už nereflektuje skutečnou cestu ideálně a při špatné přesnosti (naměřená přesnost má hodnotu 60m a více) už je cesta vykreslena hůře. Přesnost tohoto signálu navíc dost kolísá, což může způsobit po pár metrech úplnou ztrátu signálu.

Při spuštění programu a zadání cesty ze současné pozice zařízení je nutné mít již lokaci zařízení dostupnou. Získání GPS lokace může při spuštění aplikace nějakou dobu trvat, a proto byly na základě testování umístěny do uživatelského rozhraní také dvě ikony. První ikona upozorňuje získání GPS polohy zařízení. Ikona může být překryta červeným křížkem, což symbolizuje stav, kdy není lokace zařízení známa a naopak bez překrytí informuje ikona o dosažení GPS lokace. Pro úplnou informaci uživatele je v uživatelském rozhraní znázorněna také ikona připojení k internetové síti.

Testování přineslo také informaci o tom, že cesta není vykreslena do větší vzdálenosti než 150m a proto přišel na řadu výběr jednoho ze tří měřítek, které je veškeré popsáno v kapitole implementace (viz obr. 4.6). Stejně tak se narazilo i na problém špatné viditelnosti cesty v případech, kdy se blíží horizontu a díky tomuto problému je možné zvolit křížový profil zobrazované cesty.
Výsledkem testování byl učiněn závěr, že veškerá kvalita zobrazované virtuální cesty záleží především na přesnosti GPS souřadnic zařízení. Při příjmu přesných lokačních dat se vykresluje cesta do správných míst a překrývá požadovanou cestu. Pokud se při překonání cesty udržuje přesnost signálu max do 40 metrů a příliš se tento signál neztrácí, je možné aplikaci skutečně použít pro navigační účely.

Obrázek 4.6: Zobrazení úseku cesty v různých měřítkách. Na obrázku a je použito měřítko 1:1, cesta je nejpřesnější. Na obrázku b je měřítko 1:2, cesta už není tak přesně vykreslena, na posledním obrázku je zobrazeno měřítko 1:3, cesta je nejméně přesná, ale vidí s ní uživatel cestu nejdále.
Závěr

Cílem práce bylo vytvořit aplikaci, která slouží jako navigace v zařízeních s operačním systémem Android. Tato navigace je realizována pomocí rozšířené reality, kdy do obrazu reálného světa je vykreslena cesta vedoucí k zadanému cíli (adrese). K dosažení této funkčností využívá program obrazu z kamery, dat z orientačních sensorů a GPS souřadnic zařízení. Potřebná mapová data poskytuje aplikaci rozhraní Google directions API. Protože virtuální trasu není vždy možné vykreslit přesně do pozice odpovídající umístění cesty ve skutečnosti, snaží se aplikace o co nejvěrnější umístění cesty v obrazu. Účelem aplikace je však zobrazení cesty, která slouží k přibližně navigací směru cesty, proto program plní svůj úkol i při přibližném zobrazení dráhy. Aplikace je určena především pro chodce, kterým přijde vhod navigační informace poskytnutá aplikací v mobilním zařízení.

Čtenář se v tomto dokumentu mohl seznámit s kapitolami úvodu, teorie, návrhu a implementace. Část úvodu se věnovala uvedení do problematiky a seznámení s cílem práce. V sekci teorie jsou nastypey základní termíny a znalosti, které jsou využity při tvorbě této aplikace. Především se jedná o obeznámení s operačním systémem Android a vysvětlení pojmu rozšířené reality. V sekci návrhu se čtenář dočetl o koncepci a rozboru práce, o rozdělení aplikace na jednotlivé části a návrhu uživatelského rozhraní. Kapitola implementace popisovala realizaci aplikace a způsob řešení problémů při implementaci rozšířené reality, získání mapových dat a vykreslení cesty. Zároveň se zmínila o procesu vývoje a testování. Podkapitola vývoje informuje o zařízení a prostředí, ve kterém byla aplikace vyvíjena a podkapitola testování popisuje způsob, kterým byla ověřena funkčnost aplikace.


Při pokračování na vývoji této práce lze pokračovat mnoha směry. Je možné pracovat na další funkčnosti aplikace, která by měla za úkol poskytnout více informací o přesunu k cíli. Takovými údaji je myšleno např. dát uživateli informace o nejbližší zastávce městské hromadné dopravy a zobrazit jízdní řady linek dopravy, které umožní uživateli z dané zastávky se rychleji dostat k cíli. Samozřejmě je také možné věnovat se úpravám již provedené implementace, které by sloužily k lepšímu zobrazení cesty do prostoru, popř. využití jiných zdrojů informací pro výpočet pozice a orientace zařízení. Za zmínku stojí také možnost změny uživatelského rozhraní, které by bylo pro uživatele a atraktivnější. Návrhů na vylepšení aplikace může být spoustu a záleží pouze na výsledku, kterého chceme dosáhnout.
Literatura


34
Příloha A

Obsah CD

- /bin/ – instalační soubor aplikace
- /doc/ – text technické zprávy a návod k použití aplikace
- /src/ – zdrojové soubory Android projektu
- /pictures/ – obrázky pořízené při testování aplikace
- /presentation/ – plakát a demonstrační video