

**XXXII. ORSZÁGOS TUDOMÁNYOS DIÁKKÖRI
KONFERENCIA
MŰSZAKI TUDOMÁNYI SZEKCIÓ**

**SMARTVÁSÁRHELY
ANDROID APPLIKÁCIÓ**

Szerző: Tordai Tünde

Sapientia Erdélyi Magyar Tudományegyetem, Műszaki és Humántudományok Kar,
Marosvásárhely, Informatika szak, III. év

Témavezető: Dr. Antal Margit adjunktus

Sapientia Erdélyi Magyar Tudományegyetem, Műszaki és Humántudományok Kar,
Marosvásárhely, Matematika – Informatika Tanszék

Kivonat

Rohamosan fejlődő világunkban nemcsak a technika, hanem a települések is gyorsuló fejlődésnek indultak. A pár évvel ezelőtti kisváros, mára már nem is annyira „kisváros”, és a tájékozódás sem annyira egyszerű benne. Az emberi kíváncsiság azonban sok szép és új helyre csalogatja az embert, de az új dolgok ugyanakkor ismeretlenek is számunkra. Éppen ezért egy útbaigazítás rögtön megkönnyíti a közlekedést.

Ebből kiindulva született az ötlet, egy városnéző alkalmazás megvalósítása. A dolgozat célja egy olyan szoftver megvalósítása volt, mely a Marosvásárhelyre látogató turisták számára egy könnyed segítséget nyújt a városban való eligazodáshoz, és segít feltérképezni a város szépségeit.

Dolgozatomban egy rövidebb kutatást végeztem a már meglévő hasonló célú androidos alkalmazások körében, majd azt vizsgáltam, hogy az általam fejlesztett alkalmazás milyen mértékben könnyíti meg a felhasználó útbaigazodását.

Az alkalmazás lehetőséget biztosít a felhasználó számára a GoogleMaps térkép segítségével boldogulni a városban, illetve környező épületek megtalálásában. Leírásokat és képeket szolgáltat a város híres épületeiről, illetve pár órás sétákat biztosít, mely segítségével a felhasználó meglátogathatja Marosvásárhely barokk vagy akár szecessziós épületeit.

Az épületek megtalálásában segít az alkalmazás azon funkcionalitása, mely egy adott épület lefényképezése esetén jelzi, hogy melyik híres épületnél járunk. Ennek a megvalósításához OpenCv függvényeket használ, melyek segítségével képelemzést végez, kiszűri a kép fontosabb részeit, majd összehasonlítva az adatbázisban levőkkel jelzi a megfelelő találatot.

Az alkalmazás fejlesztése után, 18 felhasználóval egy élménytesztet töltöttem ki. A fő cél az alkalmazás kezelhetőségének és hasznosságának a vizsgálata volt. Végezetül, ezen eredményeket feldolgozva állapítottam meg egy végső következtetést arra vonatkozóan, hogy az általam implementált alkalmazás, bizonyos helyzetekben, mennyire lehet hiánypótló.

Abstract

In our rapidly developing world not only the science, but the settlements are developing faster than ever. A town, which was small some years before, nowadays can be a big town having a complicated road network. The curiosity of people encourages visiting new places. For a proper experience tourists need guidance.

This gave me the idea of developing a city guide application. The goal of the project was to develop such an application, which gives an easy guidance for tourists, who visit Marosvásárhely, and helps them to discover the beauty of the city.

In my paper I have made a small survey among similar apps, after that I concluded some guide-lines in order to make the navigation easier.

The application helps navigating tourists through the city using Google Maps and to discover famous places. It provides some descriptions and images about famous buildings and offers two itineraries, one for baroque buildings, and the other for secession buildings.

The other functionality of the app is the automatic recognition of the buildings. If the user takes a picture of a building, the application recognizes that building. Building recognition is performed using OpenCv functions for image analysis. Using the important parts of images (keypoints) and comparing them with images of the buildings located around the current position the application provides information regarding the recognized building.

The application development was followed by a usability testing involving a group of 18 users. Besides application's usability also its usefulness was tested. Based on the evaluation of the questionnaire I formulate the main conclusions regarding usability and usefulness of the application.

Tartalomjegyzék

1. Bevezetés	1
2. Piackutatás	3
3. Helyzetmeghatározás és Google Térkép.....	5
4. Képfelismerés.....	9
4.1 SIFT algoritmus.....	12
4.2 SURF algoritmus	12
4.3 ORB algoritmus.....	13
5. SmartVásárhely applikáció	15
5.1 UseCase diagram.....	16
5.2 Szekvencia diagram	17
5.3 Osztálydiagram.....	17
5.4 Az alkalmazás bemutatása	19
5.4.1 Városnézés, navigáció	19
5.4.2 Képfelismerés	26
6. Alkalmazás kiértékelése	29
6.1 Felhasználók által kitöltött kérdőív	29
6.2 Kérdőívek eredményeinek értékelése.....	31
7. Következtetések és továbbfejlesztési lehetőségek.....	36

1. Bevezetés

Gyorsan fejlődő világunkban egyre több lehetőség tárul elénk az új dolgok, új helyek megismeréséhez. Nem jelent ma már akadályt egy másik városba, országba, vagy éppen kontinensre látogatni. Egyetlen megtorpanást csupán az jelenthet, hogy hogyan fogunk eligazodni egy számunkra ismeretlen, idegen környezetben.

A kalandra vágyó, merész, lelkes egyének számára azonban nincsenek ilyen akadályok, amelyek meggátolnák őket abban, hogy egy nagyobb utazásnak nekivágjanak. Az ismeretlent csupán egy kihívásként, kalandként fogják fel.

Azonban nem mindig adottak olyan körülmények, hogy az ember bármennyi ideig kirándulgasson, és ilyen esetben, nem mindegy, hogy az ott eltöltött idő miben bővelkedik.

Akkor tekinthető egy kirándulás igazán tartalmasnak, ha az ember megismerkedik vagy betekintést nyer az illető település kultúrájába, szokásaiba, feltérképezi a helység szépségeit, nevezetességeit. Számos útikalauz, térkép és az internet is szolgálatunkra áll ilyen esetekben. Manapság azonban kevés az az ember, aki útikalauzzal, térképpel kezében barangol egy ismeretlen városban. Sokkal inkább azt látjuk, hogy okostelefonokat böngészve közlekedik az emberek nagyrésze. Elegáns, könnyedén használható és viszonylag gyorsan választ ad a kérdéseinkre.

Ezért is választottam az Android platformot, hiszen biztosítja az alkalmazások könnyed fejlesztését, széles körben elterjedt és nagy előszeretettel használják.

Az ismeretlen feltérképezése, egy számunkra idegen település bebarangolása során a fő célunk minél több nevezetesség meglátogatása.

Ebből a célból született az ötlet: egy Marosvásárhelyre kifejlesztett városnéző alkalmazás megvalósítása. A cél egy olyan szoftver fejlesztése volt, mely könnyed eligazodást biztosít a felhasználó számára, segít megtalálni a város nevezetességeit és ezekről bővebb információkat nyújt. Kétféle sétát is biztosít, mely által a felhasználó megtekintheti marosvásárhely barokk, illetve szecessziós épületeit.

A navigációt a GoogleMaps térkép implementálásának segítségével valósítottam meg, míg a felhasználó aktuális helyzete körül levő, különböző szolgáltatásokat nyújtó épületek (Bank, Étterem, Bevásárlóközpont stb.) megjelöléséhez a Google API szerverrel való kommunikáció volt a segítségemre. Navigálás során nagy előnyt jelent, ha a készülék rendelkezik internet hozzáféréssel, hiszen ezáltal több funkcionalitás vehető igénybe. Ha azonban nem érhető el az internet, ez sem jelent egy akadályt, hiszen offline térképet felhasználva könnyedén navigálhat tovább a felhasználó. Hátránya a kevesebb útbaigazítás hiánya.

Az alkalmazás másik fontos funkcionalitása, az automatikus épületfelismerés. Ennek folyamata úgy történik, hogy a turista által egy adott épület lefényképezése esetén, az alkalmazás felismeri az adott épületet és bővebb információkat szolgáltat erről. A gépi látás megvalósítását az OpenCv szoftvercsomag biztosította, mely a képfeldolgozó algoritmusai által lehetővé teszi a képelemzést, amely segítségével sikerült kiszűrni egy kép jellegzetes pontjait, majd felismerni azt.

Az alkalmazás jelenleg a magyar anyanyelvű közönséget célozza meg, de tervben van a többnyelvűsítés is.

2. Piackutatás

A téma megszületésekor felmerült a kérdés, hogy létezik-e a már az androidos mobilalkalmazások piacán ilyen, vagy ehhez hasonló alkalmazás. Ennek kutatása következtében érdekes eredményhez jutottam, ugyanis nem találtam kimondottan Marosvásárhelyre kifejlesztett városnéző alkalmazást.

A piackutatás folytatásaként, az androidos mobilalkalmazások piacán, a Google Play-n levő városnéző alkalmazásokat tanulmányoztam. A városnézésre kifejlesztett, felhasználók által legkedveltebb alkalmazásnak a TripAdvisor CityGuide [1] minősült, mely mindössze 60 várost foglal magába, melyek közül 25 európai város. Azonban a 25 város nem tartalmazza, nemhogy Marosvásárhelyt, de egyetlen romániai várost sem.

A romániai városokra kifejlesztett városnéző alkalmazások piaci kínálata szegényesnek minősült. Kevés olyan város van, amelyik rendelkezik ilyen alkalmazással, és személyes véleményem alapján a meglévők, funkcionalitás szempontjából, nem minősülnek igazán jól alkalmazhatónak.

Habár nem találtam olyan városnéző alkalmazást, melynek célpontja Marosvásárhely lenne, vannak azonban olyanok, melyek globálisan több várost is magukba foglalnak, többek között Marosvásárhelyt is. Egyik ilyen a Triposo [2] alkalmazás, mely több Romániai várost is tartalmaz, így Marosvásárhelyt is. Viszonylag számos szolgáltatásban bővelkedik, azonban gyengesége a szegényes, nem igazán megragadó design. Egy másik vizsgált alkalmazás, a Cityguide [3], mely navigációs funkciót biztosít több város számára is, köztük szerepel Marosvásárhely is.

A Marosvásárhelyt valamelyest érintő alkalmazások vizsgálata során megfigyelhető, hogy legtöbbjük csupán a navigálást biztosítja. Kevés olyan van, amely városnézéshez szükséges információkat is nyújt, és ha mégis, akkor ezek nem annyira részletesek.

Igaz ugyan, hogy egy városban való eligazodáshoz navigációra van szükségünk, melynek talán legjobb eszköze a Google által kifejlesztett Maps [4] alkalmazás. Azonban ha egy idegen városban vagyunk, nem csak az a legfőbb célunk, hogy adott helyről adott helyre eljussunk, hanem, hogy az út során minél többet megtudjunk az úticélunknak kiválasztott helyről, megismerjük a város nevezetességeit. Ezért is előnyös ha egy alkalmazás felsorakoztat híresebb épületeket, melyek közül nem biztos, hogy mindenik ismert számunkra.

Ezen épületekről leírást, érdekességeket, képeket tartalmaz, ahogyan a városról is, így belekóstolhatunk az ottani konyha ínycségeibe, betekintést nyerhetünk az adott település kultúrájába. Mivel ez városonként egyedi jellemző, így ilyen típusú részleteket egy navigáló alkalmazás nem biztosít.

A Google áruház piacán tanulmányozott városnéző alkalmazások, átlagosan a következő funkcionalitásokat biztosítják.

Alapvető eszközként egy térképet implementálnak, mely segítségével biztosítják a navigációt. Adott alkalmazások esetében lehetőség van csak egy adott kerület, városrész kiválasztására. Az eszköz aktuális pozíciójából kiindulva, egy menüpont keretén belül kiválasztott épülettípus esetén, a térképen elérhetővé válnak ennek a feltételnek eleget tevő, a környezetben elhelyezkedő épületek. Vannak esetek, amikor egy keresés segítségével egy adott épületre is rákereshetünk. Az épületek, szolgáltatásaik szempontjából, kategorizálva vannak (hotelek, éttermek, bevásárlóközpontok, szórakozóhelyek). Mindenik esetében, alkalmazástól függően, találunk szűkebb - bővebb információkat róluk. Van amelyik csak egy minimális információt szolgáltat egy adott helyről, míg mások képektől elkezdve egészen a látogatók véleménynyilvánításáig tartalmazznak részleteket. Az alkalmazások lehetőséget biztosítanak az adott város látnivalóinak feltérképezéséhez és némelyek pár órás, napos vagy akár több napos túrákat is tartalmaznak. Egyesek saját útvonal megtervezését, kedvelt vagy meglátogatni tervezett épületek listájának megalkotását, utazási naplót, vagy adott épületekhez vélemény megírást is biztosítanak. Átlagosan mindenik alkalmazás rendelkezik az adott várossal kapcsolatos alapszolgáltatásokkal, mint történelem, kultúra, közlekedési hálózat, pénznem, némelyek pedig mini szótárt is magukba foglalnak, segítve ezáltal a könnyebb boldogulást. Egyes alkalmazások előnyös tulajdonsága az offline módban való működés.

Bármelyik városnéző alkalmazást tekintjük mindenik arra fókuszál, hogy a felhasználónak minél részletesebb információkat nyújtson egy adott épületről és hogy minél egyszerűbb formában eljuttassa oda.

De mi történik abban az esetben, ha egy olyan épületepárhoz jut amelyek közül, nem igazán tudja eldönteni, hogy a kettő közül melyik az, amelyiket éppen megtekinteni óhajtott. Ebben lehet nagy segítségére egy olyan funkcionalitás, mely automatikus épületfelismerést biztosít. Ez esetben az adott épület lefényképezésekor az alkalmazás felismeri azt és bővebb információkat ad az adott épületről. Ezen funkcionalitás megvalósításához az OpenCv gépi látáshoz kifejlesztett szoftvercsomag biztosít megfelelő keretet.

3. Helyzetmeghatározás és Google Térkép

Alkalmazások sorozata használja nagy előszeretettel a helymeghatározás alapú szolgáltatásokat, melyekkel beazonosítják a mobil eszköz aktuális földrajzi helyzetét, annak érdekében, hogy helyzet alapú információkat szolgáltatassanak a felhasználónak.



1. Ábra:

A helymeghatározás alapú szolgáltatások (LBS – Local Bases Services) **Google Maps Logo** biztosítják a felhasználó számára, hogy személyes keresési igényeinek megfelelő információkhoz jusson. Ez egy új területet tár a fejlesztők és a mobilhálózat szolgáltatók elé, hogy olyan alkalmazások fejlesztésébe kezdjenek, melyek lényeges információkat szolgáltathatnak a felhasználónak, mint például a jelenlegi közlekedési viszonyok, útvonal adatok.

A mobil felhasználók számára a következő szolgáltatásokat biztosíthatják a helymeghatározás alapú szolgáltatások:

- a legközelebbi adott típusú épület meghatározása, mint egy bank vagy szálloda,
- üzenet küldése, mellyel értesítik a felhasználót egy adott bevásárlóközpontban levő árleszállításról vagy éppen egy forgalmi dugóról,
- barátkeresés, vagy éppen egy ellopott telefon helyzetének meghatározása.

A mobil eszköz helyzetének meghatározásához, az LBS-nek valós időben kell használnia a helymeghatározási módszereket. A módszer pontossága a megközelítés típusától függ, mely lehet egy térbeli elhelyezkedés vagy szöveges leírás.

A térbeli elhelyezkedést a földrajzi koordináták (szélességi, hosszúsági, magassági) adják meg. A földrajzi szélesség a Föld középpontját és a térbeli pontot összekötő egyenes és az egyenlítő síkja által bezárt szög (0 – 90 fokig terjed), míg a földrajzi hosszúság a greenwichi (Anglia), nemzetközileg elfogadott kezdőmeridián, és az adott ponton átmenő meridián síkja által bezárt szög (0 -180 fokig terjed). A magasság pedig a tengerszint felett méterben kifejezett érték.

A szöveges leírást az utcacím, ország és postakód együttese alkotja.

Az eszköz aktuális helyzetének meghatározása kétféleképpen történhet:

- Mobilhálózat szolgáltató (Mobile Phone Service Provider Network)

Az aktuális helyzetet három mobilcella adataiból határozza meg. Ez a legalapvetőbb és legolcsóbb kifejlesztett módszer erre a célra.

- Műholdak segítségével

A GPS (Global Positioning System) 24 műholdból álló rendszert használ, melyek a Föld körül keringenek. A GPS vevő, az eszköz és az éppen látható műholdaktól való távolság alapján határozza meg az eszköz aktuális pozícióját. A GPS jelek fogadásához szükséges, hogy a készülék rendelkezzen GPS vevővel.

Assisted-GPS (A-GPS) egy GPS alapú újabb technológia, kifejezetten olyan okostelefonokra, amelyekben van internet hozzáférés is a GPS vevő mellett. Ennek a technológiának az lényege, hogy a mobilkészülék GSM alrendszere segít a GPS-chipsetnek a legközelebbi műholdak gyors beazonosításában. Ez úgy oldható meg, hogy a mobilszolgáltató az okostelefon aktuális GSM-cellainformációja alapján, adatkapcsolat segítségével elküldi a készüléknek az adott cella fölött tartózkodó GPS-műholdak pontos helyzetét. Az A-GPS-chipset pedig ezeken a helyeken fogja keresni a műholdakat az égbolton, és nagyon gyorsan megállapítja a pontos helyzetünket.

A helyzetazonosító szolgáltatásokkal ellátott alkalmazások igen népszerűek lettek, melyek használata napjainkra már sok felhasználó esetében megszokássá vált. Gyakran térképpel összekapcsolva jelennek meg, annak érdekében is, hogy egy jobb felhasználói élményt nyújtsanak a készülék tulajdonosának a saját földrajzi helyzetéről. Az alkalmazások adott időközönként frissítik az eszköz pozíciójára vonatkozó információkat, ugyanakkor egy üzenet fogadó komponenst regisztrálnak, mely figyelmezteti a felhasználót ha egy adott körzetbe ért.

Az Android egy saját alkalmazásprogramozási interfészt (API) támogat, melynek hozzáférése van a Google térképekhez. Ennek és a helymeghatározásnak köszönhetően különböző helyeket jelölhetünk meg egy térképen.

A Google szolgáltatásai között található Google Térkép lehetőséget biztosít a telefonon megjelenítendő térképek testreszabhatóságához. A szolgáltatások igénybe vételének elengedhetetlen feltétele az API kulcs megszerzése. Az API kulcs igénylésének folyamata során az alkalmazásunk regisztrálódik a Google API Console rendszerébe. A Google API automatikus hozzáférést biztosít a Google Térkép szerverhez, adatletöltést, térkép

megjelenítést engedélyez. Lehetőséget biztosít különböző jelzőtáblák, útvonalak megjelenítéséhez, és négyféle térképnézetet támogat: normál, hibrid, szatellit, terep. A normál térkép tipikus úti térkép, melyen fontosabb épületek, útvonalak és természeti adottságok (pl. folyók) vannak jelen. Hibrid nézet esetében az eredmény nem más, mint szatellit képek úti térképpel való ötvözése. Szatellit esetében magától értetődően szatellit nézetet kapunk. Terep nézet beállítása következtében, topográfiai adatokat láthatunk, a térképen színek, kontúrvonalak, perspektivikus árnyékolások lesznek láthatóak. Lehetőség van beltéri, vagy akár utcai nézet használatára is. Ennek jellegzetessége, hogy nagymértékű nagyítás esetén különböző épületek (repülőtér, bevásárlóközpont, áruház) alaprajzát, beltéri felépítését is megjeleníti. Ez a funkcionalitás automatikusan működik, kicsinyítés esetén pedig eltűnik.

Másik kiemelt és lényeges jellemzője a helyfigyelés. Pontos adatokat szolgáltat a mobil eszköz aktuális helyzetéről, elmozgatás irányáról és irányítottságáról.[5]

A Google 2011. május 10-én hozta meg azt a döntését, miszerint engedélyezi a hozzáférési jogot a Google Places API-hoz. Ez nem más, mint egy szerver, mely adott helyekről egy webszervizben előzőleg bejegyzett információkat tartalmaznak, különleges, nevezetes, vagy kedvelt helyekről. Ezek lekérése HTTP kérésekkel történik.

Négyféle kérést kezdeményezhetünk:

1. Helykeresés – egy listát térít vissza az aktuális pozícióhoz legközelebb eső helyekről, melynek meghatározása a földrajzi szélességi és hosszúsági koordináta alapján történik.
2. Részletes információk egy adott helyről – speciális adatokat küld az adott helyre vonatkozóan. Ilyenek például a cím, kapcsolat (telefonszám), felhasználói értékelés.
3. Hely megjelölése – célja egy hely látogatottságának mérése. A legkedveltebb helyek, a helykeresési kérés végrehajtásakor, prioritással rendelkeznek.
4. Jelentés egy helyről – a felhasználó számára engedélyezi egy új hely hozzáadását, vagy egy meglévő törlését, amelyet az alkalmazás az adatbázishoz rendelt.

A kérések számát a Google Places API korlátozza. API kulccsal rendelkező alkalmazások esetében ez 1000 kérés/24 óra. Azok a felhasználók, akik személyesebb adataikat is megadták a kulcs kérvényezésekor, az API konzolon keresztül, az ők számára a kérések száma több, 100.000 kérés / 24 óra. Az azonosítás bankkártyaszám megadásával történik.

A kérések HTTP URL alapúak, és a kommunikáció JSON vagy XML formátumú adatokkal történik.

Helykeresési kérés kezdeményezésekor a következő adatokat kell megadnunk:

- földrajzi szélességi és hosszúsági koordináta, arról a helyről, amiről információt szeretnénk kapni. Továbbá egy méterben kifejezett távolság, arra vonatkozóan, hogy mekkora körzetben végezze a keresést.
- nyelv megadása - mely jelzi, hogy milyen nyelven szeretnénk az adatokat kapni,
- név megadása - kifejezés keresése esetén, szűrés céljából fontos,
- szenzor - egy igaz/hamis érték, mely jelzi, hogy a kérés egy olyan eszközről történik-e amelyen van helymeghatározó szenzor (pl. GPS).
- kulcs – az alkalmazás API kulcsa, amely az alkalmazás azonosítására szolgál. Ennek hiányában, vagy érvénytelen API kulcs esetén a szerver a kérésünket elutasítja.

Helyre vonatkozó részletek kérése esetében, a következők szükségesek:

- hivatkozás - az adott hely azonosítója (földrajzi koordináták)
- nyelv - mely jelzi, hogy milyen nyelven szeretnénk az adatokat kapni
- szenzor - egy igaz/hamis érték, mely jelzi, hogy a kérés egy olyan eszközről történik-e amelyen van helymeghatározó szenzor (pl. GPS).
- API kulcs - az alkalmazás API kulcsa, amely az alkalmazás azonosítására szolgál. Ennek hiányában, vagy érvénytelen API kulcs esetén a szerver a kérésünket elutasítja.

A Google Places API tartalmaz egy tömegközlekedési API –t is, mely tömegközlekedési dugók, a legközelebbi szállítási útvonalakról ad részletesebb információkat. [6]

4. Képfelismerés

A számítógépes látás egyre több lehetőséget szolgáltat az alkalmazás-fejlesztők számára. „A számítógépes látás az emberi látás azon funkcióit valósítja meg, amelyek a retinai kép elemzését végzik” [7].

A képfelismerés fontos építőköve a képi jellemzők megfelelő kinyerése. Ezek birtokában könnyedén megvalósítható a képösszehasonlítás és – felismerés. Az algoritmusok hatékonysága is nagymértékben ettől függ, hiszen gyakran az algoritmus csak annyira lehet hatékony amennyire hatékony a képi jellemzők detektálása. [12]

Számos számítógépes képfeldolgozó algoritmus áll rendelkezésünkre ezen jellemzők kinyeréséhez, bár különbözőek, de a feldolgozást tekintve hasonló lépéseket követnek.

A feldolgozás első lépése az előfeldolgozás, mely történhet szűrőkkel vagy akár transzformációkkal is. Az eredményül kapott kép azonban a fontos területek mellett még tartalmaz számunkra annyira nem lényeges képpontokat is. Ezen pontok szétválasztása érdekében használunk szegmentálást.

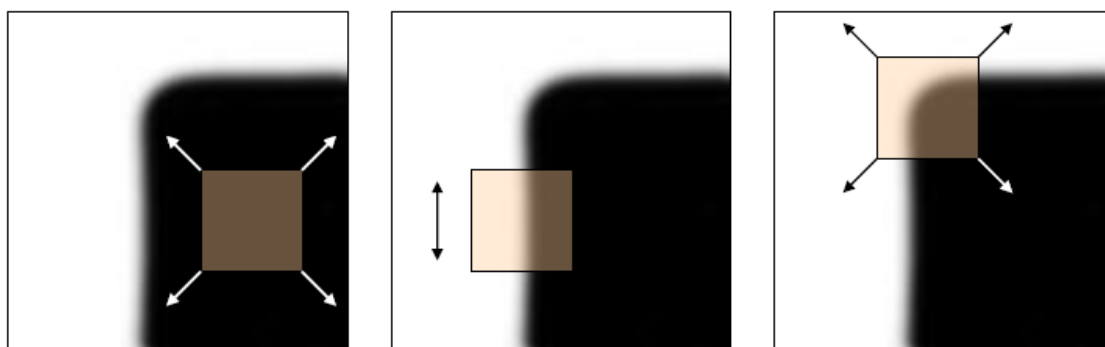
Végezetül a kép mostmár csak a számunkra fontos területet tartalmazza, azonban még mindig túl sok feldolgozandó adattal rendelkezik. Ennek megoldása érdekében „jellemző pontokat” (feature) vonunk ki a képből. [11] Jellemző pontnak a kép egy ”érdekes” részét tekintjük, ennek kinyerését pedig detektálásnak. Felmerül azonban a kérdés, hogy hogyan detektáljunk megbízható jelemzőket.

A detektáló algoritmusok célja, hogy minden képpont esetében egy lokális döntést hozzon, hogy az adott képpont egy jellemző pont vagy sem. A kapott jellemzők az eredeti kép egy részhalmazát alkotják, gyakran izolált pontok vagy egymáshoz kapcsolódó folyamatos görbék formájában. [12]

Különböző képi jellemzők vannak:

- sarok,
- él,
- régió.

Ezek detektálására egy úgynevezett csúszóablakot használunk. Az ablakot több irányba elmozgatva, ahol nagy képfüggvényváltozást észlelünk, ott lesz a sarokpontunk. Lényegében auto-korrelációt számolunk, oly módon, hogy figyeljük az ablak tartalmát egy $[u, v]$ eltolás hatására. Az él detektálásakor hasonlóan járunk el, itt azonban azt figyeljük, hogy a megfelelő irányban történt – e lényegesnek minősíthető változás, ha nem, akkor egy élpontot kaptunk. Régió detektálásakor a fő szempont, hogy bármilyen irányba mozgatva az ablakot ne legyen változás.

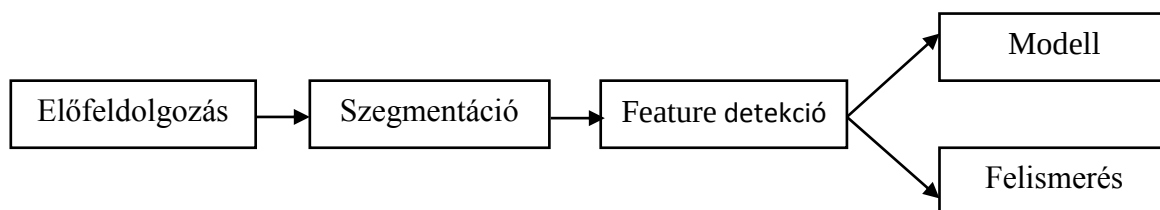


2. Ábra: Régió - , él - és sarokpont detektálás [8]

Ha azonban nemcsak egy képpel, hanem ugyanarról a látványról készült képpárral kell dolgoznunk, fontos, hogy a detektált sarokpontok invariánsak legyenek. Azaz függetlenül attól, hogy az adott kép milyen fotometriai (pl. megvilágítás) és geometriai (pl. forgatás) tulajdonságokkal rendelkezik, a detektált képpontnak megbízhatónak kell lennie.

A kinyert képpontokat egyetlen intenzitás érték jellemzi, ez azonban önmagában nem elég jellegzetes. Ebből az okból kifolyólag a pontok környezetét használjuk, mely már egyedi jellemzőket tartalmaz, azonban ezek a régiók sem skálafüggetlenek.

Jellemző pontok detektálása után a képfeldolgozó algoritmusunk számára kezelhető mennyiségű képi jellemző áll rendelkezésre. Ezek után könnyedén elvégezhető a felismerés is. Azonban nem szabad megfeledkeznünk arról sem, hogy a képi jellemzők kinyerése során hiba is történhet, azaz hamis találatokat is kaphatunk, amelyeket fel kell ismerni és kiszűrni az adathalmazból. [11]



3.Ábra: Képfeldolgozás lépései [11]

Mindezen műveletek elvégzéséhez számos OpenCv algoritmus áll a rendelkezésünkre.

Az érdekes pontokat detektáló algoritmusok különböző szűrőket használnak, melyek sarkokat, éleket, régiókat, vagy ezek kombinációit detektálják.

Jellemző pontokat detektáló algoritmus	Él	Sarok	Folt
Canny szűrő	✗		
Sobel	✗		
Harris & Stephens/Plessey detektor	✗	✗	
SUSAN	✗	✗	
FAST algoritmus		✗	✗
Gauss Laplace algoritmus		✗	✗
Hessian szűrő		✗	✗
MSER			✗
PCBR			✗

4. Ábra: Jellemző pontokat detektáló algoritmusok

Érdekes pontokat detektáló OpenCv algoritmusok

Algoritmus	Megjegyzés
FAST	FastFeatureDetector
STAR	StarFeatureDetector
SIFT	-
SURF	-
ORB	-
BRISK	-
MSER	-
GFTT	GoodFeaturesToTrackDetector
HARRIS	GoodFeaturesToTrackDetector with Harris detector
Dense	DenseFeatureDetector
SimpleBlob	SimpleBlobDetector

5. Ábra: Érdekes pontokat detektáló OpenCv algoritmusok

4.1 SIFT algoritmus

Egyik ilyen híres algoritmus a *SIFT* (*Scale Invariant Feature Transform*), mely skála – invariáns sarokpontokat detektál. Ezzel az algoritmussal könnyedén detektálhatunk sarkokat illetve régiókat. Ezeket a detektált pontokat egyetlen intenzitás érték jellemzi, ezért nem rendelkeznek eléggé jellegzetes tulajdonságokkal. Viszont ha nemcsak a különböző pontokat, hanem azok valamilyen környezetét vizsgáljuk, ez az eredmény már egyedi információnak minősül. Egy adott pont környezetének leírását képleírónak nevezzük. A SIFT leíró egy adott méretű ablakot használva képleírókat határoz meg. A két kép összehasonlítása, ezen megalkotott képleírók összehasonlításán alapszik.

Egy fontos szempontja ennek a megközelítésnek, hogy sok képi jellemzőt nyer ki, amelyek összességében szinte az egész képet lefedik. Egy tipikus 500x500 méretű kép esetében körülbelül 2000 tulajdonságot használ fel. (Ez azonban változhat a két kép tartalmától). A képpontok száma különösen fontos, kiváltképp tárgyak felismerése esetén. Ilyen esetekben a kisebb méretű tárgy, zsúfolt háttérrel való megtalálásához, mindkét képen legalább három helyes képponttalalás szükséges a megbízható azonosítás érdekében.

Képek összehasonlítása és felismerése során, a SIFT algoritmus elsősorban egy adatbázisba menti el a jellemző pontokat. Az új kép jellemző képpontjai egyenként összehasonlítódnak az előbb eltárolt jellemző pontokkal, keresve a találó képpontokat, a két kép jellemző pontjaiból alkotott vektorok, Euklideszi távolsága alapján. [13]

4.2 SURF algoritmus

SIFT algoritmus alapötletéből kiindulva született meg a *SURF* (*Speeded Up Robust Features*) algoritmus. Skála- és forgatás- invariáns jellemző pontokat és leírókat detektál. Megközelíti, sőt talán felül is múlja az előtte született algoritmusokat, hiszen figyelemmel van az ismétlődésekre, megkülönböztetésekre és robusztusságra, gyorsasági szempontból is előnyösebb. A SURF detector a Hessian mátrixon alapul [11, 1] egy nagyon alapvető közelítést használva. [14]. Ugyanakkor a képleíró Haar-waveleten alapuló eloszlást ír le a jellemző képpontok szomszédai között.

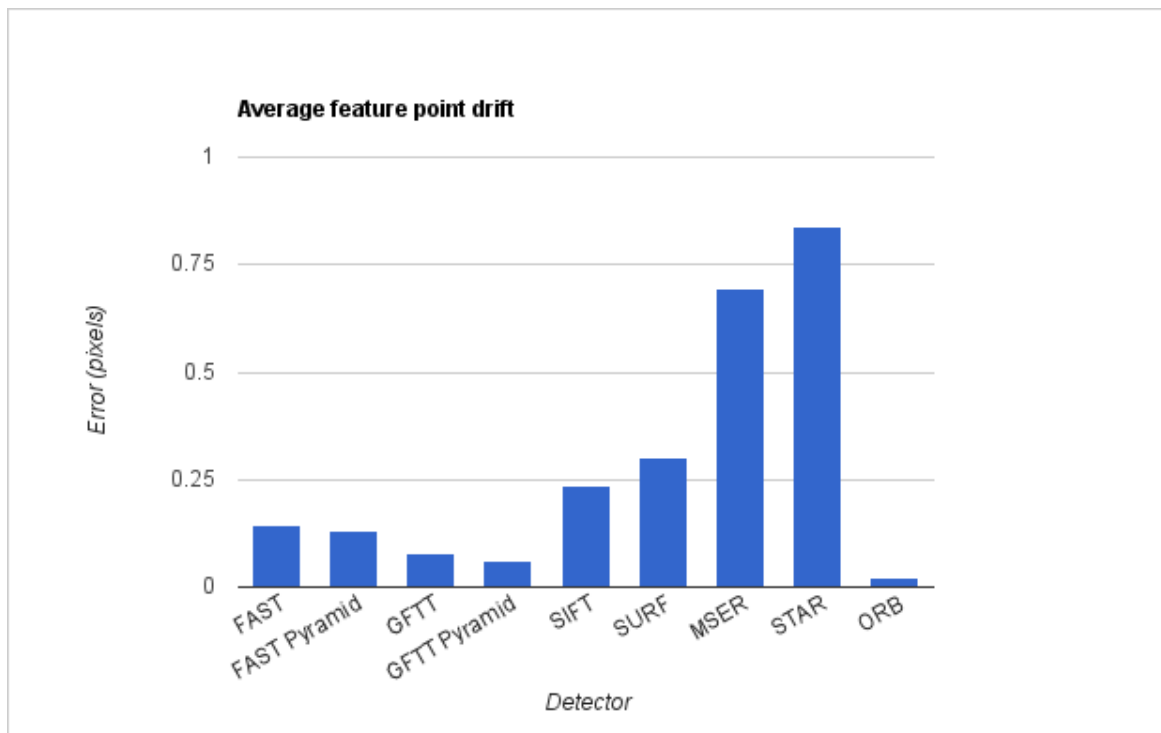
4.3 ORB algoritmus

ORB (Oriented FAST and Rotated BRIEF) a legújabb detektor, forgatás invariáns és fontos tulajdonsága, hogy zajra nem érzékeny. Egy hatékonyabb átdolgozása a SIFT és SURF detektoroknak. FAST algoritmussal detektálja a képpontokat és BRIEF képleírókat használ, igaz ugyan, hogy számos módosítással, a jobb teljesítmény növelése érdekében. A FAST segítségével képpontokat detektálunk, majd a Harris sarokpont detektálást alkalmazva kiválasztunk N darab találó képpontot. Egyetlen probléma, hogy a FAST nem robusztus a forgatást tekintve. Megoldásképpen súlyozott középpontot határoz meg, a lokális sarokpontot tekintve középpontnak. A vektor iránya ebből a pontból a középpont irányába megadja a forgatás irányát. [16]

BRIEF képleírót alkalmaz, mely eléggé újonnan jelent meg. Egyszerű bináris tesztet végez a pixelek között egy adott képterületen belül. Hasonlóan a SIFT-hez robusztus a fényre, a homályosabb képpontokra. Ugyanakkor eléggé érzékeny a síkbeli forgatásra. [15]

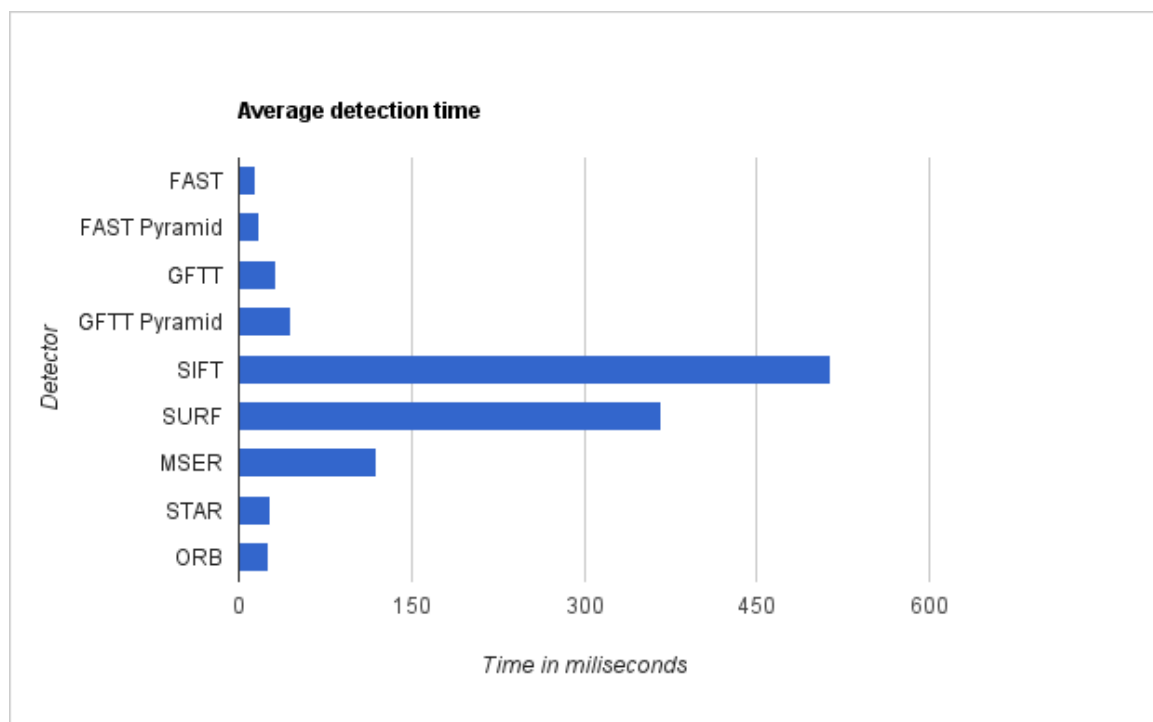
A felismerés szempontjából gyors és pontos eredményeket szolgáltat, ezért is döntöttem ez az algoritmus alkalmazása mellett. [9]

Az alábbiakban több algoritmus egyidejű összehasonlítását láthatjuk a hibaküszöböt illetve a gyorsaságot tekintve.



6. Ábra: Érdekes pontok detektálása

Forrás: <http://computer-vision-talks.com/articles/2011-07-13-comparison-of-the-opencv-feature-detection-algorithms/>



7. Ábra: Átlagos detektálási idő

Forrás: <http://computer-vision-talks.com/articles/2011-07-13-comparison-of-the-opencv-feature-detection-algorithms/>

5. SmartVásárhely applikáció

Napjainkban egyre elterjedtebbé válik az okostelefonok használata. Ha nagyobb közösségben vagyunk, szinte biztos, hogy a környezetünkben van olyan személy, aki kezében egy okostelefont tart. Ha nagyobb utazást tervezünk egy számunkra ismeretlen helyre, az okostelefonunk egyfajta idegenvezetői szolgálatot biztosít. Ezért is választottam a mobil alkalmazást, ezen belül pedig az Android platformot. Széles körben elterjedt, elérhető áron beszerezhető és minden olyan funkcionalitással rendelkezik, amely egy navigáló alkalmazás megvalósításához szükséges. Mint minden okostelefon, biztosít internethozzáférést, GPS vevőt és beépített kamerát. Az OpenCv függvénykönyvtárak is könnyedén használhatók, ennek feltétele csupán az OpenCv Manager telepítése. Az Android platform ugyanakkor lehetőséget biztosít natív kód fejlesztésére is, az úgynevezett Native Development Kitnek köszönhetően.

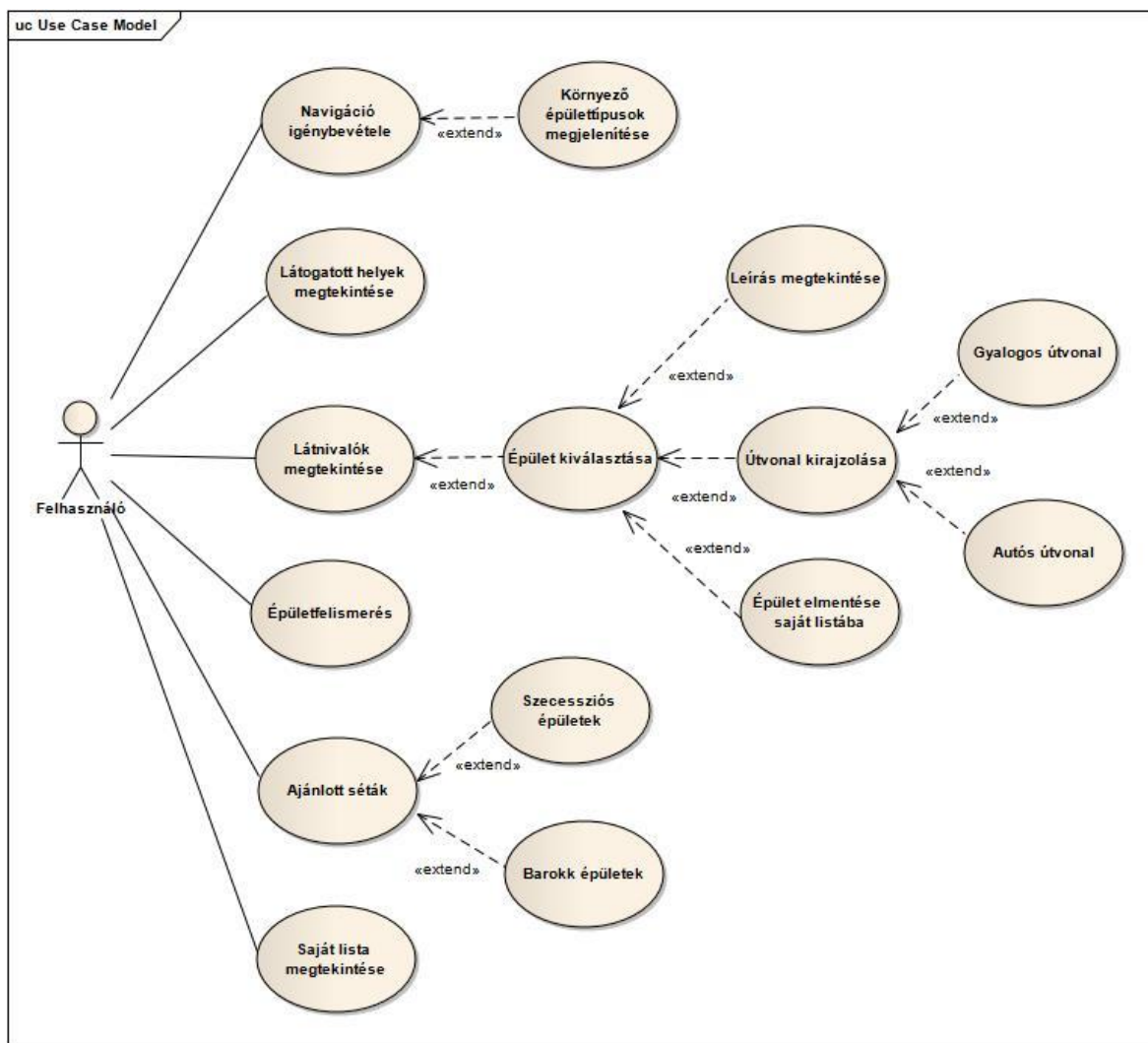
Mindezen funkcionalitásokat figyelembe véve arra a következtetésre jutottam, hogy az Android platform minden olyan lehetőséget biztosít, amely egy navigáló alkalmazás fejlesztéséhez szükséges.

Az alkalmazás működéséhez GPS vevő szükséges. Az internetkapcsolat megléte nem szükségszerű, de jelenlétében több funkcionalitás vehető igénybe. Ugyanakkor az OpenCv függvények használata miatt a készülékre telepíteni kell az OpenCV Manager programot illetve az Offline térkép eléréséhez, szükséges egy osmdroid nevű mappa, mely az offline térképet tartalmazza.

Két részből tevődik össze az applikáció: navigáció, városnézés és képfelismerés.

5.1 UseCase diagram

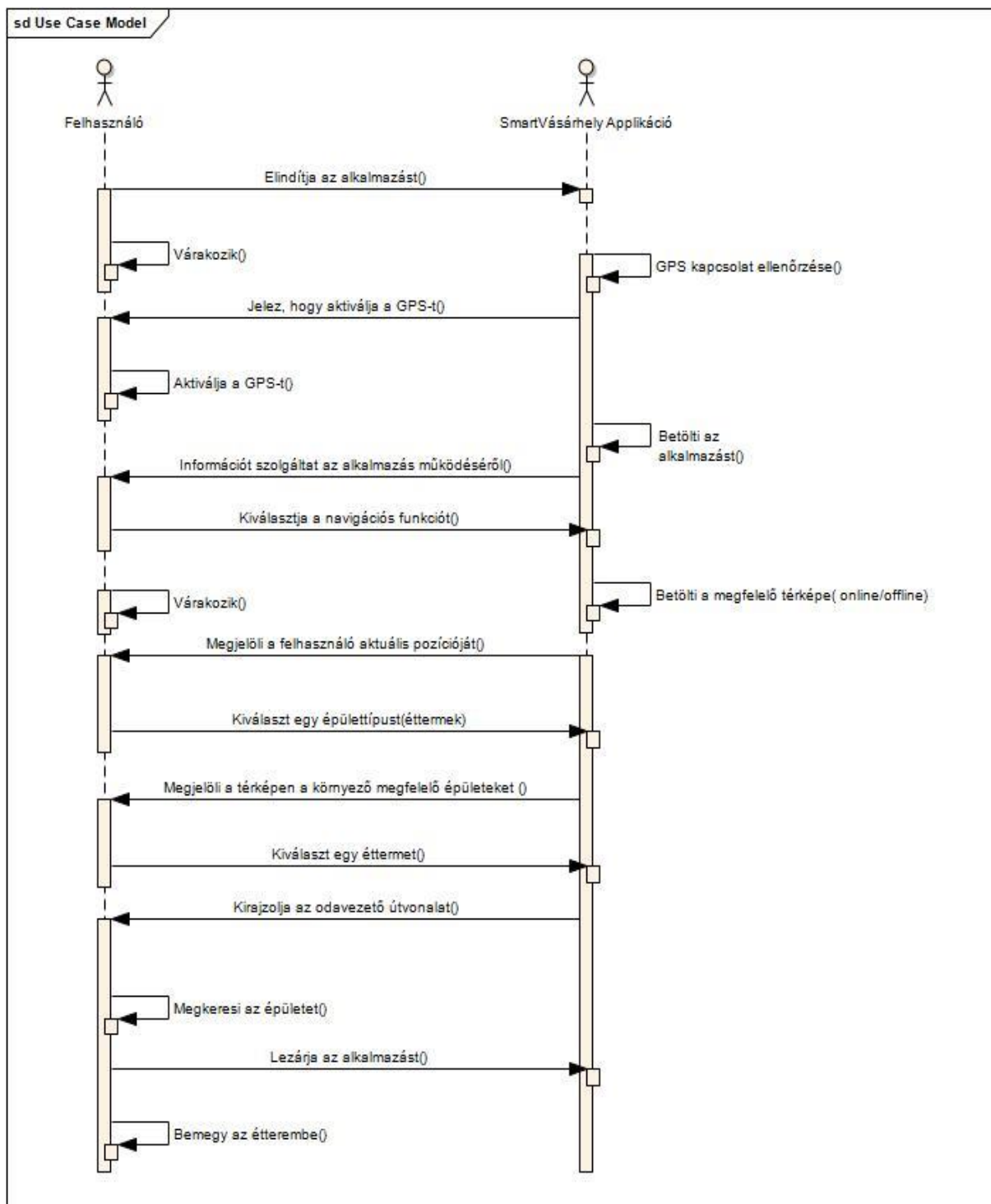
A 8. ábra a felhasználói eseteket szemlélteti.



8. Ábra: UseCase diagram

5.2 Szekvencia diagram

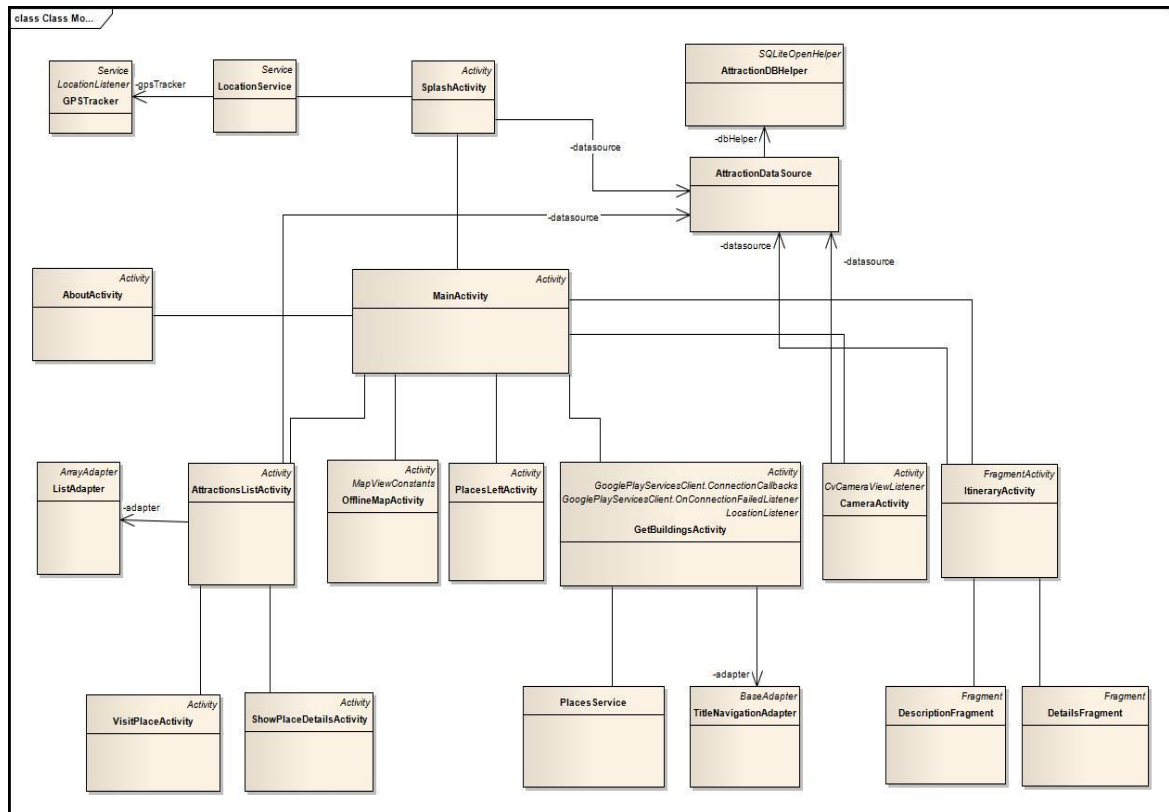
A 9. ábrán látható szekvencia diagram a navigációs funkció igénybevételének lépéseit követi végig.



9. Ábra: Szekvencia diagram

5.3 Osztálydiagram

A 10. ábrán levő osztálydiagram az alkalmazás főbb komponenseit tartalmazza. Egyik fontos komponens az aktuális pozíció lekérése, amely egy szerviz komponens segítségével történik. Másik két fontos komponens a barokk és szecessziós séta megvalósításához szükséges. A séták bővebb információinak megtekintését fragmentek segítségével valósítottam meg.



10. Ábra: Osztálydiagram

5.4 Az alkalmazás bemutatása

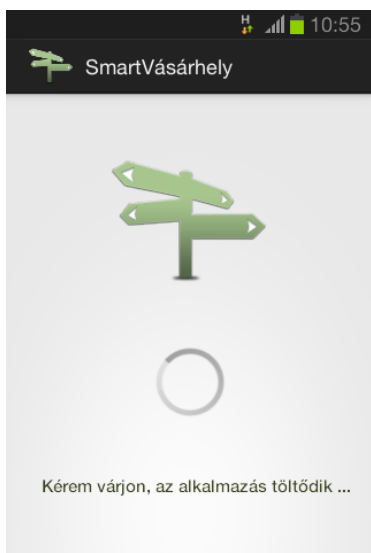
Az alkalmazás két fő részből áll: navigáció és épületfelismerés.

5.4.1 Városnézés, navigáció

Ez az a része az alkalmazásnak, mely a felhasználó városban való útbaigazításáért felel, és gondoskodik arról, hogy minél több információt szolgáltatson neki a város nevezetességeiről. Az alkalmazás mind online, mind pedig offline módban működik, nyilván az offline verzió kevesebb szolgáltatást biztosít.

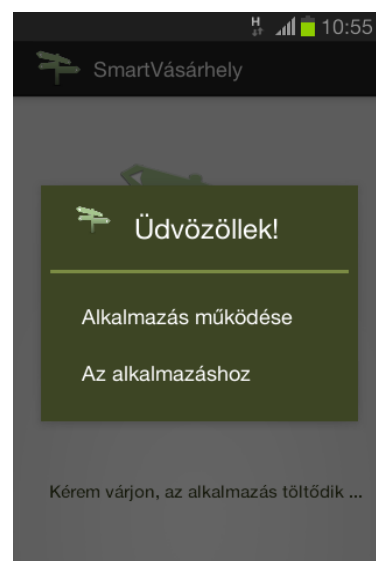
Funkció	Online mód	Offline mód
Felhasználó aktuális pozíciójának megjelölése	✓	✓
Látnivalók listájának megtekintése	✓	✓
Látnivalók bővebb információ	✓	✓
Látnivaló megjelölése a térképen	✓	✓
Látnivalóhoz vezető útvonal kirajzolása	✓	✗
Ajánlott séták térképen való megtekintése	✓	✓
Ajánlott séták galériája	✓	✓
Saját séta felépítése	✓	✓
Épülettípusok kiválasztása és az odavezető útvonal meghatározása	✓	✗
Látogatott helyek elmentése és megjelenítése	✓	✓
Épületfelismerés	✓	✓

11. Ábra: Alkalmazás online és offline funkciói



12. Ábra: Betöltő képernyő

Az alkalmazás betöltődése után üdvözlí a felhasználót, két lehetőséget kínálva fel. Az alkalmazás felhasználói útmutatójának megtekintését, illetve magát az alkalmazásnak a megtekintését.



13. Ábra: Üdvözlő képernyő

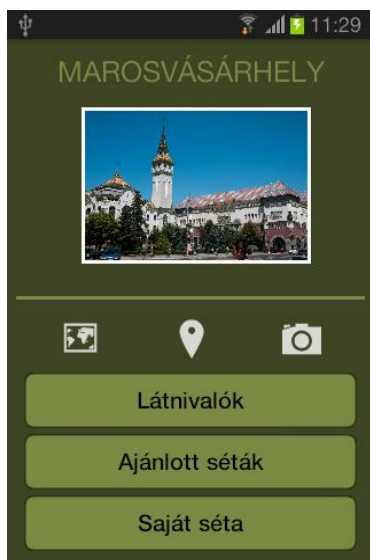


14. Ábra: Felhasználói útbaigazítás

Az alkalmazás egy *Betöltő képernyővel* indul, ahol az alkalmazás logója és a betöltő ikon jelenik meg, mely jelzi, hogy mennyi idő van még hátra az alkalmazás indulásáig. A háttérben betöltődnek az épületek adatai és földrajzi koordinátái, melyeket SQLite adatbázisban tároltam. Ugyanakkor ellenőrzi a GPS vevő státuszát, és ha az inaktív, jelzi a felhasználónak, hogy kapcsolja be.

A *felhasználói útmutató* egy rövidebb útbaigazítást nyújt az alkalmazás használatával és funkcióival kapcsolatban.

Ugyanakkor jelzi a felhasználónak, hogy online illetve offline módban is használhatja az alkalmazást. Nyilván az online verzió több funkciót biztosít.



15. Ábra: Kezdőképernyő

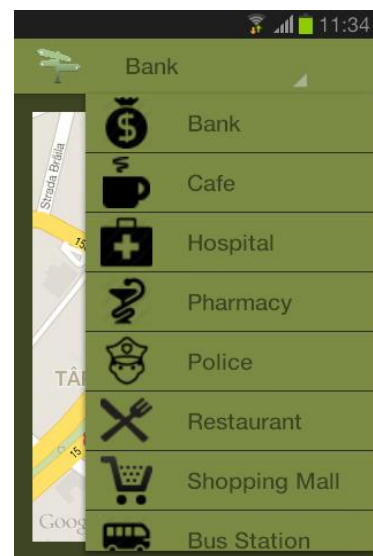
Ezután a *Kezdő képernyő* jelenik meg, mely az alkalmazás főmenüjeként szolgál. Itt vannak felsorakoztatva mindazon funkciók, melyekkel az alkalmazás rendelkezik.

Az első szükséges információ, hogy tudja a felhasználó, hogy hol is tartózkodik jelenleg, és mi van körülötte. Ez az információ a térkép ikonra kattintva érhető el.

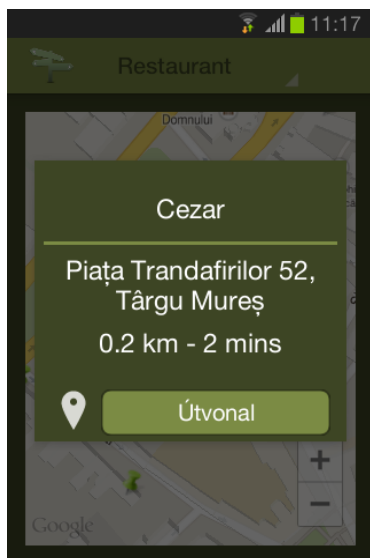
Ez esetben egy lista jelenik meg különböző épülettípusokat felsorakoztatva, melyek közül választhat a felhasználó.

Adott épülettípus kiválasztása esetén megjelennek a térképen a felhasználó környezetében levő, hasonló típusú épületek.

Adott épület kiválasztása esetén az alábbi információkat szolgáltatja az alkalmazás.

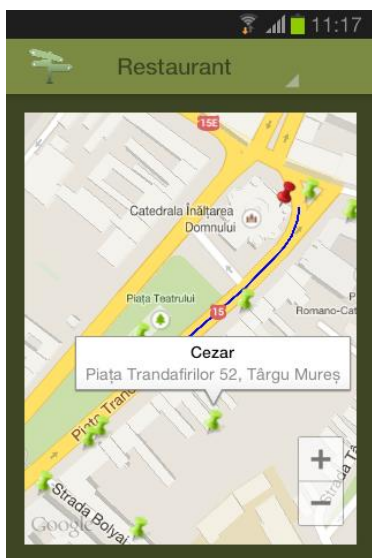


16. Ábra: Környező épülettípusok listája



17. Ábra: Részletek adott épületről

Az épület neve, címe, továbbá a távolság kilométerben és időben is kifejezve. Az odavezető útvonal hossza és ideje a Google API Szerver segítségével határozható meg.



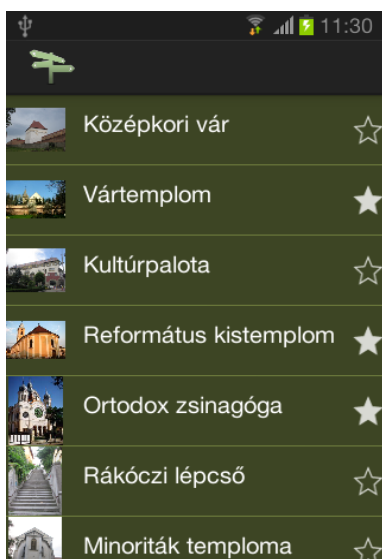
18. Ábra: Útvonal kirajzolás egy kijelölt épülethez

Az alkalmazás ugyanakkor megjegyzi azt az útvonalat, amelyet a felhasználó korábban már érintett, így bármikor visszatalál egy előző helyhez. Ennek megtekintését a középső ikonra kattintva teheti meg.

Az alkalmazás ezen helyek elmentéséhez egy Android szervizt használ, mely az alkalmazás indulásakor aktiválódik és adott időközönként lekéri az aktuális pozíciót.



19. Ábra: Érintett útvonal



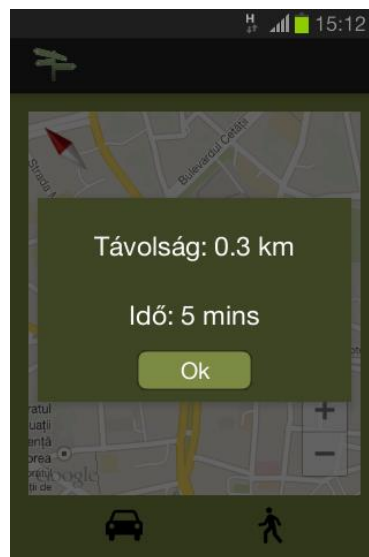
20. Ábra: Látnivalók listája

A térképen piros ikonnal kerül bejelölésre az aktuális pozíció, és zölddel a felhasználó által kiválasztott épület, továbbá az odavezető útvonal is. A térkép implementálásakor a Google térképet használtam.

Az alkalmazás keretén belül a *Látnivalókra* kattintva, egy listában felsorakoztatásra kerülnek a város nevezetességei. Minden listaelem három dolgot tartalmaz: egy kép, a nevezetesség neve és egy megjelölésre alkalmas csillag. A csillag állapota jelzi, hogy a felhasználó hozzáadta-e az adott épületet egy saját igények szerint felépíthető listához.



21. Ábra: Látnivaló - bővebb információkkal



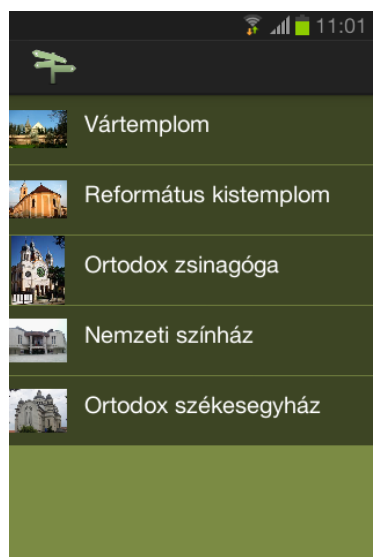
22. Ábra: Részletek az útvonallal kapcsolatosan



23. Ábra: Útvonal adott látnivalóhoz

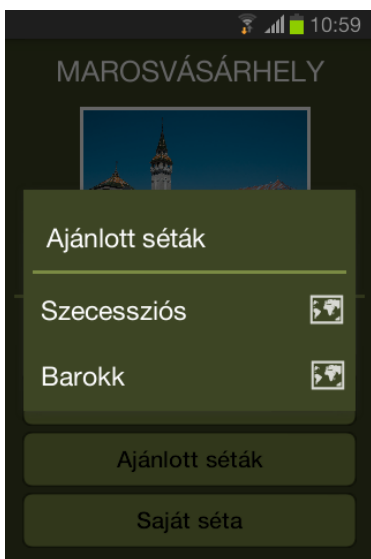
Egy listaelemre kattintva a fent látható, első képernyőre irányít az alkalmazás. A felhasználó kérheti a kiválasztott épülethez vezető autó-, vagy éppen gyalogos út megjelenítését. Ez esetben jelzi azt is, hogy milyen távolra van és mennyi időt vesz igénybe az odajutás. Továbbá egy rövidebb leírást is olvashat az aktuális épületről.

Az útvonal lekérése a Google Places Api szervernek küldött kérés eredményeként választódik ki.



24. Ábra: Saját séta

A megjelölt épületek a baloldali képen látható módon jelennek meg, ha a felhasználó a *Saját séta* gombra kattint. Bármelyik listaelemet kiválasztva, ugyanúgy egy térkép töltődik be, mint a *Látnivalók* képernyőn, és a listaelem ugyanazokkal a funkcionalitásokkal rendelkezik ebben az esetben is.

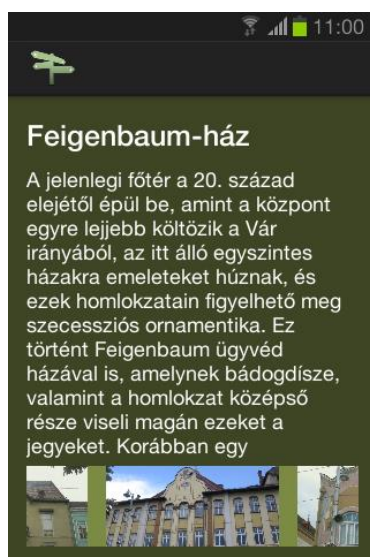


25. Ábra: Ajánlott séták

Két sétát tartalmaz az alkalmazás, mely Marosvásárhely barokk és szecessziós épületeit mutatja be. Ezek az *Ajánlott séták* menüpont alatt érhetőek el, és egy képgalériaként tekinthetőek meg.



26. Ábra: Szecessziós séta része



27. Ábra: Leírás a kiválasztott épületről



28. Ábra: Sétához tartozó épület

A térkép ikonra kattintva megjelennek az adott sétához tartozó épületek földrajzi pozíciói. Ezt szemlélteti az első ábra.

Egy adott séta esetében minden sétához tartozó épülethez egy kép és egy leírás társul, mely a második ábrán látható.

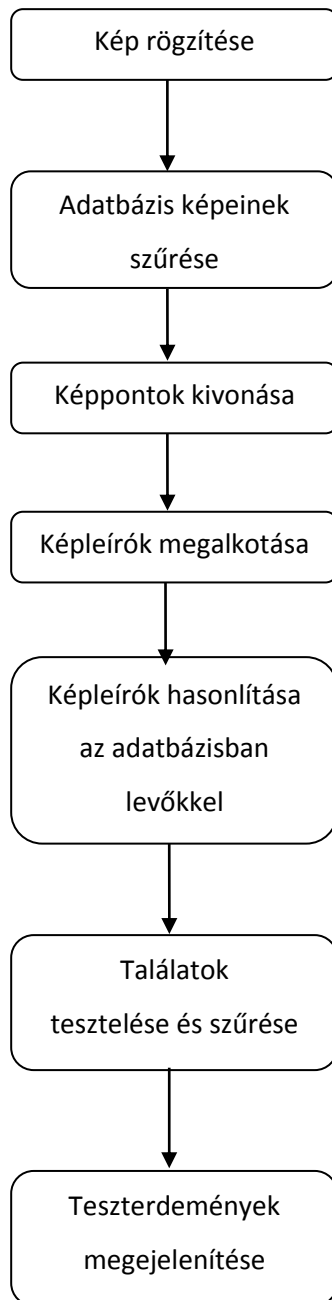
Az épülethez tartozó leírást a képre kattintva jeleníthetjük meg, ezt szemlélteti az utolsó ábra.



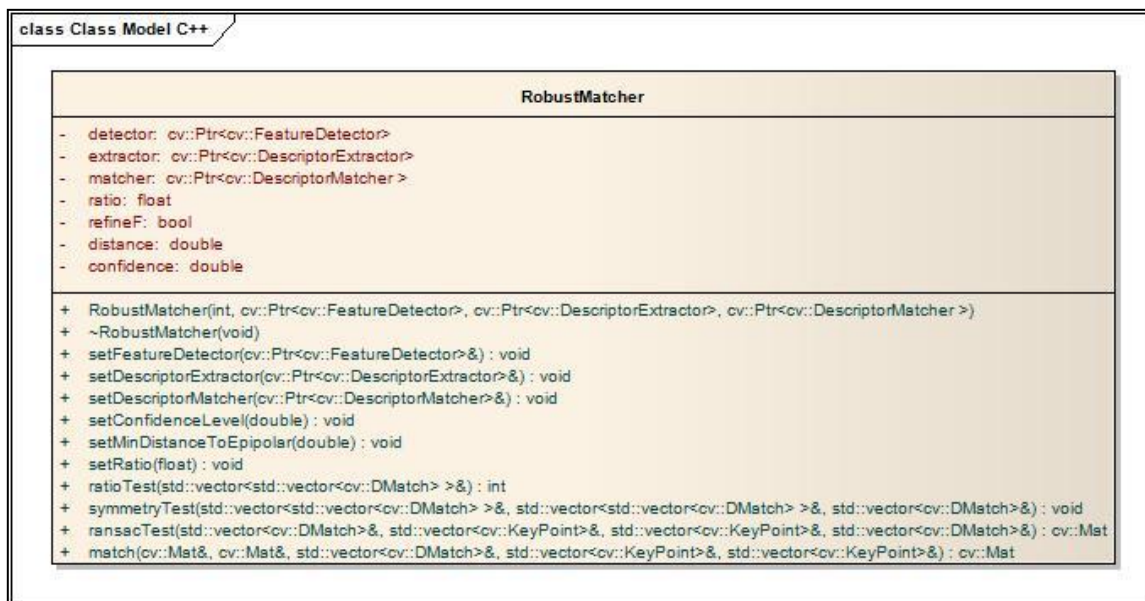
Az alkalmazás offline módban is használható. Ebben az esetben nem a Google térképet implementálja, hanem a *MOBAC* (*Mobile Atlas Creator*) programcsomaggal előállított offline térképet használja fel, amely rajta kell legyen a felhasználó készülékén.

29. Ábra: Offline térkép

5.4.2 Képfelismerés



30. Ábra: Képfelismerés lépései



31. Ábra: Képfelismerés, C++ osztály

Egy adott régió belül levő, különböző épületek, bizonyos történelmi időkeretek között való felépítése, magával hordozza az adott időszakra jellemző építészeti stílus jegyeit. Így szinte kétségtelenül kijelenthetjük, hogy az épületek egyaránt rendelkeznek egyedi és közös tulajdonságokkal. Ezért előfordulhat, hogy egymáshoz közeli épület esetén a felhasználónak fejtörést okoz, hogy melyik is a meglátogatni kívánt épület, vagy akár egy neki tetsző épület esetén kíváncsi lehet arra, hogy melyik épület is ez.

Ezt a célt szolgálja a képfelismeréses része az alkalmazásnak. Azonban a vizsgálandó képek száma eléggé tágas kört alkot, ezért az algoritmusnak hosszas dolga lenne. Ezen képek számának a csökkentése céljából, egy szűrést alkalmaztam. Mivel egy adott épületet és ugyanúgy a felhasználó helyzetét is meghatározza a földrajzi szélességi és hosszúsági koordináta, ennek a jellemzőnek a használata igencsak leszűkíti a vizsgálandó épületek körét. A felhasználó aktuális földrajzi pozíciójából kiindulva csak egy adott küszöbértéken belül eső képek kerülnek összehasonlításra.

A képek összehasonlításához a BruteForceMatcher [10] algoritmust használtam. Ennek a segítségével, az előzőleg kiszűrt képek mindenikére kiszámítom a találó képpontok számát, majd ezek közül a maximum fogja megadni az eredményt. A találó képpontok számát három szűrési eredmény után véglegesítem: arány - , szimmetria – és ransac szűrő.

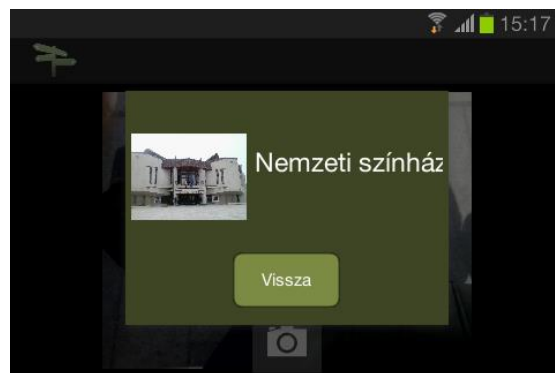
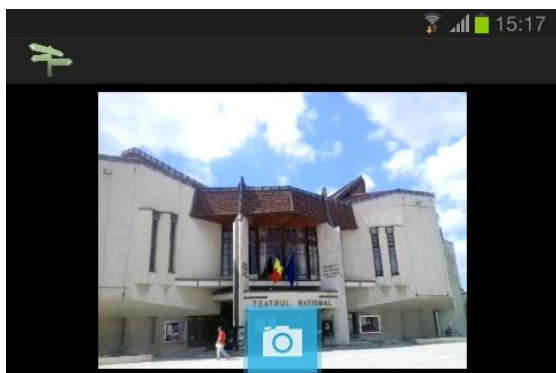
Az arányszűrő esetében mindenik képpontra két találat van, mely a leírójuk alapján kiválasztott két legnagyobb távolság értéke. Akkor minősül egy eredmény helyes találatnak, ha a két legjobb távolság közötti arány nagyobb, mint egy előzőleg meghatározott küszöbérték.

Szimmetriaszűrő használata esetén adott két hasonlítandó kép. Kezdetben az első képet viszonyítom a másodikhoz, majd fordítva. Ezáltal két kép-összehasonlítást végzek, ha ezek eredményei megegyeznek, akkor eredményhez jutottam.

Ransac szűrő működési elve, hogy a jó találatok halmazából véletlenszerűen kiválaszt nyolc értéket, ebből egy mátrixot alkot, ezt használja viszonyítási pontnak a további összehasonlítások során. Amely találatok közel esnek ehhez a ponthoz, azokat helyes eredménynek tekintjük. [10]

Minden elfogadott találatot tárolok, majd ezek közül a legjobbnak minősülő lesz a végeredmény.

A hasonlítások külön szálon hajtódnak végre, a ThreadPool nevű szálcsomagot implementálva, mely a készülék teljesítménye alapján annyi szálát futtat egyszerre, amennyit az megbír.



32 - 33. Ábra: Épületfelismerés képernyők

Az épületfelismerési szolgáltatást a kamera ikonra kattintva lehet igénybevenni. A kamerára kattintva az alkalmazás rögzíti a képet, majd találat következtében kiadja az eredményt, ellenkező esetben jelzi, hogy nincs találat.

6. Alkalmazás kiértékelése

6.1 Felhasználók által kitöltött kérdőív

Az alkalmazás összképéről alkotott megfelelő visszajelzés érdekében egy 18 kérdésből álló kérdőívet állítottam össze, melyben általános kérdéseket tettem fel az alkalmazás kinézetével, használhatóságával, átláthatóságával és hasznosságával kapcsolatban. A kérdőív megszerkesztésében a Google Docs volt a segítségemre, melyet összesen 18 személy töltött ki.

A kérdőív online változata az alábbi linken érhető el:

https://docs.google.com/forms/d/1aZH7671pK_tnRhuZ2Ljql6B1lQwd2-x_LyBpAyo47ek/viewform?usp=send_form

A kérdőív nyomtatott változata:

1. Általános kérdések					
	1	2	3	4	5
Milyen összbenyomást keltett az alkalmazás?					
Milyennek találd az alkalmazás kinézetét (Design)?					
Egységes és összehangolt-e az alkalmazás vizuális felülete, grafikája?					
Mennyire egyértelműek és érthetőek a használati útmutatók?					
Jól átláthatóak-e a rendszerüzenetek (Betűméret, betűszín, stb.) ?					
Mennyire találja hasznosnak az alkalmazást?					
Adott esetben használná-e az alkalmazást?					

2. Az alkalmazás kezelhetősége					
	1	2	3	4	5
Mennyire könnyen kezelhető az alkalmazás?					
Hamar elsajátítható-e az alkalmazás kezelése?					
Elégséges információt szolgáltat-e az alkalmazás a felhasználói útmutatóban?					
Egyértelmű-e a különböző funkcionálisok (navigáció, látnivalók, stb.) szerepe?					
Elegendő információt nyújt-e az alkalmazás, mint városnéző applikáció?					

3. Navigációs funkció					
	1	2	3	4	5
Mennyire segítette elő az alkalmazás bizonyos épületek megtalálását?					
Elégséges információkat szolgáltatott-e az alkalmazás a navigálást illetően?					
Mennyire hasznos az épületfelismerés navigáció szempontjából?					
Mennyire bőséges a látnivalók köre?					

4. Melyik az a funkcionális amelyet leginkább kedvelt?	
Navigáció	
Séták	
Épületfelismerés	

5. Megjegyzései, javaslatai, tanácsai az alkalmazást illetően?

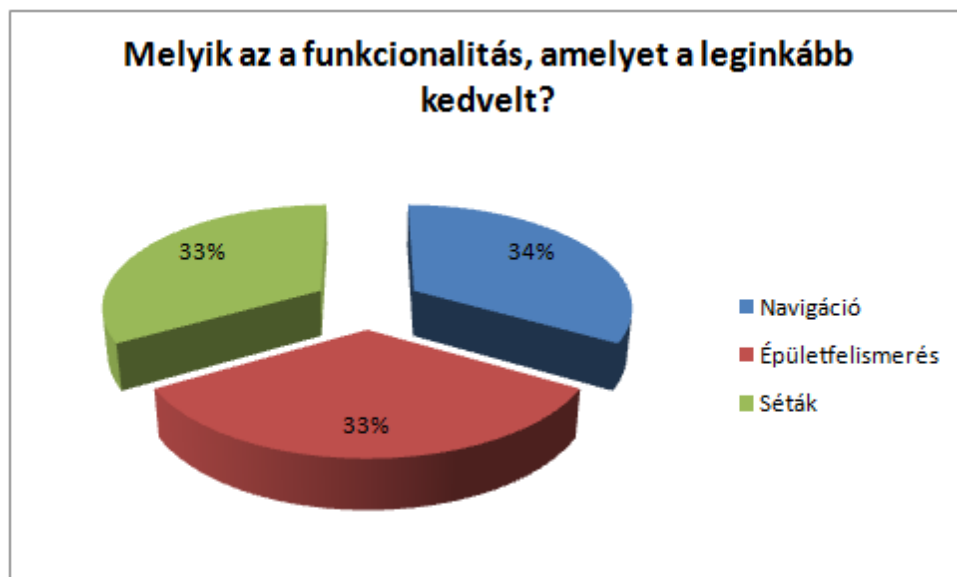
34. Ábra: Felhasználói kérdőív

6.2 Kérdőívek eredményeinek értékelése

1. Általános kérdések						
	1	2	3	4	5	Átlagérték
Milyen összbenyomást keltett az alkalmazás?	0 (0%)	0 (0%)	0 (0%)	7 (39%)	11 (61%)	4.61/5 (92,2%)
Milyennek találod az alkalmazás kinézetét (Design)?	0 (0%)	0 (0%)	0 (0%)	3 (17%)	15 (83%)	4.83/5 (96,6%)
Egységes és összehangolt-e az alkalmazás vizuális felülete, grafikája?	0 (0%)	0 (0%)	0 (0%)	6 (33%)	12 (67%)	4.67/5 (93,4%)
Mennyire egyértelműek és érthetőek a használati útmutatók?	0 (0%)	0 (0%)	2 (11%)	6 (33%)	10 (56%)	4,45/5 (89%)
Jól átláthatóak-e a rendszerüzenetek (Betűméret, betűszín, stb.) ?	0 (0%)	0 (0%)	2 (11%)	5 (28%)	11 (61%)	4,5/5 (90%)
Mennyire találja hasznosnak az alkalmazást?	1 (6%)	0 (0%)	0 (0%)	4 (22%)	13 (72%)	4,54/5 (90,8%)
Adott esetben használná-e az alkalmazást?	0 (0%)	1 (6%)	0 (0%)	3 (17%)	14(78%)	4,7/5 (94%)

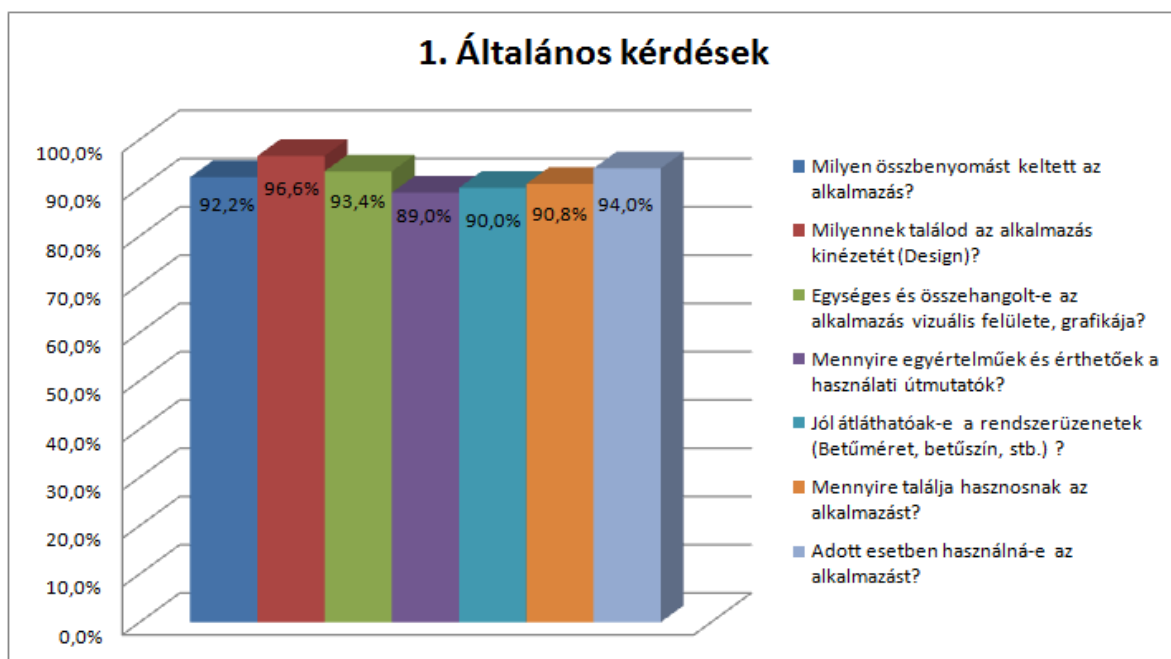
2. Az alkalmazás kezelhetősége						
	1	2	3	4	5	Átlagérték
Mennyire könnyen kezelhető az alkalmazás?	0 (0%)	0 (0%)	4(22%)	4 (22%)	10 (56%)	4,34/5 (86,8%)
Hamar elsajátítható-e az alkalmazás kezelése?	0 (0%)	1 (6%)	3 (17%)	3 (17%)	11 (61%)	4,36/5 (87,2%)
Elégséges információt szolgáltat-e az alkalmazás a felhasználói útmutatóban?	0 (0%)	0 (0%)	1 (6%)	4 (22%)	13 (72%)	4,66/5 (93,2%)
Egyértelmű-e a különböző funkcionálisok (navigáció, látnivalók, stb.) szerepe?	0 (0%)	0 (0%)	1 (6%)	6(33%)	11 (61%)	4,55/5 (91%)
Elegendő információt nyújt-e az alkalmazás, mint városnéző applikáció?	0 (0%)	0 (0%)	2 (11%)	6(33%)	10 (56%)	4,45/5 (89%)

3. Navigációs funkció						
	1	2	3	4	5	Átlagérték
Mennyire segítette elő az alkalmazás bizonyos épületek megtalálását?	0 (0%)	1 (6%)	2 (11%)	6 (33%)	9 (50%)	4,27/5 (85,4%)
Elégséges információkat szolgáltatott-e az alkalmazás a navigálást illetően?	0 (0%)	0 (0%)	1 (6%)	6 (33%)	11 (61%)	4,55/5 (91%)
Mennyire hasznos az épületfelismerés navigáció szempontjából?	0 (0%)	2 (11%)	4 (22%)	4 (22%)	8 (44%)	3,96/5 (79,2%)
Mennyire bőséges a látnivalók köre?	0 (0%)	1 (6%)	4 (22%)	3 (17%)	10 (56%)	4,26/5 (85,2%)

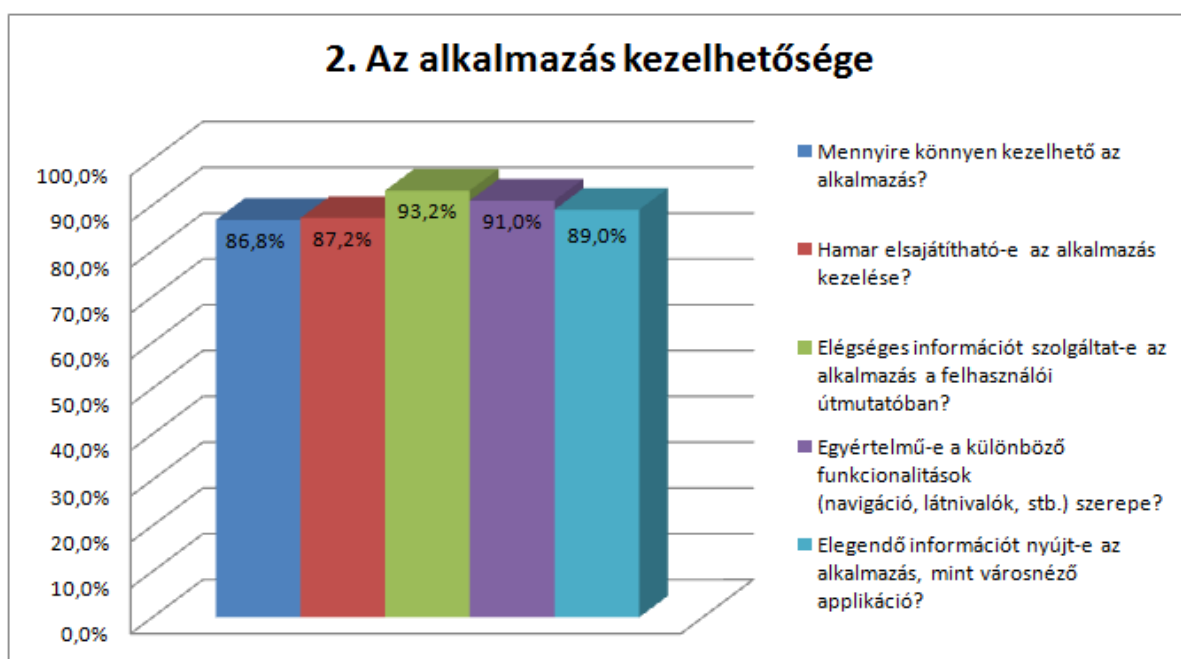


35. Ábra: Különböző funkciók kedveltsége

Az alábbi diagramot megfigyelve, a felhasználók többségének (92,2%) jó összbenyomást keltett az alkalmazás. A használati útmutatók esetében többen jelezték, hogy nem voltak elég érthetőek az útbaigazítások (89%), de ennek ellenére azonban sokan úgy vélekednek, hogy szívesen használnák adott helyzetben az alkalmazást (94%).

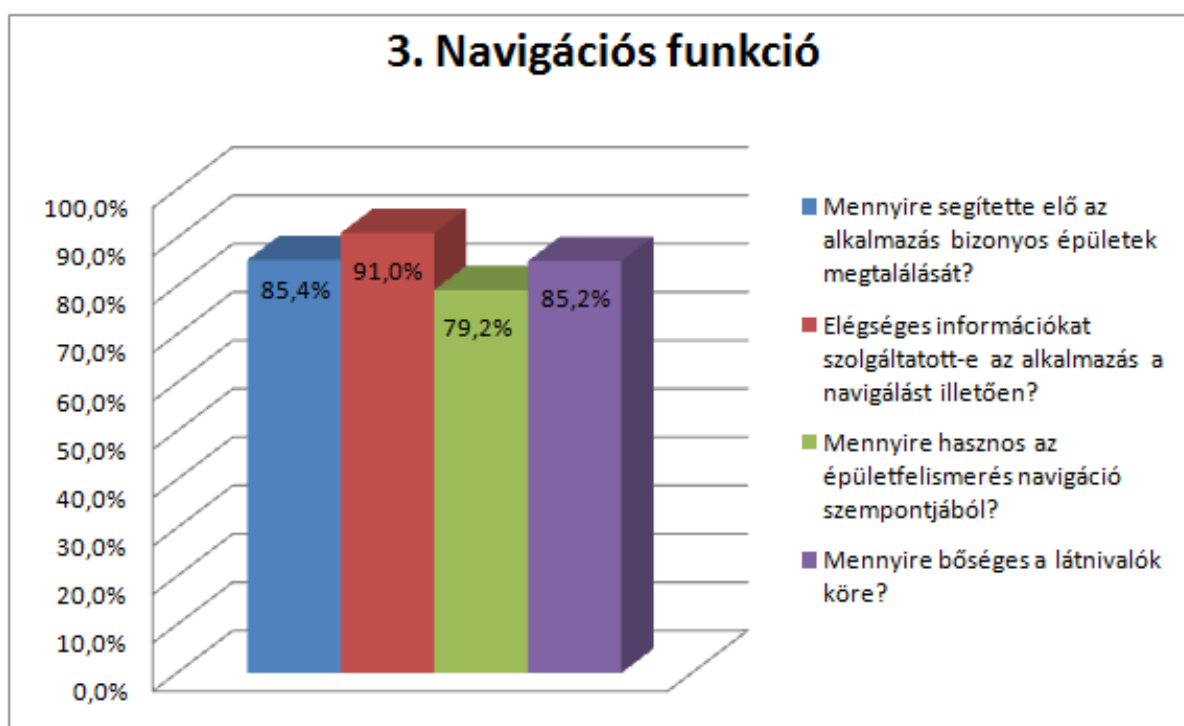


36. Ábra: Általános kérdések összesítése



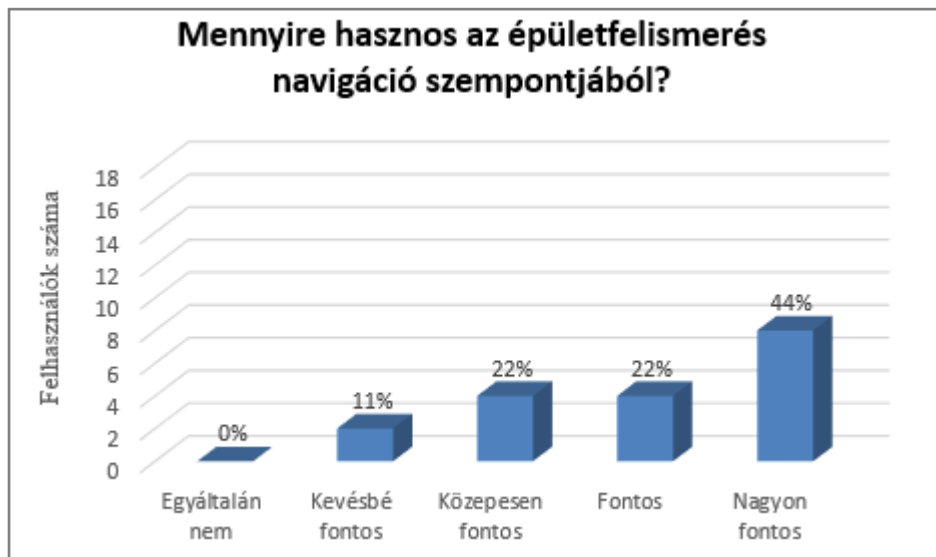
37. Ábra: Az alkalmazás kezelhetősége

A 38-as ábra diagramja az alkalmazás kezelhetőségét szemlélteti, melyet elemezve arra a következtetésre jutottam, hogy a felhasználók zömének sikerült boldogulniuk vele. Annak ellenére, hogy úgy vélik, hogy az alkalmazás elegendő információt nyújt a felhasználói útmutatóban (93,2%), mégis a kezelhetőség elsajátításának gyorsaságát nem vélik annyira gyors folyamatnak (87,2%) ugyanakkor az alkalmazás kezelhetőségének mércéje 86,8%.



38. Ábra: Navigációs funkció értékelése

Navigációs funkciót alkalmazó felhasználók 85,4%-a véli úgy, hogy nagymértékben elősegítette őket az alkalmazás bizonyos épületek megtalálásában. A Látnivalók körét viszonylag bőségesnek vélik (85,2%), azonban az épületfelismerés hasznosságát kevésbé látják potenciális tényezőnek (79,2%).



39. Ábra: Épületfelismerés hasznossága

Megfigyelhető, hogy az épületfelismerést a felhasználók 11% -a találta Kevésbé fontosnak, 22%-a Közepesen fontosnak, 22%-a Fontosnak, míg 44%-a, vagyis szinte fele Nagyon fontosnak vélte, mégis a három funkcionalitás közül, csupán 33%-nak volt ez a legkedveltebb.

7. Következtetések és továbbfejlesztési lehetőségek

A városnéző alkalmazások által nyújtott szolgáltatásokat tanulmányozva és a napjainkban egyre jobban fejlődő turizmust figyelembe véve azt a következtetést vonhatjuk le, hogy azon turisták számára, akik egy ismeretlen települést szeretnének megtekinteni előnyükre válik az ilyen típusú alkalmazások használata.

Az ehhez hasonló szoftverek nyújtotta lehetőségek nemcsak a tájékozódást könnyítik meg, de általuk időt is spórolhatunk, mely következtében lehetőségünk adódik még több látványosság megtekintéséhez.

A romániai városokra kifejlesztett ilyen típusú androidos alkalmazások száma elég csekélynek minősül az androidos mobilalkalmazások piacán, így véleményem szerint megéri időt és energiát befektetni ilyen típusú szoftverek fejlesztésébe.

A felhasználói élményeszt eredményeit figyelembe véve, számos felhasználó úgy vélekedett, hogy az alkalmazás, bizonyos élethelyzetekben (pl. idegen városban való eligazodás), elősegíti és megkönnyíti a tájékozódást, ezen helyzetekben pedig szívesen használnák is. Mindezeket szem előtt tartva az alkalmazás hiánypotló munkaként is tekinthető.

Továbbfejlesztési lehetőségként, újabb funkciókkal lehetne a projektet bővíteni:

- többnyelvűsítés,
- lehetőség biztosítása, hogy a felhasználó is bővíthesse a nevezetes épületek adatbázisát,
- séták bővítése
- épületfelismerés finomítása, gyorsítása.

Hivatkozások

- [1] <https://play.google.com/store/apps/details?id=com.tripadvisor.android.apps.cityguide.catalog> – 2014. 03. 26
- [2] <https://play.google.com/store/apps/details?id=com.triposo.droidguide.world> – 2014.03. 26
- [3] <https://play.google.com/store/apps/details?id=kz.points.cityguide> – 2014. 03. 26
- [4] <https://play.google.com/store/apps/details?id=com.google.android.apps.maps> – 2014.03. 26
- [5] <https://developers.google.com/maps/documentation/android/location>
- [6] <http://ijcsi.org/papers/IJCSI-9-1-2-237-242.pdf>
- [7] Kató Zoltán, Czúni László, Számítógépes látás, Typotex, 2011.
- [8] <http://www.inf.u-szeged.hu/~kato/teaching/IpariKepfeldolgozas/04-Correspondence.pdf>
- [9] Szigeti Péter: *Android alapú pénzfelismerés látássérültek számára*, Sapientia Erdélyi Magyar Tudományegyetem – Erdélyi Tudományos Diákköri Konferencia (ETDK), Kolozsvár, 2013. Május 23 -26.
- [10] Robert Laganieri. (2011) *OpenCV 2 Computer Vision Application Programming Cookbook*, Packt Publishing.
- [11] Nagy István: Méregző harcanyag felhő detektálásának vizsgálata digitális képfeldolgozás segítségével, 2010
- [12] http://en.wikipedia.org/wiki/Feature_detection_%28computer_vision%29
- [13] David G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints, 2004
- [14] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool: SURF: Speeded Up Robust Features
- [15] http://www.vision.cs.chubu.ac.jp/CV-R/pdf/Rublee_iccv2011.pdf
- [16] http://docs.opencv.org/trunk/doc/py_tutorials/py_feature2d/py_orb/py_orb.html