Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava 4

Zábavný systém pre spolucestujúcich v automobile

Tímový projekt

Projektová dokumentácia

Vedúci tímového projektu: Ing. Vanda Benešová, PhD.
Členovia tímu:
Bc. Patrik Polatsek
Bc. Martin Petluš
Bc. Peter Hamar
Bc. Róbert Sabol
Bc. Jakub Mercz
Bc. Lukáš Sekerák

Názov tímu: Tím č. 3 (Carlos)
Web: http://labss2.fiit.stuba.sk/TeamProject/2013/team03is-si/
Kontakt: team03.1314@gmail.com
Akademický rok: 2013/2014

Obsah

1. Úvod

Táto dokumentácia vznikla v rámci predmetu Tímový projekt pri práci na projekte s názvom Zábavný systém pre spolucestujúcich v automobile. Spomínaný projekt je riešený tímom č. 3 - Carlos. V tejto kapitole sa nachádza zadanie projektu Carlos, jeho základný koncept a ciele.

1.1 Zadanie projektu

Zábavné a informačné systémy postupne stále viac prenikajú do nášho bežného života, automobily nevynímajúc. Obohatená realita (tiež: rozšírená realita, angl. AR – augmented reality) je jednou z najmodernejších, v súčasnosti vyvíjaných metód interakcie. Jedná sa o rozšírenie vnemu skutočného reálneho sveta o virtuálne prvky vo forme textu, obrazovej, akustickej informácie a pod.

V rámci tímového projektu bude navrhnutý a implementovaný systém, ktorý pomocou kamery sníma okolie za jazdy automobilom, metódami počítačového videnia analyzuje vizuálnu informáciu spolu s GPS informáciou a na základe tejto analýzy vie poskytnúť napr. informácie zaujímavé pre turistu. Malý LED projektor spolu s transparentnou projekčnou fóliou umožňujú prezentovať vygenerované informácie (či už grafické alebo textové) tak, aby pre spolusediaceho bol vytvorený systém obohatenej reality. Interakcia s týmto systémom bude realizovaná predovšetkým pomocou mobilného telefónu, prípadne ak to hardvér umožňuje, priamou interakciou dotykom na fóliu.

1.2 Motivácia a základný koncept projektu

Zábavné a informačné systémy sa stávajú čoraz častejšie súčasťou nášho života. Jedným z najmodernejších a pre človeka najprirodzenejších spôsobov interakcie je obohatená realita, kde reálny svet je doplnený o obrazové a textové virtuálne prvky.

Cieľom tohto projektu je zmeniť bočné okienko automobilu na transparentnú projekčnú plochu, pomocou ktorej bude reálny svet dopĺňaný o ľubovoľné virtuálne informácie so zábavným i náučným zámerom. Na okienku auta nám tak vznikne rozšírená realita, ktorá nás môže informovať o našom bezprostrednom okolí.

Na prezentáciu vygenerovaných informácii použijeme malý LED projekt spolu s transparentnou projekčnou fóliou. Aplikáciu bude môcť používateľ ovládať veľmi jednoducho – pomocou hlasu a gest s mobilným telefónom.

Obr. 1. Základný koncept riešenia

Pri vývoji tohto programu využijeme programovacie jazyky C/C++, Javu pre mobilnú platformu Android, knižnicu OpenCV a OpenGL, či zariadenie Kinect.

Výsledkom tohto projektu bude prototyp interaktívneho systému pre obohatenú realitu pre spolucestujúceho v automobile.

Podstatou našej aplikácie bude rozoznávať zaujímavé objekty nasnímané kamerou na aute pomocou gps polohy auta a našej internej databázy objektov záujmu. K takto detegovanému objektu doplní do reálnej scény nielen jeho názov, ale aj iné zaujímavé textové i obrazové informácie. Napr. pri pamiatkach zobrazí otváracie hodiny, výšku vstupného a fotografie. Používateľ si na začiatku zvolí kategórie, na ktoré chce byť upozorňovaný ako napr. pamätihodnosti, reštaurácie a kaviarne. Výhodou našej aplikácie bude aj to, že nebude vyžadovať pripojenie na internet.

Aplikácia ale nebude slúžiť len ako vzdelávací systém, ale aj ako zábavný systém pre deti. Jednou z hier, o ktoré bude naša aplikácia doplnená, bude spočívať v uhádnutí názvu objektu, príp. v zodpovedaní otázky, ktorá sa bude týkať tohto objektu.

Naša aplikácia bude pozostávať z viacerých modulov. Mobilné zariadenie bude zaznamenávať naše povely a gestá, zároveň bude zisťovať našu aktuálnu gps polohu. Následne sa spracuje snímka získaná z kamery. Aplikácia bude rozoznávať objekty s využitím metód počítačového

videnia s knižnicou OpenCV. Kinect nám bude slúžiť na detekciu polohy hlavy. Informácie o polohe zdetegovaného objektu a polohy hlavy sa využijú na výpočet, na ktorej pozícii projekčnej fólie sa majú spolusediacemu v aute zobraziť virtuálne prvky. Na samotné vykreslenie informácií, či už textových alebo obrazových využijeme OpenGL, ktoré pomocou projektoru zobrazíme na bočné sklo.

Projekt je podporovaný grantom Nadácie Volkswagen.

Téma zahrňuje i vývoj:

- Výskum a vývoj pokročilých metód počítačového videnia
- Výskum a vývoj pokročilých pokročilých metód počítačovej grafiky
- Výskum a vývoj moderných metód interakcie

1.3 Ciele projektu

Primárnym cieľom projektu je vytvoriť funkčný prototyp interaktívnej aplikácie obohatenej reality, ktorá na bočné sklo automobilu zobrazí doplnkové informácie vzdelávacieho a zábavného charakteru.

Medzi základné ciele u úlohy projektu Carlos patrí:

- presné rozpoznávanie objektov
- vykonávanie aplikácie v reálnom čase
- vypočítanie polohy pre zobrazenie doplňujúcich informácií
- generovanie grafických prvkov rozšírenej reality
- ovládanie systému pomocou Android aplikácie
- ovládanie hry pomocou Android aplikácie
- poskytovanie 2 aplikácií (hier), ktoré využívajú obohatenú realitu
- získavanie GPS a časových značiek v reálnom čase pomocou Android aplikácie
- použitie lokálnej databázy pri rozoznávaní objektov záujmu na snímkach
- vypočítanie polohy hlavy z Kinectu
- integrácia komunikácie a celého systému

2. Popis projektu (1. - 3. šprint)

Popis projektu v tejto kapitole sa vzťahuje na prvé 3 šprinty. Je tu opísaný celkový pohľad na architektúru projektu Carlos. Za ním uvádzame podrobný popis komponentov, z ktorých sa projekt skladá. Ku každej časti sú v závere spomenuté úlohy, ktoré sa k jej vypracovaniu viažu. Prehľad jednotlivých úloh je opísaný po šprintoch v ďalších kapitolách.

2.1 Architektúra systému

2.1.1 Návrh architektúry

Pri projekte Carlos sme sa stretli s rôznymi technológiami a zariadeniami. V rámci požiadaviek sme mali za úlohu použiť Android zariadenie na sledovanie polohy automobilu a ovládanie aplikácie. Ďalšie zariadenie Kinect slúži na sledovanie polohy a rotácie hlavy pre presnejšie mapovanie textu na okne vozidla. Toto okno má za úlohu pomocou projektoru zobraziť rozšírenú realitu. Mali sme použiť aj ďalšie zariadenie - kameru, ktorá mala sledovať okolité prostredie. Z obrazu kamery sme mali za úlohu rozoznať objekty, napríklad ako budovy, hrady a pod.

V projekte teda vystupuje veľa pestrých technológii a ich spojenie má riešiť architektúra. Architektúra je postavená na myšlienke spojiť rôzne zariadenia a rôzne technológie. Ďalšími prioritami pri jej navrhovaní je robustnosť (možnosť rozšírenia komponentov, možnosť nahradiť komponenty inými zariadeniami ako je ďalej vysvetlené), jednoduchosť a čo najväčšia nezávislosť komponentov.

Robustnosť architektúry by nám mala priniesť výhodu najmä v ďalšom semestri, kde by naša architektúra bola pripravená na ďalšie komponenty. Prípadne ak by v tomto semestri nastali komplikácie, architektúra by sa mala jednoducho prispôsobiť.

Jednoduchosť architektúry je dôležitá najmä pre rýchle pochopenie celej podstaty systému, keďže v našom tíme je zopár ľudí, ktorí s týmito technológiami nemali skúsenosti. Preto sa bral ohľad aj na toto riziko.

Nezávislosť architektúry je našou treťou prioritou. Cieľom tejto nezávislosti je vytvoriť jednotlivé komponenty tak, aby boli čo najmenej od seba závislé, čiže aby interface medzi nimi bol čo najjednoduchší. Cieľom bolo zároveň vytvoriť taký celok komponentu, aby sa bol schopný oň postarať jeden člen tímu, ktorý by mal na starosti návrh, analýzu, implementáciu a testovanie daného komponentu.

Tieto priority sme si určili v prvých šprintoch. O týchto prioritách sme intenzívne diskutovali a až po dohode ich poradí sme ich schválili.

2.1.2 Architektúra a jej komponenty

Jednotlivé komponenty a jej pod časti sme nazvali modulmi, teda vzniklo viacero modulov, kde každý mal definovaný presný vstup, výstup modulu a jeho kompetencie.

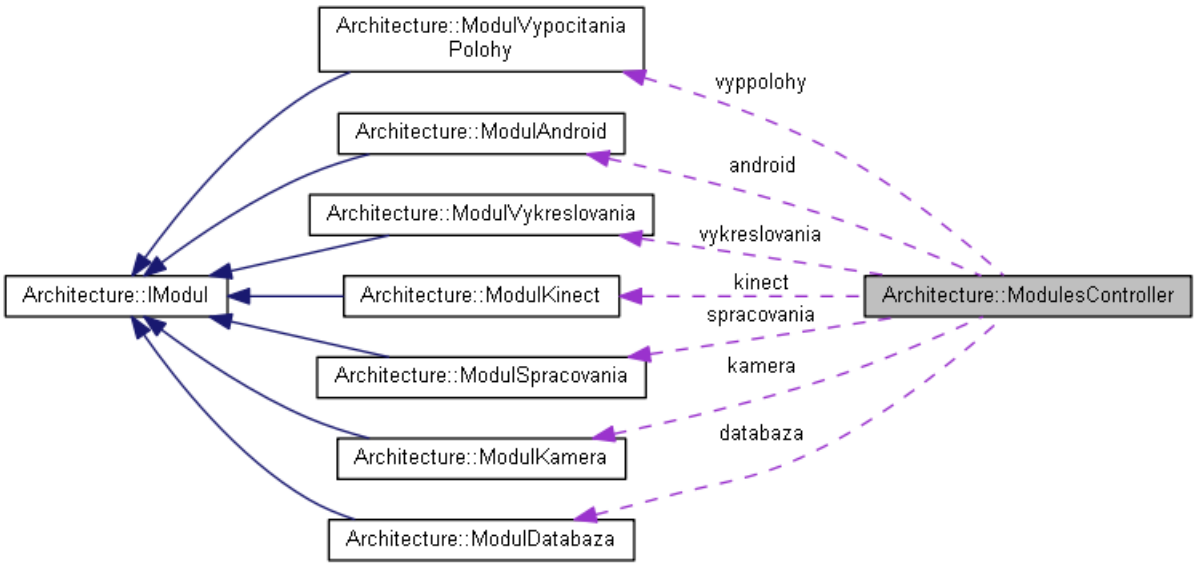
Vytvorili sme moduly:

- *Modul spracovania obrazu*
  Vstup: stream z kamery, zoznam možných objektov a cesty k ich fotografiám, maticiam deskriptorov a zoznamom keypointov
  Výstup: poloha zdetegovaných objektov
  Popis: Cieľom je detekcia polohy objektov na snímkach s využitím databázy.
- *Modul vypočítania polohy textu*
  Vstup: GPS, poloha pozerania hlavy v priestore, pozicia detegovaneho objektu, …
  Výstup: poloha textu na okne auta
  Popis: Úlohou tohto modulu je vypočitať polohu textu (prípadne inej obrazovej informácie) na okne auta z definovaných vstupov. Poloha zobrazovanej informácie na okne závisí hlavne od pohľadu používateľa na okno a polohy objektu.
- *Modul databázy*
  Vstup: Zdroj dát v databáze
  Výstup: Namapované entity cez objektový mapovač
  Popis: Cieľom je poskytnúť údaje v databáze cez službu.
- *Modul Android*
  Vstup: Interakcia človeka - gesto, hlas, dotyk na obrazovke; poloha zariadenia
  Výstup: Informácia o vykonanom geste; GPS poloha s časovým údajom
  Popis: Úlohou tohto modulu je slúžiť ako ovládač pre systém. Druhou nie menej dôležitou úlohou je slúžiť ako ovládač pre hry. Tento modul má na starosti taktiež získavanie GPS polohy a času.
- *Modul Kinect*
  Vstup: RGB video stream z Kinectu, stream hĺbkových informácií z Kinectu
  Výstup: poloha bodu pozerania v reálnom priestore
  Popis: Modul spracováva údaje z kamier Kinectu a snaží sa nájsť súradnice bodu pozerania pre používateľa. Tieto informácie sú potrebné pre správnu funkcionalitu ďalších modulov a preto ich presnosť ovplyvňuje aj presnosť týchto modulov.

- *Modul rozšírenej reality*
  Vstup: obrázok z videa, obrázok s detegovaným horizontom, vstupy z androidovej aplikácie
  Výstup: zobrazená hra na bočnom skle auta
  Popis: Úlohou tohto modulu je vytvorenie zábavnej hry, ktorú sa bude používateľ hrať pri cestovaní v aute. Hra bude využívať prvky objektov ktoré rozpoznáme.

Modul v konečnom dôsledku mal na starosti určite zariadenie, riadenie určitého zariadenia alebo väčšiu časť logiky aplikácie. Čím sme si tieto moduly mohli ľahko rozdeliť medzi team a pracovať paralelne.

Prvým krokom bolo teda definovanie modulov, tomu sme sa venovali na 1 a 2 stretnutí. Tieto definície sme najprv zapísali do náčrtov, potom diagramov a na konci sme ich implementovali.



Obr.1. Časť class diagramu, ktorý zobrazuje interface IModul a napojenie na ModulesController.

Každý modul dedí od „IModul“, čo je interface. Každý modul môže mať rôzne vstupy a výstupy. Priamo vstupy a výstupy sú zabalené v triede, ktoré sme nazvali ,,NazovModulu::In“ a ,,NazovModulu::Out“. To sme spravili, preto aby tieto vstupy a výstupy mohli byť robustné, teda ak budeme chcieť neskôr poslať novú informáciu do modulu, stačí pridať do atribútu danej triedy novú informáciu. Zároveň všetky vstupy a výstupy sa vo vlastných triedach serializujú, to chceme využiť neskôr pri debugovaniía možnosti uložiť stav aplikácie. Atribúty serializujeme do textu, ktorý vypisujeme do pomocnej konzole, čím sme si vytvorili možnosť efektívneho sledovania aplikácie a robiť záznamy o jej vykonávaní.

Mnohé moduly si medzi sebou posielajú objekty, obrázky, texty a pod. Pre tieto entity sme si vytvorili vlastne štruktúry, ktoré sú raz definované a využívajú sa vo všetkých moduloch. To pre to, lebo pri analýze technológii sme zistili, že viaceré moduly budú využívať rôzne knižnice a cieľom bolo prenechať knižnicu len danému modelu. Respektíve každému modulu si ponechať svoju knižnicu a prepojiť celý systém cez naše interfaci a naše entity.

2.1.3 Hierarchia architektúry

Pri druhom stretnutí sme sa rozhodli využiť určite metodiky pomenovania súborov a vytvoriť prehľadnú hierarchiu súborov a zložiek. V našom prípade sme použili C++ teda pre každú triedu sme vždy definovali header a cpp súbor. Kde header obsahuje presnú definíciu a cpp subor obsahuje implementáciu.

Moduly sú zadefinované vo vlastnom projekte a obsahujú svoje balíky. Názvy projektov:
com.carlos - riadiaci modul, hlavný proces a projekt
com.carlos.kinect - modul pre určenie polohy hlavy a bodu pozerania
com.carlos.tcp - modul tcp, stará sa o vytvorenie tcp servera
com.carlos.spracovanieReality - modul pre určenie polohy zdetegovaných objektov
com.carlos.vystupObrazovky - modul vykresľovania reality
com.carlos.db - modul pre databázu
com.carlos.CalculateTextPosModule - modul pre výpočet polohy textu na okne

Jednotlivé projekty zase obsahujú balíky, napríklad pre riadiaci modul sú:
architecture - entities
- modules
db - obsahuje zdrojové kódy pre prácu s databázou
main - obsahuje hlavný algoritmus a riadenie aplikácie

2.1.4 Riadiaci modul

Projekt com.carlos, teda riadiaci modul je špecifický tým, že spája všetky moduly a riadi ich. Stará sa tiež o paralelný beh modulov, pre posielanie správ a údajov. O toto pre posielanie sa stará trieda ModulesController, v súbore class.ModulesController.cpp.

Každý modul robí jeden člen tímu. Po jeho otestovaní a dopísaní dokumentácie ma za úlohu exportnuť tento modul do DLL - dynamic link library, kde tento export si pripojí naša hlavná aplikácia a začne ho používať.

V riadiacom module sa nachádza aj hlavný algoritmus. Ten sa nachádza v triede Carlos, trieda je v súbore class.Carlos.cpp. Hlavný algoritmus má sprístupnené moduly cez spomínaný ModulesController. Stream z kamery sa delí na jednotlivé snímky, snímok je vstupom do tohto algoritmu. Následne celý algoritmus zbehne a výstupom je rozšírená realita na okne auta. Pre lepšiu ilustráciu vkladáme algoritmus:

```
void Carlos::spracujJedenSnimok(Image& image) {
    // Z gps suradnic sa musi synchronizovane pockat, potom sa moze ist dalej
    // Lebo ked snimka meska, tak gps moze byt uz o par metrov dalej
    GPS gps = controller->android->getGPS();
    Rotation rotaciaHlavy = controller->kinect->getAktualnaRotaciaHlavy();

    // Modul preprocessingu
    vector<WorldObject> recepts = controller->databaza->najdiVsetkySvetoveObjektyBlizkoGPS(gps);

    // Modul spracovania
    ModulSpracovania::In spracovanie;
    spracovanie.image = image;
    spracovanie.recepts = recepts;
    ModulSpracovania::Out vysledokSpracovania = controller->spracovania->detekujObjekty(spracovanie);

    // Modul vypoctu polohy
    vector<Position> najdenePozicie; // synchronizovane
    for(uint i=0; i < vysledokSpracovania.objects.size(); i++) {
        ModulVypocitaniaPolohy::In vypocetPolohy;
        vypocetPolohy.gps = gps;
        vypocetPolohy.polohaObjektu = vysledokSpracovania.objects.at(i).position;
        vypocetPolohy.rotaciaHlavy = rotaciaHlavy;

        ModulVypocitaniaPolohy::Out polohaTextu;
        polohaTextu = controller->vyppolohy->vypocitajPolohuTextu(vypocetPolohy);
        if(polohaTextu.najdeny) {
            najdenePozicie.push_back(polohaTextu.polohaTextu);
        }
    }

    // Modul vykreslovania
    ModulVykreslovania::In vykreslovanie;
    vykreslovanie.image = image;
    vykreslovanie.najdenePozicie = najdenePozicie;
    controller->vykreslovania->vykresliObrazokSRozsirenouRealitou(vykreslovanie);
    imshow("Test", image.data);
}
```

2.1.5 Súvisiace úlohy

#57 [Architektura] Spracovanie sprav z Androidu
#35 [Architektúra] Fake ovládanie
#56 [Architektura] Riadiaci modul
#40 [Architektura] Prvotna analyza architektury
#52 [Architektura] Vygenerovanie diagramov a dokumentacie
#14 [Architektura] Nacitanie videa

#10 [Architektúra] Analýza a definovanie interfaces
#21 [Architektura] Definovanie tried pre moduly
#22 [Architektura] Nacitavanie DLL suborov
#23 [Architektura] Hlavny algoritmus
#24 [Architektura] Prototyp
#23 [Architektura] Thread cast

2.2 Modul spracovania obrazu

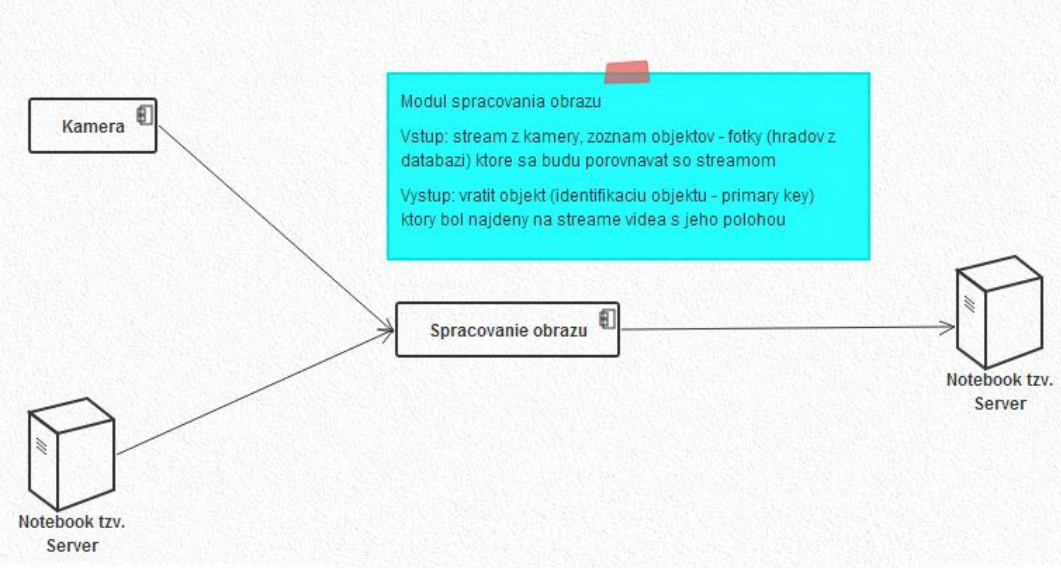
Modul spracovania obrazu slúži vo všeobecnosti na spracovanie snímok prichádzajúcich z webkamery. Jeho výstupom je v súčasnosti poloha detegovaných objektov (POI), ktoré sa porovnali s obrázkami uložených v lokálnej databáze.

2.2.1 Globálny cieľ

Vytvoriť aplikáciu, ktorá bude rozoznávať objekty zo streamu z webkamery a vracať polohy rozoznaných objektov v reálnom čase.

2.2.2 Cieľ na zimný semester

Vytvoriť prototyp aplikácie, ktorý vráti polohu najpravdepodobnejšieho objektu zo streamu z webkamery.



Obr. 1. Modul spracovania obrazu

2.2.3 Vstupy

- stream z kamery
- zoznam možných objektov a cesty k ich fotografiám, maticiam deskriptorov a zoznamom keypointov

2.2.4 Výstupy

- polohy zdetegovaných objektov

2.2.5 Analýza

OpenCV ponúka viaceré možnosti na detekciu objektov. Zanalyzoval som a zvážil som výhody a nevýhody viacerých prístupov:

- *spätná projekcia* → vytvorenie histogramu zo známeho objektu a následné vytvorenie spätnej projekcie na stream z kamery nám vytvorí pravdepodobnostnú mapu, ktorá nám ukáže ako dobre pixely zo streamu zapadajú do rozloženie farieb z nášho histogramu objektu → táto metóda je veľmi citlivá na zmeny osvetlenia a mnoho objektov záujmu (POI) má veľmi podobné histogramy, preto je táto metóda nevhodná
- *template matching* → template, v tomto prípade obrázok nášho POI sa metódou sliding window hľadá na streame z kamery, táto metóda je tak isto nevhodná, veľmi citlivá, pretože objekt môže byť napr. inak natočený a detekcia by už mohla byť neúspešná
- *deskriptory* → metóda spočíva v extrakcii dobre viditeľných bodov (keypoints) na obrázku s POI a streamu z kamery, potom sa okolie týchto bodov opíše pomocou deskriptorov (n-dimenzionálne vektory). Výhodou je že deskrirptory sú vo väčšej miere invariantné voči zmenám jasu a deformácii obrazu oproti predchádzajúcim metódam. Deskriptory objektu a stremu sú porovnávané - pre každý keypoint objektu sa nájde jeho najlepšia zhoda na streame.

Z týchto prístupov sme vybrali metódu detekcie pomocou deskriptorov ako najlepšiu.

2.2.6 Návrh

Pri návrhu modulu spracovania obrazu sa použili niektoré časti kódu od M. Račeva.

Modul bude príjmať stream z videa a množinu možných kandidátov na objekty na základe gps, ktoré na obraze môže modul zdetegovať. Ku každému kandidátovi na objekt dostane zároveň cestu k fotografii/fotografiám objektu, množine keypointov a matici deskriptorov, kde každý riadok opisuje okolie jedného bodu. Deskriptory budú vo finálnej fáze pre všetky objekty predpočítané, aby sa zvýšila rýchlosť algoritmu. Fotografie spolu s keypointami a deskriptormi sa načítajú pre všetky objekty. Pre stream z kamery nájdeme keypointy, ktoré sa opíšu rovnako pomocou deskriptorov.

Deskriptory POI a streamu sa v ďalšej fáze porovnajú a nájdu sa pre každý keypoint objektu k nemu najbližšie 2 keypointy na streame z kamery. Porovnávanie deskriptorov je založené na ich vzájomnej euklidovskej vzdialenosti (L2).
Výsledné zhody (match) sa v ďalšej fáze prefiltrujú:

- L2 medzi deskriptorom bodu objektu a jeho najbližším bodom na streame (match[i][0]) musí byť výrazne menšia než L2 s jeho 2. najbližším bodom na streame (match[i][1]) →

```
distance(match[i][0]) < threshold * distance(match[i][1])
```

- nájdeme ku každému bodu objektu jeho najbližší na streame, ale zároveň to urobíme aj opačne, t.j. pre každý bod na streame nájdeme k nemu najbližší, ak sa tieto zhody rovnajú, tak match akceptujeme →
  `match12[i].query == match21[i].train`

Pre každú fotografiu objektu sa pokúsime nájsť homografiu - maticu perspektívnej transformácie (H) pomocou RANSAC metódy a eliminujeme nevyhovujúce zhody (outliers). Tie zhody, ktoré ostali nazveme dobré zhody (good_matches). Následne určíme pomer: good_matches / matches, priemernú L2 všetkých good_matches a veľkosť zdetegovaného objektu. Z kandidátov na objekty vyberie tie, ktoré splnia hraničné hodnoty týchto podmienok. Tieto objekty považujeme za nájdené na streame.

Obr. 2. Nájdenie ”dobrých” zhôd medzi objektom a scénou.

Polohu zdetegovaného objektu určíme pomocou matice H a nájdenú pozíciu objektu modul predá na výstup ďalej na spracovanie.

Obr. 3. Detegovaný objekt

Zdetegované objekty bude možné trackovať pomocou Lucas-Kanade sledovača, ktorý nájde pozíciu bodov/objektov na nasledovnom obrázku.

2.2.7 Implementácia

Modul bol implementovaný v C++ s knižnicou OpenCV a SiftGPU. Ako deskriptory boli zvolené:

- SURF deskriptory (OpenCV)
- SIFT deskriptory vypočítavané na GPU (SiftGPU)

V súčasnosti ešte nie je implementované sledovanie objektov. Deskriptory objektov ešte nie sú dopredu vypočítané, počítajú sa vždy nanovo. Modul zatiaľ vracia polohu najpravdepodobnejšieho objektu.

2.2.8 Testovanie

Testovanie prebiehalo na vytvorených fotografiách budov ako hrad a kostol za dobrých svetelných podmienok. Pri ďalšom testovaní boli použité fotografie z Google Streetview.

2.2.9 Kalibrácia kamery

Pre správne zobrazovanie objektov je potrebné, aby sa grafické informácie premietané z

projektora zobrazovali čo najpresnejšie. Preto musíme na začiatku skalibrovať kameru pomocou nasledovnej funkcionality.

2.2.9.1 Vstup

- dva obrázky

2.2.9.2 Výstup

- matica perspektívnej transformácie (H)

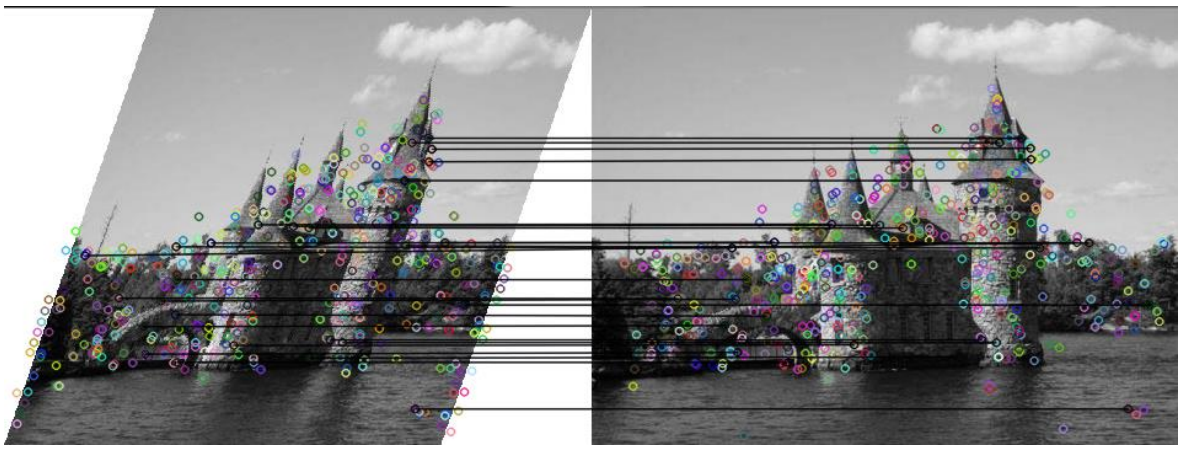
2.2.9.3 Analýza

Technika, ktorá zjednocuje viaceré obrazy, ktoré možu byť navzájom posunuté do jedného sa nazýva registrácia obrazu. Kalibrácia kamery registruje obraz z virtuálneho sveta s reálnou scénou nasnímanou s videokamerou.

Na registráciu obrazu použijeme opäť deskriptory.

2.2.9.4 Návrh

Funkcia kalibrácie kamery nájde maticu perspektívnej transformácie (H) rovnakým postupom ako v module spracovania obrazu. Tá sa potom využije na “prekrytie” oboch obrazov.



Obr. 4. Nájdené zhody medzi 2 obrázkami (1. obrázok je zdeformovaný)

Obr. 5. Transformovaný obrázok na základe matice prespektívnej transformácie H.

2.2.9.5 Implementácia

Táto funkcionalita bola implementovaná v C++ s knižnicou OpenCV. Ako deskriptory boli zvolené SURF deskriptory (OpenCV).

2.2.9.6 Testovanie

Testovanie prebiehalo na 2 rovnakých obrázkoch, pričom jeden bol zdeformovaný.

2.2.10. Súvisiace úlohy

#38 [OpenCV] preštudovať dostupné metódy na detekciu objektov
#1 [OpenCV] prvý prototyp modulu detekcie objektov
#27 [OpenCV] analyzovať a použiť kódy M. Raceva
#4 [OpenCV] detekcia objektov so streetview fotkami
#29 [OpenCV] rozoznanie objektu na obrázku
#17 [OpenCV] registrácia obrazu
#46 [OpenCV] úprava spracovania/registrácie obrazu

2.3 Modul Kinect

Pre dosiahnutie správneho umiestnenia pri zobrazovaní objektov je potreba určiť bod, z ktorého sa používateľ na okno pozerá. Tento modul má za úlohu nájdenie polohy hlavy používateľa a výpočet súradníc spomínaného bodu v priestore.

2.3.1 Globálny cieľ

Vedieť vypočítať bod pozerania používateľa s čo najväčšou presnosťou a aj za nepriaznivých podmienok.

2.3.2 Cieľ na zimný semester

Vedieť odhadnúť bod pozerania používateľa bez potreby videnia tváre.

2.3.3 Vstup

- RGB video stream z Kinectu
- stream hĺbkových informácií z Kinectu

2.3.4 Výstup

- poloha bodu pozerania v reálnom priestore (súradnice x, y, z)

2.3.5 Analýza

Pre určenie polohy hlavy a následne aj hľadaného bodu sme uvážili ako vhodný nástroj Kinect, ktorý poskytuje ako farebný obraz, tak aj hĺbkovú informáciu, vhodnú pre výpočet súradníc bodu a tiež pre niektoré techniky rozpoznávania objektov. Pre zjednodušenie práce s týmto nástrojom sme si vybrali knižnicu freenect, súčasť projektu OpenKinect.org.

Samotné detekovanie polohy hlavy, tváre a potrebného bodu je vykonávané cez knižnicu OpenCV pomocou kaskádových klasifikátorov vhodných na detekciu objektov vo videu.

2.3.6 Návrh

Tento modul bude prímať z kinectu dáta obrazové aj hĺbkové a to pomocou spomenutej knižnice freenect. Tá tieto dáta zachytí a spracuje a následne odošle na spracovanie knižnicou OpenCV, ktorá nájde tvár. Na konci modul odhadne bod pozerania na tvári.

2.3.7 Implementácia

Implementácia prebehla v C++ pomocu knižníc freenect a OpenCV. Ako pomocný súbor pre rozpoznávanie tváre bol použitý súbor haarovych kaskád.
Vstupom pre tento modul je samotný obraz z kinectu a to vrátane hĺbkovej mapy, posunutej aby sedela s obrazom.
Výstupom sú súradnice x, y, z bodu z ktorého sa používateľ pozerá.

Aktuálne modul podporuje len detekciu tváre a určenie bodu pomocou odhadu (polovica šírky tváre, ⅔ výšky). V ďalšom vývoji je plánované nájdenie celej hlavy a teda detekcia bez videnia tváre a tiež spresnenie nájdených súradníc.

2.3.8 Testovanie

Modul bol úspešne testovaný na 4 tvárach, kde približne dokázal určiť potrebný bod. Tiež bola testovaná presnosť hĺbkovej mapy na daný bod pomocou ručných meraní ozajstnej vzdialenosti.

2.3.9 Súvisiace úlohy

#43 [Kinect] nainštalovať knižnicu pre Kinect
#08 [Kinect] hĺbkova informácia
#33 [Kinect] detekcia tváre
#50 [Kinect] Výpočet súradníc bodu z pohľadu kinectu
#44 [Kinect] Rozpoznávanie hlavy

2.4 Modul vypočítania polohy textu

Pri zobrazovaní informácií o detegovanom objekte na premietacie plátno (okno vozidla) je potrebné rátať s viacerými vstupmi a pozíciu informácie na okne, prípadne nejakého virtuálného objektu premietaného na okno je potrebné upravať podľa toho v akom uhle sa práve na daný detegovaný objekt pozerá daný človek cez okno a ako ďaleko je od okna. Úlohou tohto modulu je práve vypočítanie správnej pozície informácie alebo objektu na okne vozidla

2.4.1. Globálny cieľ

Vytvoriť modul, ktorý bude rešpektovať všetky dole vymenované vstupy a bude správne vypočítavať polohu text aj pri prudkých zmenách rýchlosti auta a prudkých zmenách smeru auta.

2.4.2. Cieľ na zimný semester

Vypočítavať polohu textu pre pohybujúce sa auto, pričim nepočítať s prudkými zmenami rýchlosti alebo prudkých zmien smeru auta.

2.4.3. Vstupy

- GPS objektu
- GPS auta
- vzdialenosť kamery od okna
- pozícia kamery v okne v rámci x
- (x1,y1),(x2,y2) = dva rohy (lavý horný a pravý dolný) detegovaný objekt na okne
- (x,y,z) hlavy = bod pozerania hlavy v priestore

2.4.4. Výstupy

- (x,y) - stred polohy pre informáciu o danom objekte

2.4.5. Analýza

Pre vypočítanie správnej polohy informácie na okne sme identifikovali využitie nasledovných vstupov:

1. GPS objektu
2. GPS auta
3. vzdialenosť kamery od okna
4. pozícia kamery v okne v rámci x
5. (x1,y1),(x2,y2) = dva rohy (lavý horný a pravý dolný) detegovaný objekt na okne
6. (x,y,z) hlavy = bod pozerania hlavy v priestore

Modul nebude využívať žiadne externé knižnice a jeho výstupom, bude len jeden bod (x,y) - poloha v rámci okna auta, kde sa nachádza stred polohy informácie pre daný detegovaný objekt. Stred súradnicovej sústavy bude najlepšie umiestniť do lavého horného rohu okna auta (premietacieho plátna). V prvej implemntácii tohoto prototypu sme sa zameráme len na ideálnu situáciu a to takú, že vozidlo sa pohybuje po priamke a neuvažujeme žiadne zákruty ani prudké zmeny rýchlosti, ktoré by mohli v značne krátkom časovom okamihu ovplyvniť polohu informácie.

2.4.6. Návrh

Vstupy 1. a 2. súžia na vypočítanie priamej vzdialenosťi objektu od auta na náslendé ďalšie využie tejto vzdialenosti v algoritme. Vstupy 3. a 4. slúžia na to, aby sme vedeli správne identifikovať uhol, pod akým sa používateľ pozerá na objekt a tak správne vypočítať kolmú vzdialenosť objektu od vozidla spolu s predchádzajúcou vypočítanou priamou vzdialenosťou. Vsup 5. je už samotná pozícia objektu na okne zistená algoritmom na rozpoznávanie objektov. Všetky tieto vypočítané dáta využijeme už pri vypočítaní samotnej polohy informácie spolu so vstupom 6. o polohe pozerania hlavy v priestore.

2.4.7. Implementácia

Popísaný algoritmus je implementovaný v programovacom jazyku C++ bez využitia akýchloľvek externých knižníc. Výstupom algoritmu je bod (x,y), kde na okne sa nachádza stred polohy pre informáciu o danom objekte.

2.4.8. Testovanie

Testovanie prebiehalo vytvorením 3D scény v programovacom jazyku Java Script s využitím knižnice Processing.js. 3D scéna obsahovala improvizované okno vozidla, hlavu človeka a detegovaný objekt. Úlohou v tejto scéne bolo vypočítanie správnej polohy informácie na okne z

už spomenutých vstupov. GPS v tomto prípade bolo vynechané z výpočtov a namiesto toho sme využili pozíciu v objektu a vozidla priestore. Testovanie na tejto scéne prebehlo úspešne a algoritmus bol po tomto úspešnom otestovaní prepísaný do jazyka C++, aby sme ho mohli využiť v našom projekte.

2.4.9. Súvisiace úlohy

#5 [Poloha] Modul na výpočet polohy objektov na okne

2.5 Modul Android

Modul ovládania systému, ktorý zahŕňa ovládanie spúštania aplikácii, ktoré systém obsahuje alebo slúži ako ovládač pre hranie hry na projekčnej ploche je inteligentný telefón s operačným systémom Android. v súčasnosti je jeho výstupom TCP soket s konkrétnou informáciou, ktorú sme na ovládači zvolili. Môže to byť pohyb prstu na obrazovke v 4 smeroch - hore, dole, doľava a doprava, druhou možnosťou je nakláňanie telefónu v 4 smeroch a treťou možnosťou je hlasový povel v anglickom jazyku.

2.5.1 Globálny cieľ

Vytvorenie aplikácie, ktorá bude primárne slúžit na zapínanie / prepínanie medzi funkciami v systéme. Druhou nie menej dôležitou úlohou aplikácie je slúžiť ako ovládač pre hru v systéme. Modul android ma na starosti taktiež získavanie GPS polohy auta s časovou stopou, ktorú bude odosielať iným modulom v systéme.

2.5.2 Cieľ na zimný semester

Prispôsobenie aplikácie Controller, ktorú sme prevzali z minuloročného tímového projektu tak, aby slúžila naším potrebám. Taktiež je potrebné doplnenie ovládania hry s využitím už implementovaných funkcií, ktoré už aplikácia má. Vytvorenie GPS modulu, ktorý bude získavať aktuálnu polohu auta spolu s časovou stopou, ktorá sa v systéme bude využívať.

2.5.3 Vstupy

- gestá - dotyk na obrazovke
- hlas
- nakláňanie zariadenia - gyroskop, akcelerometer
- poloha zariadenia

2.5.4 Výstupy

- informácia o vykonanom geste
- GPS poloha zariadenia s časovou stopou

2.5.5 Analýza

Ako vhodný nástroj resp. technológiu na ovládanie celého systému sme po vzájomnej dohode zvolili Android zariadenie - inteligentný telefón / tablet. Hlavnou výhodou je dotykový displej, na ktorom môžeme implementovať rozpoznávanie rôznych giest a pohybov. Druhou možnosťou je využitie gyroskopu a akcelerometra pre ovladanie pohybov v hre. Inteligentný telefón nám samozrejme ponúka aj iné možnosti, ktoré môžeme využiť ako je napríklad rozpoznávanie hlasu a hlasové povely. Tie sú však v offline režime obmedzené na anglický jazyk, ale pre jednoduché povely - “start”, “run”, “one” … by sme to dokázali využiť, keďže najväčším problémom môže byť výslovnosť a akcent daného používateľa. Ďalšou výhodou tejto platformy je aj to, že časť našeho systému využíva aktuálnu GPS polohu, čo je pomerne jednoducho implementovateľné a môžeme to kľudne zahrnúť do tejto aplikácie.

2.5.6 Návrh

Tento modul bude z telefónu odosielať TCP sokety s informáciou o danom geste, či už je to pohybové, hlasové alebo iné gesto. Aplikácia jednak poslúži ako ovládač pre súštanie rôznych možností v systéme, ale aj ako ovládač hier.

2.5.7 Implementácia

Implementácia prebehla v jazyku Java v prostredí Android Development Tool (Eclipse prispôsobený vývoju pre Android). Ako už bolo spomenuté vyššie čiastočne som využil funkčnú implemntáciu z iného tímového projektu, ktorú som však do značnej miery musel upravovať, aby to vyhovovalo naším potrebám. Implementoval som taktiež TCP komunikáciu, kde Android aplikácia sa správala ako klient a Java Server aplikáciu, ktorá príjmala sokety sa tvárila ako server. Výstupom tejto implementácie je odoslaný soket po vykonaní daného gesta.

2.5.8 Testovanie

Testovanie prebehlo na Android zariadeni Samsung Galaxy SII, ktoré komunikovalo s lokálnym serverom a odosielalo na neho sokety. Sokety boli úspešne prijaté.

2.5.9 Android Aplikácia pre zaznamenávanie GPS polohy

Samostatnú časť modulu tvorí aj aplikácia pre záznam GPS polohy .

V našom projekte je veľmi potrebná informácia aktuálnej GPS polohy našeho auta z dôvodu rozpoznania objektov v danej oblasti. Pre dosiahnutie správnej GPS informácie bolo potrebné vytvoriť nasledovný modul ktorý má na starosti zachytenie správnej GPS informácie. Jedná sa o konkrétne samostatnú android aplikáciu, ktorá má za úlohu získanie súradníc ktoré sa získavajú pravidelne každú sekundu. Táto aplikácia v budúcnosti bude súčasťou celého Android modulu a nebude vysupovať ako samostatná aplikácia.

2.5.9.1 Analýza

Pre určenie GPS polohy našeho auta a jej následné uloženie sme uvážili ako najvhodnejší nástroj/zariadenie použiť Android smartfón. Vätšina dnešných Android telefónov obsahuje GPS prijímač a internetové pripojenie ktoré môže zmenšiť odchýlku polohy nameranú smartfónom od reálnej polohy. Ďalšia výhoda využitia smartfónu spočíva v tom, že sme vedeli dopredu, že smartfón bude použitý na ovládanie celej aplikácie a užívateľ jednoducho musí mať jeden pri sebe. Výhoda zistenia polohy android smartfónom bola aj v samotnej implementácie, pretože Android development obsahuje prehľadné API pre získanie GPS polohy.

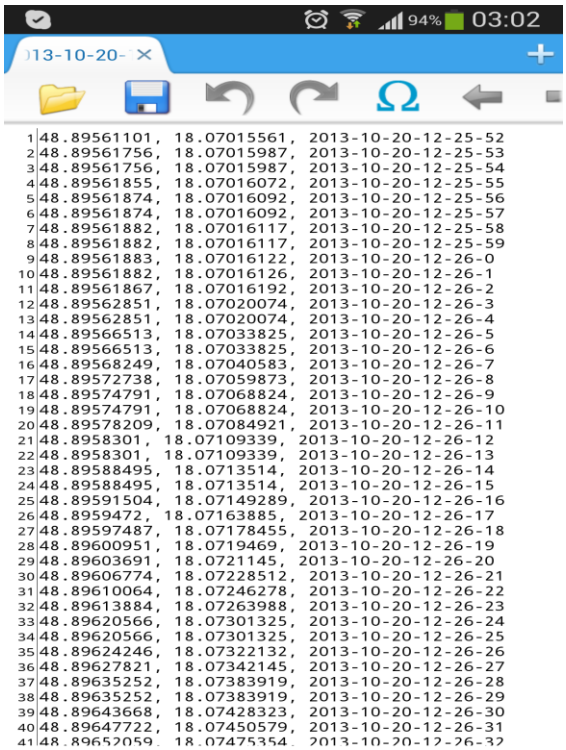
2.5.9.2 Návrh

Aplikácia bude prijímať z GPS prijímača dáta geografického súradnicového systému t.j. informáciu o zemepisnej šírke a zemepisnej dĺžke (z angl. „langitude“ a „longittude“). Keďže naše auto sa občas bude hýbať vačšou rýchlosťou, to znamená naša poloha sa bude zmenou rýchlosti neustále meniť, je potrebné mať stále informácie o aktuálnej polohe, takže sme sa rozhodli získať novú polohu každú sekundu.

2.5.9.3 Implementácia

Implementácia prebehla v Jave (Android SDK) pomocou viacerých knižníc, využili sme najma framework API `android.location.` Taktiež sme použili triedu LocationManager a Location čo nám podstatne uľahčilo prácu.

Výstupom danej aplikácie je súbor s GPS súradnicami v danom formáte:
(1.stlpec Latitude 2.stlpec Longitute 3.stlpec timestamp)



Obr. 1 Ukážka výstupu aplikácie

Ako je možné vidieť, každý riadok obsahuje informáciu GPS v polohy v danej sekunde. Aktuálna verzia tejto aplikácie zatiaľ podporuje iba lokálne ukladanie GPS informácie v telefóne.

2.5.9.4 Testovanie

Aplikácia bola úspešne testovaná na 4 Android smartfónoch, kde sme dokázali vytvoriť pár testovacích záznamov. Presnosť bola otestovaná s aplikáciou Google Earth. Odchýlka bola zväčša pár metrov.

Obr. 2 Ukážka dizajnu aplikácie

2.5.9.5 Súvisiace úlohy

#36 [Mobil] ukladať gps polohu

2.5.10 Súvisiace úlohy

#41 [Mobil] ukladanie gest
#16 [Mobil] demo pre rozpoznávanie gest
#37 [Mobil] analýza kódov na rozoznávanie gest
#32 [Mobil] TCP komunikácia z Androidu

2.6 Modul rozšírenej reality

Tento modul sa bude starať o zobrazenie hry na bočné okno auta a v neskoršej fáze aj na zobrazenie identifikovaných informácií o viditeľných objektoch. Vykonávať sa to bude za pomoci LED projektora ktorý bude umiestnený priamo v aute.

2.6.1. Globálny cieľ

Vytvorenie minimálne dvoch verzií hier pre používateľov. Prvá bude orientovaná skôr na zábavu. Druhá hra bude mať za úlohu naučiť niečo zaujímavé používateľa a následne ho aj vyskúšať. Taktiež bude potrebné klásť dôraz na to aby sa v hre využívali informácie o rozpoznaných objektoch.

2.6.2. Cieľ na zimný semester

Vytvorenie prvej funkčnej hry, zameranej na zábavu. Pokúsiť sa hru na-implementovať čo najvhodnejšie a tak aby ďalšie rozširovanie - pridávanie hier nebolo problematické. Taktiež dbať aj na používateľa a spracovať hru čo najzábavnejším spôsobom (zaujímavý model, možnosť výberu modelu, rebríčky najlepších...).

2.6.3. Vstupy

- obrázok z videa
- obrázok s detegovaným horizontom
- vstupy z androidovej aplikácie

2.6.4. Výstupy

- zozbrazená hra na bočnom skle auta

2.6.5. Analýza

Keďže cieľom je vytvorenie hry, hľadali sme vhodnú knižnicu za pomoci ktorej by sme túto hru implementovali. Avšak s prihliadnutím na to, že podstatná časť projektu bude implementovaná v jazyku C++ a v IDE Visual studio, rozhodli sme sa zvoliť OpenGL.

Táto knižnica je dostatočne komplexná a existuje pre ňu značné množstvo tutoriálov a dokumentácie. Taktiež použijeme aj ďalšie knižnice ako sú glut, freeglut, glew a mnohé ďalšie ktoré nám uľahčia prácu s OpenGL.

Ďalšia analýza spočívala vo voľbe čo najvhodnejšej hry na implementovanie. Vytvorili sme jednoduchú hru, ktorá formou otázok zisťovala vedomosti hráča a na konci mu zobrazila skóre. Vďaka čomu sme sa lepšie naučili používať knižnicu OpenGL. Taktiež všetci členovia tímu sa mali zamyslieť nad nápadom hry. Takto sme ziskali mnohé nápady. Viac o tom je napísané v samotnom návrhu.

2.6.6 Návrh

Pre zimný semester sme sa dohodli, že vytvoríme hru ktorou by sa mal používateľ baviť. Na stretnutiach sme zvažovali rôzne možnosti hry. Od logických, vedomostných až po letecké. Ako najvhodnejšiu hru sme vybrali lietadlo, ktoré letí v prostredí kde sa nachádza auto. Taktiež sme zvažovali viaceré možnosti ktoré je možné s týmto lietadlom vykonať.

Ako najlepší nápad hry sme vybrali lietadlo letiace nad horizontom, kde je cieľom nahrať čo najviac bodov a dôjsť čo najďalej. Čím používateľ letí bližšie horizontu tím viac bodov dostáva, avšak je aj bližšie pádu. Ak používateľ klesne pod horizont resp. narazí tak hra končí. Počas letu sa hráčovi pripočítavajú body. Na konci sa zobrazí tabuľka, kde si zadá meno a uvidí ako dobre / zle v celkovom rebríčku dopadol. Celá hra by sa mala ovládať pomocou mobilu.

2.6.7. Implementácia

Ako IDE sme si zvolili Visual Studio 2012. Na implementáciu hry použijeme OpenGL a viaceré ďalšie knižnice ktoré nám uľahčujú prácu so samotným OpenGL.

Aktuálne použité knižnice:

- FreeImage
- FreeType
- Glut
- FreeGlut
- Glew
- Glm

Všetky knižnice bolo potrebné u seba si skompilovať a následne ich pridať do modulu rozšírenej reality. V tomto momente už máme načítaný 3D model. Model je otexturovaný. Formát 3D modelu sme použili .obj súbor spojený s .mtl súborom v ktorom sú zaznačené údaje o textúre ktorá sa má použiť na model. Aktuálne sa dokončilo zobrazenie meniacich sa obrázkov ktoré získame z videa. Scéna už je kompletne vytvorená. Taktiež už je možné hýbať modelom po osiach x a y za pomoci kláves w,a,s,d.

V implementácii sa bude ďalej prokračovať, najviac času zaberie implementácia logiky hry a jej otestovanie, kompletná integrácia s android zariadením, integrácia s modulom spracovania obrazu, ďalej napojenie na databázu, počítanie skóre, prípadne zvažujeme aj možnosť ďať používateľovi na výber z viacerých typov lietadiel.

2.6.8. Testovanie

Bol vytvorený prototyp ktorý spĺňal všetky vytýčené cieľe. Tento prototyp bol otestovaný na všetky možné stavy ktoré môžu v hre nastať. Avšak je dôležité v testovaní pokračovať keďže ešte nie vytvorené všetko.

2.6.9. Súvisiace úlohy

#42 [OpenGL] Naštudovať základy OpenGL
#6 [OpenGL] Odskúšať objavujúce sa a miznúce objekty
#45 [OpenGL] Vypisovanie textu v okne
#18 [OpenGL] Práca na jednoduchej hre
#28 [OpenGL] Pridanie 3D modelu
#34 [OpenGL] Pohyby 3D modelom
#54 [OpenGL] Pripravenie scény za 3D modelom
#55 [OpenGL] Držať lietadlo nad horizontom

2.7 Modul databázy

Modul databázy, zahŕňa inštaláciu databázy, jej kompletnú prípravu. Ďalej vytvorenie dátového modelu, naplnenie ho dátami. V ďalšej fáze ma tento modul sprístupniť databázu pre iné modely v podobe služby. Zároveň tento modul ma vytvoriť možnosť perzistentne uchovávať informácie o rôznych entitách. Ktoré entity sa budú uchovávať sú popísané v dátovom modely. Vytvorenie dátového modelu je sekcii analýza a návrh.

2.7.1. Globálny cieľ

Vytvorenie modulu, ktorý bude mať na starosti riadenie databázy, uchovávanie informácií a jej opätovné získavanie. Celá funkcionalita bude v podobe služby. Služba bude sprístupnená len pre riadiaci modul, ktorý bude preposielať informácie ďalej.

2.7.2 Cieľ na zimný semester

Vytvorenie databázovej služby, ktorá bude spĺňať základnú funkcionalitu. Teda pripojenie na databázu, uloženie informácii a získavanie informácii. Zároveň cieľom je aj vytvorenie prvej verzie databázového modelu a naplnenie fake údajmi pre testovanie. V rámci tohto modulu je potrebné vytvoriť objektový mapovač a poskytnúť údaje v databáze v podobe entít.

2.7.3. Analýza

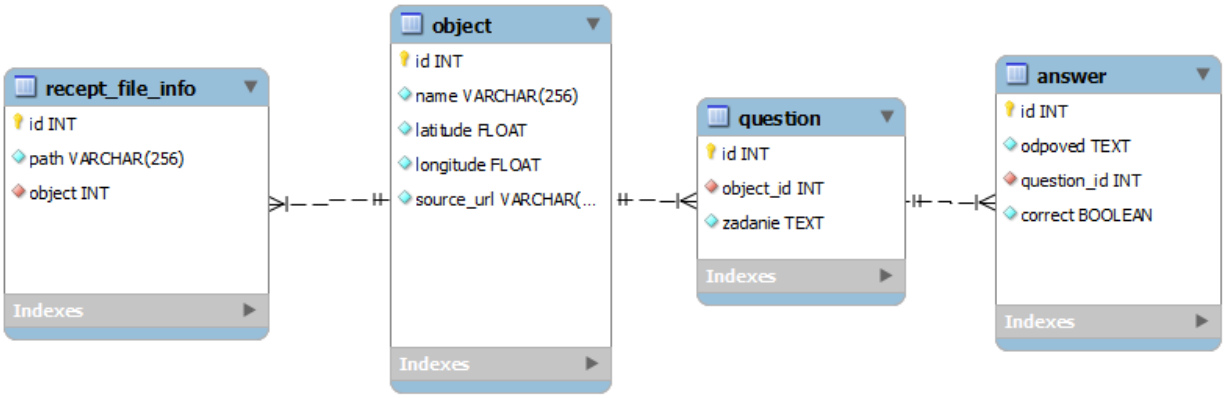
Základom modulu je databáza, takže v rámci analýzy bolo dôležité nájsť databázu, ktorá bude spĺňať naše požiadavky. V rámci analýzy sme sa pozreli na MySql, Oracle, sqlite, postgresql. Všetky databázy okrem sqlite, majú zložitú inštaláciu respektíve majú bežať na servery. Komunikácia nášho programu so serverom by bola komplikovaná a ak by aplikácia bola umiestnená mimo serveru nastali by rôzne komplikácie so sieťovou konfiguráciou a hlavne veľkou latenciou. Predpokladá sa, že naša databáza bude mať 5-10 tabuliek. Pre objektový mapovač sme našli viacero knižníc.

V ďalšej fáze analýzy sme skúmali, aký databázový model potrebujeme. V rámci diskusie medzi členmi sme sa dohodli na modeli a ten uvádzame v kapitole Návrhu.

2.7.4. Návrh

V analýze sme sa v skratke pozreli na dostupné riešenia. Pozreli sme sa na klady a zápory. Nakoniec sme na základe dostupných faktov vybrali sqlite. Najmä preto, že práca s ňou je jednoduchá a pre tento počet tabuliek je postačujúca. Pre tuto databázu existuje C++ API, takže aplikácia sa dokáže priamo napojiť na zdroj dát a priamo pracovať s databázou, bez sieťovej konektivity. Preto databáza môže byť dodaná priamo s aplikáciou.

Mnohé objektové mapovače, ktoré sme našli v procese analýzy boli príliš komplexné pre väčšie systémy. Nakoniec sme vybrali knižnicu sweetql.



Obr.1. Dátový model databázy

2.7.5. Implementácia

Implementácia prebehla v jazyku C++, vo visual studiu prostredí. Pre prácu s databázou bolo použité sqlilite API. Implementácia bola v podstate jednoduchá, spočívala len v prepojení sqlite API s knižnicou sweetql. Zároveň sme tieto dve knižnice zabalili do jednej triedy, do databázovej služby.

V rámci implementácie sme naplnili databázu falošnými údajmi.

2.7.6. Testovanie

Testovanie prebehlo v spolupráci s hlavným modulom, ktorému bola poskytnutá databázová služba. Služba sa pripojila na databázu, úspešne vyhľadala a stiahla údaje z databázy.

2.7.7. Súvisiace úlohy

#53 [Databaza] OOP Mapovac a DB Service
#26 [Databaza] Zadefinovanie entit ktore su v databaze
#25 [Databaza] C++ Wrapper, pripojit sa na databazu
#11 [Databaza] Nainštalovať databázu

3. Prvý šprint

Začiatok šprintu: 23.9.2013

#1 [OpenCV] prvý prototyp modulu detekcie objektov
Zodpovedná osoba: Patrik Polatsek

Na základe analýzy možných spôsobov detekcie objektov sa vytvorí prvý prototyp modulu detekcie objektov. Tento modul sa pokúsi nájsť na obrázku možné objekty pomocou deskriptorov. Na porovnanie sa využije lokálna zložka s.

#2 Vytvoriť webstránku projektu
Zodpovedná osoba: Martin Petluš

Cieľom tejto úlohy bolo vytvoriť web stránku, tímu, ktorá bude slúžiť na prezentáciu projektu (materiálov a informácií k nemu) a zároveň aj jeho členov.

#58 Redmine
Zodpovedná osoba: Martin Petluš

Rozbehať na serveri Redmine, vytvoriť na Redmine projekt k tímovému projektu.

#48 Rozbehanie servera
Zodpovedná osoba: Martin Petluš

Nainštalovať na virtuálny server operačný systém a pripraviť ho na inštaláciu ďalších potrebných programov, ktoré sa budú využívať v tímovom projekte.

#36 [Mobil] ukladať gps polohu
Zodpovedná osoba: Juraj Jarábek

Vytvorenie samostanej GPS aplikácie pre zaznamenávanie GPS polohy spolu s časovým razítkom. Tento submodul resp. aplikácia bude neskôr súčasťou celého Android modulu.

#37 [Mobil] analýza kódov na rozoznávanie gest
Zodpovedná osoba: Róbert Sabol

Stretnutie s kolegom Bc. Richardom Sámelom a prebratie minuloročného projektu Controller. Následne zanalyzovanie aplikácie a pochopenie aké gestá sa dajú využívať.

#38 [OpenCV] preštudovať dostupné metódy na detekciu objektov
Zodpovedná osoba: Patrik Polatsek

Naša aplikácia bude rozoznávať zo snímiek objekty záujmu. Na to je potrebné analyzovať dostupné metódy na detekciu objektov a učiť ich výhody a nevýhody. Z týchto metód je treba vybrať najvhodnejšiu metódu rozoznania objektov pre náš systém.

#39 [OpenCV] nastudovať základy OpenCV
Zodpovedná osoba: Martin Petluš

Naštudovať si základy OpenCV, pre budúcu prácu na projekte. Vyskúšať si jednoduché príklady s použitím knižnice OpenCV. Nainštalovať si OpenCV.

#40 [Architektura] Prvotná analýza architektúry
Zodpovedná osoba: Lukáš Sekerák

V rámci tejto analýzy sme sa pozerali na možnosti architektúry. Akú architektúru vytvoríme a ako budú jednotlivé komponenty podelené a ako budú spolupracovať.

#41 [Mobil] ukladanie gest
Zodpovedná osoba: Róbert Sabol

Prerobenie aplikácie Controller tak, aby dokázala gestá ukladať lokálne na mobilné zariadenie namiesto ukladania na server ako to bolo predpripravené.

#42 [OpenGL] naštudovať základy OpenGL
Zodpovedná osoba: Peter Hamar

V tejto časti analýzi pôjde o dôkladnejšie zoznámenie sa s knižnicou OpenGL. Naštudovanie základnych funkcií, nainštalovanie vývojového prostredia a vytvorenie prvého projektu vo VS s využitím knižnice OpenGL. Taktiež oboznámenie sa s prácou s textúrami, ich načítanie a zobrazenie.

#43 [Kinect] nainštalovať knižnicu pre Kinect
Zodpovedná osoba: Jakub Mercz

Pre uľahčenie prístupu k dátam, ktoré kinect ponúka bude použitá knižnica freenect, ktorú je potrebné správne nainštalovať a dôkladne preskúmať jej používanie.

#45 [OpenGL] Vypisovanie textu v okne

Zodpovedná osoba: Peter Hamar

Úlohou je pridanie textu na obrázok / textúru. Taktiež si otestovať rôzne druhy typov písma a samozrejme poziciovanie textu v okne.

4. Druhý šprint

Začiatok šprintu: 14.10.2013

#3 natočiť videá s gps informáciou
Zodpovedná osoba: Juraj Jarábek

Natočiť niekoľko video záznamov mimo mesta a v meste, ktoré budeme môcť neskôr použiť spolu s gpd informáciou, na ktorú nám poslúži už vytvorená android aplikácia.

#4 [OpenCV] detekcia objektov so streetview fotkami
Zodpovedná osoba: Patrik Polatsek

Doterajšie riešenie, ktoré rozoznáva objekty na obrázkoch treba otestovať na obrázkoch získané z Google Streetview. Toto testovanie je potrebné a záver vyhodnotiť.

#5 [Poloha] modul na výpočet polohy objektov na okne
Zodpovedná osoba: Martin Petluš

Cielom úlohy bolo vytvoriť prototyp, ktorý bude vypočítavať polohu textu na okne. Modul bude počítať len s ideálnou situáciou (nie prudké zmeny rýchlosti ani smeru).

#6 [OpenGL] Odskúšať objavujúce sa a miznúce objekty
Zodpovedná osoba: Peter Hamar

Cieľom tejto úlohy je vytvoriť pohybujúci sa text. Otestovať rôzne druhy zložitejších pohybov a taktiež vyskúšať si zobrazujúci a miznúci text pomocou interpolácie.

#7 [Mobil] demo pre rozpoznávanie giest
Zodpovedná osoba: Juraj Jarábek

Oboznámiť sa s projektom od kolegu Bc. Richarda Sámela. Pochopiť štruktúre projektu a pomôcť Robovi s vytvorením dema pre rozpoznávanie giest.

#8 [Kinect] hĺbkova informácia
Zodpovedná osoba: Jakub Mercz

Táto úloha zahŕňa spracovanie hĺbkovej informácie z kinectu na merateľnú jednotku (milimetre) a jej namapovanie na RGB stream, aby oba vstupy spolu korelovali.

#9 [Architektúra] Definovať aké vstupy a výstupy má každý modul
Zodpovedná osoba: všetci členovia tímu, ktorí pracujú na moduloch

Väčšina členov tímu pracuje samostatne na svojich moduloch. Pre potrebu vytvorenia komplexnej architektúry projektu je ale potrebné zadefinovať vstupy a výstupy každého modulu. Tie sa následne využijú na vytvorenie prvého návrhu architektúry.

#10 [Architektúra] Analýza a definovanie interfaces
Zodpovedná osoba: Lukáš Sekerák

Po tom čo sme definovali vstupy a vystupy modelov. Je potrebne zadefinovať interfaci medzi modulmi. Je potrebné zadefinovať aj základne entity ktoré vystupujú pri komunikácií modulov. Zároveň to všetko treba implementovať.

#11 [Databáza] Nainštalovať databázu
Zodpovedná osoba: Lukáš Sekerák

V rámci modulu databázi, je potrebné spraviť prvý krok a to vytvoriť sqlite databázu.

#16 [Mobil] demo pre rozpoznávanie gest
Zodpovedná osoba: Róbert Sabol

Ukázať aplikáciu a oboznámiť zvyšok tímu s tým ake gestá je možné využiť a aké možností pre ovládanie systému modul Android poskytne.

#17 [OpenCV] registrácia obrazu
Zodpovedná osoba: Patrik Polatsek

Pre správnu funkčnosť celého systému je dôležité, aby boli dolnkové informácie zobrazované na bočnom skle čo najpresnejšie. Preto je na začiatku potrebné skalibrovať kameru. Prístup, ktorý sa pri tom využije sa nazýva registrácia obrazu, ktorá sa pokúsi “prekryť” viaceré obrázky do jedného.

#18 [OpenGL] Práca na jednoduchej hre
Zodpovedná osoba: Peter Hamar

Keďže v projekte plánujeme vytvoriť hru, je potrebné vytvoriť nejaký jednoduchší prototyp hry na to aby sme sa naučili lepšie pracovať s OpenGL. Navrhli sme hru s náučným kontextom. Zobrazovalo sa 10 obrázkou, zobrazila sa otázka ktorá sa týkala objektu na obrázku spoločne s 3 možnosťami a úlohou bolo vybrať správu. Hráčovi sa postupne počítalo skóre.

#20 [TP Cup] Prihláška do súťaže
Zodpovedná osoba: Patrik Polatsek

Náš tím sa rozhodol zúčastniť v súťaži TP Cup 2014. K tomu je potrebné vypracovať prihlášku, kde sa uvedie základný koncept a motivácia k projektu.

#27 [OpenCV] analyzovať a použiť kódy M. Račeva
Zodpovedná osoba: Patrik Polatsek

Na detekciu objektov na obrázku je možné použiť diplomovoú prácu M. Račeva a jeho kódy. Tie treba analyzovať a podľa možností zapracovať do modulu spracovania obrazu.

#28 [OpenGL] Pridanie 3D modelu
Zodpovedná osoba: Peter Hamar

Keďže v hre plánujeme použiť 3D objekty bolo dôležité nejaký objekt načítať. Po preštudovaní formátov 3D modelov sme sa rozhodli použiť .obj súbor. Bolo potrebné skompilovať viaceré knižnice ktoré sme následne použili. Výsledkom bol pridaný 3D model lietadla s namapovanou textúrou.

#29 [OpenCV] rozoznanie objektu na obrázku
Zodpovedná osoba: Patrik Polatsek

Modul spracovania obrazu sa pokúsi rozoznať možné objekty na snímke. V tejto fáze je potrebné vyselektovať z možných kandidátov na objekty tie, ktoré sú skutočne na obrázku.

#31 [TCP] TCP modul do C++ projektu
Zodpovedná osoba: Lukáš Sekerák

V neskoršej fáze vývoja, sme sa zistili, že nie je dobré ak android komunikuje priamo s riadiacim modulom. Preto sme sam rozhodli ďalší modul, ktorý bude mať funkcionalitu TCP servera. Na tento server sa pripojí android a bude mu posielať správy. TCP server v prípade prijatia správy prepošle správu riadiacemu modulu. V tejto úlohe ide hlavne o riešenie týchto problémov a ich implementáciu.

#32 [Mobil] TCP komunikácia z Androidu
Zodpovedná osoba: Róbert Sabol

Keďže modul Android bude so systémom komunikovať cez TCP server, je potrebné implementovať komunikáciu Android modulu s TCP serverom, ktorému modul Android odošle informáciu o vykonanom geste v tvare TCP soketu.

#33 [Kinect] detekcia tváre
Zodpovedná osoba: Jakub Mercz

Pre prvú verziu detekcie bodu pozerania je potreba detekovať tvár a v tej určiť odhadom bod pozerania.

5. Tretí šprint

Začiatok šprintu: 4.11.2013

#30 Doxygen
Zodpovedná osoba: Juraj Jarábek

Naštudovať Doxygen tutoriály a vygerovať vzorovú HTML dokumentáciu. Taktiež vytvoriť súbor pravidiel dokumentovania kódu pre všetkých členov tímu.

#34 [OpenGL] Pohyby 3D modelom
Zodpovedná osoba: Peter Hamar

Cieľom je vytvoriť pohyby 3D modelu, aby lietadlo simulovalo let. Zatiaľ pomocou klávesnice. Snažiť sa o dosiahnutie čo najrealnejšieho pohybu lietadla.

#35 [Architektúra] Fake ovládanie
Zodpovedná osoba: Lukáš Sekerák

Keďže android modul, nie je ešte priprvený rozhodli sme sa nahradiť android “dump” modulom. Ktorým by sa tiež ovladalá aplikácia ale len za pomoci klávesnice.

#46 [OpenCV] úprava spracovania/registrácie obrazu
Zodpovedná osoba: Patrik Polatsek

Modul spracovania obrazu a funkciu registráciu obrazu je potrebné nateraz sfinalizovať a zapojiť do celkového projektu.

#51 [OpenCV] detekcia horizontu
Zodpovedná osoba: Patrik Polatsek

Hra s lietadlom bude spočívať v tom, že lietadlo nesmie padnúť pod horizont. Na to je potrebné horizont na obrázku zdetegovať. Táto úloha vyžaduje analýzu možných detekcií horizontu a výber najvhodnejšej metódy pre tento projekt.

#54 [OpenGL] Pripravenie scény za lietadlom
Zodpovedná osoba: Peter Hamar

Na to aby sme mohli verne simulovať let lietadla je potrebné pripraviť scénu. Cieľom je zobrazenie videa priamo za modelom lietadla. Video sa bude zobrazovať za pomoci obrázkov resp. textúr v reálnom čase. Taktiež už bude možný pohyb po scéne.

#55 [OpenGL] Držať lietadlo na horizontom
Zodpovedná osoba: Peter Hamar

Cieľom je držať lietadlo nad horizontom. Lietadlo prirodzene klesá k zemi a hráč ho stlačením klávesy dvíha vyššie. Horizont v tomto bode ešte nebude reálny, len nejaký ukážkový.

#50 [Kinect] Výpočet súradníc bodu z pohľadu kinectu
Zodpovedná osoba: Jakub Mercz

Cieľom je preštudovať a implementovať výpočet súradníc v priestore na základe súradníc na obrazovke a hĺbkovej informácie.

#44 [Kinect] Rozpoznávanie hlavy
Zodpovedná osoba: Jakub Mercz

Pre detekciu aj v prípade že nie je možné detekovať tvár, je treba implementovať detekovanie celej hlavy z hĺbkovej mapy.

#59 Inštalácia gerritu
Zodpovedná osoba: Martin Petluš

Nainštalovať na server program gerrit pre tímové code review.

#52 [Architektúra] Vygenerovanie diagramov a dokumentácie

V rámci tohto šprintu vznikla potreba vygenerovanie diagramov a dokumentácie zo zdrojovéko kódu. V tejto úlohe zodpovedný člen si mal naštudovať možnosti ako generovať dokumentáciu z CPP kódu. Výsledné diagramy mali byť použité na ďalšom stretnutí pri prezentácii.

#53 [Databáza] OOP Mapovač a DB Service
Zodpovedná osoba: Lukáš Sekerák

V tejto úlohe bolo potrebné dokončiť OOP Mapovač, teda dodefinovať všetky entity, dokončiť a otestovať implementáciu OOP Mapovača. Ďalej bolo potrebné celú logiku databázového modulu zakryť do služby, ktorá by bola prístupná ostatným modulom.

#60 [Dokumentacia] Projektová dokumentácia a Dokumentácia riadenia
Zodpovedná osoba: všetci členovia tímu

K prvému kontrolnému bodu je potrebné opísať priebežný stav projektu a opísať jednotlivé moduly riešenia. Rovnako je potrebné opísať priebeh manažovania v našom tíme.

6. Popis projektu (4. - 6. šprint)

Popis projektu v tejto kapitole zahŕňa štvrtý až šiesty šprint. Sú tu opísané zmeny v celkovej architektúre ako i v jednotlivých moduloch, ktoré vznikli po prvom kontrolnom bode. Ku každej časti sú v závere spomenuté úlohy, ktoré sa k jej vypracovaniu viažu. Prehľad jednotlivých úloh je opísaný po šprintoch v ďalších kapitolách.

6.1 Architektúra systému

V tejto časti došlo k malým zmenám v architektúre. V hlavnom programe pribudla podpora viacerých vlákien, takže mnohé moduly môžu bežať paralelne. V rámci tejto novinky samozrejme pribudla správa vlákien a potrebná synchronizácia.

Druhou dôležitou novinkou bola podpora konfigurácie programu a jednotlivých modulov. Počas vývoja vznikla potreba mnohé časti kódu konfigurovať, načítavať konštanty zo súboru a pod. Keďže viaceré moduly potrebovali takúto konfiguráciu, dohodli sme sa, že je to spoločná vlastnosť a má to riešiť architektúra, ktorá by modulom poskytla konfiguračné nastavenia.

V tejto fáze sa naďalej pokračovalo v integrácii modulov, na konci tejto fázy by mal byť prototyp hotový a prepojený so všetkými modulmi. Dôležitou úlohou architekta bola najmä podpora, vysvetlenie prepojenia a pomoc pri generovaní modulov.

6.1.1 Súvisiace úlohy

#66 [Carlos] Integracia modulov
#65 [Architektura] Konfiguracia pre Carlos
#56 [Architektura] Riadiaci modul

6.2 Modul spracovania obrazu

Modul spracovania obrazu vykonáva funkcie z oblasti počítačového videnia, ktoré spracúvajú obraz zo vstupu.

Tento modul v súčasnosti poskytuje nasledovné funkcie:

- *detekcia objektov*
- *kalibrácia kamery*
- *detekcia horizontu*

V tejto kapitole je podrobne opísaná **detekcia horizontu**, ktorá predstavuje novú funkcionalitu tohto modulu. Detekcia horizontu bola vytvorená pre potreby hry *Lietadlo*, ktorá je podrobnejšie opísaná v časti *6.5 Modul rozšírenej reality*. Táto funkcia na vstupnom obraze zdeteguje horizont a nájde tie časti, ktoré zodpovedajú oblohe.

6.2.1 Vstupy

- stream z kamery

6.2.2 Výstupy

- bitmapa veľkosti streamu, kde biele pixely reprezentujú oblohu

6.2.3 Analýza

Jeden z možných spôsobov detekcie horizontu je využitie hranového detektora. Na detekciu hrán sme vyskúšali nasledovné metódy:

- *Canny*
- *Prewitt*
- *Sobel*

Tieto metódy sme vyskúšali aplikovať na obrázkoch, kde je prítomný horizont pri rôznych podmienkach. Najlepšie výsledky dosahoval *Canny detektor*, ktorý bol následne implementovaný do tohto modulu.

6.2.4 Návrh

Na čiernobiely obrázok je najprv aplikovaný Gaussian Blur filter na potlačenie šumu. Potom sa pomocou Canny detektora nájdu hrany na obrázku, čím vznikne binárny obraz s detegovanými hranami. Treshold pri tomto detektore je dostatočne vysoký, aby na oblakoch neboli rozoznané hrany. Na tomto obrázku sa potom nájdu kontúry, pričom algoritmus predpokladá, že nad najvrchnejšou kontúrou sa nácháza obloha.

Bitmapa oblohy sa zostrojí nasledovne. Obraz s kontúrami sa prechádza postupne zhora po stĺpcoch. Algoritmus zafarbuje pixely na bielo, pokým nenájde prvú kontúru v danom stĺpci. Zvyšné pixely sú potom čierne.

Obr.1 Pôvodný obrázok a obrázok s detegovanou oblohou (biela farba)

6.2.5 Implementácia

Táto funkcia bola implementovaná v C++ s knižnicou OpenCV. Funkcia, ktorá príjme vstupný obraz, vráti binárny obrázok so zdetegovanými regiónom oblohy.

V tomto štádiu funkcia predpokladá, že na obrázku je vždy prítomný horizont a obloha.

6.2.6 Testovanie

Na testovanie bolo použité viacero obrázkov s prítomnou oblohou zachytených prevažne pri dobrých svetelných podmienkach. Funkcia bola otestovaná na obrázkoch s jasnou oblohou ako i oblačnou.

6.2.7 Súvisiace úlohy

#51 [OpenCV] detekcia horizontu
#68 [OpenCV] fake funkcionality modulu spracovania obrazu
#69 [OpenCV] funkcia vracajúca polohu oblohy
#70 [OpenCV] úprava funkcionality detekcie objektov do projektu
#81 [OpenCV] finalizácia detekcie horizontu

6.3 Modul Kinect

V module kinectu pokračoval vývoj v smere zisťovania súradníc v reálnom priestore, aby bolo možné odosielať reálne údaje na ďalšie spracovanie. V tejto kapitole je opísaná inicializačná fáza programu, ktorá tvorí hlavnú časť potrebnú pre zisťovanie týchto súradníc.

6.3.1 Vstupy

- RGB video stream z Kinectu
- stream hĺbkových informácií z Kinectu
- 4 nekolineárne body v reálnom priestore (ich súradnice)

6.3.2 Výstupy

- transformačná matica M

6.3.3 Analýza

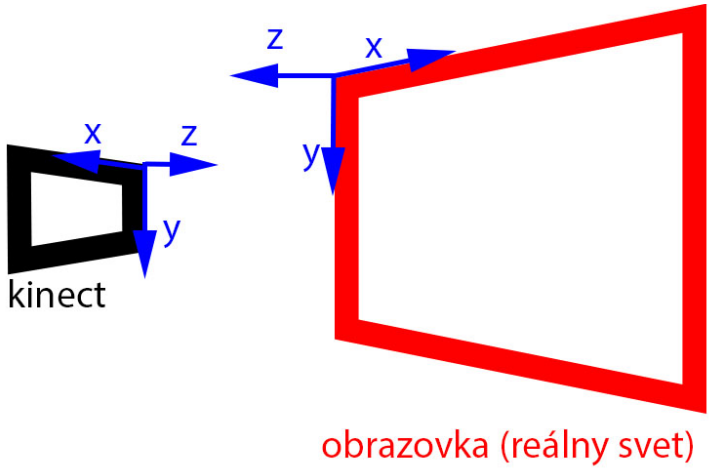
Modul doteraz bol schopný zaznamenať súradnice bodov len z pohľadu kinectu. Pre účely celého systému je ale potreba prispôsobiť tieto súradnice inej súradnicovej sústave, s ktorou už systém pracuje. Preto je potrebné nájsť transformáciu medzi bodmi týchto 2 súradnicových sústav. Na vypočítanie tejto tranformácie je potrebné získať 4 nekolineárne body z každej z týchto sústav. Tieto body usporiadané do matíc vyhovujú modelu transformácie v 3D pristore, ktorého rovnica je:

$$y = M*x$$

kde y je matica súradníc výslednej sústavy, x je matica súradníc východzej sústavy a M je daná transformačná matica. Úpravou tejto rovnice dostaneme rovnicu

$$M = y*x^{-1}$$

v ktorej už poznáme alebo vieme vypočítať všetky členy na pravej strane rovnice.



Obr.1 Náčrt riešeného problému

6.3.4 Návrh

Modul počas inicializácie štyrikrát zaznamená súradnice terča, ktorého stred sa vždy nachádza v bode odpovedajúcom bodu v súradniciach reálneho sveta. Po zaznamenaní všetkých 4 bodov prebehne výpočet transformačnej matice, ktorá sa uloží pre ďalšie použitie v programe.

6.3.5 Implementácia

Pri implepmentácii boli využité funkcie knižnice OpenCV, ktorá ponúka inverziu matíc a tiež násobenie matíc - obe funkcie potrebné pre výpočet transformačnej matice.

Detekcia bodov z pohľadu kinectu je vykonávaná pomocou detekcie terčíka pozostávajúceho z niekoľkých sústredných kruhov. Na detekciu je využitá funkcia HoughCircles z knižnice OpenCV schopná detekovať stred kruhu.

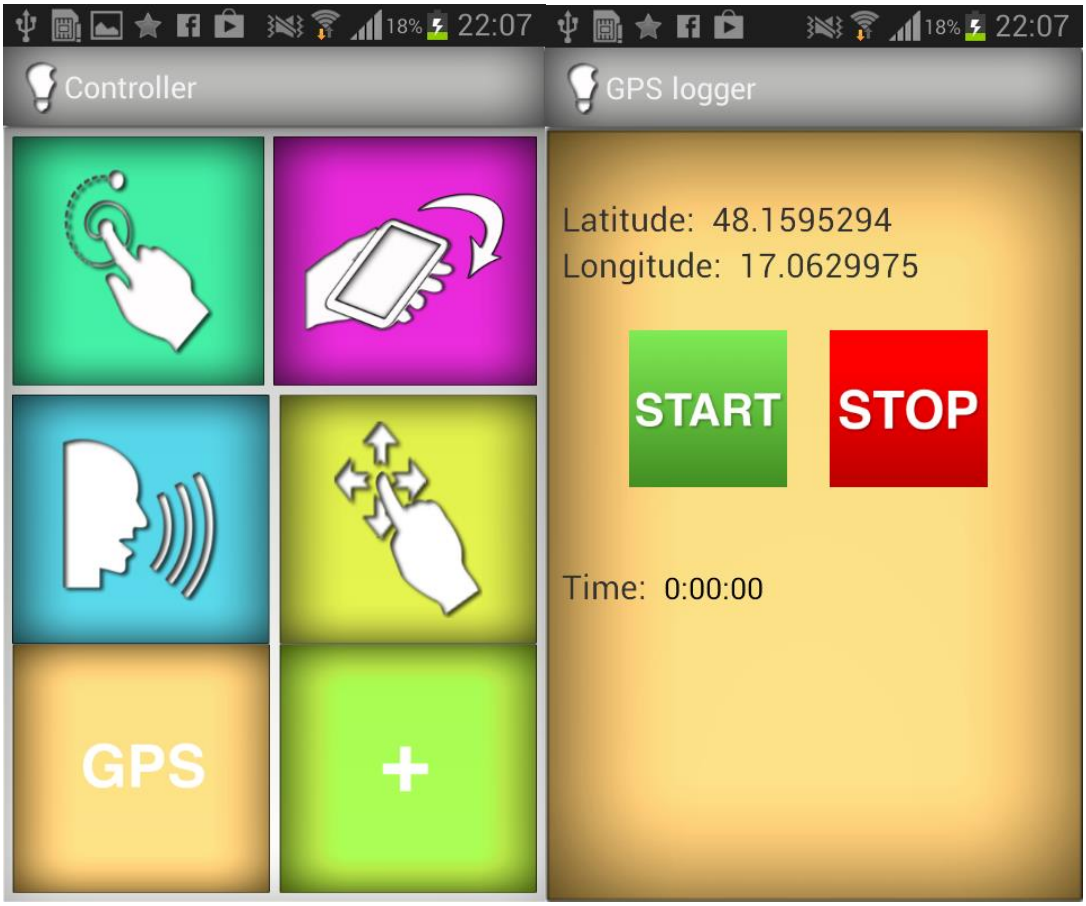
Na ukotvenie bodov v reálnom priestore bol použitý systém nitiek. Tento systém využíva možnosť definovať bod v priestore 3 vektormi a tvoria ho teda pre každý bod 3 nitky, z ktorých každá je uchytená v inom bode. Pre bod so súradnícami [x, y, z] by to boli nitky:
1. uchytená v bode [0, 0, 0] s dĺžkou ako vektor [x, y, z]
2. uchytená v bode [x, 0, 0] s dĺžkou ako vektor [0, y, z]
3. uchytená v bode [0, y, 0] s dĺžkou ako vektor [x, 0, z]

6.3.6 Súvisiace úlohy

#72 [Kinect] Dummy modul
#76 [Kinect] Výpočet transformačnej matice
#89 [Kinect] Detekcia bodov pre inicializáciu
#90 [Kinect] Určenie vstupných bodov pre inicializáciu v reálnom priestore

6.4 Modul Android

V tomto module bolo vykonaných niekoľko menších zmien a taktiež boli do aplikácie doplnené ďalšie funkčné vlastnosti. Jednou z nich bolo začlenenie GPS zaznamenávanie zo samostatnej aplikácie priamo do hlavnej Android časti. S týmto začlenením súvisel aj grafický návrh GPS časti aplikácie tak, aby korešpondovala s grafickým návrhom celej aplikácie. S pridávaním GPS modulu do aplikácie súvisel aj refaktoring pôvodného zdrojového kódu.



Obr.1 Obrazovky Android časti

Druhým doplnkom aplikácie bolo vytvorenie záložného ovládania, ktoré bude slúžiť ako sekundárne ovládanie pre hru. Týmto doplnkom sme chceli hlavne predísť neúspešnému ovládaniu hry z dôvodu zlyhania niektorého z predchadzajúcich možností ovládania - hlasové, dotykové alebo pohybové ovládanie.

Ďalšou zmenou v tomto module bolo prepracovanie serverovej časti z dôvodu nesprávnej funkčnosti. Serverovú časť sa podarilo úspešne implementovať tak, aby posielala správne dáta TCP serveru.

Neočakávanú chybu v časti rozpoznávania hlasu bolo potrebné opraviť a implementovať časť rozpoznávania hlasu inak.

6.4.1 Testovanie

Implementované časti boli otestované úspešne. Špeciálne bolo otestované ovládanie hry pomocou dotykového ovládania, ktoré dopadlo kladne.

6.4.2 Súvisiace úlohy

#78: [Mobil] Prerobit server cast
#73: [Mobil] Pridanie GPS modulu
#74: [Mobil] Prisposobenie vzhladu GPS
#75: [Mobil] Vytvorit zalozne ovladanie
#77: [Mobil] Voice recognition
#71: [Mobil] Refaktoring GPS logger
#83: [Mobil] Pripravit GPS instalacku na web
#84: [Mobil] Otestovanie ovladania hry
#108: [Mobil] vlozenie doxygen komentarov

6.4.3 Komunikácia - Modul Android a Riadiaci modul

Pri klasickom testovaní na statickom mieste android zariadenie posiela rovnakú GPS súradnicu. Keďže testovanie v teréne nie je vždy možné vytvorili sme si GPS súbor s raz nahranými súradnicamy. Tieto GPS súradnice sa teda načítavajú zo súboru a táto funkcionalita je implementovaná v TCP serveri, aby čo najviac simulovala posielanie súradnic z androidu. Táto časť sa naimplementovala a otestovala.

Druhou veľkou úlohou bolo konečné prepojenie android zariadenia s tcp serverom a následne preposielanie správ s tcp servera do hlavného modulu. Táto úloha čiastočne zasahuje do celkovej architektúry a aj do modulu hry. Preto sa na tejto úlohe podieľal architekt a aj človek zodpovedný za modul hry. Na začiatku tejto fázy bol TCP server pripravený, dôležitou úlohou bolo definovanie správ, ktoré sa budú posielať a otestovanie posielania všetkých správ.

6.4.3.1 Súvisiace úlohy

#67 [Android] Fake gps pozicia
#57 [Architektura] Spracovanie sprav z Androidu
#105 [Architektura] Prepojenie modulu hry - tcp servera - androidu

6.5 Modul rozšírenej reality

Modul rozšírenej reality vykonáva funkcie z oblasti počítačovej grafiky, ktoré vykresľujú upravený obraz zo vstupu.

Tento modul v súčasnosti poskytuje nasledovné funkcie:

- *zobrazenie scény s 3D modelom*
- *zobrazenie videa - prostredia, v ktorom sa nachádza auto*
- *prepínanie medzi stavmi hry - úvodná obrazovka, hranie hry, game over, skóre*
- *implementovanú hernú logiku, zatiaľ bez horizontu*

V tejto kapitole je podrobne opísaná pridaná funkcionalita počas šprintov 4 - 6.

6.5.1 Vstupy

- obrázky z videa
- ovládanie z android časti

6.5.2 Výstupy

- zobrazené video na bočnom skle auta ovládané pomocou mobilu - android

6.5.3 Analýza

Logiku hry Lietadlo je potrebné naimplementovať čo najlepšie, aby sme dosiahli čo najreálnejšie výsledky. Je potrebné brať do úvahy fakt, že lietadlo má určité rozmery a stačí ak sa akoukoľvek časťou lietadla dotkne okraja resp. horizontu. Následne, by hra mala skončiť.

Celú hru Lietadlo je potrebné rozkúskovať do viacerých stavov a zamyslieť sa aj nad spôsobom reštartu.

6.5.4 Návrh

Ako najvhodnejšie riešenie logiky hry je celé lietadlo zabaliť do kvádra a zisťovať či sa tento kváder dotýka pozadia. Ak áno, tak lietadlo je na hernom pláne. Ak nie, tak hra končí, lebo vyletelo mimo obraz. Obdobne je to potrebné implementovať aj pre horizont.

Navrhované stavy sú:

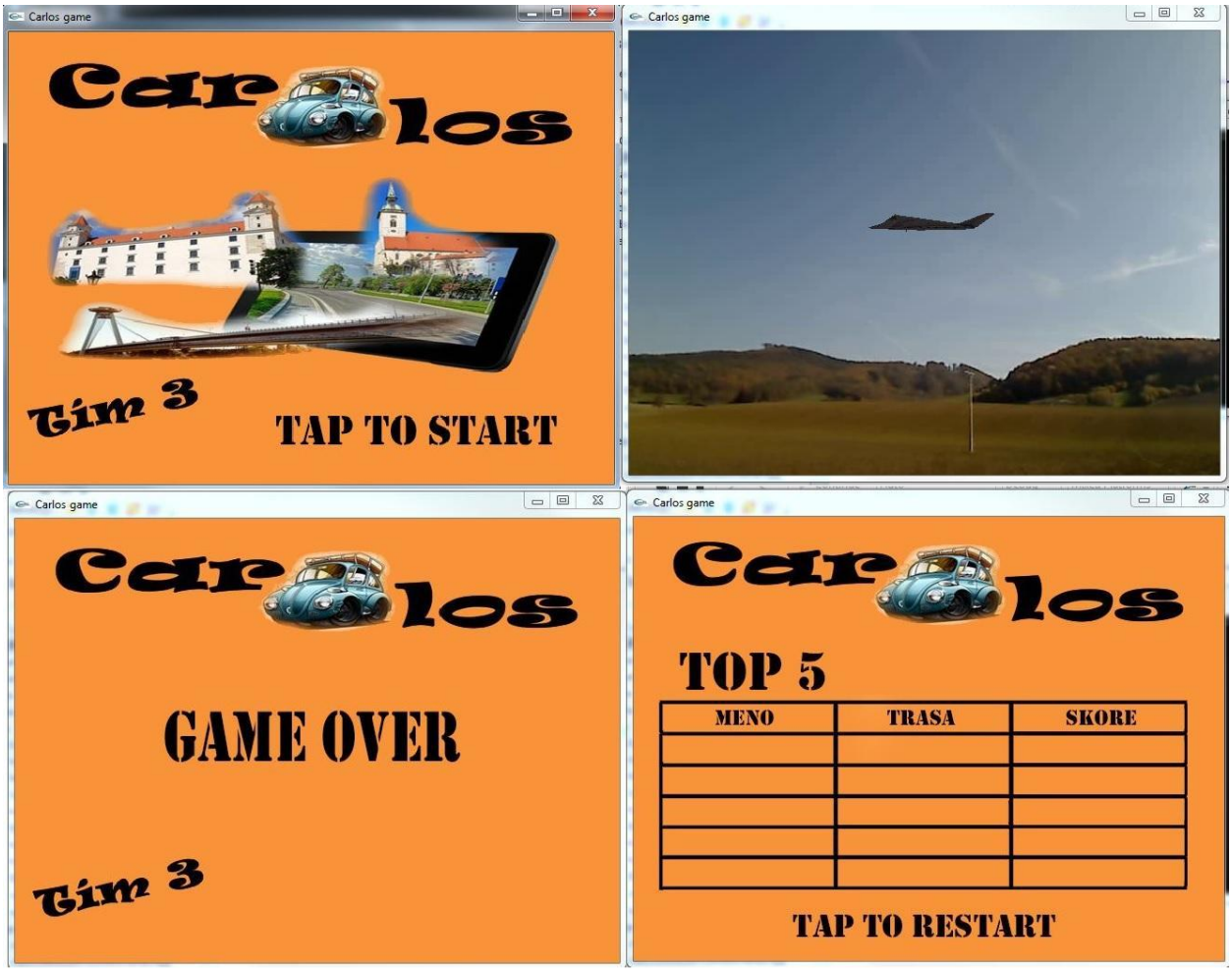
- úvodná obrazovka
- hranie hry
- game over
- obrazovka so skóre

Taktiež je potrebné upraviť spôsob ovládania, prepojiť sa s Lukášovou časťou a začať prijímať ovládanie z jeho modulu.

6.5.5 Implementácia

V tomto momente už je naimplementovaná počiatočná logika hry, ktorá zisťuje, kde sa lietadlo nachádza. Sú vytvorené všetky navrhnuté stavy zo sekcie 6.6.3. Hru je už možné hrať, aj keď nie je zobrazený horizont a nepočíta sa ani skóre.

Ovládanie sa už prijíma z hlavného modulu, ktorý je opísaný v sekcii 2.1.4.



Obr.2 Jednotlivé stavy hry - Lietadlo. Vľavo hore - úvodná obrazovka, vpravo hore - hranie hry, vľavo dole - game over a vpravo dole - obrazovka skóre.

6.5.6 Testovanie

Na testovanie bolo použité video, ktoré sme si nahrali. Otestovali sme aj jednotlivé stavy hry, prijímanie obrázkov a príkazov. Tieto príkazy sa používajú na ovládanie lietadla. Testovanie dopadlo podľa očakávaní a prípadne problémy boli odstránené.

6.5.7 Súvisiace úlohy

#61 [OpenGL] Dokoncit zobrazenie videa
#63 [OpenGL] Vytvorenie hernej logiky
#64 [OpenGL] Zobrazenie textu v hre
#103 [OpenGL] Napojenie na git, gerrit a doxygen komentare, spojenie s lukasovou castou
#104 [OpenGL] Opravenie spojenia s lukasovou castou
#86 [OpenGL] Vlozenie doxygen komentarov
#87 [OpenGL] Nastavenie kniznic
#88 [OpenGL] Prepojit sa s robom - ovladanie pomocou mobilu

6.6 Modul databázy

V module databázy bola doimplenetovaná databázova služba, ktorá pokrýva všetku prácu nad databázou. Databáza bola naplnená potrebnými údajmi. Tieto nové časti boli riadne otestované a pre prvú verziu prototypu je táto časť hotová.

6.6.1 Súvisiace úlohy

#53 [Databaza] OOP Mapovac a DB Service

7. Štvrtý šprint

Začiatok šprintu: 18.11.2013

#56 [Architektúra] Riadiaci modul
Zodpovedná osoba: Lukáš Sekerák

Náš hlavný program beží pod určitým operačným systémom. Keďže aplikácia beží neustále, bolo potrebné naimplementovať cyklus, v ktorom sa aplikácia nezastaví, teda len za určitých podmienok. Zároveň v rámci tohto cyklu musí aplikácia pravidelne komunikovať s oper. systémom. V tejto úlohe sa riešili vyššie spomenuté problémy.

#57 [Architektúra] Spracovanie správ z Androidu
Zodpovedná osoba: Lukáš Sekerák

Mnohé správy, ktoré sa príjmajú z android zariadenia majú textovú formu. Zvyšok modulov však používa správy definované v ENUM. Preto je v tejto úlohe potrebné tieto správy prerobiť. Zároveň takáto komunikácia bude prebiehať cez viacero vlákien a túto operáciu treba synchronizovať.

#61 [OpenGL] Dokončiť zobrazenie videa
Zodpovedná osoba: Peter Hamar

Na bočné sklo auta je potrebné zobraziť to, čo je práve vidieť za oknom. Preto musíme v hre zobrazovať na pozadí video. Video musíme rozkúskovať do obrázkov a tie následne zobrazovať. Problém bol vyriešený iba sčasti, keďže pre správny beh boli potrebné vlákna. Dokončené v nasledujúcom šprinte.
#63 [OpenGL] Vytvorenie hernej logiky
Zodpovedná osoba: Peter Hamar

Bolo potrebné začať vytvárať hernú logiku a určovať pozíciu lietadla v rámci obrazovky. Taktiež sme v tomto bode rozdelili hru do viacerých stavov a dané stavy na seba vzájomne poprepájali. Týmto sme dosiahli to, že hru už je možné hrať.

#64 [OpenGL] Zobrazenie textu v hre
Zodpovedná osoba: Peter Hamar

Keďže sme si definovali do akých stavov sa môže hra dostať, bolo potrebné si pripraviť jednotlivé textúry - nakresliť ich. Vytvoriť spôsob reštartu hry a taktiež sa zamerať na to, aby bolo možné jednoduchým spôsobom zobraziť text v okne.

#65 [Architektúra] Konfigurácia pre Carlos
Zodpovedná osoba: Lukáš Sekerák

Počas vývoja vznikla potreba mnohé časti kódu konfigurovať, načítavať konštanty zo súboru a pod. Keďže viaceré moduly potrebovali takúto konfiguráciu, dohodli sme sa, že je to spoločná vlastnosť, ktorú má riešiť architektúra, ktorá by modulom poskytla konfiguračné nastavenia. V tejto úlohe sa riešila konfigurácia.

#66 [Carlos] Integrácia modulov
Zodpovedná osoba: Lukáš Sekerák

V tejto fáze sa naďalej pokračovalo v integrácii modulov. Dôležitou úlohou architekta je v tejto úlohe najmä podpora, vysvetlenie prepojenia a pomoc pri generovaní modulov.

#67 [Android] Fake gps pozícia
Zodpovedná osoba: Lukáš Sekerák

Pri klasickom testovaní na statickom mieste android zariadenie posiela rovnakú GPS súradnicu. Keďže testovanie v teréne nie je vždy možné, vytvorili sme si GPS súbor s raz nahranými súradnicami. Tieto GPS súradnice sa teda majú načítavať zo súboru a táto funkcionalita má byť naimplementovaná v TCP servery, aby čo najviac simulovala posielanie súradnic z androidu.

#68 [OpenCV] fake funkcionality modulu spracovania obrazu
Zodpovedná osoba: Patrik Polatsek

Cieľom tejto úlohy je definovať formát funkcií, ktoré budú volané z riadiacej časti. Ďalej je potrebné vytvoriť fake verzie týchto funkcií.

#69 [OpenCV] funkcia vracajúca polohu oblohy
Zodpovedná osoba: Patrik Polatsek

V úlohe #51 sa vybral na detekciu horizontu Canny hranový detektor. V tejto úlohe je potrebné vytvoriť reálnu funkciu, ktorá vráti polohu oblohy pre potrebu hry “Lietadlo”.

#70 [OpenCV] úprava funkcionality detekcie objektov do projektu
Zodpovedná osoba: Patrik Polatsek

Vytvorenú funkciu detekcie objektov v rámci modulu spracovania obrazu je potrebné zakomponovať do spoločného solution.

#103 [OpenGL] Napojenie na git, gerrit a doxygen komentáre, spojenie s Lukášovou časťou
Zodpovedná osoba: Peter Hamar

Cieľom tejto úlohy bolo začať používať git a gerrit. Zdrojový kód je potrebné komentovať pomocou nástroja doxygen, čiže bolo potrebné povkladať komentáre do kódu. Pre zobrazenie videa bolo potrebné sa prepojiť s hlavnou (Lukášovou) časťou. Toto prepojenie nefungovalo úplne správne a bude potrebná oprava v naslednujúcom šprinte.

#95 [Support] Nainštalovať programy na pc v škole
Zodpovedná osoba: Martin Petluš

Na novo zakúpený pc bolo treba nainštalovať vývojové prostredia pre C++/C, Javu. Tiež bolo treba nainštalovať a nastaviť OpenCV ako aj všetky ostatné programy nutné pre základnú prácu s pc.

#94 [Support] Dokončenie Gerritu
Zodpovedná osoba: Martin Petluš

Dokončiť inštaláciu Gerritu na serveri. Otestovať jeho funkčnosť a prihlásenie nových používateľov. Napísať návod na jeho použitie ostatnými členmi tímu.

8. Piaty šprint

Začiatok šprintu: 28.11.2013

#71 [Mobil] Refaktoring GPS logger
Zodpovedná osoba: Róbert Sabol

Pri začlenovaní GPS aplikácie do hlavnej Android časti sa našli isté nedostatky v pomenovaní premenných, preto bolo potrebné kód zrefaktorovať.

#72 [Kinect] Dummy modul
Zodpovedná osoba: Jakub Mercz

Pre možnosť fungovania systému aj bez potreby fyzického zariadenia kinectu a jeho konfigurácie, bol vytvorený dummy modul odosielajúci falošné informácie.

#73 [Mobil] Pridanie GPS modulu
Zodpovedná osoba: Róbert Sabol

Tažisko tejto úlohy bolo hlavne v tom, aby samostatne vytvorená GPS aplikácia bola zahrnutá v hlavnej Android aplikácii Controller. Bolo potrebné vytvoriť novú aktivitu v aplikácii ako aj xml súbory na layout a pod.

#74 [Mobil] Prispôsobenie vzhladu GPS
Zodpovedná osoba: Róbert Sabol

Keďže grafika aplikácie je navrhnutá tak, aby všetky aktivity boli podobne graficky spracovane, bolo potrebné pridanú GPS časť graficky upraviť tak, aby korešpondovala ku grafike celej aplikácie.

#75 [Mobil] Vytvoriť záložné ovládanie
Zodpovedná osoba: Róbert Sabol

Aj keď ovládanie pomocou pohybových senzorov telefónu alebo pomocu rôznych dotykových gest, je veľmi efektné’pre koncového používateľa, pre eliminovanie rizika zlyhania ovládania pomocou týchto senzorov, sme boli nútení vytvoriť záložné ovladanie, ktoré obsahuje len tlačidlá s danými akciami.

#76 [Kinect] Výpočet transformačnej matice
Zodpovedná osoba: Jakub Mercz

Implementácia a anlýza možností transformácie medzi 2 súradnícovými sústavami, ktorej výsledkom bola transformačná matica M

#77 [Mobil] Voice recognition
Zodpovedná osoba: Róbert Sabol

Pri testovaní hlasových povelov nastal problém pri rozpoznávaní príkazov preto bolo potrebné ošetriť isté chybové stavy.

#78 [Mobil] Prerobiť server časť
Zodpovedná osoba: Róbert Sabol

Dodaná serverová časť v Android aplikácii nevyhovovala naším potrebám, keďže pracovala s JSON-mi, preto sa úplne prerobila na TCP komunikáciu, ktorá korektne dokázala komunikovať s C++ serverom.

#81 [OpenCV] finalizácia detekcie horizontu
Zodpovedná osoba: Patrik Polatsek

Na základe poznatkov získaných pri úlohe #51 a #69 dokončiť funkciu, ktorá bude na vstupnom obraze hľadať polohu oblohy.

#82 [OpenCV] začlenenie modulu spracovania obrazu do projektu
Zodpovedná osoba: Patrik Polatsek

Začleniť všetky 3 funkcie modulu spracovania obrazu do spoločného solution.

#96 [OpenCV] Git - použiť systémové premenné namiesto priamych ciest (OpenCV)
Zodpovedná osoba: Martin Petluš

Odskúšať, či sa dajú použiť v projekte Visual Studia systémové premenné ku knižnici OpenCV namiesto absolútnej cesty k OpenCV.

#104 [OpenGL] Opravenie spojenia s Lukášovou časťou
Zodpovedná osoba: Peter Hamar

Je potrebná oprava spojenia medzi modulom hry a hlavnou časťou. Nefungovali správne 2 okná súčasne - textúry.

9. Šiesty šprint

Začiatok šprintu: 5.12.2013

#83 [Mobil] Pripravit GPS inštalačku na web
Zodpovedná osoba: Róbert Sabol

Kvôli potrebám natáčania testovacieho videa bolo potrebné aby GPS logger fungoval ako samostatná aplikácia. Preto bol modul GPS vyňatý z Controller-a a skompilovaný do samostatnej APK aplikácie, ktorá sa dá jednoducho nainštalovať.

#84 [Mobil] Otestovanie ovládania hry
Zodpovedná osoba: Róbert Sabol

Po sfunkčnení hry bolo potrebné otestovať, či ovládanie pomocou Android aplikácie bude dostatočne dobré a rýchle pre naše potreby. Android aplikáciu sme s hrou spojili TCP serverom tak, že aplikácia vytvorila spojenie so serverom a odosielala TCP sokety serveru, ktorý ich následne odosielal modulu s hrou.

#86 [OpenGL] vloženie doxygen komentárov
Zodpovedná osoba: Peter Hamar

Po rozšírení a úprave kódu bolo potrebé opraviť a dopísať doxygen komentáre.

#87 [OpenGL] Nastavenie knižníc
Zodpovedná osoba: Peter Hamar

Keďže modul hry - Lietadlo používa vlastné knižnice a chceme mať projekt uložený na gite, aby si ho vedeli spustiť aj ostatní členovia tímu, je potrebné v projekte použiť systémové premenné. Taktiež uploadnuť tieto knižnice na google drive pre ostatných členov tímu a dať im zoznam systémových premenných ktoré potrebujú.

#88 [OpenGL] prepojiť sa s Robom - ovládanie pomocou mobilu
Zodpovedná osoba: Peter Hamar

Doposiaľ bolo ovládanie riešené pomocou klávesnice. Teraz je potrebné sa prepojiť s časťou android, cez hlavný - riadiaci modul - ovládanie už bude posielané spolu s obrázkom.

#89 [Kinect] Detekcia bodov pre inicializáciu
Zodpovedná osoba: Jakub Mercz

Detekcia bodov pomocou kinectu pre účely transformačnej matice.

#90 [Kinect] Určenie vstupných bodov pre inicializáciu v reálnom priestore
Zodpovedná osoba: Jakub Mercz

Určenie bodov v reálnom priestore pre úcely transformačnej matice

#91 [Support] Git spraviť na PC v škole - rozbehať projekt
Zodpovedná osoba: Martin Petluš

Nastaviť Git v škole na náš projekt. Nastaviť všetky systémové premenné. Otestovať, či sa dá projekt spustiť.

#97 [OpenCV] opraviť modul spracovania obrazu
Zodpovedná osoba: Patrik Polatsek

Pri testovaní funkcie detekcie objektov bolo objavených niekoľko chýb. Cieľom tejto úlohy bolo zistené úlohy opraviť.

#98 [OpenCV] opraviť chybu s heap vo vs 2012
Zodpovedná osoba: Patrik Polatsek

Po presune modulu spracovania obrazu do spoločného solution bola objavená chyba OpenCV vo vs 2012 na 64bit systémoch pri vykonávaní niektorých funkcií akou je napríklad detekcia keypointov a ich opis do matice deskriptorov a označená ako RtlFreeHeap error. Je potrebné zabezpečiť kompatibilitu spomínaných funkcií aj po presune modulu z vs 2010 na vs 2012.

#101 [OpenCV] vloženie doxygen komentárov
Zodpovedná osoba: Patrik Polatsek

Celý projekt modulu spracovania obrazu je potrebné okomentovať doxygen značkami.

#102 [Dokumentacia] 2. kontrolny bod
Zodpovedná osoba: všetci členovia tímu

K druhému kontrolnému bodu je potrebné znovu odovzdať Projektovú dokumentáciu a Dokumentáciu riadenia, kde treba uviesť nové informácie a zmeny oproti prvému kontrolnému bodu, ktoré boli zadokumentované v rámci úlohy #60.

#105 [Architektura] Prepojenie modulu hry - tcp servera - androidu
Zodpovedná osoba: Lukáš Sekerák

V rámci dokončenia prepojenia modulu hry - tcp servera - androidu je potrebné dodefinovať správy, ktoré sa budú posielať. Tieto správy poslať z android zariadenia do tcp servera a odtiaľ do modulu hry. Všetko je potrebné riadne otestovať a zabezpečiť.

#105 [Architektura] Doxygen komentare
Zodpovedná osoba: Lukáš Sekerák

Niektoré časti zdrojového kódu nie sú riadne okomentované. Je potrebné doplniť anotáciu pre doxygen. Komentáre, ktoré existujú treba skontrolovať, či spĺňajú anotáciu.

#107 [TextPosModule] Vloženie Doxygen komentárov
Zodpovedná osoba: Martin Petluš

Okomentovať zdrojové kódy v module Doxygen značkami.

#108: [Mobil] vlozenie doxygen komentarov
Zodpovedná osoba: Róbert Sabol

Celý projekt modulu Android bolo potrebné okomentovať doxygen značkami.

10. Popis projektu (7. - 10. šprint)

Popis projektu v tejto kapitole zahŕňa siedmy až desiaty šprint tímového projektu. Sú tu opísané všetky zmeny v architektúre a v jednotlivých moduloch, ktoré sa vykonali počas letného semestra tímového projektu. Ku každej časti sú v závere spomenuté úlohy, ktoré sa k jej vypracovaniu viažu. Prehľad jednotlivých úloh je opísaný po šprintoch v ďalších kapitolách.

10.1 Architektúra systému a riadiaci modul

V architektúre systému sa väčšina prepojení medzi modulmi nezmenila. Oproti minulému semestru sa zmenili len špecifické časti. Tieto zmeny súviseli najmä so zmenou požiadaviek. Napríklad sa zmenil spôsob posielania videa, kde sa rozhodlo, že je oveľa vhodnejšie posielať video v slučke. Teda po ukončení videa sa video opätovne spustí, aby sme tak simulovali neustálu jazdu automobilu.

Ďalej v hernom systéme pribudli nové stavy hry. Týchto stavov bolo pomerne veľa a zložitosť tejto časti narastala. Preto sa opäť rozhodlo, že je vhodnejšie pripraviť systém na spravovanie týchto stavov a prepínanie medzi nimi uľahčiť. Vďaka tomuto systému automatického spravovania, mohol vývojár implementovať nový stav oveľa rýchlejšie. Staré stavy prešli refaktorizáciou.

10.1.1 Súvisiace úlohy

#127 [Kamera] Osetrit koniec streamu z kamery
#128 [TCP] Osetrit chyby
#129 [TCP] Pridat funkcionalitu
#130 [Databaza] Implementacia selektu
#131 [Repozitar] Spravit poriadok s kniznicami
#135 [Carlos] Refaktorizacia kodu
#147 [Carlos] Testovanie
#151 [Carlos] Stream kamery
#168 [Hra] Zrefaktorovat stavy
#173 [Kamera] Video zacyklit

10.1.2 Zhrnutie

Architektúra bola navrhnutá podľa úvodnych požiadaviek. Bola pripravená robustne, aby dokázala pružne reagovať na zmeny. V prvom semestri teda potrebovala vysokú pozornosť. V druhom semestri sme mali možnosť viackrát vidieť, že takýto systematický a plánovaný prístup nám pomohol. Vďaka nemu sme mnohé časti kódu implementovali rýchlo, bez chýb a bez problémov v prepojení modulov.

V budúcnosti nie je potrebné túto časť ďalej vylepšovať. To iba v prípade, ak dôjde k ďalším zmenám požiadaviek. Riadiaci modul je dokonca pripravený na nasadenie do reálneho prostredia.

10.2 Modul spracovania obrazu

Objekty ako dopravné značky a stromy spôsobovali v hre Lietadlo prílišný nárast horizontu, na čo používateľ nestihol včas zareagovať. Preto bola v module spracovania obrazu počas letného semestra upravená *detekcia horizontu*.

Existujúca detekcia objektov pomocou štandardných Surf deskriptorov nebola rýchlostne ani kvalitatívne postačujúca. Preto bol zmenený spôsob *detekcie objektov* a rovnako došlo k dôkladnejšiemu testovaniu tejto detekcie.

10.2.1 Analýza

Detegovaný horizont je potrebné dodatočne spracovať, aby z neho boli odstránené prudké navýšenia.

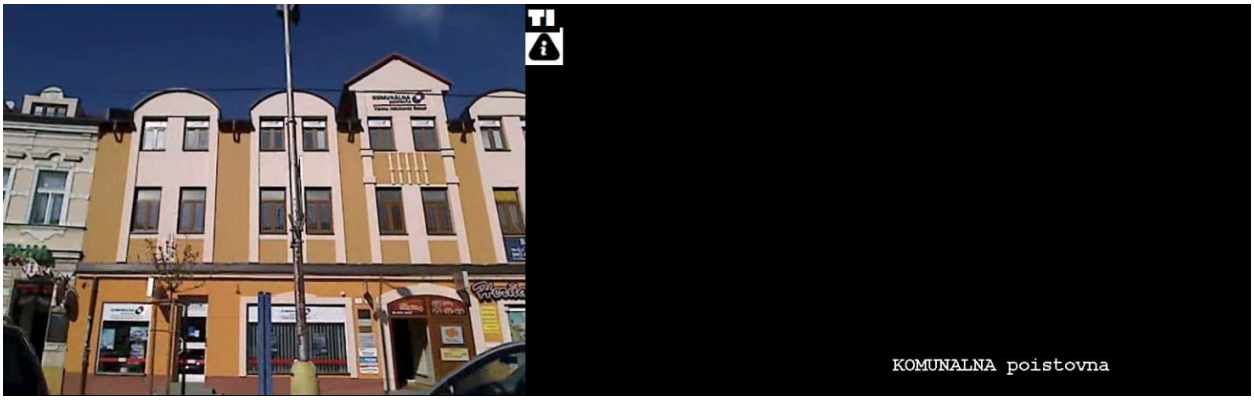
Štandardné implementácie deskriptorov nevyhovovali z hľadiska rýchlosti na detegovvanie objektov. Preto sa Surf deskriptory z knižnice OpenCV nahradili, Sift deskriptormi z knižnice SiftGPU, ktoré bežia na grafickej karte NVidia s CUDA. Výhodou týchto deskriptorov je, že sú robustnejšie a rýchlejšie.

10.2.2 Návrh

Po detekcii horizontu pomocou hranového detektora je tento horizont dodatočne upravený. Výškový profil horizontu sa analyzuje za účelom nájdenia vysokých lokálnych extrémov, kde horizont prudko vzrastie a následne prudko klesne. Takéto lokálne extrémy sú odstránené nájdením priemernej výšky horizontu v okolí tohto extrému. Zároveň je profil horizontu porovnávaný s predchádzajúcim snímkom za účelom odstránenia nepresností v detekcii spôsobených napríklad odrazom slnka. Ak sa výškový profil týchto horizontov v určitých oblastiach veľmi odlišuje, pre výšku v tomto bode sa potom použije priemerná hodnota z okolia predchádzajúcej snímky. Aby bolo možné lietadlom prejsť cez každú oblasť, v detekcii horizontu sme nastavili maximálnu výšku, ktorú horizont môže mať.

Obr. 1 Detekcia horizontu v hre Lietadlo. Horizont je znázornený červenou čiarou a zelená čiara predstavuje maximálnu možnú výšku horizontu.

Na extrakciu príznakov a následné hľadanie zhôd bol použitý SIFT GPU deskriptor. Ďalšia logika v detekcii horizontu zostala zachovaná. Deskriptory vytvorené z fotografií objektov v našej lokálnej databáze sú kvôli zrýchleniu celého procesu dopredu predpočítané a uložené v súbore, ktorý sa pri detekcii načíta.



Obr. 2 Detekcia objektu v turistickom sprievodcovi. Aktuálna snímka okolia je porovnaná s objektami v bezprostrednej blízkosti z databázy pomocou SIFT GPU deskriptorov.

10.2.3 Implementácia

Na implementáciu tohto modulu bola použitá knižnica OpenCV. Na extrakciu príznakov a porovnávanie deskriptorov bol použitý GPU SIFT deskriptor a SIFT matcher z knižnice SiftGPU. Pre správne fungovanie detekcie je potrebné mať grafickú kartu NVidia s podporou CUDA.

10.2.4 Testovanie

Za účelom testovania a upravovania detekcie objektov boli vytvorené pomocou webkamery s 30 FPS 2 typy videozáznamov - v meste a mimo mesta.

10.2.4 Súvisiace úlohy

#109 [OpenCV] Testovanie/uprava detekcie objektov na zaklade videa
#110 [OpenCV] Casova analyza detekcie objektov
#111 [OpenCV] Testovanie/uprava detekcie horizontu na zaklade videa
#132 [OpenCV] "Vypnutie" horizontu
#133 [Repozitar] Pripravit OpenCV pre RELEASE verziu vo vsetkych moduloch
#134 [OpenCV] Nacitanie/zapis deskriptorov zo/do suboru
#138 [OpenCV] Analyza thresholdu pre vyber zdetegovaneho objektu
#139 [OpenCV] Vyber zdetegovanych objektov
#148 [OpenCV] SiftGPU verzia - extrakcie deksriptorov
#152 [OpenCV] Detekcia horizontu - odstranenie chyb v detekcii
#149 [OpenCV] SiftGPU verzia - zakomponovanie do projektu
#150 [OpenCV] SiftGPU verzia - deskriptor matcher
#172 [OpenCV] Analyza detekcie pre video "Bratislava"

#179 [OpenCV] Detekcia objektov - video "Bratislava"
#180 [OpenCV] Detekcia objektov - video "Trencin"

10.2.5 Zhrnutie

Modul spracovania obrazu vykonáva funkcie z oblasti počítačového videnia, ktoré spracúvajú obraz zo vstupu.

Tento modul poskytuje nasledovné funkcie:

- *rozpoznávanie objektov a určenie ich polohy*
- *detekcia horizontu*

Detekcia objektov si však vyžaduje ďalšie a dôkladnejšie testovanie, príp. úpravy v algoritme detekcie. Nesprávne rozpoznávanie objektov bolo väčšinou spôsobené vysokou podobnosťou objektov alebo nevhodnými fotografiami objektov, ktoré neobsahovali žiadne charakteristické rysy objektu, príp. veľkú časť snímky objektu tvorilo pozadie. Úspešnosť detekcie sa samozrejme znižovala v závislosti od natočenia snímaného objektu voči fotografii objektu v lokálnej databáze.

10.3 Modul Kinect

Pri priebežnom testovaní sa ukázali problémy v pôvodnej implementácii, ktoré bolo treba odstrániť zmenou prístupu k metódam knižnice Freenect.

Kalibrácia celého modulu pre ukotvenie súradníc v priestore určenom oknom auta sa ukázala problematická a preto bol upravený spôsob vypočítavania polohy v priestore.

10.3.1 Analýza

Pri spúšťaní na niektorých systémoch narazil modul na deadlock, z ktorého sa síce niekedy dostal, ale celkovo spustenie modulu trvalo obvykle viac než 30 minút. Tento problém bol zakorenený v metódach obsiahnutých vo wrapperi knižnice pre C++ a OpenCV.

Ukotviť súradnice do priestoru sa ukázalo ako veľmi obtiažna úloha s veľkými nepresnosťami. Preto bolo treba nájsť lepší spôsob kalibrácie systému na súradnicovú sústavu nami zvolenú.

10.3.2 Návrh

Kvôli zložitosti problému a nedostatočnej dokumentácii knižnice bolo potrebné metódy spôsobujúce deadlock kompletne obísť. Pre tento účel boli použité metódy knižnice nižšej úrovne, ktoré pristupujú už priamo k zásobníkom a teda neobsahujú kód v ktorom by bol deadlock možný. Pri týchto metódach ale bolo treba do programu doplniť semafóry, ktoré kontrolovali prácu so zásobníkmi a zabraňovali ich neželanému prepísaniu.

Pri sledovaní pohybu hlavy sme sa rozhodli zamerať na pohyb dopredu a dozadu, teda v našej súradnicovej sústave, určenej ľavým horným rohom okna, pohyb po osi x. Na základe tejto skutočnosti bol výstup a spôsob kalibrácie modulu upravený. Súradnice pre os y a z boli menené na pevné a pri posielaní na ďalšie spracovanie sa menila iba súradnica pre os x. V tomto prípade už nebolo potrebné robiť kompletnú transformáciu medzi súradnícovými sústavami. Miesto toho bola zvolená metóda interpolácie medzi krajnými bodmi v ktorých sa používateľ môže nachádzať. Tieto 2 body je treba pred prvým spustením systému v jeho aktuálnej konfigurácii (umiestnenie kinectu a používateľa) odmerať pomocou konfigurátora fungujúceho ako samostatná aplikácia čisto pre tento účel.

10.3.3 Implementácia

Pre implementáciu oboch častí bola použitá hlavne knižnica OpenCV a tiež knižnica freenect potrebná pre prístup k údajom z kinectu.

10.3.4 Testovanie

Testovanie prebiehalo priebežne od začiatku semestra v laboratóriu, kde bol celý systém rozložený.

10.3.5 Súvisiace úlohy

#118 [Kinect] Testovanie a úprava kalibrácie kinectu
#119 [Kinect] Analýza nových spôsobov kalibrácie kinectu
#120 [Kinect] Implementácia nového spôsobu kalibrácie kinectu
#145 [Kinect] Experimentácia s kalibráciou
#146 [Kinect] Riešenie problémov kalibrácie
#160 Dlhé spúšťanie kinectu
#171 [Kinect] prerobenie bodov kalibrácie

10.3.6 Zhrnutie

Modul Kinect deteguje pozíciu tváre používateľa a pomocou hĺbkovej informácie z kinectu vypočíta jej polohu v priestore. Interpoláciou potom nájde súradnice tejto polohy vzhľadom na ľavý horný roh okna. Táto informácia je následne poskytnutá ďalším modulom na spracovanie a upravenie polohy zobrazených informácií.

Pri ďalšom pokračovaní v projekte by bolo vhodné nahradiť detekciu tváre detekciou hlavy s použitím zložitejších metód. Pre nepresnosti vznikajúce pri detekcii by bolo tiež vhodné pridať kontrolu výstupu modulu, aby sa zabránilo posielaniu informácii pri zlých detekciách.

10.4 Modul vypočítania polohy textu

Modul vypočítania polohy textu sa zaoberá vypočítaním polohy textu na okne automobilu s využitím vstupov z Kinectu (poloha hlavy), polohou detegovaných objektov a rôznych konštánt.

Jediná zásadná zmena, ktorá sa týkala zmeny v našom module bola implementácia posúvania virtuálnych prvkov na okne vzhľadom na pozíciu hlavy získanej zo zariadenia Kinnect.

Inak zmeny, ktoré sme okrem implementácie získania polohy hlavy v okne spravili, sú len normalizovanie vstupných a výstupných hodnot modulu. Tieto zmeny tu nebudeme detailne opisovať, kedže sú okomentované v kóde a sú veľmi jednoduché.

10.4.1 Analýza

Tento modul je teda zodpovedný na vypočítanie polohy hlavy v rámci okna na aute. Výstupom má byť hodnota v intervale od -1 do 1, kde -1 znamená že hlava používateľa sa nachádza v úplne ľavej časti okna. Hodnota 1 znamená, že hlava používateľa sa nachádza v úplne pravej časti okna.

Keďže kód, ktorý sa bude starať o výpočet polohy hlavy v rámci okna na aute máme už implementovaný, úlohou bude tento kód skombinovať tak, aby nám dával požadovaný výstup.

Jedinou podstatnou úlohou bola voľba bodu za oknom auta, na ktorý sa bude ako keby používateľ neustále pozerať a bude nám slúžiť na výpočet polohy hlavy v rámci okna. Ako tento bod sme zvolili plátno, na ktoré sa premieta natočené video.

10.4.2 Návrh

Pri návrhu sme museli porozmýšľať, ako budú ostatné moduly k tejto hodnote polohy hlavy v rámci pozície okna pristupovať. Rozhodli sme sa, že ju implementujeme ako metódu nášho modulu, ktorá má na vstupe získanú pozíciu polohy hlavy v priestore.

10.4.3 Implementácia

Navrhnutú funkcionalitu sme implementovali takým spôsobom, ako sú implementované iné metódy v moduloch. Zaviedli sme základnú metódu do základného (nie skutočného modulu) a potom sme už do svojho modulu (skutočného) zaviedli metódu, ktorá už skutočne počíta túto hodnotu a preťažuje pôvodnú metódu.

10.4.4 Testovanie

Testovanie prebiehalo v zimnom semestri implementovanom testovacom prostredí v prehliadači a následne na to aj so skutočnými vstupmi zo všetkých modulov. Testovanie prebehlo úspešne a neboli zaznamenané veľmi zlé výstupy modulu.

10.4.5 Súvisiace úlohy

#125 [TextPosModule] Testovanie vstupu a výstupu do modulu
#126 [TextPosModule] Analyza kalibracie
#181 [Modul vypocitania polohy] Ziskat polohu hlavy v ramci okna
#183 [Modul vypocitania polohy] Testovanie vypoctu pozicie hlavy zo vsetkymi vstupmi

10.4.6 Zhrnutie

Modul vypočítania polohy textu vykonáva funkcie, ktoré súvisia s počítaním polohy textu na okne vozidla a má na vstupe polohu hlavy, polohu detegovaných objektov a rôzne konštanty.

Tento modul poskytuje nasledovné funkcie:

- *vypočítanie polohy textu na okne auta z rôznych vstupov*
- *vypočítanie polohy hlavy v rámci pozície okna*

Úspešnosť vypočítania polohy bola pri testovaní a aj skutočnom nasadení ohodnotená ako na dobrej úrovni. V prípade ďalšej práce na projekte by sme vedeli vyskúšať iné prístupy k vypočítaniu polohy s využitím ďalších rôznych pomocných vstupov.

10.5 Modul Android

Modul android sa v letnom semestri skompletizoval tak, aby úlohy, ktoré má na starosti vedel bezchybne vykonať a tak používateľovi znásobiť pozitívny zážitok z používania vytvoreného systému.

Implementovali sa tri zásadne zmeny. Jednou z nich bolo rozpoznávanie hlasu v slovenskom jazyku. Druhou zmenou v aplikácii bolo pridanie spúštacích tlačidiel na spustenie hry alebo turistickej časti systému. Treťou veľmi dôležitou zmenou bolo pridanie interakcie aplikácie s turistickou časťou systému.

Ostatné zmeny boli len kozmetické, ktoré len zlepšovali funkčnosť aplikácie. Boli to napríklad zmeny príkazov, prisbôsobenie grafiky, refaktoring s cieľom zrychliť rýchlosť aplikácie alebo kalibrácia pohybového ovládania. Poprípade uľahčenie zmeny IP adresy servera, s ktorým komunikuje aplikácia.

10.5.1 Analýza

V rámci systému sme narazili na pár otázok, ktoré bolo potrebne riešiť. Jednou z nich bola interakcia so systémom ako takým. Bolo potrebné vyriešiť ako bude fungovať prepínanie medzi hrou/hrami a inými časťami systému akou je napriklad časť so zobrazovaním turistických informácii. Keďže na ovládanie hry sme už využívali android aplikáciu, rozhodli sme sa doimplementovať aj časť na prepínanie medzi systémovými aplikáciami.

Ďalším z problémov našej android aplikácie bolo rozpoznávanie hlasu len v anglickom jazyku, kvôli tomu, aby sme mohli fungovať offline (bez internetu). Ako sme už spomínali, anglický jazyk sme museli obmedziť na jednoduché jednoslovné príkazy z dôvodu správnej vyslovnosti. To sa nám javilo ako nepohodlné, preto sme sa zhodli na tom, že rozpoznávanie hlasu bude lokalizované na slovenský jazyk, k čomu je síce potrebný internet, ale usúdili sme, že sa nenecháme obmedziť takouto podmienkou, keďže predpokladáme, že internet v mobile, poprípade v aute, drvivá väčšina naších cieľových zákazníkov bude mať.

10.5.2 Návrh

V module android sme navrhli prepínanie medzi hrou a aplikáciou na zobrazovanie turistických informácii. Najprv sme na prepínanie navrhli ovládací prvok “prepínač”, avšak po otestovaní sme sa zhodli, že v rámci grafiky našej aplikácie a pohodlnosti ovládania bude vhodnejšie implementovať dve veľké tlačidlá na prepínanie v duchu grafického návrhu celej aplikácie.

Taktiež sme navrhli ako bude aplikácia komunikovať s turistickou časťou systému. Rozhodli sme sa pre odosielanei TCP soketov, tak ako to bolo aj v prípade hry. Vybrali sme pár príkazov/viet, na ktoré vie aplikácia na zobrazovanie turistických informácií reagovať. Rozpoznávanie príkazov v slovenskom jazyku súviselo s touto úlohou, keďže odosielame také isté príkazy.

10.5.3 Implementácia

Implementácia prebehla v jazyku Java v prostredí Android Development Tool (Eclipse prispôsobený vývoju pre Android). Bolo implementované rozpoznávanie hlasových povelov v slovenskom jazyku, ktoré funguje na princípe prehľadávania zoznamu podporovaných príkazov. Ak sa rozpozná hlasový povel, ktorý sa v zozname nájde, tak sa pomocou TCP soketu odošle kód tohto rozpoznaného príkazu. Podobne sa tento kód príkazu odošle aj po kliknutí na interaktívne tlačidlo na obrazovke, ktoré slúži na interakciu s aplikáciou zobrazujúcou turistické informácie. Prepínanie medzi aplikáciami systému je implementované pridaním dvoch tlačidiel s jednoznačným popisom na čo dané tlačidlo slúži na hlavnú obrazovku.

10.5.4 Testovanie

Testovanie prebehlo na Android zariadeni Samsung Galaxy SII a HTC ONE, ktoré komunikovali s lokálnym serverom a odosielali na neho sokety. Testovaná bola najmä funkcionalita ovládania hry pomocou mobilu, rozpoznávanie hlasových povelov, interakcia s turistickou časťou a pohybové gestá na ovládanie hry, ktoré boli aj kalibrované. Taktiež bolo otestované prepínanie medzi aplikáciami. Testovanie dopadlo úspešne vo všetkých prípadoch. Avšak parkrát sa objavil problém s neskoršou odozvou ovládania lietadla, čo bolo zapríčinené pomalším WI-FI spojením.

10.5.5 Súvisiace úlohy

#112 [Mobil] Zadavanie IP adresy v aplikacii
#113 [Mobil] Nastavenie pohyboveho ovladania
#114 [Mobil] Prijatie signalu o spusteni videa
#115 [Mobil] Odosielanie GPS dat
#116 [Mobil] Vytvorenie obrazovky na interakciu s turistickou appkou
#117 [Mobil] Pridanie prvkov na interakciu
#143 [Mobil] Interakcia s turistickou aplikaciou
#144 [Mobil] Fixnut odosielanie Command prikazov
#157 [Mobil] Hlasova interakcia s turistickou castou
#169 [Mobil] Pridat prepinanie medzi turistom a hrou
#170 [Mobil] Refactorovat vsetko zle
#175 [Mobil] Testovanie interakcie
#176 [Mobil] Spustanie aplikacie

10.5.6 Zhrnutie

Modul Android je funkčným protoypom pre ovládanie systému do auta - konkrétne hry a aplikácie zobrazujúcej turistické informácie. Je naprogramovaný tak, že je ľahko rozširiteľný o ďalšiu funkcionalitu, čo je do budúcnosti veľké plus.
V takomto stave tento modul poskytuje nasledovné funkcie:

- *ovládanie hry*
    - *pomocou ovládania dotyk, hore, dole, doľava, doprava*
    - *pomocou pohybu zariadením hore/dole*
- *interakciu s aplikáciou zobrazujúcou turistické informácie*
    - *pomocou interakčných prvkov (tlačidiel na špecialnej obrazovke)*
    - *pomocou hlasových povelov v slovenčine*
- *prepínanie medzi hrou a aplikáciou zobrazujúcou turistické informácie*

10.6 Modul rozšírenej reality

Modul rozšírenej reality sa zaoberá vykreslením rozšírenej reality na bočné sklo auta. V letnom semestri sme sa zamerali na vykreslenie reálne detegovaného horizontu. Vytvorenie ochranného pásma pre lietadlo.

Taktiež sme sa rozhodli podporovať viacero modelov lietadiel. Ako vhodné rozšírenie sme sa rozhodli vytvoreniť meranie času letu. Bolo potrebné aj implementovať posúvanie okna vzhľadom na pozíciu hlavy získanej zo zariadenia kinect. V neposlednom rade sme aj v rámci tohto modulu riešili optimalizáciu spúšťania celej aplikácie.

Poslednou časťou ktorá patrí do modulu rozšírenej reality je aplikácia turistické info, ktorá zobrazuje dodatočné informácie o detegovaných okolitých objektoch.

10.6.1 Analýza

Určité objekty vo videu ktoré sú viditeľné iba krátko a sú tak vysoké, že prechádzajú celou obrazovkou je potrebné upraviť. Teda je potrebné vytvoriť ochranné pásmo, ktoré bude zobrazené zelenou farbou.

Pre zvýšenie atraktivity hry je vhodné poskytnúť viacero modulov lietadiel. Po dohode sme dospeli k záveru, že je vhodné poskytnúť aspoň 3 rôzne modely. Vždy pri spustení aplikácie sa model zvolí náhodne. Taktiež ako vhodné rozšírenie sme definovali zobrazovať nejaký druh skóre, ktorý by motivoval hráča k lepším výkonom.

Spúšťanie aplikácie sme identifikovali ako značne problematické a užívateľsky málo príjemné. Z toho dôvodu sme sa rozhodli ho vylepšiť, aby bolo jednoduché a intuitívne.

10.6.2 Návrh

V rámci modulu vykresľovania rozšírenej reality sme sa v časti návrh rozhodli vykonať viacero vylepšení. Zobrazovať reálny detegovaný horizont v hre. Tento horizont bude zobrazený červenou farbou, ktorá štandardne indikuje nebezpečenstvo zatiaľ čo ochranné pásmo zobraziť zelenou, ktorá indikuje bezpečie.

Pre používateľa spraviť hru zaujímavejšou pomocou viacerých modelov lietadiel, skóre a taktiež lepšieho nastavenia konštánt hry. Poskytnúť mu ešte aj iný mód okrem lietadla - turistické info. Turistické info bude poskytovať doplňujúce informácie o rozoznaných objektoch.

Vylepšiť interakciu s mobilom a spríjemniť ovládanie. Ďalšou časťou je zabezpečiť posúvanie okna s vykreslenou rozšírenou realitou podľa pozície hlavy. Cieľom je zarovnanie reality a rozšírenej reality. Túto pozíciu zasielať spoločne s každým prijatým snímkom a predísť tak problémom s chýbajúcou pozíciou.

10.6.3 Implementácia

Na implementáciu bola použitá knižnica OpenGL. Pomocou tejto knižnice je implementovaná prakticky kompletná funkcionalita modulu vykresľovania. Avšak využívajú sa v nej aj iné knižnice na zjednodušenie práce. Konkrétne ide o knižnice ako freeimage, freetype, glm, glew, freeglut atď. Všetky knižnice sú uložené v zložke s projektom.

10.6.4 Testovanie

Testovanie modulu hry a turistického infa prebiehalo na 2 verziách videí. Jedno video bolo sa sústreďovalo na detekciu horizontu a druhé na okolité objekty. Taktiež prebiehalo aj testovanie interakcie s mobilom a nastavovanie konštánt.

10.6.5 Súvisiace úlohy

#121 [OpenGL] prevzatie a zobrazenie horizontu na scene
#122 [OpenGL] zakomponovanie horizontu do hry
#123 [OpenGL] testovanie ovladania
#140 [OpenGL] zmenit farbu lietadla
#141 [OpenGL] podporovat viacero druhov lietadiel
#142 [OpenGL] zobrazit v okne iba lietadlo a horizont
#153 [OpenGL] vytvorit ochranne pasmo
#154 [OpenGL] zmenit farbu horizontu
#155 [OpenGL] Zobrazit cas hrania hry
#161 [OpenGL] sfinalozovat hru lietadlo
#162 [OpenGL] opravit algoritmus narazu
#177 [OpenGL] optimalizovat spustanie aplikacie
#178 [OpenGL] otestovat realne posuvanie obrazu
#186 [OpenGL] refaktorovat a doplnit doxygen

Úlohy súvisiace s aplikáciou turistického infa:

#124 [OpenGL] Základné zobrazovanie textu
#136 Analyza modulu hry
#137 Zobrazovat turisticke info
#163 Objektovo orientovany mapovac
#164 Zastarale graficke ovladace
#165 Vysvietit detekovany objekt
#166 Ako sa bude volit hra, alebo turisticke info
#167 Oprava renderovania textu o turistickom infe
#182 [Modul hra] Prepinanie medzi hrou a turistickym info v lubovolnom case
#184 [Modul hra] Testovanie turistickeho infoa spolu s detekciou objektov

10.6.6 Zhrnutie

Modul vykresľovania rozšírenej reality sa stará o kompletné zobrazenie rožšírenej reality na bočné sklo auta. Zahŕňa 2 módy lietadlo a turistické info. Obe módy sú funkčné avšak je možné ich ďalšie rozšírenie. Ako veľmi vhodné je vylepšenie posúvania okna aplikácia vzhľadom na pozíciu hlavy, keďže aktuálne dosiahnutý pohyb nie je úplne plynulý.

Kompletná architektúra je navrhnutá tak, že je možné pridávať ďalšie módy (aplikácie). Všetko záleží len od kreativity.

V súčasnosti modul vykresľovania poskytuje:

- Hru lietadlo (detegovaný horizont, interakcia s android zariadením)

Mód turistické info (rozoznávanie objektov, interakcia s android zariadením)

11. Siedmy šprint

Začiatok šprintu: 17.2.2014

#109 [OpenCV] Testovanie/uprava detekcie objektov na zaklade videa
Zodpovedná osoba: Patrik Polatsek

Na základe mestského videa "Trenčín" je potrebné otestovať, príp. upraviť existujúci algoritmus detekcie objektov s cieľom dosiahnuť čo najlepší pomer medzi true positive a false positive rate.

#110 [OpenCV] Casova analyza detekcie objektov
Zodpovedná osoba: Patrik Polatsek

S cieľom priblížiť sa reálnemu času pri detekcii objektov je potrebné identifikovať a analyzovať výpočetne najnáročnejšie časti algoritmu a navrhnúť spôsob optimalizácie kódu.

#111 [OpenCV] Testovanie/uprava detekcie horizontu na zaklade videa
Zodpovedná osoba: Patrik Polatsek

Na základe mestského videa "Trenčín" je potrebné otestovať, príp. upraviť existujúci algoritmus detekcie horizontu s cieľom získať čo najpresnejší horizont.

#112 [Mobil] Zadavanie IP adresy v aplikacii
Zodpovedná osoba: Róbert Sabol

Na základe zistenia počas testovania sme úsúdili, že by bolo vhodné vedieť si zmeniť IP adresu servera v aplikácii. IP adresa sa dá zmeniť v aplikácii na obrazovke nastavení.

#113 [Mobil] Nastavenie pohyboveho ovladania
Zodpovedná osoba: Róbert Sabol

Po zistení, že ovládanie hry pohybom zariadením sme zistili, že je potrebné ovládanie nakalibrovať.

#114 [Mobil] Prijatie signalu o spusteni videa
Zodpovedná osoba: Róbert Sabol

Aplikácia potrebovala vedieť, kedy sa spustí video, aby tým pádom vedela, kedy má začať odosielať predpripravené informácie o GPS polohe.

#115 [Mobil] Odosielanie GPS dat
Zodpovedná osoba: Róbert Sabol

Odosielanie aktuálnych GPS dát do systému cez TCP server.

#116 [Mobil] Vytvorenie obrazovky na interakciu s turistickou appkou

Zodpovedná osoba: Róbert Sabol

Vytvorenie novej obrazovky s grafikou, ktorá obsahuje interakčné prvky (tlačidlá) na interakciu s aplikáciou zobrazujúcou turistické informácie.

#117 [Mobil] Pridanie prvkov na interakciu
Zodpovedná osoba: Róbert Sabol

Pridanie interakčných prvkov na obrazovku so správnymi hodnotami.

#118 [Kinect] Testovanie a úprava kalibrácie kinectu
Zodpovedná osoba: Jakub Mercz

Testovanie a odstraňovanie nedostatkov prvej verzie kalibrácie pomocou nití a terčíkov, pre ukotvenie bodov na presných súradniciach v priestore.

#119 [Kinect] Analýza nových spôsobov kalibrácie kinectu
Zodpovedná osoba: Jakub Mercz

Analýza ďalších spôsobov uchytenia bodov v priestore, pre účely kalibrácie.

#120 [Kinect] Implementácia nového spôsobu kalibrácie kinectu
Zodpovedná osoba: Jakub Mercz

Implementácia nájdených nových metód.

#121 [OpenGL] prevzatie a zobrazenie horizontu na scene
Zodpovedná osoba: Peter Hamar

Pre potreby modulu hry je potrebné prevziať reálny horizont z hlavného modulu, spracovať ho a vykresliť.

#122 [OpenGL] zakomponovanie horizontu do hry
Zodpovedná osoba: Peter Hamar

Cieľom tejto úlohy je zakomponovanie horizontu do logiky hry. Lietadlo ho má vnímať ako prekážku, do ktorej keď narazí, hra končí.

#123 [OpenGL] testovanie ovladania
Zodpovedná osoba: Peter Hamar

Kompletná interakcia s hrou a teda aj s modulom vykreslovania prebieha pomocou androidového zariania. Cieľom je otestovať kompletné ovládanie a nakalibrovať hodnoty, ktoré sa týkajú fyziky lietadla a sveta (rýchlosť stúpania, klesania, silu gravitácie atď.).

#124 [OpenGL] Základné zobrazovanie textu

Zodpovedná osoba: Martin Petluš

Implementovať základné zobrazovanie textu s využitím knižnice OpenGL, kde vstupom je poloha textu a reťazec textu.

#125 [TextPosModule] Testovanie vstupu a výstupu do modulu
Zodpovedná osoba: Martin Petluš

Opätovné testovanie modulu vypočítania polohy textu, kvôli jeho dôležitej úlohe v aplikácií.

#126 [TextPosModule] Analyza kalibracie
Zodpovedná osoba: Martin Petluš

Analyzovať kód modulu kvôli kalibrácií spojenej s Kinectom.

#127 [Kamera] Osetrit koniec streamu z kamery
Zodpovedná osoba: Lukáš Sekerák

Pri ukončení streamu z kamery sa hlavný modul nesprávne ukončil, kde vznikli ďalšie chyby. Problém bolo potrebné odstrániť.

#128 [TCP] Osetrit chyby
Zodpovedná osoba: Lukáš Sekerák

V TCP servery vznikala vážna chyba pri odpojení používateľa. Chybu bolo potrebné nájsť a opraviť.

#129 [TCP] Pridat funkcionalitu
Zodpovedná osoba: Lukáš Sekerák

V TCP servery chýbala funkcionalita pre príjmanie GPS polohy zo zariadenia.

#130 [Databaza] Implementacia selektu
Zodpovedná osoba: Lukáš Sekerák

Do našej databázy sa dopytuje hra na zoznam objektov vo svete na základe polohy. Výsledkom selektu je zoznam namapovaných objektov. Tento selekt a mapovanie bolo potrebné implementovať.

#131 [Repozitar] Spravit poriadok s kniznicami
Zodpovedná osoba: Lukáš Sekerák

Mnohé knižnice sa nepoužívali. Ďalšie boli zle umiestnené. Tieto knižnice bolo potrebné umiestniť na správne miesto.

#132 [OpenCV] "Vypnutie" horizontu
Zodpovedná osoba: Patrik Polatsek

V prípade, že pre lietadlo nie je dostatočný priestor na prelet, horizont sa vypne.

#133 [Repozitar] Pripravit OpenCV pre RELEASE verziu vo vsetkych moduloch
Zodpovedná osoba: Patrik Polatsek

Celý projekt je potrebné uraviť z Debug verzie na Release, ktorá je rýchlejšia.

#134 [OpenCV] Nacitanie/zapis deskriptorov zo/do suboru
Zodpovedná osoba: Patrik Polatsek

Kvôli zrýchleniu procesu detekcie objektov budú deskriptory vytvorené z fotografií objektov v lokálnej databáze dopredu predpočítavané a uložené v súbore, ktorý sa bude len načítavať. V prípade, že takýto súbor neexistuje, bude automaticky po prvom spustení vytvorený.

#135 [Carlos] Refaktorizacia kodu
Zodpovedná osoba: Lukáš Sekerák

Mnohé triedy a metódy boli premenované a presunuté. Zložitosť kódu to vyžadovala.

#136 Analyza modulu hry
Zodpovedná osoba: Martin Petluš

Bolo potrebné vykonať analýzu kódu modulu hry, kvôli nasledujúcej implementácií turistického infa.

12. Ôsmy šprint

Začiatok šprintu: 5.3.2014

#137 Zobrazovat turisticke info
Zodpovedná osoba: Martin Petluš

Úlohou bolo rozšíriť kód v hre o základné zobrazovanie turistického infa.

#138 [OpenCV] Analyza thresholdu pre vyber zdetegovaneho objektu
Zodpovedná osoba: Patrik Polatsek

Na základe pozorovaní je potrebné určiť prahovú hodnotu zdetegovania objektov.

#139 [OpenCV] Vyber zdetegovanych objektov
Zodpovedná osoba: Patrik Polatsek

V súčasnosti modul spracovania obrazu vracia len 1 najpravdepodobnejší objekt zdetegovaný na snímke. To je potrebné zmeniť tak, aby modul vrátil všetky zdetegované objekty.

#140 [OpenGL] zmenit farbu lietadla
Zodpovedná osoba: Peter Hamar

Keďže sa na bočné sklo zobrazuje iba čierna plocha s detegovaným horizontom a ochranným pásmom súčasné lietadlo (čierne) sa javí ako nevýrazné. Z toho dôvodu je potrebné zmeniť farbu lietadla na oranžovú.

#141 [OpenGL] podporovat viacero druhov lietadiel
Zodpovedná osoba: Peter Hamar

Modul hra podporuje iba jeden model lietadla. Pre zvýšenie atraktivity hry vhodné podporovať viacero druhov lietadiel, aspoň 3. Tieto modely sa budú vyberať náhodne pri spustení aplikácie.

#142 [OpenGL] zobrazit v okne iba lietadlo a horizont
Zodpovedná osoba: Peter Hamar

Cieľom je zobrazenie v okne vykreslovania iba lietadlo s detegovaným horizontom a teda vypnutie textury reálneho sveta na pozadí.

#143 [Mobil] Interakcia s turistickou aplikaciou
Zodpovedná osoba: Róbert Sabol

Sfunkčnenie odosielania kódov s informáciou o aké turistické informácie mám záujem.

#144 [Mobil] Fixnut odosielanie Command prikazov
Zodpovedná osoba: Róbert Sabol

Pri rozširovaní aplikácie sa objavil “bug”, ktorý zapríčinil zlé rozpoznávanie príkazov na ovládanie hry, čo zapríčinilo nefukčnosť ovládania.

#145 [Kinect] Experimentácia s kalibráciou
Zodpovedná osoba: Jakub Mercz

Testovanie a experimentovanie s metódami kalibrácie.

#146 [Kinect] Riešenie problémov kalibrácie
Zodpovedná osoba: Jakub Mercz

Riešenie chýb a problémov v programe, ktoré sa ukázali pri testovaní a experimentovaní.

#147 [Carlos] Testovanie
Zodpovedná osoba: Lukáš Sekerák

Aktuálny prototyp bolo potrebné otestovať. Išlo o integračné testovanie.

#148 [OpenCV] SiftGPU verzia - extrakcie deksriptorov
Zodpovedná osoba: Patrik Polatsek

Kvôli priblíženiu k reálnemu času pri detekcii obejktov je potrebné nahradiť štandardné deskriptory z OpenCV GPU SIFT deskriptormi z knižnice SiftGpu.

#151 [Carlos] Stream kamery
Zodpovedná osoba: Lukáš Sekerák

Stream z kamery nebol implementovaný správne. Bolo potrebné ošetriť situácie, keď sa kamera odpojí.

#152 [OpenCV] Detekcia horizontu - odstranenie chyb v detekcii
Zodpovedná osoba: Patrik Polatsek

Cieľom tejto úlohy je odstránenie nepresností v detekcii horizontu porovnaním predchádzajúcej snímky.

#153 [OpenGL] vytvorit ochranne pasmo
Zodpovedná osoba: Peter Hamar

Keďže sa vo videách vyskytovali objekty, ktoré prechádzali celou obrazovkou a horizont nebolo možné detegovať bolo to potrebné riešiť. Lietadlo potrebovalo určitú minimálnu šírku (maximálnu výšku horizontu), aby mohlo preletieť. Z toho dôvodu sme sa rozhodli vytvoriť ochranné pásmo, jehošírka je 120 pixelov od vrchu okna.

#154 [OpenGL] zmenit farbu horizontu

Zodpovedná osoba: Peter Hamar

Pôvodná farba horizontu nebola správna, teda neindikovala ohrozenie. Naším cieľom teda bolo zmeniť farbu horizontu na červenú.

#155 [OpenGL] Zobrazit cas hrania hry
Zodpovedná osoba: Peter Hamar

Cieľom tejto úlohy bolo poskytnúť skóre dosiahnuté počas letu. Toto skóre je poskytnuté formou do dĺžky (častu) letú v sekundách.

#156 Zabezpečiť štipce + výkres
Zodpovedná osoba: Peter Hamar

Pre potreby prototypu bolo potrebné zabezpečiť štipce a výkres. Tieto pomôcky sa použili na vytvorenie provizórneho okna auta na plexisklo.

#157 [Mobil] Hlasova interakcia s turistickou castou
Zodpovedná osoba: Róbert Sabol

Upravenie hlasovej interakcie s turistickou časťou systému a pridanie slovenskej lokalizácie. Taktiež bol dohodnutý a pridaný zoznam podporovaných príkazov.

#158 [TP Cup] Upravit clanok na iit src
Zodpovedná osoba: Patrik Polatsek

Na základe pripomienok je potrebné upraviť článok na konferenciu iit src.

#159 [Repozitar] RELEASE verzia projektu
Zodpovedná osoba: Patrik Polatsek

Úprava fungovania celého projektu vo verzii Release.

#163 Objektovo orientovany mapovac
Zodpovedná osoba: Martin Petluš

Analýza použitého objektovo orientovaného mapovača použitého v projekte na vybratie objektov z databázy, kvôli jeho využitiu v turistickom infe.

#164 Zastarale graficke ovladace
Zodpovedná osoba: Martin Petluš

Aktualizácia zastaralých grafických ovládačov na osobnom notebooku. Aktualizácia je potrebná kvôli nutnému testovaniu aplikácie na osobnom notebooku.

13. Deviaty šprint

Začiatok šprintu: 19.3.2014

#149 [OpenCV] SiftGPU verzia - zakomponovanie do projektu
Zodpovedná osoba: Patrik Polatsek

Modul spracovania obrazu je potrebné upraviť tak, aby bolo možné voliteľne použiť na detekciu knižnicu SiftGpu.

#150 [OpenCV] SiftGPU verzia - deskriptor matcher
Zodpovedná osoba: Patrik Polatsek

Existujúci matcher deskriptorov sa v projekte nahradí implementáciou z knižnice SiftGpu.

#160 Dlhé spúšťanie kinectu
Zodpovedná osoba: Jakub Mercz

Prerobenie prístupu k prostriedkom knižnice freenect, keďže predošlé spôsobovali deadlock, kvôli ktorému spúšťanie modulu trvalo aj 30 minút.

#161 [OpenGL] sfinalozovat hru lietadlo
Zodpovedná osoba: Peter Hamar

Cieľom tejto úlohy je dokončenie všetkých nedotiahnutých častí hry, odstránenie pohybov lietadla do strán a lepšie nastavenie konštánt letu.

#162 [OpenGL] opravit algoritmus narazu
Zodpovedná osoba: Peter Hamar

Algoritmus nárazu lietadla do horizontu obsahoval chybu a teda lietadlo v niektorých špecifických situáciách neočakávanie havarovalo. Cieľom je nájdenie a odstránienie tejto chyby.

#165 Vysvietit detekovany objekt
Zodpovedná osoba: Martin Petluš

Vymysieť a implementovať spôsob, akým sa bude detegovaný objekt na obrazovke vysvieťovať ešte pred tým, než sa spýtame na otázku a tak sa namiesto tohto vysietenia vypíše text.

#166 Ako sa bude volit hra, alebo turisticke info
Zodpovedná osoba: Martin Petluš

Vymyslieť riešenie voľby turistického infa alebo hry a priamo ho aj do kódu implementovať.

#167 Oprava renderovania textu o turistickom infe
Zodpovedná osoba: Martin Petluš

Text sa v niektorých prípadoch z nejasných príčin vôbec na obrazovke nezobrazil. Bolo potrebné nájsť v kóde túto chybu a odstrániť ju.

#168 [Hra] Zrefaktorovat stavy
Zodpovedná osoba: Lukáš Sekerák

Doterajšie stavy hry boli implementované pomocou funkcií. Tieto funkcie sa prepínali pomocou switchu. Táto časť bola nahradená interfacom a použitím polymorfizmu.

#169 [Mobil] Pridat prepinanie medzi turistom a hrou
Zodpovedná osoba: Róbert Sabol

Vytvorenie grafických a funkčných prvkov na prepínanie medzi hrou a turistickou časťou systému. Najprv to bol prepínač, neskôr zmenený na dve tlačidlá s jasným popisom na čo slúžia.

#170 [Mobil] Refactorovat vsetko zle
Zodpovedná osoba: Róbert Sabol

V záujme lepšej prehľadnosti zdrojového kódu a zároveň zjednodušenia rozšíriteľnosti bolo potrebne refaktorovať "zlý" kód.

#172 [OpenCV] Analyza detekcie pre video "Bratislava"
Zodpovedná osoba: Patrik Polatsek

Na základe nového mestského videa "Bratislava" sa analyzuje a upraví detekcia objektov. Všetky objekty v tomto videu je rovnako potrebné doplniť aj do lokálnej databázy objektov - základné informácie, GPS a fotografie.

#181 [Modul vypocitania polohy] Ziskat polohu hlavy v ramci okna
Zodpovedná osoba: Martin Petluš

Kvôli posúvaniu virtuálnych prvkov na obrazovke je potrebné implemetovať funkcionalitu v module vypočítania polohy textu, ktorá nám bude počítať pozíciu polohy hlavy v rámci okna na aute v intervale od -1 do 1.

14. Desiaty šprint

Začiatok šprintu: 2.4.2014

#171 [Kinect] prerobenie bodov kalibrácie
Zodpovedná osoba: Jakub Mercz

Implementácia kalibrácie a zmena spracovania vstupu na interpolačnú metódu.

#173 [Kamera] Video zacyklit
Zodpovedná osoba: Lukáš Sekerák

Video stream sa posielal do konca zaznamu. Je oveľa vhodnejšie restartovať stream a posielať video opäť.

#174 Navrh plagatu
Zodpovedná osoba: Róbert Sabol

Keďže sme sa zúčastnili študentskej vedeckej konferencie IIT SRC 2014, bolo potrebné navrhnúť a vytvoriť plagát, ktorý nás mal prezentovať.

#175 [Mobil] Testovanie interakcie
Zodpovedná osoba: Róbert Sabol

Testovanie interakcie a ovládania hry.

#176 [Mobil] Spustanie aplikacie
Zodpovedná osoba: Róbert Sabol

Objavili sa problémy a otázky so spúštaním celého systému, preto bolo uvažované, či sa náš systém nebude spúšťať pomocou Android aplikácie.

#177 [OpenGL] optimalizovat spustanie aplikacie
Zodpovedná osoba: Peter Hamar

Keďže sa aplikácia problematicky spúšťala, bolo potrebné to v značnej miere zjednodušiť. Cieľom tejto úlohy je definovať konkrétne ktoré okná sa budú kde spúšťať a nastaviť ich zobrazenie na celú obrazovku.

#178 [OpenGL] otestovat realne posuvanie obrazu
Zodpovedná osoba: Peter Hamar

Okno s vykreslenou rozšírenou realitou sa má prispôsobovať pozícii hlavy detegovanej pomocou kinnectu. Je potrebné vykonať reálne otestovanie posúvania okna.

#179 [OpenCV] Detekcia objektov - video "Bratislava"

Zodpovedná osoba: Patrik Polatsek

Cieľom tejto úlohy je otestovať, príp. upraviť detekciu objektov pre video “Bratislava”, ku ktorému bola naplnená DB objektov.

#180 [OpenCV] Detekcia objektov - video "Trencin"
Zodpovedná osoba: Patrik Polatsek

Cieľom tejto úlohy je otestovať, príp. upraviť detekciu objektov pre video “Trenčín”, ku ktorému bola naplnená DB objektov.

#182 [Modul hra] Prepinanie medzi hrou a turistickym info v lubovolnom case
Zodpovedná osoba: Martin Petluš

Bolo potrebné implementovať prepínanie medzi stavom hry a stavom turistického infa v ľubovolnom čase.

#183 [Modul vypocitania polohy] Testovanie vypoctu pozicie hlavy zo vsetkymi vstupmi
Zodpovedná osoba: Martin Petluš

Úlohou bolo otestovanie správneho výpočtu pozície hlavy so všetkými skutočnými vstupmi, teda aj tými s Kinectu.

#184 [Modul hra] Testovanie turistickeho infoa spolu s detekciou objektov
Zodpovedná osoba: Martin Petluš

Keďže bolo implementované zobrazovanie textu o objektoch, bolo potrebné túto funkcionalitu priamo otestovať spolu s real-time detekciou objektov a tak odhaliť prípadné chyby a opraviť ich.

#185 Dotaznik
Zodpovedná osoba: Patrik Polatsek

K súťaži TP Cup je potrebné vyplniť dotazník.

#186 [OpenGL] refaktorovat a doplnit doxygen
Zodpovedná osoba: Peter Hamar

Počas semestra sa v modelu hry urobilo veľké množstvo zmien a pridalo sa dosť kódu je potrebné aktualizovať a doplniť doxygen komentáre. Kód je vhodné aj zrefaktorovať.

#187 [OpenCV] Simuntanny obraz hry Liedadlo na prezentaciu
Zodpovedná osoba: Patrik Polatsek

Pre účely iit src je potrebné pripraviť videosnímky na ktorých bude simultánne videozáznam na ľavej strane a virtuálne prvky na pravej strane.

15. Celkový pohľad

15.1 Opis problémovej oblasti a prehľad riešenia

Zábavné a informačné systémy sa stávajú čoraz častejšie súčasťou nášho života. Jedným z najmodernejších a pre človeka najprirodzenejších spôsobov interakcie je obohatená realita, kde reálny svet je doplnený o obrazové, textové alebo zvukové virtuálne prvky.

Obr 1. Prehľad systému

Rozšírená realita nám umožňuje veľmi jednoducho sprostredkovať turistické informácie o okolí pri jazde automobilom v reálnom čase alebo využiť realitu ako základ pre rôzne interaktívne hry. Automobil sa takýmto spôsobom môže stať interaktívnym turistickým sprievodcom alebo zábavným systémom.

Cieľom tohto projektu je zmeniť bočné okienko automobilu na transparentnú projekčnú plochu, pomocou ktorej bude reálny svet dopĺňaný o ľubovoľné virtuálne informácie so zábavným i náučným zámerom. Na okienku auta nám tak vznikne rozšírená realita, ktorá nás môže informovať o našom bezprostrednom okolí.

Výsledkom projektu je prototyp interaktívneho systému pre obohatenú realitu pre spolucestujúceho v automobile.

Na prezentáciu vygenerovaných informácii sa použije malý LED projektor spolu s transparentnou projekčnou fóliou, ktorá sa pripevní na bočné sklo automobilu. Aplikáciu používateľ ovláda veľmi jednoducho – prostredníctvom mobilného telefónu, pohybom alebo hlasom (obrázok č. 1).

Náš systém s názvom Carlos pozostáva z 2 základných režimov, ktoré predstavujú možnosti ako sa dá využiť rozšírená realita:

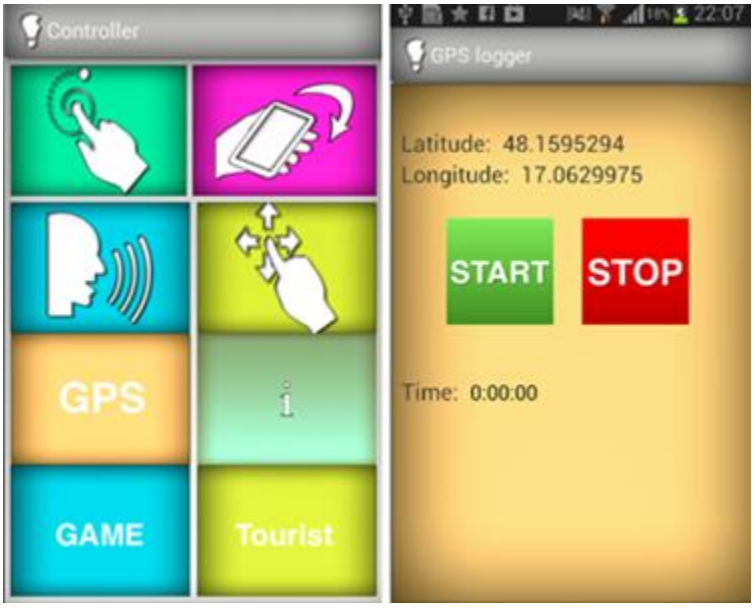
1. **Informačný** – zobrazovanie turistických informácií
2. **Zábavný** – hra Lietadlo

Základom nášho systému je rozoznávanie zaujímavých objektov nasnímaných kamerou v reálnom čase. K takto detegovanému objektu doplní do reálnej scény nielen jeho názov, ale aj iné zaujímavé textové informácie. Napr. pri pamiatkach zobrazí otváracie hodiny, výšku vstupného a základné historické informácie. Výhodou našej aplikácie bude aj to, že nebude vyžadovať pripojenie na internet, keďže všetky údaje o objektoch sú uložené v lokálnej databáze.

Carlos nepredstavuje len informačný, ale aj zábavný systém. Preto je súčasťou systému aj hra s názvom Lietadlo, ktorá využíva reálne prostredie. Účelom tejto hry je ovládať prostredníctvom mobilného telefónu lietadlo s cieľom udržať ho čo najdlhšie nad zdetegovaným horizontom.

Celý systém sa ovláda cez mobilné zariadenie so systémom Android (obrázok č. 2). Pomocou tejto aplikácie sa rovnako používateľ prepína medzi turistickou a hernou časťou systému.

Náš systém deteguje pomocou snímok z kamery na aute objekty ako napr. pamiatky, reštaurácie, kaviarne a hotely. V rámci detekčnej fázy systém porovná snímky s využitím aktuálnej GPS pozície a internej databázy objektov záujmu. Detekcia začína so selekciou potenciálnych objektov, ktoré sa môžu na snímke zdetegovať. Z databázy sa vyberú tie objekty, ktoré sú v blízkosti GPS pozície získavanej v pravidelných intervaloch z mobilného zariadenia (obrázok č. 2 vpravo).



Obr 2. Obrazovky mobilnej aplikácie.

Následne sa vykoná detekcia objektov, ktorá spočíva v extrakcii príznakov a porovnaním *deskriptorov*, ktoré opisujú okolie kľúčových bodov (obrázok č. 3 vľavo). Výhodou je, že deskriptory sú vo väčšej miere invariantné voči zmenám jasu a deformácii obrazu. V našom

systéme využívame SIFT deskriptory pre grafické karty NVidia s CUDA z knižnice SiftGPU. Na zrýchlenie celého procesu detekcie sú deskriptory pre objekty v databáze dopredu predpočítané a načítavané zo súboru. Po extrakcii deskriptorov z aktuálnej snímky okolia sa hľadajú zhody medzi touto snímkou a potenciálnymi objektmi.

Po úspešnej detekcii objektov sa určí ich poloha na snímke pomocou *homografie* (obrázok č. 3 vpravo) s využitím tzv. RANSAC (RANdom SAmple Consensus) metódy.

Obr 3. Detekcia objektu pomocou deskriptorov a určenie jeho polohy s využitím homografie.

Kvôli správnemu zobrazeniu virtuálnych informácií na obrazovke využíva náš systém zariadenie Kinect. Pomocou video- a hĺbkovej informácie získanej z Kinect-u sa určí aktuálna pozícia hlavy používateľa, aby boli obrazové informácie zobrazené na skle automobilu čo najpresnejšie voči aktuálnemu pohľadu používateľa (obrázok č. 4).

Po úspešnej detekcii objektov na aktuálnej snímke sa zobrazí v hornom ľavom rohu informácia, že v okolí boli identifikované objekty. Používateľ si môže pomocou mobilnej aplikácie – stlačením príslušného tlačidla alebo hlasom zobraziť názvy objektov, príp. základné informácie o týchto objektoch.

Obr 4. Posun zobrazených informácií na základe polohy hlavy.

Nakoniec sa po prepočítaní pozícií všetkých virtuálnych elementov zobrazia tieto turistické informácie v textovej alebo obrazovej podobe na bočnom skle automobilu pomocou projektora.

Carlos rovnako ponúka aj leteckú hru, kde hráč musí udržať lietadlo nad horizontom (obrázok č. 5). Na detekciu horizontu je použitý hranový detektor *Canny*, ktorý na snímke určí oblasti oblohy (obrázok č. 6). Horizont je následne upravený tak, že sú z neho odstránené prudké zvýšenia, ktoré sú pravdepodobne spôsobené dopravnou značkou, príp. stromom.

V prípade pádu lietadla sa na obrazovke zobrazí dĺžka letu. Let lietadla je ovládaný jednoduchým nakláňaním mobilného zariadenia, príp. pohybom.

Obr 5. Ukážky z hry "Lietadlo".

Obr 6. Detekcia regiónov oblohy (biela farba).

15.2 Realizácia riešenia

Na implementáciu riešenia využívame programovacie jazyky C/C++, Javu pre mobilnú platformu Android a knižnice OpenCV, OpenGL a Freenect. Implementácia systému zahŕňa viaceré algoritmy z oblasti *počítačovej vízie*, *počítačovej grafiky* a *interakcie typu človek-počítač*.

Náš systém Carlos využíva viaceré periférne zariadenia:

1. *kamera* na zachytenie snímok, ktoré budú následne spracované detekčnými algoritmami,
2. *Kinect* na sledovanie pozície hlavy pre správne umiestnenie virtuálnych elementov na obrazovke vzhľadom na aktuálny uhol pohľadu používateľa,
3. *mobilný telefón* s operačným systémom Android na získavanie aktuálnych GPS súradníc, ktoré sa využijú na selekciu objektov z databázy, ktoré sú v bezprostrednej blízkosti a na ovládanie celého systému,
4. *projektor* na zobrazenie virtuálnych informácií na sklo automobilu s transparentnou fóliou.

Náš systém pozostáva z viacerých modulov, ktorých štruktúru znázorňuje obrázok č. 7:

- **Riadiaci modul** predstavuje centrálny modul, ktorý spravuje a synchronizuje zvyšné moduly. Riadi ich vstupy a výstupy, prijíma snímky z kamery a zobrazuje virtuálne informácie cez projektor.

- **Modul spracovania obrazu** deteguje objekty a ich polohu na snímkach pomocou deskriptorov a určuje oblasti na snímke, ktoré zodpovedajú oblohe pomocou hranového detektora.

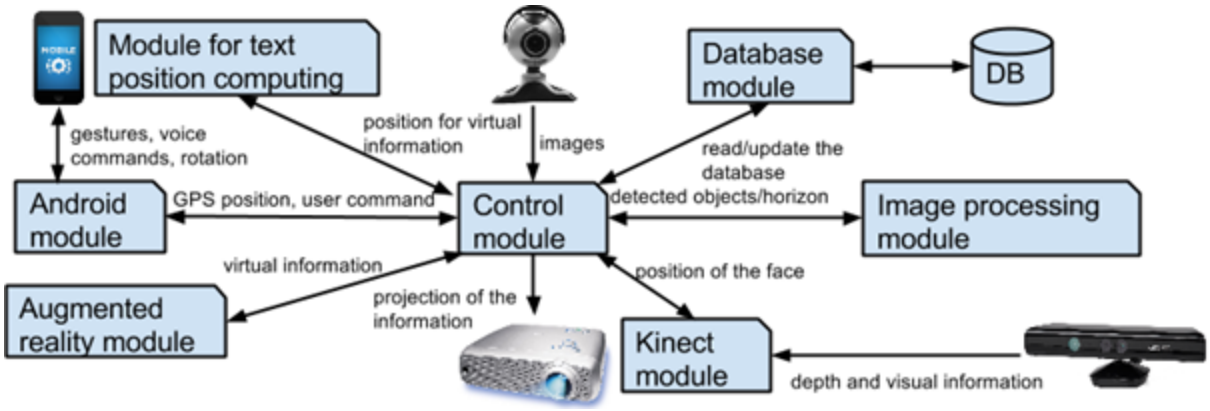
- **Modul Kinect** spracováva hĺbkovú a video informáciu z Kinect-u na určenie polohy hlavy.

- **Modul výpočítania polohy textu** vypočítava na základe informácie o polohe objektov na snímke a pozícii hlavy používateľa presné umiestnenie virtuálnych prvkov na skle automobilu.

- **Modul Android** predstavuje Java aplikáciu pre mobil s OS Android, ktorý ponúka rozhranie na ovládanie celého systému (hlas, rotácia zariadenia) a zároveň zaznamenáva aktuálnu GPS polohu.

- **Modul rozšírenej reality** generuje informácie, ktoré majú byť zobrazované cez projektor. Zároveň implementuje leteckú hru, ktorá využíva rozšírenú realitu.

- **Modul databázy** spravuje internú databázu objektov záujmu, ktoré obsahuje fotografie, GPS pozície, predpočítané deskriptory a základné informácie o objektoch.



Obr 7. Základná štruktúra nášho systému.

Výsledkom nášho projektu je funkčný prototyp systému pre spolujazdca s rozšírenou realitou. Prototyp obsahuje leteckú hru ako aj interaktívneho turistického sprievodcu.

Na testovanie sme mali k dispozícii počítač s grafickou kartou NVidia, zariadenie Kinect a sklo s transparentnou fóliou. Na simulovanie jazdy v automobile sme využili 2 projektory (obrázok č. 8). Jeden z nich zobrazoval doplnkové informácie na sklo a druhý premietal na stenu videozáznam „okolia“. Pomocou webkamery s 30 FPS sme si vytvorili 2 typy záznamov – v meste a mimo mesta.

Obr 8. Ukážka testovania systému.

V tomto simulovanom prostredí sme otestovali posúvanie zobrazovaných informácií voči aktuálnej zdetegovanej polohy hlavy. Na videosnímkach mimo mesta bola odskúšaná hra Lietadlo a v rámci nej detegovanie horizontu a ovládanie lietadla mobilným telefónom. Mestské videá slúžili na testovanie turistického sprievodcu a rozpoznávania objektov.

16. Používateľská príručka

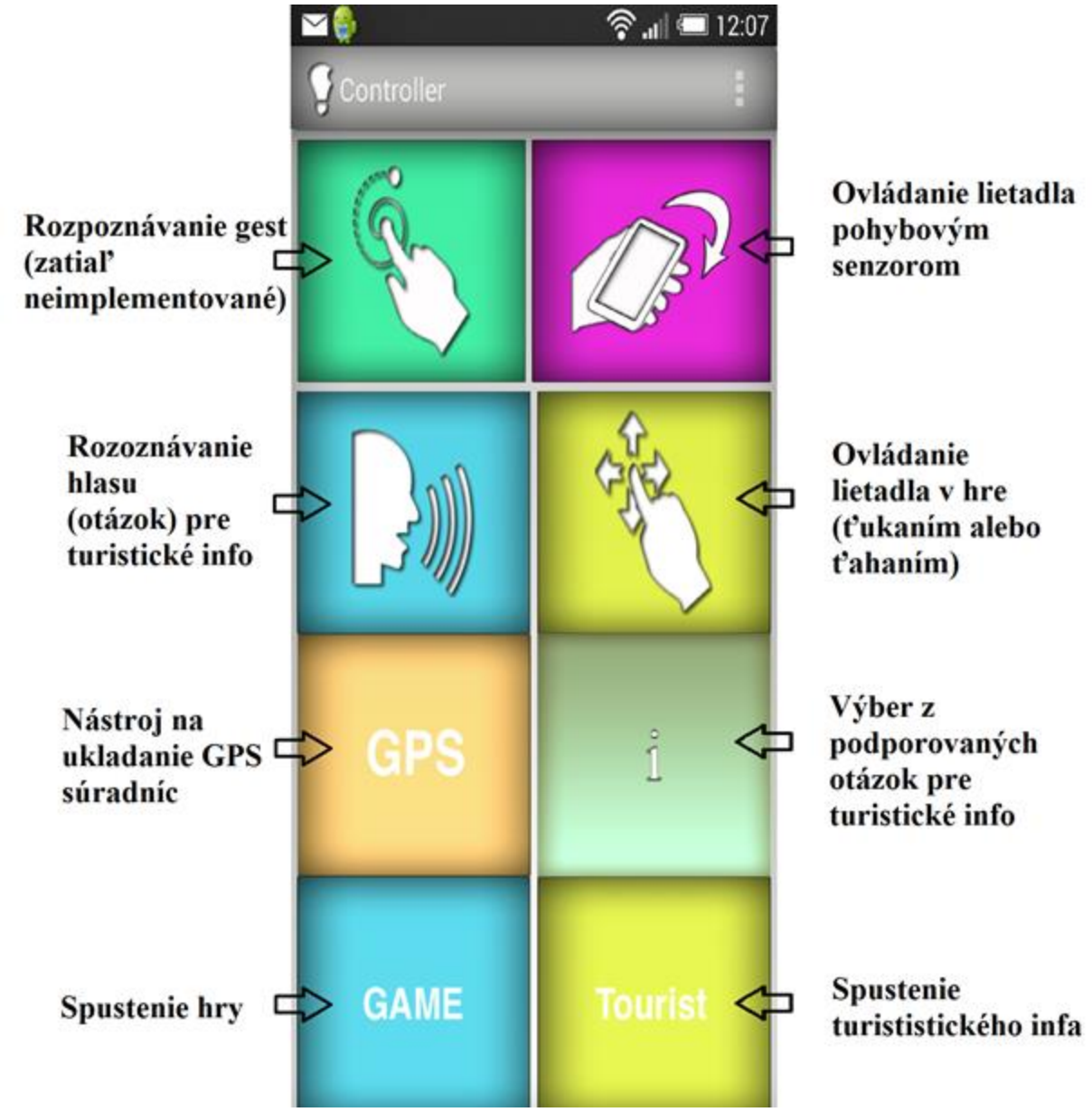
Potrebný hardvér a softvér na spustenie projektu:

- Počítač s wifi modulom
- Grafickú kartu cuda
- Zariadenie kinect
- 2 projektory na premietanie obohatenej reality a sveta za oknom auta
- Mobilné zariadenie s Androidom a nainštalovanou aplikáciou Controller
- Nainštalovaný nástroj connectify [1]

Postup spustenia produktu:

1. V nástroji connectify vytvoríme a spustíme hotspot sieť.
2. Mobilným zariadením sa na túto sieť pripojíme.
3. Pomocou príkazového riadku zistíme IP adresu vytvorenej siete.
4. Nastavíme IP adresu vytvorenej siete v konfiguračnom súbore ktorý nájdeme v: ..\CarlosAR_VW\data\configuration.xml
5. Túto IP adresu nastavíme spolu s portom do mobilnej aplikácie Controller v mobilnom zariadení
6. Na počítači spustíme aplikáciu Carlos.exe.

Zábavný systém Carlos ponúka 2 režimy. Prvým je hra, kde je cieľom udržať lietadlo nad horizontom čo najdlhšie a dosiahnuť tak lepší čas. Druhým režimom je turistické info. Pomocou neho sa má používateľ dozvedieť viac o rozoznaných objektoch za oknom auta. Kompletná interakcia so systémom je riadená len pomocou mobilného zariadenia. Konkrétny spôsob ovládania je zobrazený na obrázku 1.



Obr. 1. Aplikácia Controller systému Carlos

[1] http://www.connectify.me/

17. Zhrnutie

Primárnym cieľom nášho projektu bolo poukázať na možnosti, ktoré nám ponúka rozšírená realita, či už na edukačné alebo herné účely. Vytvorený prototyp by sa dal využiť nielen v automobiloch, ale aj iných dopravných prostriedkoch.

Rozšírená realita ponúka obrovské možnosti uplatnenia v interaktívnych systémoch. Modularita nášho systému umožňuje jednoduché rozšírenie o ďalšiu funkcionalitu.

Jednou zo zamýšľaných funkcií, ktoré sa nestihli implementovať bola napríklad kvízová hra, kde by používateľ musel uhádnuť, ktorý objekt je v jeho okolí alebo zodpovedať na otázku, ktorá sa tohto objektu týka.

Vytvorené riešenie predstavuje len prototyp. Preto by sa v ďalších fázach projektu musel vyriešiť spôsob napájania a uloženia periférnych zariadení v automobile. Prototyp by sa dal po ďalších úpravách rovnako využiť aj ako systém pre vodiča, kde by sa rozšírená realita zobrazila na čelnom skle.

Príloha A. Výsledok testovania treťou stranou

Náš vytvorený produkt sme odovzdali tímu číslo 5 (ARVIS) na ohodnotenie a získanie spätnej väzby. Kedže produkt je orientovaný pre použitie širšou verejnosťou, ktorá má často malé alebo žiadne skúsenosti s obohatenou realitou, získanie spätnej väzby je veľmi dôležité. Takto získaná spätná väzba bude slúžiť pri ďalšom vývoji produktu.

Nových používateľov produktu sme zanechali v miestnosti samotných bez akejkoľvek pomoci členmi nášho tímu len s vytvorenou používateľskou príručkou, ktorá je dostupná na stránke tímu Carlos v časti *Dokumenty* (Kapitola 16). Používatelia boli následne po otestovaní nášho produktu vyzvaní na vyplnenie nasledovného dotazníka.

Carlos - dotazník

Hodnotiteľ: Tím č. 5 (ARVIS)

1. Ako hodnotíte nápad projektu?

Nápad tohto projektu hodnotíme veľmi pozitívne. Auto je veľmi vhodným miestom na použitie obohatenej reality, pretože skoro každá dnešná rodina vlastní minimálne jedno auto a v aute trávime podstatne viac času ako kedysi.

2. Ako ste boli spokojní s voľbou *Turistického infa*?

Voľba *Turistického infa* je prehľadná. Nachádza sa na hlavnej obrazovke mobilnej aplikácie. Je implementovaná prostredníctvom softvérového tlačidla, ktoré je dostatočne veľké a teda sme nezaznamenali problém s jeho používaním. Avšak voľba mohla byť skôr implementovaná prostredníctvom prepínača medzi hrou a infom, kedže by bolo zretelnejšie už len z pohľadu na mobilný telefón, či máme zvolenú hru alebo *Turistické info*.

3. Ako ste boli spokojní s voľbou hry *Lietadlo*?

Všetky spomenuté pozitíva aj negatíva spomenuté v predchádzajúcej otázke platia aj pre túto otázku.

4. Ako ste boli spokojní s ovládaním hry *Lietadlo* pohybom?

K obrazovke, na ktorej sa ovládalo lietadlo pohybom sme sa dopracovali cez tlačidlo na hlavnej oprazovke aplikácie. Chvíľku nám trvalo, kým sme pochopili, že sa cez toto tlačidlo dopracujeme k uvedenej obrazovke. Táto vlastnosť mohla byť v aplikácii viac zvýraznená. Lietadlo sme ovládali posúvaním prstu smerom hore po dostatočne veľkej ploche. Reakcie lietadla na pohyb boli však niekedy oneskorené.

5. Ako ste boli spokojní s ovládaním *Turistického infa* hlasom?

Ovládanie hlasom bolo na dostačujúcej úrovni. Otázky kladené aplikácii boli vo väčšine prípadov správne rozoznané. Niekedy bolo potrebné dlhšiu dobu počkať na odpoveď aplikácie. Celkovo však hlasové ovládanie hodnotíme veľmi pozitívne. Otázky na detegované objekty boli vhodne zvolené.

6. Ako ste boli spokojní s ovládaním hry *Turistického infa* dotykom?

Ovládanie *Turistického infa* dotykom hodnotíne opäť pozitívne. I keď veľkosť tlačidiel nebola zvolená najvhodnejšie. Tlačidla mohli zaberať viac z dostupne voľnej plochy na displeji. Po stlačení tlačidla sa opäť odpoveď v hlavnej aplikácií zobrazila až trošku neskôr.

7. Ako ste boli spokojní s voľbou otázok *Turistického infa*?

Druh otázok bol zvolený vhodne. 2 otázky, ktoré ponúkal systém, vystihujú základnú podstatu, ktorú by sa chcel používateľ dozvedieť o detegovanom objekte. Viac otázok by mohlo pôsobiť zmätočne.

8. Ako ste boli spokojní so zobrazovaním textu *Turistického infa*?

Text sa na obrazovke zobrazoval relatívne na správnom mieste. Niekedy sa vyskytli situácie, že bol úplne mimo miesta, kde sa nachádzal objekt. To sa dá však tolerovať, pretože sa jedná o prototyp aplikácie a pravdepodobne je takéto počítanie polohy textu dosť zložitou operáciou.

9. Ako ste boli spokojní s posúvaním virtuálnch prvkov voči polohe hlavy?

Posúvanie virtuálnych prvkov reagovalo celkom dobre na zmenu polohy hlavy. Objekty, ktoré sa mali posúvať sa občas posúvali trhane a nie plynulo. Toto spôsobovalo nepríjemný dojem, že objekty skáču po obrazovke, no po dlhšej dobe si na to používateľ zvykol, keďže sa naučil pohybovať hlavou tak, aby bol posun virtuálnych prvkov na obrazovke plynulý.

10. Neporozumeli ste niečomu?

Občas nám spôsobovalo problém prepínanie sa medzi *Turistickým infom* a hrou *Lietadlo*. Problém bol v tom, že reakcie boli oneskorené a tlačidlo sme museli viackrát stláčať, čím sme boli mylne v predstave, že aplikácia nereaguje na našu žiadosť. Tento problém bol však spôsobený oneskorenou reakciou kvôli zložitosti a komplexnosti aplikácie.

11. V čom vidíte nedostatky nášho produktu?

Hlavný nedostatok (ktorý je aj jednoznačne badať na produkte) vidíme v tom, že sa jedná o prototyp produktu v takej fáze vývoja, že by bolo potrebné ešte veľa času na testovanie a upravovanie.

12. Čo sa Vám najviac páči na našom produkte?

Na produkte sa nám najviac páčil jeho nápad. Obohatená realita v aute nie je v súčasnosti až tak často využívaná, aj keď na Internete je možné nájsť i zrealizované príklady. V prípade skutočného nasadenia takéhoto produktu do aút by určite išlo o veľmi úspešný a atraktívny produkt a tento nápad je vhodným aj pre startup.

13. Čo by ste zlepšili na našom produkte?

Do produktu by sme pridali určite viac rôznorodých hier. Tie by určite zaujali maloletých používateľov. Takisto *Turistické info* by sa mohlo rozšíriť o ďalší druh poskytovaných informácií, ďalší okruh otázok alebo nejakú hru formu kvízu.

Príloha B. Záverečná správa TP Cup 2014

Táto príloha predstavuje Záverečnú správu vypracovanú v rámci súťaže TP Cup 2014.

Záverečná správa – TP Cup 2014

Tím: Tím č. 3 (Carlos)
Téma: Zábavný systém pre spolucestujúcich v automobile (AUTO)
Vedúci: Ing. Vanda Benešová, PhD.
Členovia: Bc. P. Polatsek, Bc. M. Petluš, Bc. J. Mercz, Bc. L. Sekerák, Bc. P. Hamar, Bc. R. Sabol
Web: http://labss2.fiit.stuba.sk/TeamProject/2013/team03is-si/
Kontakt: team03.1314@gmail.com

Abstrakt

Zábavné a informačné systémy sa stávajú čoraz častejšie súčasťou nášho života. Jedným z najmodernejších a pre človeka najprirodzenejších spôsobov interakcie je obohatená realita, kde reálny svet je doplnený o obrazové, textové alebo zvukové virtuálne prvky.

Rozšírená realita nám umožňuje veľmi jednoducho sprostredkovať turistické informácie o okolí počas jazdy automobilom alebo využiť realitu ako základ pre rôzne interaktívne hry. Automobil sa takýmto spôsobom môže stať interaktívnym turistickým sprievodcom alebo zábavným systémom.

Cieľom tohto projektu je zmeniť bočné okno automobilu na transparentnú projekčnú plochu, pomocou ktorej je reálny svet dopĺňaný o ľubovoľné virtuálne informácie so zábavným i náučným zámerom. Na okne auta nám tak vzniká rozšírená realita, ktorá nás informuje o našom bezprostrednom okolí.

Výsledkom projektu je prototyp interaktívneho systému pre obohatenú realitu pre spolucestujúceho v automobile s názvom Carlos. Na prezentáciu vygenerovaných informácii je požitý malý LED projektor spolu s transparentnou projekčnou fóliou, ktorá je pripevnená na bočné sklo automobilu. Aplikáciu môže používateľ ovládať veľmi jednoducho – prostredníctvom mobilného telefónu.

Základom našej aplikácie je rozoznávanie kamerou nasnímaných zaujímavých objektov. K takto detegovanému objektu doplní do reálnej scény nielen jeho názov, ale aj iné zaujímavé informácie. Napr. pri pamiatkach zobrazí históriu, otváracie hodiny a výšku vstupného. Výhodou našej aplikácie bude aj to, že nebude vyžadovať pripojenie na internet.

Carlos nepredstavuje len informačný, ale aj zábavný systém. Preto je súčasťou systému aj hra, ktorá využíva rozšírenú realitu. Účelom tejto hry je ovládať prostredníctvom mobilného telefónu lietadlo s cieľom udržať ho čo najdlhšie nad horizontom.

Implementácia systému zahŕňa viaceré algoritmy z oblasti počítačovej vízie, počítačovej grafiky a interakcie človeka s počítačom. Systém využíva viaceré periférie ako kameru na zachytenie snímok, zariadenie Kinect na sledovanie pozície hlavy, mobilný telefón s operačným systémom Android na získavanie aktuálnych GPS súradníc a na ovládanie celého systému a nakoniec projektor na zobrazenie virtuálnych informácií na sklo automobilu s transparentnou fóliou.

1. Opis problémovej oblasti a prehľad riešenia

Zábavné a informačné systémy sa stávajú čoraz častejšie súčasťou nášho života. Jedným z najmodernejších a pre človeka najprirodzenejších spôsobov interakcie je obohatená realita, kde reálny svet je doplnený o obrazové, textové alebo zvukové virtuálne prvky.

Rozšírená realita nám umožňuje veľmi jednoducho sprostredkovať turistické informácie o okolí pri jazde automobilom v reálnom čase alebo využiť realitu ako základ pre rôzne interaktívne hry. Automobil sa takýmto spôsobom môže stať interaktívnym turistickým sprievodcom alebo zábavným systémom.

Cieľom tohto projektu je zmeniť bočné okienko automobilu na transparentnú projekčnú plochu, pomocou ktorej bude reálny svet dopĺňaný o ľubovoľné virtuálne informácie so zábavným i náučným zámerom. Na okienku auta nám tak vznikne rozšírená realita, ktorá nás môže informovať o našom bezprostrednom okolí.

Výsledkom projektu je prototyp interaktívneho systému pre obohatenú realitu pre spolucestujúceho v automobile.

Na prezentáciu vygenerovaných informácii sa použije malý LED projektor spolu s transparentnou projekčnou fóliou, ktorá sa pripevní na bočné sklo automobilu. Aplikáciu používateľ ovláda veľmi jednoducho – prostredníctvom mobilného telefónu, pohybom alebo hlasom (obrázok č. 1).

Obrázok 1. Prehľad systému

Náš systém s názvom Carlos pozostáva z 2 základných režimov, ktoré predstavujú možnosti ako sa dá využiť rozšírená realita:

1. **Informačný** – zobrazovanie turistických informácií
2. **Zábavný** – hra Lietadlo

Základom nášho systému je rozoznávanie zaujímavých objektov nasnímaných kamerou v reálnom čase. K takto detegovanému objektu doplní do reálnej scény nielen jeho názov, ale aj iné zaujímavé textové informácie. Napr. pri pamiatkach zobrazí otváracie hodiny, výšku vstupného a základné historické informácie. Výhodou našej aplikácie bude aj to, že nebude vyžadovať pripojenie na internet, keďže všetky údaje o objektoch sú uložené v lokálnej databáze.

Carlos nepredstavuje len informačný, ale aj zábavný systém. Preto je súčasťou systému aj hra s názvom Lietadlo, ktorá využíva reálne prostredie. Účelom tejto hry je ovládať prostredníctvom mobilného telefónu lietadlo s cieľom udržať ho čo najdlhšie nad zdetegovaným horizontom.

Celý systém sa ovláda cez mobilné zariadenie so systémom Android (obrázok č. 2). Pomocou tejto aplikácie sa rovnako používateľ prepína medzi turistickou a hernou časťou systému.

Náš systém deteguje pomocou snímok z kamery na aute objekty ako napr. pamiatky, reštaurácie, kaviarne a hotely. V rámci detekčnej fázy systém porovná snímky s využitím aktuálnej GPS pozície a internej databázy objektov záujmu. Detekcia začína so selekciou potenciálnych objektov, ktoré sa môžu na snímke zdetegovať. Z databázy sa vyberú tie objekty, ktoré sú v blízkosti GPS pozície získavanej v pravidelných intervaloch z mobilného zariadenia (obrázok č. 2 vpravo).

Obrázok 2. Obrazovky mobilnej aplikácie.

Následne sa vykoná detekcia objektov, ktorá spočíva v extrakcii príznakov a porovnaním *deskriptorov*, ktoré opisujú okolie kľúčových bodov (obrázok č. 3 vľavo). Výhodou je, že deskriptory sú vo väčšej miere invariantné voči zmenám jasu a deformácii obrazu. V našom systéme využívame SIFT deskriptory pre grafické karty NVidia s CUDA z knižnice SiftGPU. Na zrýchlenie celého procesu detekcie sú deskriptory pre objekty v databáze dopredu predpočítané a načítavané zo súboru. Po extrakcii deskriptorov z aktuálnej snímky okolia sa hľadajú zhody medzi touto snímkou a potenciálnymi objektmi.

Po úspešnej detekcii objektov sa určí ich poloha na snímke pomocou *homografie* (obrázok č. 3 vpravo) s využitím tzv. RANSAC (RANdom SAmple Consensus) metódy.

Obrázok 3. Detekcia objektu pomocou deskriptorov a určenie jeho polohy s využitím homografie.

Kvôli správnemu zobrazeniu virtuálnych informácií na obrazovke využíva náš systém zariadenie Kinect. Pomocou video- a hĺbkovej informácie získanej z Kinect-u sa určí aktuálna pozícia hlavy používateľa, aby boli obrazové informácie zobrazené na skle automobilu čo najpresnejšie voči aktuálnemu pohľadu používateľa (obrázok č. 4).

Po úspešnej detekcii objektov na aktuálnej snímke sa zobrazí v hornom ľavom rohu informácia, že v okolí boli identifikované objekty. Používateľ si môže pomocou mobilnej aplikácie – stlačením príslušného tlačidla alebo hlasom zobraziť názvy objektov, príp. základné informácie o týchto objektoch.

Obrázok 4. Posun zobrazených informácií na základe polohy hlavy.

Nakoniec sa po prepočítaní pozícií všetkých virtuálnych elementov zobrazia tieto turistické informácie v textovej alebo obrazovej podobe na bočnom skle automobilu pomocou projektora.

Carlos rovnako ponúka aj leteckú hru, kde hráč musí udržať lietadlo nad horizontom (obrázok č. 5). Na detekciu horizontu je použitý hranový detektor *Canny*, ktorý na snímke určí oblasti oblohy (obrázok č. 6). Horizont je následne upravený tak, že sú z neho odstránené prudké zvýšenia, ktoré sú pravdepodobne spôsobené dopravnou značkou, príp. stromom.

V prípade pádu lietadla sa na obrazovke zobrazí dĺžka letu. Let lietadla je ovládaný jednoduchým nakláňaním mobilného zariadenia, príp. pohybom.

Obrázok 5. Ukážky z hry "Lietadlo".

Obrázok 6. Detekcia regiónov oblohy (biela farba).

2. Realizácia riešenia

Na implementáciu riešenia využívame programovacie jazyky C/C++, Javu pre mobilnú platformu Android a knižnice OpenCV, OpenGL a Freenect. Implementácia systému zahŕňa viaceré algoritmy z oblasti *počítačovej vízie*, *počítačovej grafiky* a *interakcie typu človek-počítač*.

Náš systém Carlos využíva viaceré periférne zariadenia:

1. *kamera* na zachytenie snímok, ktoré budú následne spracované detekčnými algoritmami,
2. *Kinect* na sledovanie pozície hlavy pre správne umiestnenie virtuálnych elementov na obrazovke vzhľadom na aktuálny uhol pohľadu používateľa,
3. *mobilný telefón* s operačným systémom Android na získavanie aktuálnych GPS súradníc, ktoré sa využijú na selekciu objektov z databázy, ktoré sú v bezprostrednej blízkosti a na ovládanie celého systému,
4. *projektor* na zobrazenie virtuálnych informácií na sklo automobilu s transparentnou fóliou.

Náš systém pozostáva z viacerých modulov, ktorých štruktúru znázorňuje obrázok č. 7:

- **Riadiaci modul** predstavuje centrálny modul, ktorý spravuje a synchronizuje zvyšné moduly. Riadi ich vstupy a výstupy, prijíma snímky z kamery a zobrazuje virtuálne informácie cez projektor.

- **Modul spracovania obrazu** deteguje objekty a ich polohu na snímkach pomocou deskriptorov a určuje oblasti na snímke, ktoré zodpovedajú oblohe pomocou hranového detektora.

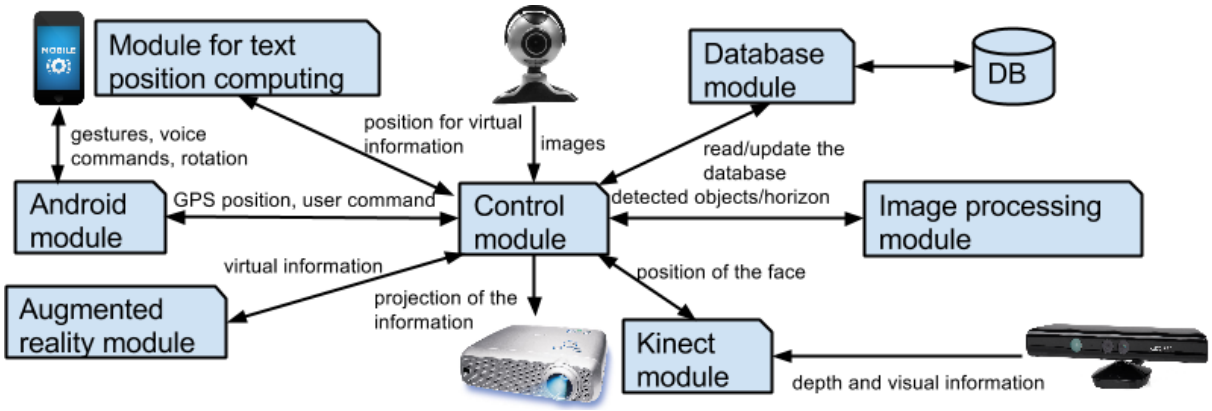
- **Modul Kinect** spracováva hĺbkovú a video informáciu z Kinect-u na určenie polohy hlavy.

- **Modul výpočítania polohy textu** vypočítava na základe informácie o polohe objektov na snímke a pozícii hlavy používateľa presné umiestnenie virtuálnych prvkov na skle automobilu.

- **Modul Android** predstavuje Java aplikáciu pre mobil s OS Android, ktorý ponúka rozhranie na ovládanie celého systému (hlas, rotácia zariadenia) a zároveň zaznamenáva aktuálnu GPS polohu.

- **Modul rozšírenej reality** generuje informácie, ktoré majú byť zobrazované cez projektor. Zároveň implementuje leteckú hru, ktorá využíva rozšírenú realitu.

- **Modul databázy** spravuje internú databázu objektov záujmu, ktoré obsahuje fotografie, GPS pozície, predpočítané deskriptory a základné informácie o objektoch.



Obrázok 7. Základná štruktúra nášho systému.

Výsledkom nášho projektu je funkčný prototyp systému pre spolujazdca s rozšírenou realitou. Prototyp obsahuje leteckú hru ako aj interaktívneho turistického sprievodcu.

Na testovanie sme mali k dispozícii počítač s grafickou kartou NVidia, zariadenie Kinect a sklo s transparentnou fóliou. Na simulovanie jazdy v automobile sme využili 2 projektory (obrázok č. 8). Jeden z nich zobrazoval doplnkové informácie na sklo a druhý premietal na stenu videozáznam „okolia“. Pomocou webkamery s 30 FPS sme si vytvorili 2 typy záznamov – v meste a mimo mesta.

Obrázok 8. Ukážka testovania systému.

V tomto simulovanom prostredí sme otestovali posúvanie zobrazovaných informácií voči aktuálnej zdetegovanej polohy hlavy. Na videosnímkach mimo mesta bola odskúšaná hra Lietadlo a v rámci nej detegovanie horizontu a ovládanie lietadla mobilným telefónom. Mestské videá slúžili na testovanie turistického sprievodcu a rozpoznávania objektov.

Detekcia objektov si však vyžaduje ďalšie a dôkladnejšie testovanie, príp. úpravy v algoritme detekcie. Nesprávne rozpoznávanie objektov bolo väčšinou spôsobené vysokou podobnosťou objektov alebo nevhodnými fotografiami objektov, ktoré neobsahovali žiadne charakteristické rysy objektu, príp. veľkú časť snímky objektu tvorilo pozadie. Úspešnosť detekcie sa samozrejme znižovala v závislosti od natočenia snímaného objektu voči fotografii objektu v lokálnej databáze.

3. Zhrnutie

Primárnym cieľom nášho projektu bolo poukázať na možnosti, ktoré nám ponúka rozšírená realita, či už na edukačné alebo herné účely. Vytvorený prototyp by sa dal využiť nielen v automobiloch, ale aj iných dopravných prostriedkoch.

Rozšírená realita ponúka obrovské možnosti uplatnenia v interaktívnych systémoch. Modularita nášho systému umožňuje jednoduché rozšírenie o ďalšiu funkcionalitu.

Jednou zo zamýšľaných funkcií, ktoré sa nestihli implementovať bola napríklad kvízová hra, kde by používateľ musel uhádnuť, ktorý objekt je v jeho okolí alebo zodpovedať na otázku, ktorá sa tohto objektu týka.

Vytvorené riešenie predstavuje len prototyp. Preto by sa v ďalších fázach projektu musel vyriešiť spôsob napájania a uloženia periférnych zariadení v automobile. Prototyp by sa dal po ďalších úpravách rovnako využiť aj ako systém pre vodiča, kde by sa rozšírená realita zobrazila na čelnom skle.

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava 4

Zábavný systém pre spolucestujúcich v automobile

Tímový projekt

Dokumentácia riadenia projektu

Vedúci tímového projektu: Ing. Vanda Benešová, PhD.
Členovia tímu:
Bc. Patrik Polatsek
Bc. Martin Petluš
Bc. Peter Hamar
Bc. Róbert Sabol
Bc. Jakub Mercz
Bc. Lukáš Sekerák

Názov tímu: Tím č. 3 (Carlos)
Web: http://labss2.fiit.stuba.sk/TeamProject/2013/team03is-si/
Kontakt: team03.1314@gmail.com
Akademický rok: 2013/2014

Obsah

1. Úvod

Tento dokument poskytuje informácie o riadení projektu s názvom Zábavný systém pre spolucestujúcich v automobile, ktorý je vypracovaný v rámci predmetu Tímový projekt. Tento projekt je riešený tímom č. 3 - Carlos.

Tím Carlos je pod vedením Ing. Vandy Benešovej, PhD. v nasledovnom zložení:

- Bc. Patrik Polatsek - vedúci tímu
- Bc. Martin Petluš - manažér kvality
- Bc. Peter Hamar - manažér podpory vývoja
- Bc. Róbert Sabol - manažér monitorovania projektu
- Bc. Jakub Mercz - manažér rozvrhu
- Bc. Lukáš Sekerák - manažér rizík
- Bc. Marianna Mušinská - manažér ľudských zdrojov
- Bc. Juraj Jarábek - manažér dokumentovania

Pozn. Bc. Marianna Mušinská a Bc. Juraj Jarábek už nie sú súčasťou tímu.

Úvodná časť popisuje náš tím, jej členov a ich kvalifikácie. Obsahuje Zoznam kompetencií tímu, ktorý bol vypracovaný pri uchádzaní sa o tímový projekt.

Jadro dokumentu tvoria metodiky na dolnej úrovni vytvorené v rámci predmetu Manažment v informačných systémoch/Manažment v softvérovom inžinierstve a stručné opisy jednotlivých manažérskych oblastí..

Dokument rovnako obsahuje jednotlivé zápisy zo stretnutí, ktoré dokumentujú jednotlivé stretnutia tímu. V každom zápise je uvedené čo bolo cieľom tohto stretnutia a aké boli úlohy jednotlivých členov tímu.

2. Zoznam kompetencií tímu

V tejto časti sa nachádza Zoznam kompetencií tímu, ktorý sa odovzdával za účelom získania projektu na začiatku zimného semestra. Obsahuje predstavenie tímu Carlos spolu s ponukami k 3 projektom. V závere tejto kapitoly sa nachádza poradie prefencií k projektom.

2.1 Predstavenie tímu

Patrik Polatsek (IS).
- ovláda: C/C++, Java, SQL, OpenCV
- bakalárska práca: Detekcia žmurknutia používateľa počítača (prostredníctvom snímkov z webkamery sa detekovalo žmurknutie v C++ programe s využitím knižnice OpenCV)
- pracovná pozícia: SAP ABAP developer

Martin Petluš (IS).
- ovláda: C/C++, Java, SQL, JavaScript
- bakalárska práca: Strojové učenie ohodnocovacej funkcie pre vyhľadávač (vo webovom vyhľadávači sme zaznamenávali dopyty používateľov a na základe zaznamenaných dát, sme sa naučili modely pre používateľov, podľa ktorých sme používateľom zoraďovali výsledky vo vyhľadávači, Java a Ruby on Rails)
- pracovná pozícia: vývíjanie webovej aplikácie, chat klient, video konferencie (HTML5)

Jakub Mercz (IS).
- ovláda: C/C++, Java, SQL, XML
- bakalárska práca: Vytváranie databázových dopytov v prirodzenom jazyku (aplikácia na základe vstupu v prirodzenom jazyku vytvorila SQL dopyt pre databázu s využitím parsovania a prehľadávania štruktúry databázy)

Lukáš Sekerák (IS).
- ovláda: C/C++, Java, MysqlSQL, OracleSql, PHP, Ruby, XML, JavaScript, HTML5, UML, CSS, Android, jQuery, phoneGap. 1 rok Android skúsenosti. 7 rokov web developer.
- bakalárska práca: Interaktívna vizualizácia informačnej siete (Aplikácia ktorá zobrazovala vzťahy vyše 40 miliónov entít. Entity boli zobrazované vrcholmi v grafe a predstavovali rôzne osoby, tel. čísla, mená, informácie,...)
- pracovná pozícia: CIIT.at Java web junior developer + Android developer

Peter Hamar (IS).
- ovláda: C/C++, PHP, Java, SQL, Ruby on Rails, XML

- bakalárska práca: Správa citácií (automatizovanie tvorby citácií a bibliografických odkazov v existujúcom nástroji Annota)

Róbert Sabol (SI).
- ovláda: C/C++, Java, MySQL, C#, Objective-C, XML, HTML, JavaScript, Android development, iOS development, UML
- bakalárska práca: Prepojenie TV vysielania a sociálnych sietí (Android aplikácia)
- pracovná pozícia: iOS developer

Juraj Jarábek (SI).
- ovláda: C, Java, SQL, Android development, Perl, Python
- bakalárska práca: Aplikácia pre záznam nálady s prvkami gemifikácie pre operačný systém Android
- pracovná pozícia: Java Developer

Marianna Mušinská (IS).
- ovláda: C/C++, Java, SQL
- bakalárska práca: Databázový systém webshopu (webová aplikácia slúžiaca ako webshop)

2.2 Téma č. 1 - 2 Zábavný systém pre spolucestujúcich v automobile

Náš tím by chcel vytvoriť program, ktorý bude obohacovať realitu o virtuálne prvky. Aplikácia by mohla pomocou gps a rozoznávaním objektov z internej databázy upozorňovať a zvýrazňovať zaujímavé objekty v okolí vodiča. Aplikácia by zobrazovala základné informácie o okolitých objektoch, príp. ich vzdialenosť od auta, pričom údaje získa z internetu a zo svojej databázy (napr. pri pamiatkach zobrazí otváracie hodiny, výšku vstupného, fotografie vo forme pútača). Používateľ si na začiatku zvolí kategórie, na ktoré chce byť upozorňovaný (pamätihodnosti, reštaurácie, kaviarne, …). Aplikácia bude doplnená aj o hru pre deti, ktoré by prostredníctvom android mobilného telefónu ovládali postavičku, ktorá by sa pohybovala v zobrazovanej realite.

V našom tíme máme človeka, ktorý pracoval v rámci svojej bakalárskej práce s knižnicou OpenCV, kde rozoznával žmurkanie používateľa pred webkamerou, pričom na detekciu využil viaceré nezávislé metódy. Väčšina tímu mala v bakalárskom štúdiu predmet Počítačová grafika, kde sme sa naučili základy grafiky a prácu s knižnicou OpenGL. Traja členovia tímu v rámci svojej bakalárskej práce vytvorili aplikáciu pre platformu Android.

2.3 Téma č. 2 - 6 Virtuálna FIIT na mobile

Téma virtuálna FIIT na mobile nás zaujala tým, že presne definuje čo by bolo našou úlohou. Vytvorili by sme interaktívnu aplikáciu s pekným používateľským prostredím a grafikou. Krásnu grafiku by nám spravila Marianna, ktorá má talent pre grafické cítenie.

Hlavným cieľom by bolo rozšíriť aplikáciu medzi najväčší počet študentov. Preto navrhujeme spraviť anketu (najlepšie teda medzi prvákmi / novými študentami), spýtať sa kolegov aké sú ich požiadavky a podľa týchto požiadaviek vytvoriť aplikáciu. Predpokladáme, že typický používatelia by boli noví študenti, ktorí našu budovu nepoznajú a majú problém s orientáciou.

Ďalšie nápady sú:

- plnohodnotná navigácia spolu s QR kódmi
- spolupráca aplikácie s hladnystudent.sk, mhd, možno okolitým prostredím
- rôzne metadáta učební a kancelárií, napríklad kapacita, názov, aktuálna výučba v triede, v prednáške
- prepojená aplikácia s AIS študenta, navigácia, kde má študent najbližšie cvičenie
- prepojenie so sociálnymi sieťami - (priatelia, skupiny (krúžky), poloha priateľov)

- chat v rámci svojho krúžku

V tíme máme 3 expertov na android, 2 ďalších na JavaScript, jeden člen tímu má skúsenosti s parsovaním hladnystudent.sk. Zároveň máme experta na QR kódy a vyššie spomenutú grafičku. Sme teda dosť silný team pre túto tému a schopní zrealizovať akýkoľvek nápad.

2.4 Téma č. 3 - 5 Vizualizácia informácií v obohatenej realite

Náš tím by chcel vytvoriť aplikáciu, vďaka ktorej by bol používateľ schopný interagovať napr. s grafmi v obohatenej realite. S grafmi by používateľ pracoval prostredníctvom nasnímaných špeciálnych značiek, príp. pohybom rúk.

V našom tíme máme človeka, ktorý pracoval v rámci svojej bakalárskej práce s knižnicou OpenCV, kde rozoznával žmurkanie používateľa pred webkamerou, pričom na detekciu využil viaceré nezávislé metódy. Väčšina tímu mala v bakalárskom štúdiu predmet Počítačová grafika, kde sme sa naučili základy grafiky a prácu s knižnicou OpenGL. Traja členovia tímu v rámci svojej bakalárskej práce vytvorili aplikáciu pre platformu Android.

2.5 Zoradenie všetkých tém podľa priority

1. Zábavný systém pre spolucestujúcich v automobile
2. Virtuálna FIIT na mobile
3. Vizualizácia informácií v obohatenej realite
4. Analýza výsledkov výskumu
5. Webový komunitný systém otázok a odpovedí
6. Digital SweatShop
7. Trojdimenzionálne UML
8. Distribuované počítanie na FIIT
9. Prehliadka kódov v tímových projektoch
10. Sledovanie pohľadu pri používaní aplikácií
11. Monitor programátora v IDE
12. Interaktívne hry na mobile s multimediálnym obsahom
13. 3D Robotický futbal

3. Úlohy členov tímu

V tejto kapitole sa nachádza prehľad jednotlivých členov tímu spolu s ich primárnymi úlohami a manažérskymi rolami.

Bc. Patrik Polatsek - vedúci tímu
Zameranie:

- práca s knižnicou OpenCV
- práca na úlohách súvisiacimi s počítačovým videním
- práca na detekcii a rozoznávaní objektov zo snímok

Bc. Martin Petluš - manažér kvality
Zameranie:

- technologická podpora tímu
- práca na module počítajúcom polohu textu na okne
- zodpovedný za virtuálny server tímu

Bc. Peter Hamar - manažér podpory vývoja
Zameranie:

- práca s knižnicou OpenGL
- práca s 3D objektami a scénou
- vytvorenie hry, ktorá bude využívať prvky počítačového videnia

Bc. Róbert Sabol - manažér monitorovania projektu
Zameranie:

- práca na Android module
- práca na rozpoznávaní gest
- vytvorenie TCP modulu v Module Android

Bc. Jakub Mercz - manažér rozvrhu
Zameranie:

- práca s údajmi z Kinectu
- detekcia tváre, hlavy

Bc. Lukáš Sekerák - manažér rizík
Zameranie:

- návrh celej architektúry, integrácia modulov
- práca s hlavným algoritmom, riadiacim modulom

- vytvorenie TCP serverového modulu, databázového modulu

Bc. Marianna Mušinská - manažér ľudských zdrojov

Bc. Juraj Jarábek - manažér dokumentovania
Zameranie:

- návrh a implementácia časti Android modulu
- dokumentácia zdrojových kódov (Doxygen)

4. Metodiky (dolná úroveň)

V tejto kapitole sa nachádzajú jednotlivé metodiky členov tímu Carlos, ktoré boli vypracované k predmetu Manažment v informačných systémoch/Manažment v softvérovom inžinierstve.

V nasledovnej tabuľke sa nachádza prehľad metodík a ich autorov.

Metodika	**Autor**
Metodika evidencie nových úloh v Redmine	Polatsek
Metodika písania Jednotkových (Unit) testov	Hamar
Metodika - Dokumentácia zdrojových kódov za použitia Doxygen (Manažment zdrojových kódov)	Sekerák
Zdielanie zdrojov v Google Drive	Petluš
Metodika - Manažment požiadaviek na zmenu	Sabol
Metodika plánovania úloh	Mercz
Metodika prehliadky kódov vo dvojiciach	Mušinská
Metodika - Manažment verzií zdrojového kódu	Jarábek

4.1 Metodika evidencie nových úloh v Redmine

Autor: Patrik Polatsek

4.1.1 Úvod

Táto metodika definuje pravidlá a postupy pri zakladaní úloh vo webovom nástroji na manažment projektov Redmine. Po schválení a priradení úloh na stretnutí pred ďalším šprintom je potrebné úlohy zaevidovať do systému procesmi spomínaných v tejto metodike.

Pravidlami definovanými v tejto metodike sa riadi manažér plánovania, ktorý je za evidenciu nových pridelených úloh do systému Redmine a s nimi súvisiacich činností zodpovedný.

4.1.1.1 Použité pojmy a skratky

- *Redmine*: open-source webový nástroj na manažment projektov a sledovanie chýb, ktorý zahŕňa okrem iného aj Ganttov graf a kalendár
- *Scrum*: metóda agilného vývoja softvéru
- *Šprint*: základná časová jednotka vo vývoji v Scrume. Každému šprintu predchádza plánovanie, kde sú identifikované úlohy, ktoré sa majú do konca šprintu vykonať.

4.1.1.2 Roly a zodpovednosti

V Tabuľke 1 sú uvedené roly, ktoré vystupujú v tejto metodike a ich zodpovednosti relevantné s touto metodikou.

Rola	**Zodpovednosť**
Manažér plánovania	● Zaevidovanie a pridelenie novej úlohy ● Zaevidovanie nového šprintu

Tabuľka 1. Roly a zodpovednosti

4.1.1.3 Príbuzné metodiky

- Metodika plánovania
- Metodika zdielania zdrojov na Google Drive

4.1.2 Vytvorenie novej úlohy v Redmine

Vstup: identifikovaná úloha na stretnutí tímu
Výstup: vytvorená nová úloha
Zodpovedný: manažér plánovania

Na stretnutí tímu sa pri novom šprinte identifikujú nové úlohy, ktoré sa následne priradia ako je opísané v *Metodike plánovania*. Pri založení novej úlohy v systéme Redmine, ktorá nie je priamou podúlohou už k existujúcej úlohe, sa manažér plánovania riadi nasledovnými krokmi, ktoré sú zobrazené na Obrázku 1. V prípade vytvorenia novej podúlohy manažér plánovania postupuje krokmi opísanými v časti *Vytvorenie novej podúlohy k existujúcej úlohe* v tejto metodike.

1. Prihlásiť sa do systému Redmine.
2. Vybrať si príslušný projekt.
3. V prípade, že nie je v systéme zadefinovaný šprint, do ktorého sa má úloha pridať, je potrebné pred vykonaním nasledovného kroku vytvoriť tento šprint ako je opísané v tejto metodike v časti *Vytvorenie nového šprintu* od kroku číslo 3.
4. Kliknúť na tlačidlo *New issue*. Zobrazí sa okno zobrazené na Obrázku 1.
5. Vybrať typ úlohy v časti *Tracker* ako je uvedené v Tabuľke 2.

	Prípady použitia
Feature	● úlohy týkajúce sa vývoja ● implementačné úlohy ● úlohy týkajúce sa testovania
Support	● úlohy týkajúce sa analýzy ● úlohy týkajúce sa návrhu ● tvorba dokumentácie ● všeobecné úlohy týkajúce sa chodu tímového projektu

Bug	• opravy zistených chýb

Tabuľka 2. Typy úloh

6. Napísať výstižný názov úlohy v časti *Subject* v nasledovnom tvare: *[modul] popis*, kde
   - *modul* charakterizuje zaradenie úlohy jedným slovom – opisuje modul alebo oblasť, ktorej sa úloha týka
   - *popis* opisuje výstižne (max. 5 slovami) pointu úlohy.
7. Opísať štruktúrovane úlohu v časti *Decsription* nasledovne:
   1. Popis prvej časti úlohy
   2. Popis druhej časti úlohy
   3. …

   Jedna úloha je rozdelená číselne do samostatných bodov, na ktoré sa potom člen tímu pri aktualizácii stavu svojej úlohy odkazuje.
8. Ponechať *Status* úlohy na defaultný – *New*.
9. Zvoliť prioritu úlohy podľa jej dôležitosti v časti *Priority* podľa Tabuľky 3.

Priorita úlohy	**Prípady použitia**
Normal	Štandardná úloha, ktorá by mala byť vyriešená podľa vopred zadaného termínu
High	Dôležitá úloha, ktorá musí byť vyriešená do ďalšieho týždňa
Low	Úloha nie je dôležitá a jej vyriešenie nie je kľúčové pre chod tímu a systému
Urgent	Veľmi dôležitá úloha, ktorú treba riešiť prioritne
Immediate	Závažná úloha, ktorá musí byť vyriešená okamžite po jej vytvorení

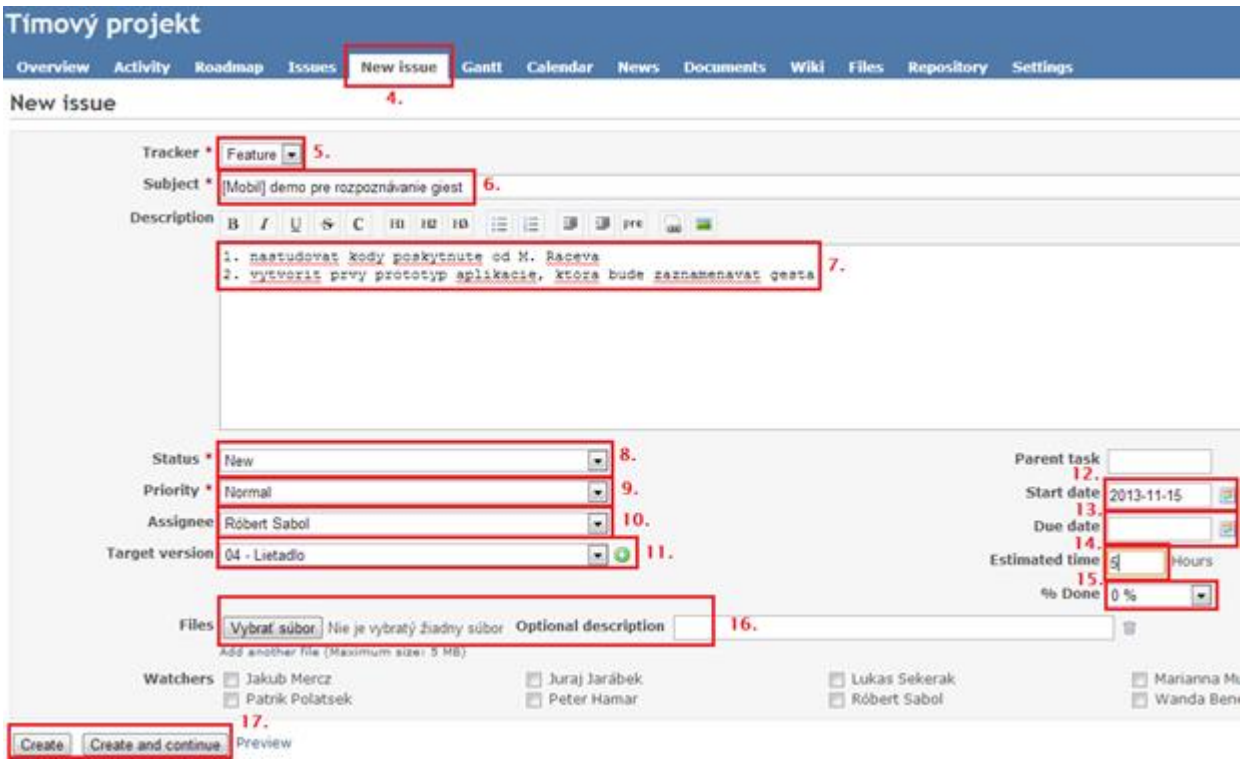
Tabuľka 3. Priority úloh

10. Priradiť úlohu zvolenému členovi tímu v časti *Assignee*.
11. Priradiť úlohu do šprintu v časti *Target version*.
12. Zvoliť počiatočný dátum v časti *Start date* na dátum, kedy bola úloha členovi tímu zadaná.
13. V prípade, že má byť úloha vyriešená skôr než je koniec šprintu, vybrať tento dátum v časti *Due date*.
14. Vyplniť časť *Estimated* time v prípade, že sa na úlohu stanoví odhadovaný čas v hodinách.
15. Percentuálne dokončenie úlohy v časti *% done* ponechať na 0.
16. Ak k vypracovaniu danej úlohy existuje súvisiaci dokument, vložiť tento dokument cez tlačidlo *Vybrať súbor* v časti *Files* a vložiť k nemu max. 5 slovný popis v časti *Optional*

Description. V prípade, že sa bude prikladať viac súborov, stlačiť tlačidlo *Add another file* a postup v tomto kroku opakovať. Do systému Redmine je možné vkladať súbory len s max. veľkosťou 5 MB. V opačnom prípade je potrebné vložiť tento súbor do zdielaného priečinku na Google Drive podľa postupu opísanej v *Metodike zdielania zdrojov na Google Drive*. K takto vloženému súboru vygenerujeme link, ktorý vložíme na koniec popisu úloh v časti *Description* nasledovne: *Príloha X: link,* kde

- *X* je poradové písmeno prílohy
- *link* je link k súboru na Google Drive

17. Kliknúť na tlačidlo *Create* alebo *Create and continue*, ak sa bude zadávať ešte ďalšia úloha do systému.
18. V prípade, že má úloha priamo súvisieť s existujúcou úlohou, ktorá je už zaevidovaná v systéme, je potrebné postupovať krokmi v časti *Vytvorenie závislosti medzi úlohami* v tejto metodike od kroku 3.



Obrázok 1. Postup pri vytváraní novej úlohy v Redmine

4.1.2.1 Vytvorenie nového šprintu

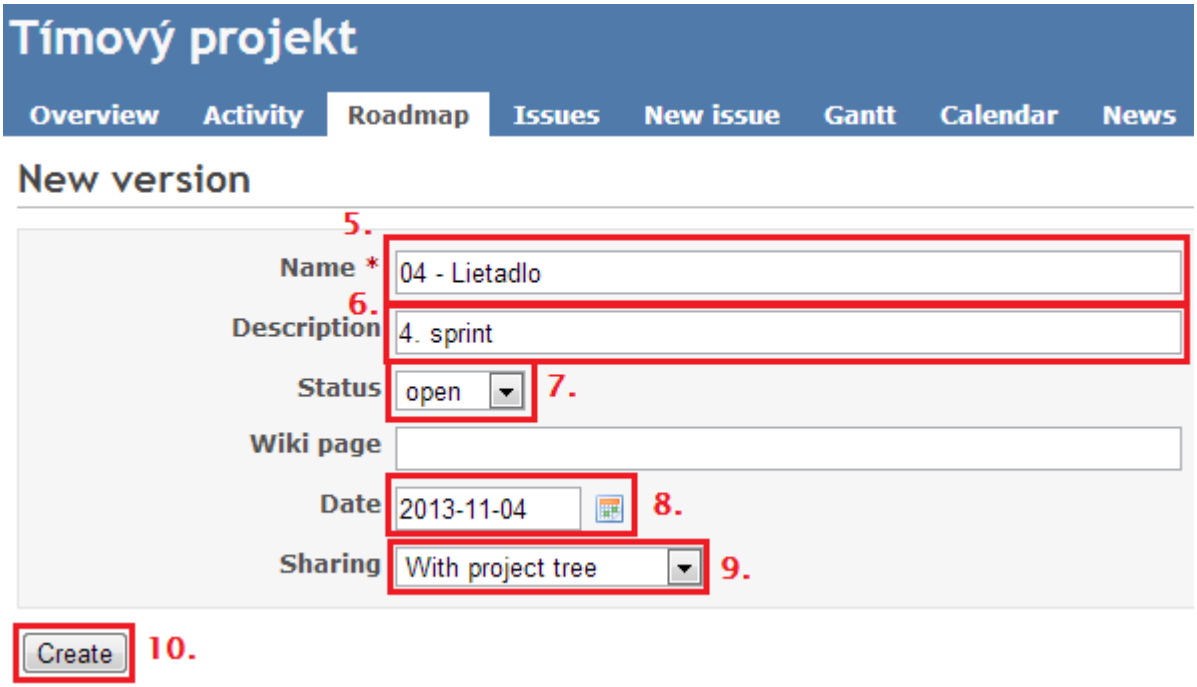
Vstup: ukončenie predchádzajúceho šprintu, dokument s novými identifikovanými úlohami

Výstup: zadefinovaný šprint v Redmine

Zodpovedný: manažér plánovania

Pri zadefinovaní nového šprintu do systému Redmine musí manažér plánovania postupovať týmito krokmi, ktoré sú zobrazené aj na Obrázku 2:

1. Prihlásiť sa do systému Redmine.
2. Vybrať si príslušný projekt.
3. Kliknúť na tlačidlo *Settings* – *Versions*.
4. Kliknúť na *New version.* Zobrazí sa okno z Obrázku 2.
5. Zadať názov šprintu v časti *Name* v nasledovnom tvare: *por.č.* – *meno*, kde
   - *por.č.* je 2-ciferné poradové číslo šprintu
   - *meno* je jednoslovný názov šprintu.
6. Do časti *Description* napísať *X. šprint*, kde X je poradové číslo šprintu.
7. Ponechať stav šprintu na otvorený – *open* v časti *Status.*
8. V časti *Date* zadať dátum mítingu, ktorému začiatok šprintu predchádzal.
9. *Sharing* nastaviť na *With project tree*.
10. Kliknúť na tlačidlo *Create*.



Obrázok 2. Postup pri zakladaní šprintu v Redmine

4.1.2.2 Vytvorenie novej podúlohy k existujúcej úlohe

Vstup: identifikovaná nová podúloha k existujúcej úlohe

Výstup: vytvorená nová podúloha v Redmine

Zodpovedný: manažér plánovania

V prípade zaevidovania novej úlohy do systému Redmine, ktorá je podúlohou k existujúcej úlohe, musí manažér plánovania postupovať nasledovne:

1. Prihlásiť sa do systému Redmine.
2. Vybrať si príslušný projekt.
3. Kliknúť na tlačidlo *Issues*.
4. V prípade potreby si vyselektovať úlohy pomocou filtra cez ich stav a potvrdiť cez *Apply*.
5. Kliknúť na úlohu, ktorá má byť nadúlohou tejto úlohy.
6. Kliknúť na *Add* v časti *Subtasks* (bod A. na Obrázku 3).
7. Pokračovať v tejto metodike od kroku 3 v časti *Vytvorenie novej úlohy v Redmine*.

Obrázok 3. Pridanie podúlohy alebo závislosti k úlohe v Redmine

4.1.2.3 Vytvorenie závislosti medzi úlohami

Vstup: identifikovanie závislosti medzi novou zaevidovanou úlohou a inou existujúcou úlohou
Výstup: vytvorená závislosť medzi úlohami
Zodpovedný: manažér plánovania

Ak pri zadaní úloh do systému Redmine je medzi úlohami identifikovaná vzájomná závislosť, manažér plánovania postupuje nasledovnými krokmi:

1. Prihlásiť sa do systému Redmine.
2. Vybrať si príslušný projekt.
3. Kliknúť na tlačidlo *Issues*.
4. V prípade potreby si vyselektovať úlohy pomocou filtra cez ich stav a potvrdiť cez *Apply*.
5. Kliknúť na úlohu, ktorá má s úlohou súvisieť.

6. Kliknúť na *Add* v časti *Related issues* (bod B na Obrázku 3).
7. Zadať typ závislosti úlohy, s ktorou úloha súvisí z možností, ktoré uvádza Tabuľka 4.

Závislosť	**Prípad použitia**	**Následky**
Related to	Vytvorenie prepojenia (linku) na súvisiacu úlohu, ďalšie závislosti medzi úlohami nie sú.	-
Duplicates	Vyriešenie súvisiacej úlohy vyrieši aj túto úlohu.	Zmena stavu súvisiacej úlohy na *closed* automaticky uzavrie aj túto úlohu.
Duplicated by	Vyriešenie tejto úlohy vyrieši aj súvisiacu úlohu (opak *Duplicates*).	Zmena stavu tejto úlohy na *closed* automaticky uzavrie aj súvisiacu úlohu.
Blocks	Úloha, ktorej vyriešenie je potrebné na vyriešenie súvisiacej úlohy.	Pokým táto úloha nemá stav *closed* nie je možné uzavrieť súvisiacu úlohu.
Blocked by	Na vyriešenie tejto úlohy je potrebné vyriešiť súvisiacu úlohu (opak *Blocks*).	Pokým súvisiaca úloha nemá stav *closed* nie je možné uzavrieť túto úlohu.

Tabuľka 4. Závislosti medzi úlohami

8. Zadať ID závislej úlohy.
9. V prípade, že úloha súvisí aj s ďalšou úlohou, opakovať postup od kroku 5.

4.2 Metodika písania jednotkových (Unit) testov

Autor: Peter Hamar

4.2.1 Úvod

Cieľom tejto metodiky je definovať jednotné spôsoby písania testov pre tím Carlos. K jednotlivým štádiám testovania táto metodika definuje zodpovedné osoby a taktiež jednotlivé výstupy pre konkrétne časti. Vývojovým nástrojom je Visual Studio a teda táto metodika bude určovať pravidlá písania unit testov v tomto vývojovom prostredí.

4.2.2 Zoznam pojmov

- Unit test – jednotkový test, slúži na overenie určitej malej časti (jednotky) funkcionality.
- Visual Studio – vývojové prostredie, používané kvôli dobrej podpore jazyka C++.

4.2.3 Zoznam skratiek

- VS – Visual Studio

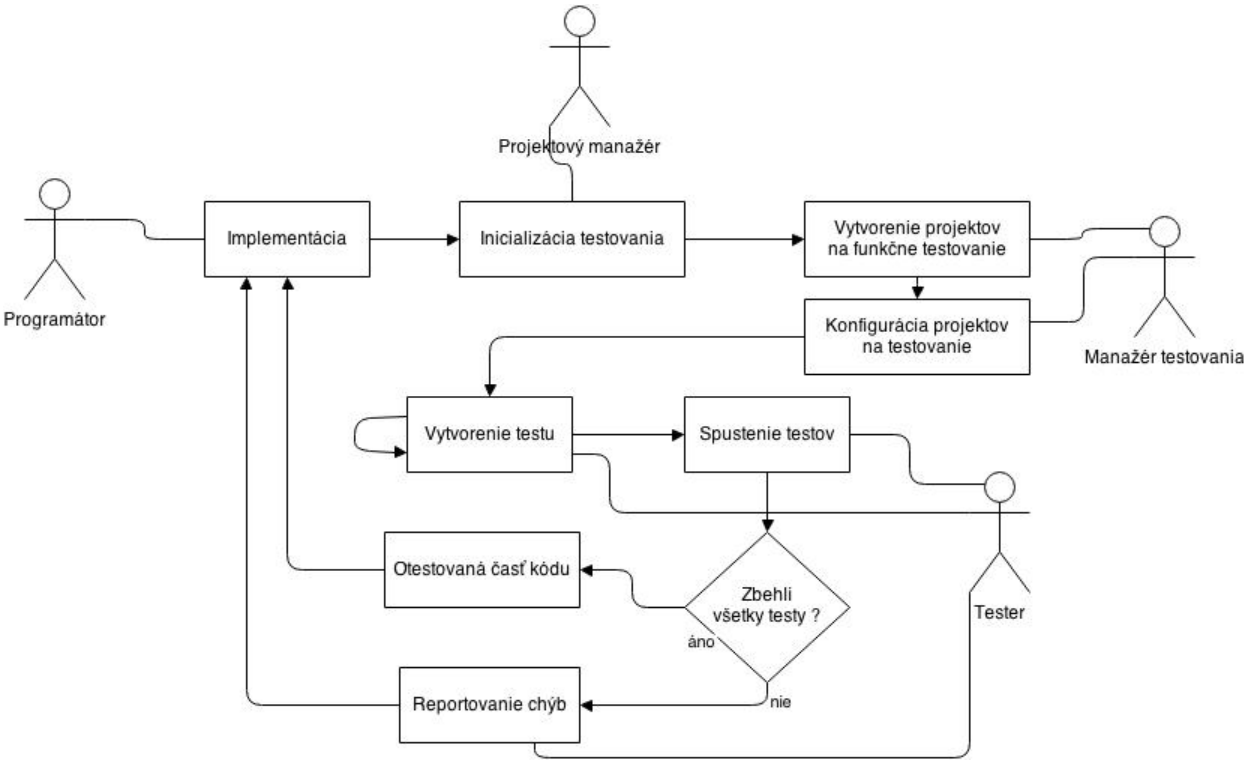
4.2.4 Zoznam nadväzujúcich metodík

- metodika riešenia problémov
- metodika kompilácie kódu

4.2.5 Zodpovednosť za procesy a momenty spustenia procesov

Proces	Zodpovednosť	Kedy
1.Inicializácia Unit testovania	Projektový manažér	Začatie fázy implementácie
2.Vytvorenie projektov na funkčné testovanie	Manažér testovania	Po inicializácii testovania
3.Konfigurácia projektov na testovanie	Manažér testovania	V momente vytvorenia projektov na testovanie
4.Vytvorenie testu	Tester	Po nastaveni projektov
5.Spustenie testu	Tester	V momente ukončenia vytvárania testu
6.Reportovanie chýb	Tester	Po spusteni testu/ov

4.2.6 Diagram



4.2.7 Podrobný opis jednotlivých procesov

1. Inicializácia unit testovania

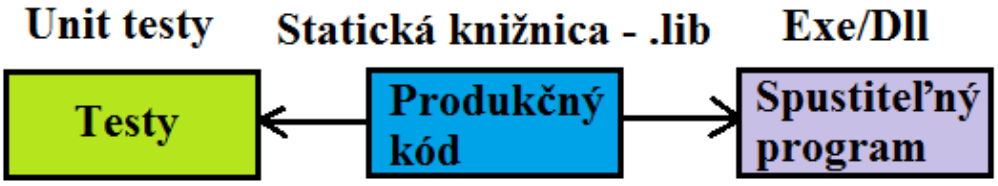
Vstup: Časť funkčného kódu.
Výstup: Overený funkčný kód.
Zodpovedný: Projektový manažér.

Popis: Cieľom tohto procesu je, aby Manažér testovania nastavil všetky potrebné detaily vo vývojovom nástroji – VS 2012. Vďaka čomu bude možné následne vytváranie a spúšťanie samotných testov.
Projektový manažér informuje Manažéra testovania, že je potrebné pripraviť projekt na písanie jednotkových testov. Tomuto procesu predchádza započatie fázy implementácie. Následne nasleduje proces vytvorenia projektov na funkčné testovanie.

2. Vytvorenie projektov na funkčné testovanie

Vstup: Vytvorený projekt vo VS
Výstup: Upravený projekt na písanie jednotkových testov.
Zodpovedný: Manažér Testovania.

Popis:
Manažér testovania spustí projekt, v ktorom sa bude vytvárať kód. Je potrebné vytvoriť 3 projekty v jednom solution. Schéma podľa ktorej je potrebné projekty vytvoriť :



- Produkčný kód – tento projekt by mal byť už vytvorený. Vytvorí sa hneď v momente začatia implementácie. Mal by byť v jazyku C++, vytvorený ako „Empty project“ a ako „Win32 Console Application“.
- Unit Testy –
    1. V otvorenom „solution“ vo VS je potrebné kliknúť pravým tlačidlom na „solution“.
    2. V roletovom menu vybrať možnosť „Add->New Project“.

3. Následne sa zobrazí okno, kde je potrebné vybrať typ projektu a to konkrétne „Visual C++->Test->Native Unit Test Project“.
4. Výber potvrďte kliknutím na tlačidlo „ok“.

- Spustiteľný program -
    1. V otvorenom „solution“ vo VS je potrebné kliknúť pravým tlačidlom na „solution“.
    2. V roletovom menu vybrať možnosť „Add->New Project“.
    3. Následne sa zobrazí okno, kde je potrebné vybrať typ projektu, a to konkrétne „Visual C++->Win32-> Win32 Console Application“.
    4. Potom sa zobrazí ďalšie okno, kde je potrebné kliknúť na tlačidlo „next“. Pokračujte zaškrtnutím políčka „Empty project“, tým sa znefunkční políčko „Precompiled Header“. Výber potvrdíme tlačidlom „finish“.

Tento proces je ukončený v momente keď v jednom „solution“ sú vytvorené všetky 3 projekty. Následne sa pristúpi k procesu konfigurácie projektov.

3. Konfigurácia projektov na testovanie

Vstup: Projekty vo VS.
Výstup: Nakonfigurované projekty, pripravené na vytváranie jednotkových testov.
Zodpovedný: Manažér Testovania.

Popis: Cieľom je konfigurácia jednotlivých projektov, pre správne písanie unit testov.

- **Produkčný kód**
    1. Vo vytvorenom solution kliknite pravým tlačidlom myši na projekt s produkčným kódom.
    2. Následne v roletovom menu zvoľte možnosť properties. Po tejto voľbe sa zobrazí okno s nastaveniami projektu.
    3. V ľavom hornom rohu nastavte voľbu „Configuration“ na „All Configurations“.
    4. V nastavovaní pokračujte vybratím možnosti „Configuration Properties“ a následne vyberte možnosť “General“.
    5. V pravej časti okna, voľbu „Configuration Type“ nastavte na „Static library (.lib)“. Výber potvrďte stlačením tlačidla „Apply“ a ešte raz to celé potvrďte
       kliknutím na tlačidlo „ok“. Zobrazené okno sa zavrie a uloží vaše nastavenia

- **Unit Testy a Exe/DLL** – Postup pre konfiguráciu oboch projektov je skoro identický, teda je ho potrebné vykonať pre oba projekty.

    1. Pravým tlačidlom myši kliknite na prvý z projektov a z roletového menu vyberte možnosť „Properties“.
    2. Možnosť „Configuration“ nastavte na „All Configurations“.
    3. Pokračujte zvolením možnosti „Common Properties“, ktoré je dostupné v okne pod predchádzajúcou voľbou.
    4. V zobrazenom „submenu“ zvoľte možnosť „Framework and References“.
    5. Kliknite na tlačidlo „Add New Reference“ a v zobrazenom okne vyberte statickú knižnicu – Produkčný kód, voľbu potvrdte tlačidlom „ok“.
    6. Následne zvoľte možnosť C/C++ v zobrazenom okne.
    7. V zobrazenom pravom panely zvoľte možnosť „Additional Include Directories“ a následne vyberte voľbu „edit“.
    8. Pridajte nový riadok a to „$(SolutionDir)\ProductLibrary;%(AdditionalIncludeDirectories)“. S tým, že „ProductLibrary“ prepíšte na názov projektu s produkčným kódom. Voľbu potvrďte kliknutím na tlačidlo „ok“.

- **Iba pre Unit Test project:**
    1. Je potrebné kliknúť voľbu „Configuration Properties“.
    2. Zvoliť „General“ a zmeniť hodnotu poľa „Common Language Runtime Support“ na hodnotu „/clr“.
    3. Všetko potvrdiť tlačidlom „apply“a ok.
- **Iba pre Exe/DLL:**
    1. Je potrebné kliknúť na hlavnú voľbu „Configuration Properties“.
    2. V zobrazenom pravom okne zmeňte pole „Configuration Type“ na hodnotu „Application (.exe)“ resp. „Dynamic library (.dll)“.
    3. Všetko podtvrdte tlačidlom „apply“ a „ok“.

Týmto je konfigurácia projektov ukončená a sú pripravené na písanie jednotkových testov. Po tomto procese sa pristúpi k písaniu unit testov.

4. Vytvorenie unit testu

Vstup: Vytvorené solution s potrebnými projektami pripravenými na písanie jednotkového testu.
Výstup: Projekt s napísaným jednotkovým testom.

Zodpovedný: Tester.

Popis:

1. Vo vytvorenom projekte na písanie unit testov je potrebné vytvoriť súbor s príponou .cpp a to tak, že pravým tlačidlom myši kliknite na Source Files v unit test projekte a z roletového menu vyberte „Add->New Item“.
2. Následne sa zobrazí okno a v ňom zvoľte Visual C++/Test/C++ Unit Test Class.
3. Vhodne pomenujte .cpp súbor, konkrétne podľa metodiky programovacích štandardov.
4. Keď máme .cpp súbor vytvorený, je v ňom vložený vygenerovaný kód s 1 metódou. Naplňte telo tejto metódy a vytvorte tak samotný jednotkový test. Test sa vytvára v jazyku C++, teda pre písanie testov je potrebná znalosť tohto jazyka. Tento proces môže prebiehať viackrát za sebou resp. súčasne. Z toho vyplýva, že sa vytvorí viacero testov naraz. Po napísaní samotného testu/ov sa pristúpi k ich spusteniu.

5. Spustenie unit testov

Vstup: Solution s napísaným testom/i.
Výstup: Otestované jednotlivé časti kódu.
Zodpovedný: Tester.

Popis: Cieľom tohto procesu je spustenie jednotkových testov vo Visual Studiu.

1. V zapnutom vývojovom prostredí, veberte z horného menu možnost „Test“. Následne sa zobrazí menu.
2. V ňom zvoľte možnosť „Windows->Test Explorer“. V pravej časti obrazovky sa zobrazí panel, na ktorom je vidieť zoznam testov.
3. Testy spustíte kliknutím na tlačidlo „Run All“, ktoré sa nachádza na ľavej časti tohto panelu. Potom sa testy vykonajú. Všetky testy ktoré prebehli úspešne sú označené zelenou farbou. Testy ktoré nezbehli, naopak červenou. Samozrejme tento proces sa môže viac krát opakovať.

V prípade, že nechcete spustiť všetky testy máte na výber z viacerých možností. V zobrazenom panely sa nachádza tlačidlo „Run...“. Kliknutím na neho sa zobrazí ďalšie menu, v ktorom je možné spustenie iba nezbehnutých testov, nových testov, úspešne zbehnutých testov a naposledy spustených testov.

Ak nastane situácia, že všetky testy prebehli správne, tak pre tento moment fáza testovania na určitý okamih končí. Avšak iba do okamihu, kedy bude potrebné otestovať

ďalšie funkčné jednotky kódu. Kde následne môže vzniknúť problém aj s predtým funkčným testom.

V prípade, že existujú nejaké testy, ktoré nezbehli správne je potrebné pristúpiť k procesu reportovania chýb.

6. Reportovanie chýb

Vstup: Nesprávne fungovanie určitej jednoty kódu – nezbehnutý test.
Výstup: Správa pre programátora.
Zodpovedný: Tester.

Popis: Tento proces nastáva v prípade, že existujú testy, ktoré sa nevykonali správne. Cieľom tohto procesu je odstránenie chýb v kóde. Tester o každom nevykonanom teste vytvorí správu. Vytvorenú správu následne zašle programátorovi.

Správa pre programátora obsahuje:

- Meno odosielateľa
- Nefunkčný test
- Report o chybe
- Dátum do kedy je potrebná oprava

4.3 Metodika - Dokumentácia zdrojových kódov za použitia Doxygen (Manažment zdrojových kódov)

Autor: Lukáš Sekerák

4.3.1 Úvod

Cieľom tejto metodiky je definovať postup, ako správne vygenerovať dokumentáciu z anotácie a v krátkosti popísať danú anotáciu. Metodika je určená pre tímový projekt. Tento projekt sa nazýva „Carlos“ a je napísaný v jazyku C++. Z tohto dôvodu sme pre generovanie zvolili štandardný nástroj doxygen, ktorý je značne populárny pre tento jazyk. Vďaka „doxygenu“ je možné dokumentáciu generovať automatizovane. Metodika je určená pre tvorcov zdrojových kódov, najmä programátorov. Taktiež pre manažéra dokumentácie a manažéra kvality. Manažéri majú za úlohu dohliadať na dodržiavanie tejto metodiky.

4.3.2. Zoznam pojmov

- ➔ **Doxygen** – Štandardný nástroj pre generovanie dokumentácie, vo viacerích programovacích jazykoch. Podporuje C++, C#, PHP, Java, Python. Dokumentáciu dokáže vygenerovať vo formáte PDF, HTML ale aj v iných.
- ➔ **C++** - Programovací jazyk.
- ➔ **Carlos** – Názov nášho tímu, pre ktorý sa vyvíja metodika.
- ➔ **Doxygen wizard** - Pomocný nástroj, ktorý je súčasťou doxygen inštalačného balíka. Nastroj umožňuje použiť doxygen cez jednoduché používateľské prostredie bez znalosti špeciálnych príkazov.

4.3.3. Nadväzujúce metodiky

- ➔ - Metodika prehliadka a kontrola zdrojového kódu
- ➔ - Metodika zdrojových kódov (všeobecne)

4.3.4. Role

Zoznam rolí, pre ktoré je vhodná táto metodika.

Rola	Úloha
Programátor	Má za úlohu kvalitne anotovať triedy, funkcie, všeobecne povedané zdrojový kód. Pri čom má využívať normy a značky používané doxygen nástrojom a zároveň má dodržiavať pravidlá tejto metodiky.
Manažér dokumentácie	Jeho hlavnou úlohou je vytvorenie, teda vygenerovanie dokumentácie. Vygenerovanie ma byť realizované za pomoci doxygen nástroja.
Manažér kvality	Jeho úlohou, je sledovať prácu manažéra dokumentácie a programátorov. Sledovať či si svoje úlohy plnia správne.

4.3.5. Procesy

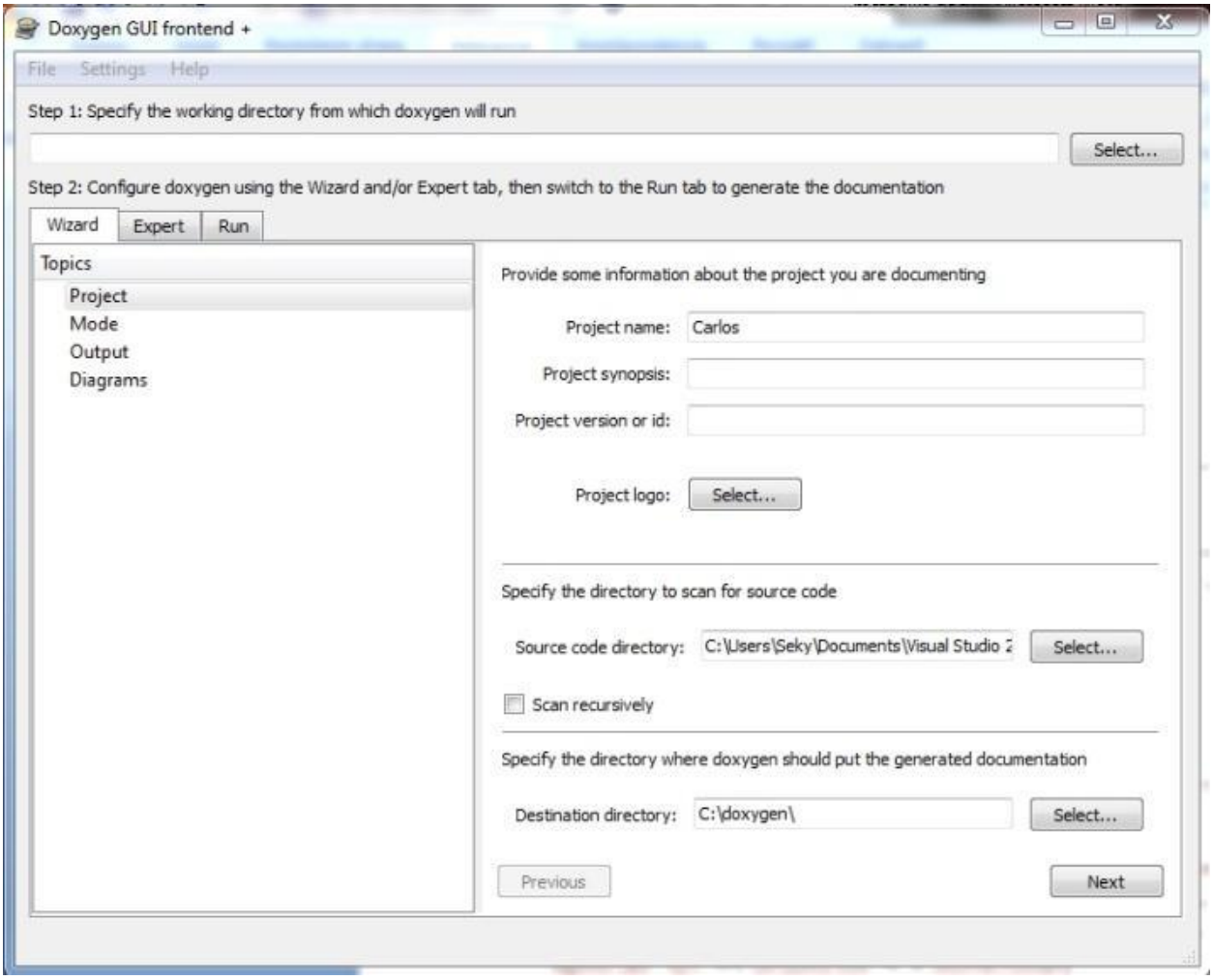
4.3.5.1. Generovanie dokumentácie

Vstup: Kvalitne zdokumentovaný zdrojový kód
Výstup: Vygenerovaná dokumentácia (napríklad v HTML formáte)
Role v procese: Manažér dokumentácie
Opis nástroja: V tomto procese je hlavnou postavou manažér dokumentácie, ktorého úlohou je použiť doxygen wizard, nástroj na generovanie dokumentácie. Program sa skladá z 3 častí:

- **Horná časť** – obsahuje cestu k priečinku, kde je doxygen nainštalovaný
- **Ľavá časť okno so zo zoznamom** – obsahuje zoznam kategórií nastavení
- **Pravá časť okna** – sústava nastavení pre kategóriu

Proces:

1. Spustite doxygen wizard, ktorý nájdete v inštalačnom balíku.
2. Zobrazí sa okno aplikácie, kde je defaultne nastavená kategória „Project“.



Obr. 1 – Kategória nastavenia ,,Project“

2.1. V hornej časti obrazovky nájdite sekciu „Step 1 Specify the working directory from which doxygen will run“. V tejto sekcii kliknite na tlačidlo „Select“. Následne sa zobrazí okno, v ktorom je potrebné nájsť súbor „Doxygen.exe“. Tento súbor sa náchádza v priečinku, do ktorého ste predtým nainštalovali „doxygen“.

2.2. V pravej časti obrazovky nastavte pole s názvom „Project name“ na hodnotu „Carlos“.

2.3. V pravej časti obrazovky nastavte pole s názvom „Project version“ na aktuálnu verziu projektu.

2.4. V pravej časti obrazovky kliknite na „Select...“ tlačidlo, ktoré je v riadku s Project logo. Následne v okne nájdite logo (súbor) pre projekt „Carlos“.

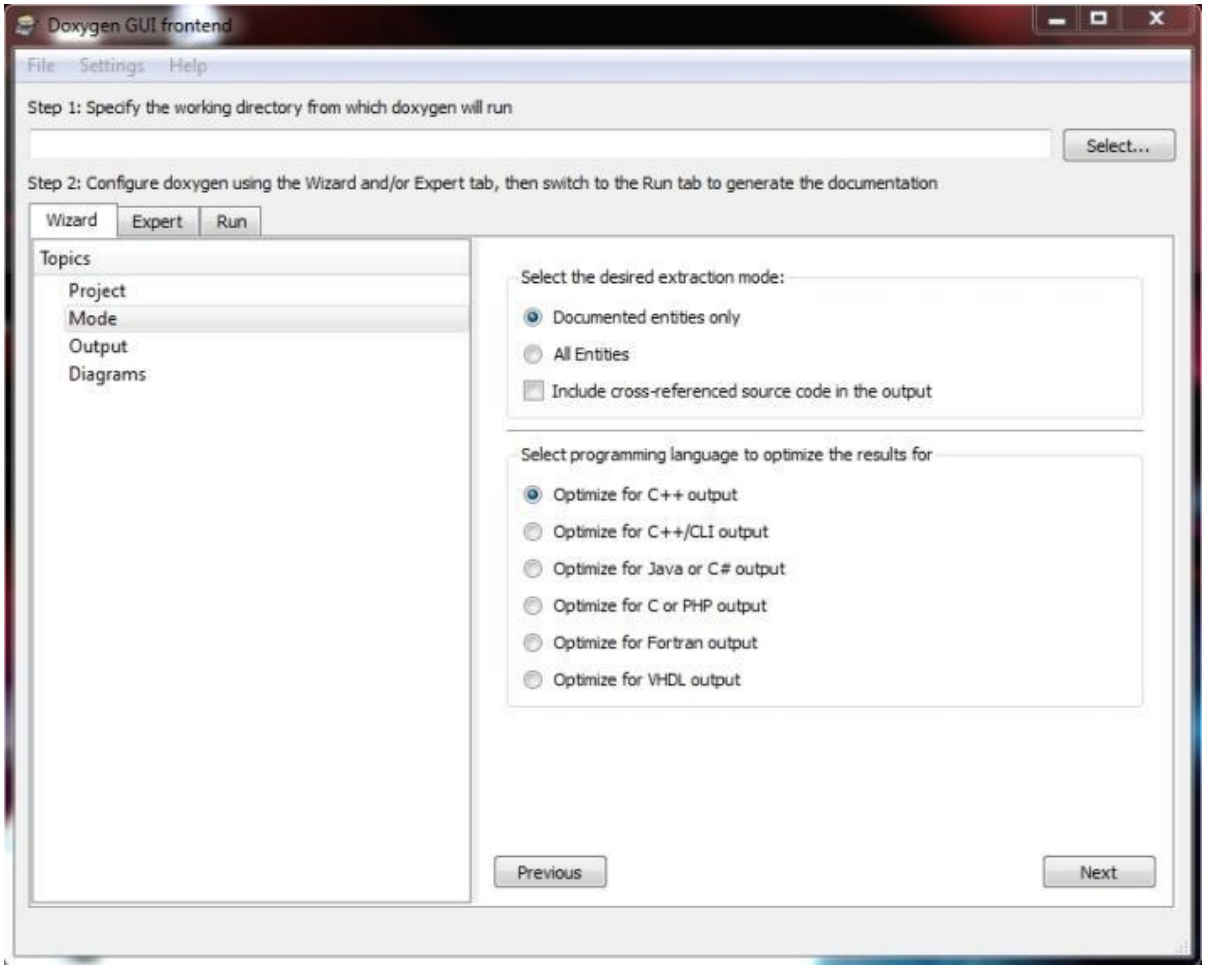
2.5. Pokračujte nájdením položky „Source code directory“, kde kliknite v rovnakom riadku na „Select ...“. Po tejto akcii sa zobrazí okno, v ktorom treba nájsť priečinok ku projektu, pre ktorý, má byť vygenerovaná dokumentácia. Následne potvrďte výber tlačidlom „ok“.

2.6. Pole „Scan recursively“ zaškrtnite, nachádza sa v strede pravej časti okna.

2.7. Následne kliknite na tlačidlo „Select...“, ktoré sa nachádza pri poli „Destination directory“. Opäť sa zobrazí okno, v ktorom je možné prechádzať priečinky v počítači. Zvoľte priečinok, do ktorého sa má výsledná dokumentácia uložiť. Svoj výber opäť potvrďte tlačidlom „ok“.

2.8. Všetky ostatné nastavenia v tomto okne sú voliteľné, pre potreby tímu postačuje ak tieto hodnoty ostanú nezmenené.

3. Kliknite na tlačidlo „Next“ v pravom dolnom rohu okna. Takto sa presuniete na nastavenie pre kategóriu „Mode“.

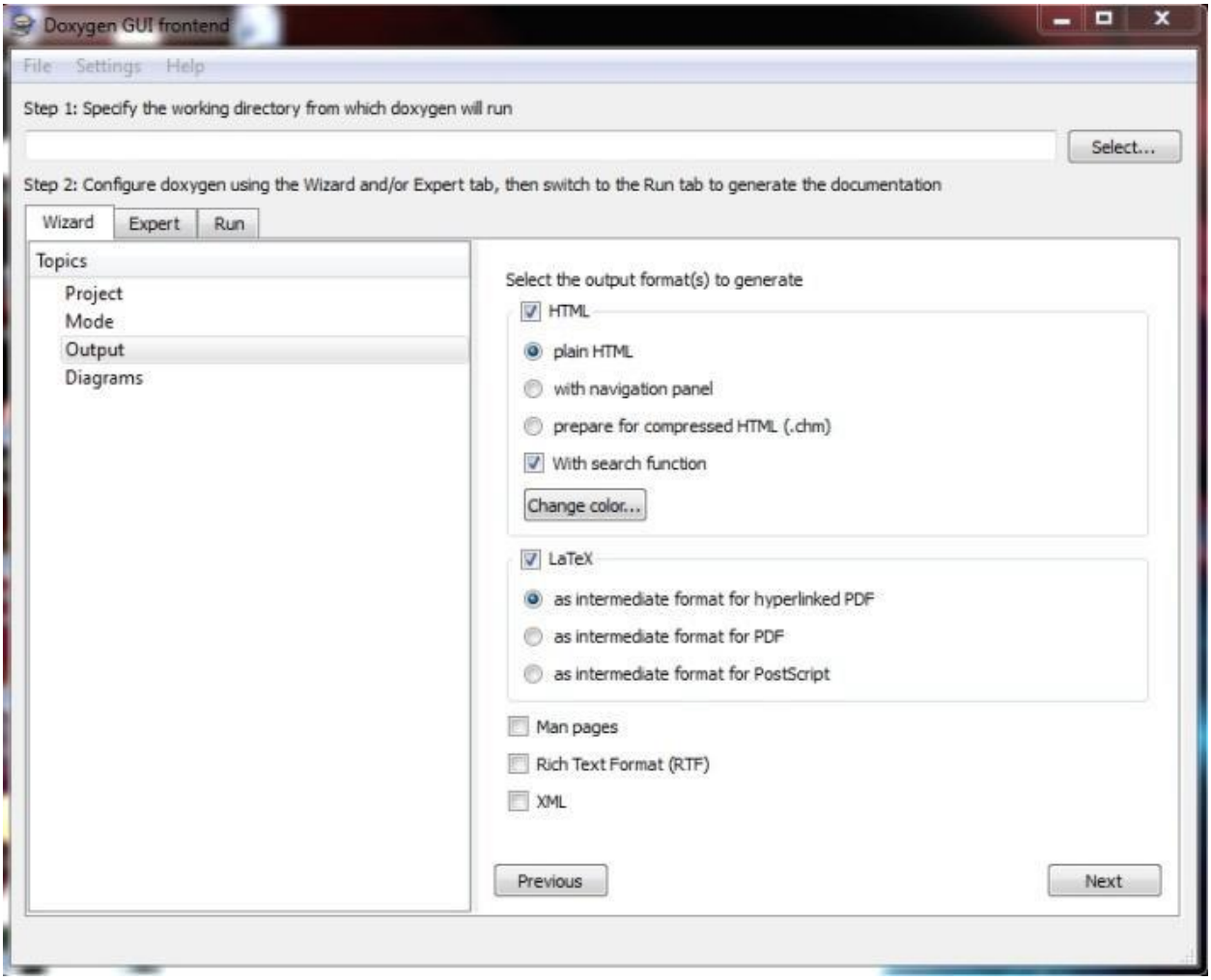


Obr. 2 – Kategória nastavenia „Mode“

3.1. V tomto bode je možné vybrať ku ktorým častiam kódu bude vygenerovaná dokumentácia a teda je tu možnosť výberu. Je možné vygenerovať dokumentáciu ku všetkým entitám v kóde, keď zvolíte „All Entities“ v pravej časti okna. Druhou možnosťou je vygenerovať dokumentáciu len k entitám, ktoré obsahujú anotáciu a to voľbou „Documented entities only“. Odporúčaná možnosť je „Documented entities only“.

3.2. V ďalšej časti vyberte možnosť „Optimus for C++ output“. V prípade iného projektu je potrebné túto možnosť zmeniť na tú, v ktorom programovacom jazyku je projekt napísaný.

4. Kliknite na tlačidlo „Next“ v pravom dolnom rohu aplikácie. Čím sa presuniete na nastavenie kategórie „Output“.

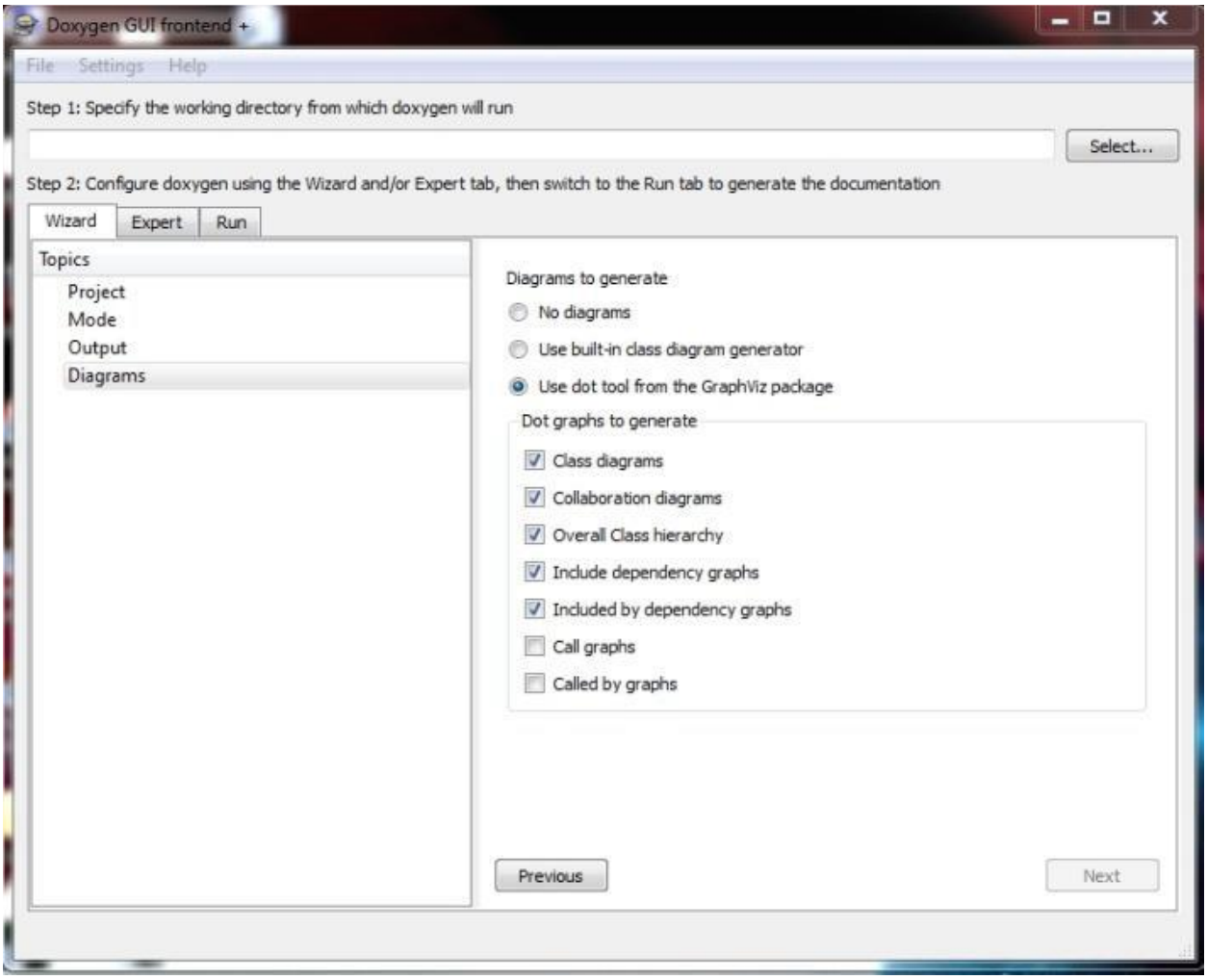


Obr. 3 – Kategória nastavenia Output

4.1. V nastavení kategórie „Output“ je možné nastaviť výstupný formát dokumentácie. Ako najvhodnejší formát pre potreby tímového projektu je formát html. Z toho dôvodu je potrebné zaškrtnúť práve toto políčko.

4.2. Všetky ďalšie nastavenia v tomto okne sú dobrovoľné. Môžete súčasne vygenerovať aj dokumentáciu v iných formátoch napríklad „Latex“. Odporúča sa však nemeniť ostatné hodnoty.

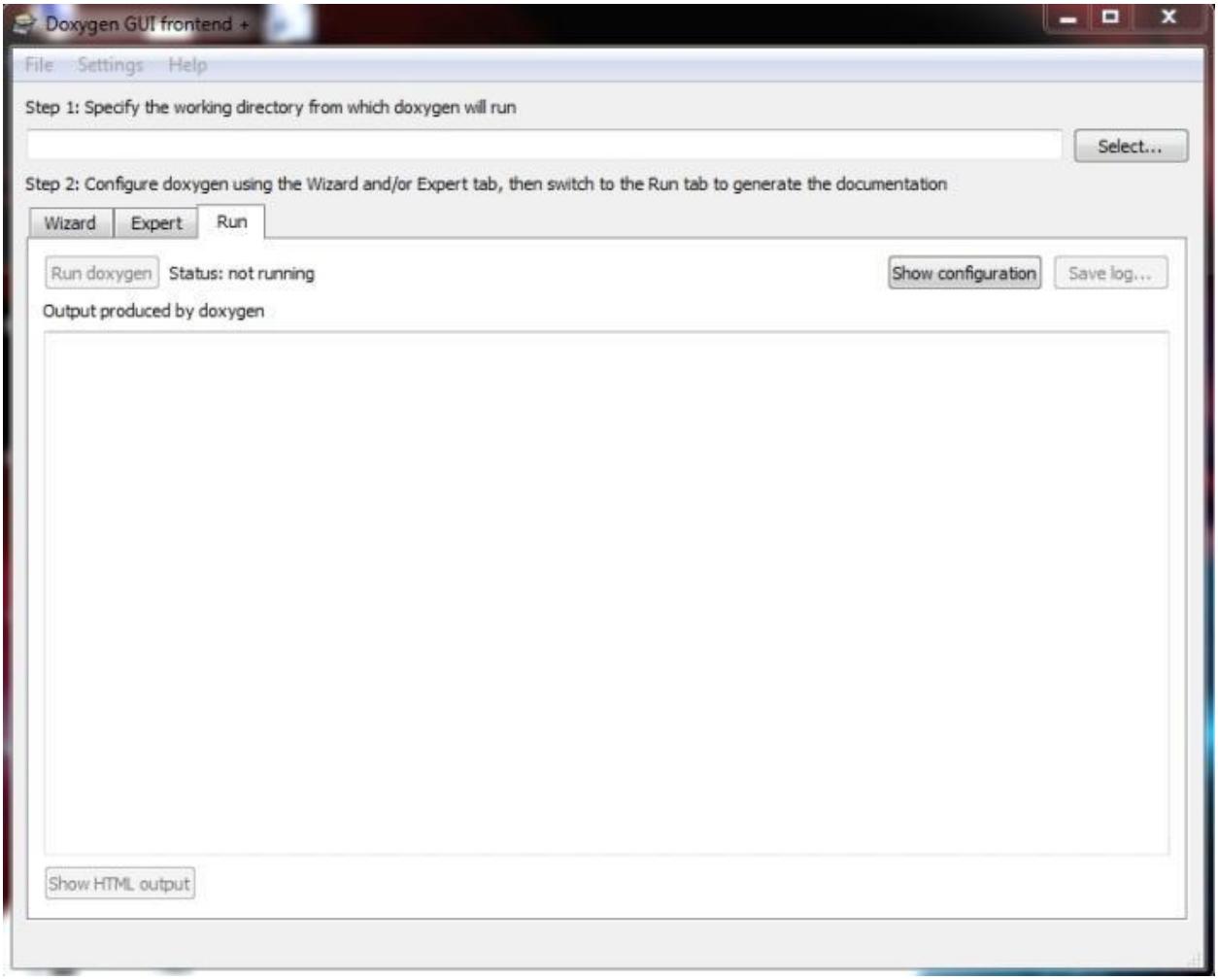
5. Kliknite na tlačidlo „next“ v pravom dolnom rohu. Tým sa presuniete na obrazovku nastavení pre kategóriu „Diagrams“.



Obr. 4 – Kategória nastavenia „Diagrams“

Zaškrtnite voľbu „No diagrams“. V tejto časti sa nachádzajú nastavenia pre generovanie diagramov zo zdrojového kódu. Na generovanie diagramov je potrebný nástroj ,, DOT“. Tento nástroj však nie je súčasťou inštalačného balíka. Generovanie diagramov zo zdrojového kódu, je súčasťou metodiky tvorby diagramov zo zdrojového kódu.

6. V hornej časti obrazovky kliknite na tab „Run! Tým sa zobrazí posledná časť. V tejto časti kliknite na „Run doxygen“. Čím sa spustí proces generovania.



Obr. 3 – Tab „Run“.

7. Následne počkajte, kým doxygen vygeneruje dokumentáciu. Skontrolujte v strede okna záznam, či nenastali chyby pri generovaní. V prípade objavenia konfiguračnej chyby, je vašou úlohou zopakovať tento proces. Respektíve opraviť konfiguračnú chybu.

8. Proces končí, dokumentácia je vygenerovaná.

4.4 Metodika zdielania zdrojov v Google Drive

Autor: Martin Petluš

4.4.1 Úvod

Tento dokument popisuje metodiku na dolnej úrovni pri procese zdielania zdrojov (zložiek, súborov) v službe Google Drive. Dokument je vhodný pre projekty a tímy, ktoré vyvíjajú softvérový projekt v malom rozsahu a s počtom ľudí v tíme v rozsahu od 8 do 12.

4.4.1.1 Dedikácia metodiky

Metodikou opisovanom v tomto dokumente sa riadi každý člen tímu.

4.4.1.2 Príbuzné metodiky

- Evidencia úloh v Redmine

4.4.1.3 Role a zodpovednosti

V tabuľke 1 sú vymenované role a ich zodpovednosti, ktoré sa objavujú v tejto popísanej metodike.

Rola	**Zodpovednosť**
Člen tímu	● Študent ● Vytvára zdielanú zložku ● Vytvára jednotlivé zápisnice v Google Drive ● Vytvoriť zápisnicu v stanovenom časovom limite ● Je zodpovený za to, aby zápisnica obsahovala všetky potrebné údaje a bola v dohodnutej klavite

Tabuľka 1: Role a zodpovednosti

4.4.1.4 Slovník pojmov

- **Google, Inc.** – je americká spoločnosť zameraná hlavne na internetový trh založená v roku 1998. Spoločnosť založili Larry Page a Sergej Brin. Jej hlavným produktom je internetový vyhľadávač Google.

- **Google Drive** – je služba na ukladanie a synchronizáciu súborov poskytovaná spoločnosťou Google. Táto služba umožňuje ukladanie súborov do cloud-u, zdielanie a kolaboratívne editovanie súborov.
- **Zdielaná zložka** – je taká zložka, ktorá je zdielaná medzi viacerými používateľmi služby Google Drive. Títo používatelia majú rôzne nastavené práva na tejto zložke, ktoré ich obmedzujú v tom čo môžu zo zložkou robiť.
- **Zdielaný súbor** – je taký súbor, ktorý je zdielaný medzi viacerými používateľmi sližby Google Drive a títo používateľa majú rôzne práva pri editovaní tohto súboru.

4.4.2 Zdielanie zdrojov v Google Drive

Táto kapitola sa zaoberá jednotlivými procesmi zdielania zdrojov v službe Google Drive medzi členmi tímu.

4.4.2.1 Vytvorenie zápisnice

Na každom stretnutí tímu k tímovému projektu sa robia poznámky, do ktorých sa zapisuje, ktoré úlohy kto vyriešil, preberajú sa úlohy, ktoré je treba vyriešiť. Tiež sa do poznámok zapisujú úlohy, ktoré sa prideľujú jednotlivým členom tímu. Preberajú sa taktiež rôzne problémy s tímovým projektom, ktoré sa vyskytli, zapisuje sa, kto bol prítomný na stretnutí k tímovému projektu. Tieto poznámky, je treba spracovať do dokumentu, ktorý obsahuje, všetky potrebné náležitosti a je na potrebnej odbornej úrovni. Tento dokument sa volá *Zápisnica* a po jeho vytvorení musia mať k tomuto dokumentu prístup všetci členovia, ak by bolo potrebné spraviť nejaké úpravy samostatne každým členom (spresnenie úlohy, ...).

Proces vytvorenia *Zápisnice* v službe Google Drive (ktorú využívame na zdielanie zápisnice medzi členmi tímu) sa skladá z viacerých podúloh, ktoré nasledujú pri vytvorení prvej zápisnice sekvenčne za sebou. Ak už existuje zdielaná zložka v Google Drive medzi členmi tímu, krok vytvorenia tejto zložky môžeme preskočiť.

4.4.2.1.1 Vytvorenie zdielanej zložky

Vstup: Emailové adresy členov tímu na Google účet
Výstup: Vytvorená zdielaná zložka v Google Drive medzi členmi tímu
Zodpovedný: Člen tímu

Kroky:

1. Prihlásenie sa do služby Google Drive
2. V ľavej časti menu si vyberieme možnosť *My Drive*
3. V ľavej časti obrazovky klikneme na tlačidlo *CREATE*

4. Z ponúknutých možností si vyberieme *Folder*
5. V popupe zadáme zvolené meno zložky (napríklad názov *Zápisnice*)
6. Klikneme na tlačídko *Create*
7. V zozname vecí v našom Google Drive sa nám objaví nová zložka, ktorú sme práve vytvorili
8. Túto novovytvorenú zložku si vyznačíme
9. V hornej časti nad zoznamom vecí v Google Drive vyberieme možnosť *More*
10. V ponúknutom menu vyberieme možnosť *Share* a opäť vyberieme možnosť *Share* na ktorú klikneme
11. V popupe máme možnosť nastaviť viditeľnosť zložky a to tým, že klikneme na tlačidlo *Change*
12. Ponúknuté možnosti zahŕňajú:

Možnosť	**Práva**
Public on the web	Ktokoľvek na Internete bude môcť vidieť túto zložku. Zložka sa môže objaviť vo výsledkoch vyhľadávačov.
Anyone with the link	Ktokoľvek s linkom na túto zložku bude vidieť túto zložku.
Private	Sme jedinou osobou, ktorá má prístup k tomuto dokumentu. Môžeme pridať ďalšie osoby, ktoré budú mať prístup k tejto zložke.

13. Z ponúkaných možností zvolíme prístup *Private*
14. Klikneme na tlačidlo *Save*
15. Nachádzame sa opá v predchádzajúcom popupe
16. Tu v časti *Invite people* máme možnosť pridať používateľov na zdielanie tejto zložky
17. Do textového pola zadáme emailové adresy členov tímu
18. Pridaným členom tímu máme možnosť nastaviť dva typy prístupu k zložke:

Možnosť	**Práva**
Can edit	Používatelia môžu do zložky pridávať veci, mazať veci (dokumenty), pridávať ďalších používateľov (ak je im to povolené), vidieť ostatných používateľov tejto zložky, alebo editovať existujúce dokumenty v zložke.

Can view	Používatelia môžu vidieť súbory, zložky, stiahnuť súbory.

19. Vyberieme možnosť *Can edit*, aby ostaní členovia mali možnosť do zložky pridávať svoje vlastné zápisnice, ktorí budú vytvárať
20. Nakoniec klikneme na tlačidlo *Share & save*
21. Posledným krokom, sme vytvorili zdielanú zložku, do ktorej budú môcť ostatní členovia tímu pridávať nové zápisnice

4.4.2.1.2 Vytvorenie zápisnice v zdielanej zložke

Vstup: Poznámky zo stretnutia tímu
Výstup: Vytvorená zápisnica v zdielanej zložke zo stretnutia tímu
Zodpovedný: Člen tímu

Kroky:

1. Prihlásenie sa do služby Google Drive
2. V ľavej časti obrazovky klikneme na tlačidlo *Shared with me*
3. V zozname vecí, ktoré sú so mnou zdielané sa prepneme do zdielanej zložky, v ktorej sa nachádzajú zápisnice
4. Klikneme na tlačidlo *CREATE* v ľavej časti obrazovky
5. Z ponúknutých možností vyberieme možnosť *Document*, na ktorú klikneme
6. V popupe klikneme na tlačidlo *Create & Share*
7. V novom tabe prehliadača sa nám objaví nový dokument
8. V ľavej hornej časti dokumentu klikneme na možnosť *File*
9. Z ponúknutých možností vyberieme možnosť *Rename…*
10. Do textového pola zadáme názov zápisnice v nasledujúcom tvare: *Zapisnica **XX**-**Y***:

Premenná	**Hodnota**
XX	● Naberie hodnotu ZS, ak sa nachádzame v zimnom semestri ● Naberie hodnotu LS, ak sa nachádzame v letnom semestri
Y	● Ak sa jedná o prvú zápisnicu, tak naberá hodnotu 1

	• Inak naberá hodnotu, o jedna väčšiu ako je najvyššia hodnota, zo všetkých zápisníc

11. Klikneme na tlačidlo *OK* a týmto novovytvorenú zápisnicu premenujeme
12. Každá zápisnica musí dodržiavať tie isté formátovacie pravidlá (vyzerať tak isto), ako predchádzajúca zápisnica. Pri vytváraní prvej zápisnice, môžeme formát zvoliť ľubovoľne.
13. Každá zápisnica, musí obsahovať hlavičku (prípadne úvodnú stranu), ktorá musí obsahovať tieto údaje:
    - **názov zápisnice** – vo formáte *Zápis Y. stretnutia tímu č. 3*, kde **Y** naberá takú hodnotu, ako je uvedené v tabuľke vyššie
    - **autor zápisu** – kto je autorom tejto zápisnice
    - **dátum** – dátum, kedy sa konalo stretnutie tímového projektu
    - **miestnosť** – miestnosť na škole, v ktorej sa konalo stretnutie tímového projektu
    - **kto je prítomný** – postupne vymenované, ktorý členovia tímu boli na danom stretnutí tímového projektu prítomní na tomto stretnutí
14. Ďalej telo zápisnice musí obsahovať nasledujúce údaje:
    - **priebeh stretnutia** – v krokoch popísané, ktoré sú v časovej postupnosti, popísané čo sa na stretnutí tímového projektu riešilo
    - **tabuľka s pridelenými úlohami –** tabuľka, ktorá obsahuje, id úlohy, meno člena a aká úloha na vypracovanie mu bola pridelená
    - **tabuľku s minulými úlohami –** v akom stave sa nachádzajú úlohy z zadané na predchádzajúcom stretnutí

Po vytvorení zápisnice v zdielanej zložke, a po jej vyplnení obsahom, je teda pripravená na prípadné úpravy druhými členmi tímu. Táto vytvorená zápisnica sa umiestňuje aj na stránku tímu a z tejto zdielanej zložky, ju môže hociktorý člen tímu zobrať a pridať na stránku

4.5 Metodika - Manažment požiadaviek na zmenu

Autor: Róbert Sabol

4.5.1 Úvod

4.5.1.1 Vymedzenie obsahu

Účelom tejto metodiky je definovať a opísať procesy pri zadanej zmene. Tými procesmi je prijatie požiadavky na zmenu, analýza danej požiadavky, implementácia uvedených zmien a následné testovanie.

V metodike sú vymenované a opísané osoby, ktoré sa touto metodikou riadia. V opise sú uvedené ich roly a zodpovednosti v procese zmeny, ktorá vyplýva z požiadavky, ako aj opis súvisiacich procesov, ich vstupov a výstupov.

4.5.1.2 Dedikácia metodiky

Metodika je určená pre osoby, ktoré sa s procesom manažmentu zmien priamo stretávajú. Sú to osoby, ktoré sa zúčastňujú procesu vytvárania požiadavky na zmenu, jej následneho spracovania, zanalyzovania, implementácie, ktorá je spojená s testovaním. Zaverečnou fázou tohto procesu je odovzdanie zmenenej časti systému. Osoby, ktoré sa tohto procesu zúčastňujú sú uvedené v tabuľke č. 1, v ktorej je aj opis podprocesov, ktorých sa zúčastňuju a tiež činností, z ktorých sa tieto procesy skladajú.

4.5.1.3 Vymedzenie pojmov

- *požiadavka na zmenu* - informácia o žiadosti / požiadavke na funkčnosť, ktorá do momentu uverejnenia požiadavky nebola dohodnutá
- *preberací protokol* - protokol o prebrati zmeny zákaznikom
- *issue* - problém
- *bug* - chyba
- support - označenie požiadavky ako žiadosť o podporu
- *task* - úloha
- *change request* - označenie úlohy ako požiadavky na zmenu
- *unit test* - jednotkový test, slúži na overenie určitej malej časti funkcionality.
- *release* - označenie verzie systému, ktorá musí byť dodaná v stanovený dátum

4.5.1.4 Roly a ich zodpovednosti

Rola	Proces	Činnosť
Zákazník	prijatie požiadavky	- vytvorenie novej požiadavky - kategorizácia požiadavky - pridanie subjektu požiadavky - pridanie opisu požiadavky - pridanie priority požiadavky
	preberacie protokoly o zmene	- podpísanie preberacích protokolov
Projektový manažér	akceptačné testovanie zmeny	- odhad ceny zmeny - vykonanie akceptačných testov - vytvorenie správy o akceptačnom testovaní
	odovzdanie zmeny	- odovzdanie zmeny zákazníkovi
	preberacie protokoly o zmene	- vytvorenie preberacieho protokolu - vyplnenie preberacieho protokolu
Projektový analytik	analýza požiadavky na zmenu	- analýza rozsahu zmeny - odhad času zmeny - analýza zmenených modulov
Programátor	implementácia zmeny	- implementácia zmeny - testovanie počas implementácie
	odovzdanie zmeny	- odovzdanie funkčnej zmeny
Tester	testovanie zmeny	- testovanie zmeny - unit testy

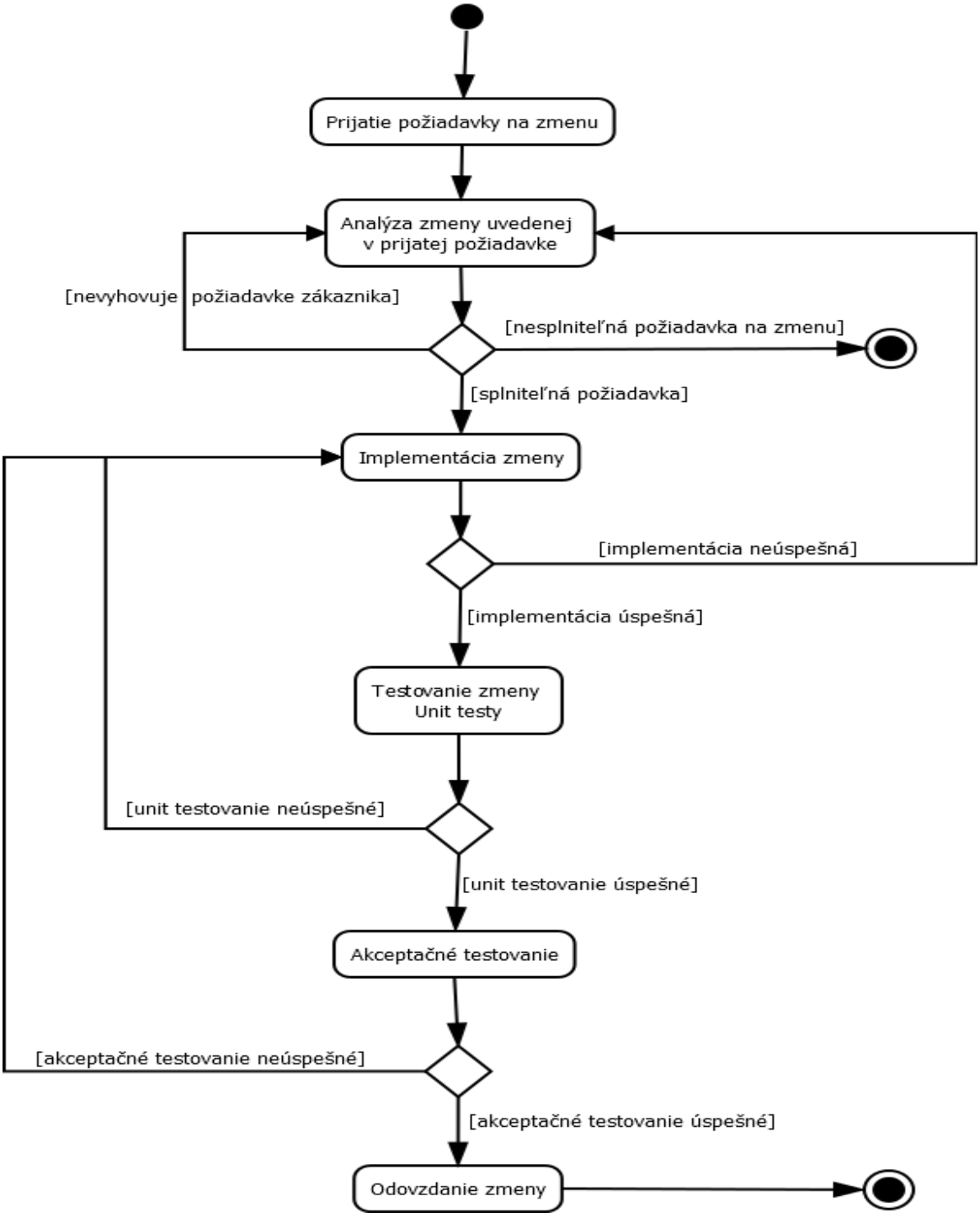
Tabuľka č. 1

4.5.1.5 Nadväzujúce metodiky

- metodika tvorby analýzy
- metodika unit testovania
- metodika riadenia úloh
- metodika plánovania

4.5.2 Opis procesov

Na nasledujúcom obrázku obr. 1 je zobrazený diagram činností, ktoré sa vykonávajú počas pocesu zmeny v projekte od prijatia požiadavky na zmenu až po odovzdanie zmeny.



Obrázok č. 1

4.5.2.1 Prijatie požiadavky na zmenu

Vstup: Požiadavka na zmenu
Výstup: Záznam o požiadavke v systéme Redmine
Zodpovedná osoba: Zákazník

Popis: Hlavným ťažiskom tejto činnosti je získať požiadavku na zmenu od zákazníka. Vstupom ako je uvedené v štruktúrovanom prehľade je samotná požiadavka na zmenu od zákazníka. Výstupom je zaevidovaný záznam o žiadosti v systéme Redmine. Vytvorený záznam musí mať vyplnené všetky potrebné polia, ktoré sú uvedené v tabuľke č.2, v ktorej je v krokoch opísaný proces prijímania požiadavky. Po tomto procese je požiadavka úspešne zaevidovaná.

Krok	Opis kroku
1	Vytvorenie novej požiadavky
2	Pridanie kategórie pre požiadavku
3	Pridanie subjektu pre požiadavku
4	Pridanie opisu požiadavky
5	Pridanie priority požiadavky

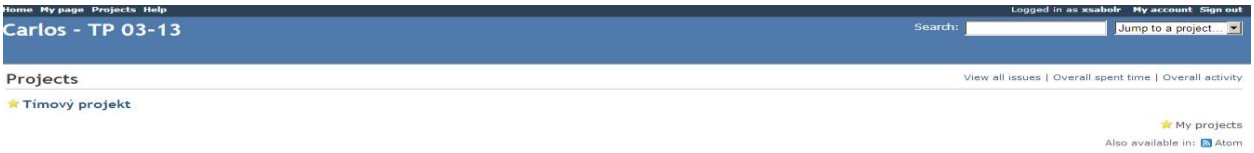
Tabuľka č. 2

4.5.2.1.1 Vytvorenie novej požiadavky

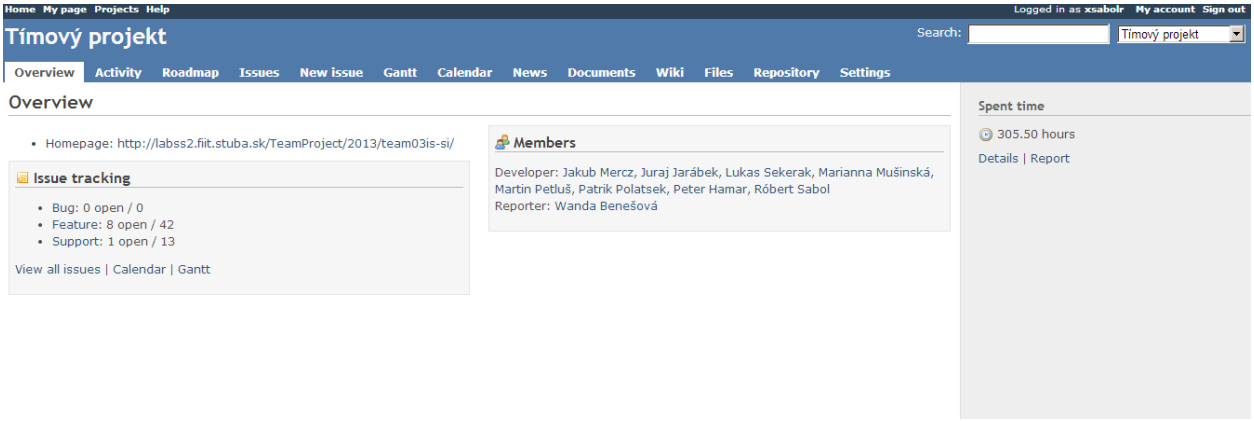
Vytvorenie požiadavky sa vykonáva v systéme Redmine. Dôležitou podmienkou v tomto kroku je to, že zákazník musí byť v systéme registrovaný.

Priebeh činností zákazníka v systéme Redmine:

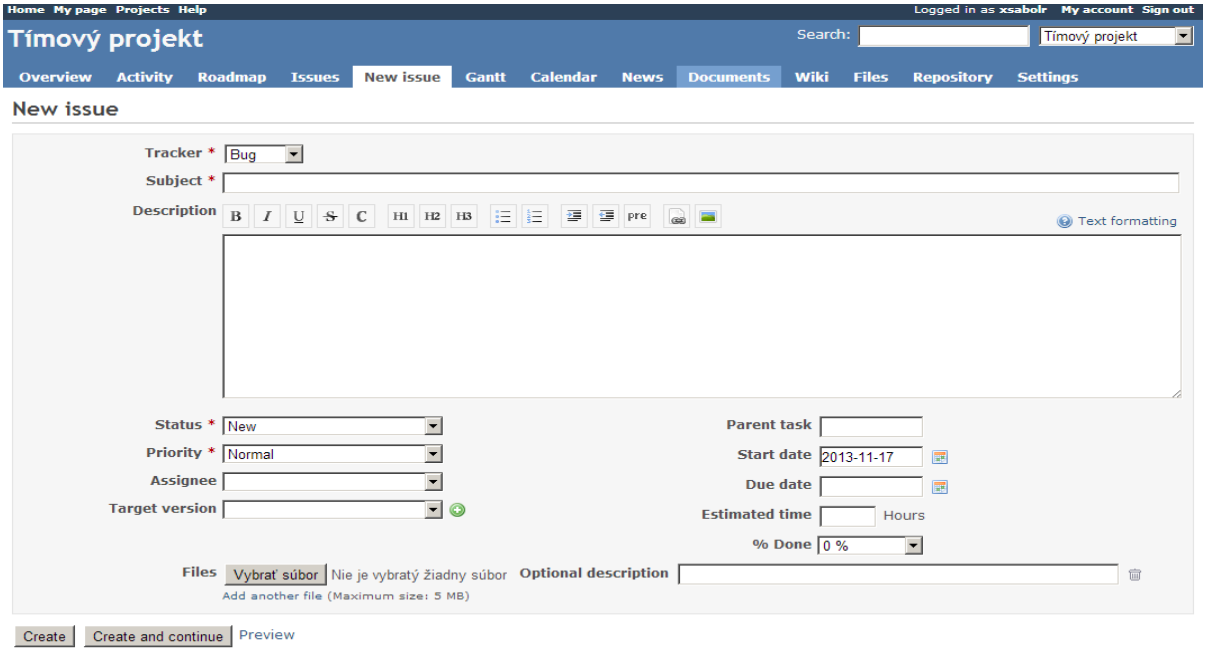
- vyberie projekt, v ktorom chcem zaevidovať požiadavku, kliknutím na názov projektu (Obr. č. 2.)
- na ďalšej obrazovke (Obr. č. 3), na ktorej je prehľad daného projektu, klikne na položku “New issue” v hornom menu tejto obrazovky
- následne sa zobrazí obrazovka s formulárom (Obr. č. 4), ktorý zákazník vypĺní



Obrázok č. 2



Obrázok č. 3



Obrázok č. 4

4.5.2.1.2 Pridanie kategórie pre požiadavku

Požiadavke je potrebné priradiť kategóriu z týchto 4 typov:

- chyba (bug)
- podpora (support)
- úloha (task)
- požiadavka na zmenu (change request)

Priebeh činností zákazníka v systéme Redmine:

- v rozbaľovači “Tracker” zvolí možnosť “požiadavka na zmenu” (change request)

4.5.2.1.3 Pridanie subjektu pre požiadavku

Subjekt (predmet) by mal byť čo najvýstižnejší, aby bolo možné požiadavku ľahko identifikovať a preto je vhodné zvoliť takýto formát.

[<Modul v systéme>] <Názov>

Napríklad:

[Architektúra] Spracovanie správ z Androidu

Hodnoty, ktoré môžu jednotlivé polia obsahovať sú uvedené v nasledujúcej tabuľke (Tab. č. 3).

Modul v systéme	Architektura Kinect Mobil OpenCV Poloha Dokumentacia
Názov	opis požiadavky bez diakritiky v rozsahu max. 10 slov

Tabuľka č. 3

4.5.2.1.4 Pridanie opisu požiadavky

Opis požiadavky na zmenu musí byť presný a výstižný.

Obsahuje:

- modul, ktorého sa zmena týka
- opis zmeny
- dôvod zmeny

4.5.2.1.5 Pridanie priority požiadavky

Dôležitým pre pridanie novej požiadavky je zadať prioritu. Prioritu vyberieme z rozbaľovača, ktorý je vo formulári pod opisom na ľavej strane.

Môžeme vyberať z 5 stupňov priority:

- Low (nízka)
- Normal (normálna)
- High (vysoká)
- Urgent (urgentná)
- Immediate (okamžitá)

Najzávažnejšie sú požiadavky s prioritou “Urgent” a “Immediate”, ktoré je nutné riešiť v aktuálnom release resp. okamžite, aby sa mohlo pokračovať v projekte.

4.5.2.2 Analýza zmeny uvedenej v prijatej požiadavke

Vstup: Záznam o požiadavke v systéme Redmine
Výstup: Dokument s analýzou zmeny s časovým a finančným odhadom
Zodpovedná osoba: Projektový analytik

Popis: Podstatou tejto činnosti je analýza požiadavky na zmenu, ktorá je už riadne evidovaná v systéme Redmine. Po zanalyzovaní sa rozhodne, či sa nejedná o požiadavku, ktorá je v absolútnom rozpore s analýzou. V takomto prípade ide o nepochopenie systému a požiadavka je následne zamietnutá.Ak zákazník aj po zamietnutí požiadavky stále trvá na tejto zmene, požiadavka sa mu vráti na modifikovanie tak, aby nebola v rozpore so súčasnou analýzou.
Požiadavka môže byť tiež označená ako chyba - to sa stane v prípade ak je nejaká časť systému zle impelentovaná. Požiadavka sa v tomto prípade označí ako chyba a je odoslaná osobe, ktorá je zodpovedná za túto

implementáciu.
Existuje ešte jeden prípad, kedy sa žiada o zmenu už implementovanej časti. V tomto prípade sa na základe analýzy rozhodne o aky zásah do systému ide. Prebehne analýza rozsahu požadovaných zmien, časový a finančný odhad. K tomuto odhadu sa môže vyjadriť zákazník v systéme Redmine.
Požiadavka s analýzou je následne posunutá programátorom. V prípade nesúhlasu s finančným odhadom zo strany zákazníka je požiadavka zamietnutá poprípade zaslaná na prehodnotenie zákazníkovi.

4.5.2.3 Implementácia zmeny

Vstup: Záznam o požiadavke v systéme Redmine, Analýza požiadavky na zmenu
Výstup: Implementovaný modul, ktorý spĺňa požiadavku na zmenu
Zodpovedná osoba: Programátor

Popis: Hlavnou úlohou tejto činnosti je implementovať funkcionalitu, ktorá bola uvedená v požiadavke na zmenu. Modul, v ktorom sa táto implementácia vykonávala je po ukončení implementácie posunutý testerom na vykonanie unit testovania. Vývoj daného modulu má na starosti jeden z programátorov - senior developer. Ak nastane prípad, že sa modul na základe analýzy nedá implementovať, je spolu s opisom problému zaslaný spať projektovému analytikovi.

4.5.2.4 Testovanie zmeny - Unit testy

Vstup: Implementovaný modul
Výstup: Správa o úspešnom unit testovaní
Zodpovedná osoba: Tester

Popis: Podstatou tejto činnosti je otestovanie funkčnosti implementovanej zmeny a to tak, aby sa na základe tejto zmeny nenarušila spolupráca s ostatnými modulmi v systéme. Pomocou unit testov sa zistí, či zmenený modul dokáže správne fungovať v systéme a spolupracovať s ostatnými časťami systému. Ak nastane prípad,že modul neprejde niektorým z unit testov, tak je vrátený na doimplementovanie. Výstupom testovania je správa o úspešnosti unit testov, ktorá je následne prílohou pre akceptačné testovanie. Unit testovanie sa vykonáva na základe metodiky o unit testovaní.

4.5.2.5 Akceptačné testovanie

Vstup: Správa o úspešnosti unit testovania, akceptačné testy
Výstup: Správa o úspešnom akceptačnom testovaní
Zodpovedná osoba: Projektový manažér

Popis: Hlavným ťažiskom tejto činnosti je vykonanie akceptačných testov projektovým manažérom, ktorý tak učinní na základe vykonanej analýzy. Na základe toho, či zmena prejde akceptačným testovaním je aj rozhodnutie, či sa môže zmena odovzdávať a prejsť k podpísaniu preberacích protokolov. Prílohou k akceptačným testom je aj správa o unit testovaní, ktorá je technickým dôkazom o funkčnosti systému po zahrnutí požadovanej zmeny. Ak nastane zlyhanie akceptačných testov, tak projektový manažér odošle správu o zlyhaní projektovému analytikovi, aby zistil príčinu zlyhania a napravil to.

4.5.2.6 Odovzdanie zmeny

Vstup: Správa o úspešnosti akceptačného testovania
Výstup: Podpísané preberacie protokoly
Zodpovedná osoba: Projektový manažér, zákazník

Popis: V tejto činnosti je na základe úspešného akceptačného testovania podísaný preberací protokol o implementovanej zmene. V preberacom protokole je uvedený aj dodatok o nasadení zmeny v konkrétnom release, podmienky financovania, prípadne poznámky. Zmenený modul je odoslaný zodpovednej osobe, ktorá sa postará o nasadenie v danom termíne.

4.6 Metodika plánovania úloh

Autor: Jakub Mercz

4.6.1 Úvod

Cieľom tejto metodiky je definovať presné pravidlá a postupy pre plánovanie úloh a ich následné zaznamenia na stretnutiach k tímovému projektu, ktorých sa zúčastňuje aj product owner. Tieto stretnutia majú okrem iného za úlohu zhodnotenie vypracovania predchádzajúcich úloh a naplánovanie úloh ďalších.

Metodika je určená pre všetkých členov tímu, špeciálne postavenie má člen aktuálne poverený písaním zápisnice.

4.6.1.1 Použité pojmy

- *Product owner* – pozícia definovaná v scrum-e, pre účely tohto projektu je ním vedúca projektu

4.6.1.2 Použité skratky

- *PO* – Product owner

4.6.1.3 Nadväzujúce metodiky

- Metodika evidencie nových úloh v Redmine

4.6.1.4 Role

Rola	Popis	Zodpovednosť
Člen tímu	-	prezentácia výsledkov, identifikácia úloh, premiestňovanie úloh na tabuli
Zapisovateľ	Člen tímu aktuálne poverený písaním zápisnice	zapisovanie úloh a ich stavov do zápisnice

4.6.1.5 Nástroje

1) Softvérové

a) Textový editor podľa uváženia zapisovateľa

2) Fyzické
   a) Veľké papiere rozdelené na zóny, zavesené v Laboratóriu počítačového videnia a počítačovej grafiky, ktoré slúžia ako tabuľa
   b) Prilepovacie lístočky a fixy rôznych farieb

Obrázok 1 - *Fyzické nástroje*

4.6.2 Procesy

4.6.2.1 Dokončenie úlohy

Vstup: výsledky práce medzi stretnutiami
Výstup: zaznamený aktuálny stav úlohy na tabuli a v tabuľke v zápisnici
Zodpovedný: člen tímu, zapisovateľ

Popis:
Postupne v poradí určenom PO prezentuje každý člen tímu svoje výsledky od posledného stretnutia a na základe diskusie s PO a rozhodnutia PO vyhodnocuje úlohu ako dokončenú.

1. Zapisovateľ si pripraví tabuľku vychádzajúc z tabuľky pridelených úloh, ktorá sa nachádza v zápisnici z predchádzajúceho stretnutia. Do tejto tabuľky pripojí ďalší stĺpec s názvom „Stav“.
2. Člen tímu povie názov pridelenej úlohy z predchádzajúceho stretnutia.
3. Člen tímu opíše čo najpodrobnejšie postup a výstup jeho snaženia od posledného stretnutia.
   a. Ak ho k tomu vyzve PO, člen tímu vizuálne odprezentuje aktuálny stav jeho modulu.
4. Člen tímu povie koľko času strávil danou úlohou.
5. Člen tímu vedie diskusiu s PO, či je úloha splnená alebo je nutná ďalšia práca na danej úlohe.

a. Ak bola úloha splnená

- Člen tímu nájde prilepovací papierik s danou úlohou a premiestni ho do sekcie DONE na tabuli.
- Zapisovateľ zapíše do tabuľky ako stav úlohy „vyriešené“

b. Ak úloha nebola splnená

- Zapisovateľ zapíše do tabuľky ako stav úlohy „v procese riešenia“
- Ak nastali špecifické udalosti týkajúce sa riešenia danej úlohy, zapisovateľ ich tiež popíše do stĺpca Stav

6. V prípade, že člen tímu má ďalšiu úlohu, ktorá ešte nebola spomenutá, pokračuje bodom 2 s nespomenutou úlohou.
7. Po prejdení všetkých úloh aktuálneho člena tímu, PO vyzve ďalšieho člena tímu k reprezentácií svojich výsledkov.

Keď všetci členovia tímu prejdu týmto procesom, zapisovateľ by mal mať hotovú tabuľku, ktorá sa podobá na *Tabuľku 1*.

ID	Pridelené členovi	Opis úlohy	Stav
1	Martin	študovať OpenCV, vytvoriť prvú verziu webovej stránky tímu	vyriešené
2	Lukáš	pozrieť sa na architektúru aplikácie podrobnejšie, nainštalovať databázu	v procese riešenia, nemohol nainštalovať kvôli nefunkčnosti serveru
3	Patrik	porovnávať rôzne metódy detekovania objektov, na nejakej snímke skúšať detekovanie objektov, vyskúšať či je lepšie hladať objekt nanovo na každom snímku, ale sledovať objekt	vyriešené
4	Robo	skúsiť ukladať gestá do lokálnej databázy a skúsiť odoslať informácie	v procese riešenia, nemal potrebný HW

		o gestách na lokálny server	
5	Marianna a Peťo	preštudovať si OpenGL, pracovať na nejakej jednoduchšej úlohe (napríklad vypisovať texty na rôzne miesta na obrazovke, pracovať s nejakým 3D modelom, dopĺnať text do obrázku)	vyriešené
6	Jakub	nainštaluje knižnicu pre prácu s Kinnectom a spraví experimenty na snímanie hlavy z rôznych pohľadov Kinnectu	vyriešené
7	Juraj	doriešiť aplikáciu na záznam GPS, pozrieť sa spolu s Robom na ukladanie gest v lokálnej databáze	vyriešené
8	všetci členovia	každý si podrobnejšie premyslí nejaký druh hry (napríklad zábavného typu, vzdelávacieho typu, atd.), najmä také, ktoré dokážu zabaviť deti	vyriešené

Tabuľka 1 - *Príklad tabuľky stavu úloh v zápisnici*

4.6.2.2 Zaznamenanie novej úlohy

Vstup: diskusia o príbehu aktuálneho šprintu
Výstup: zaznamené nové úlohy na tabuli a v tabuľke v zápisnici
Zodpovedný: člen tímu, zapisovateľ

Popis:
Po diskusii členov tímu a PO o príbehoch aktuálneho šprintu (práve prebiehajúci alebo začínajúci šprint) jednotlivý členovia tímu identifikujú úlohy pre svoj modul a po diskusii s PO ich zaznamenajú.

1. Zapisovateľ si pripraví tabuľku pridelených úloh so stĺpcami „ID“, „Pridelené členovi“ a „Opis úlohy“

2. Každý člen tímu si zoberie farebné papieriky a fixy s farbou, ktorá mu bola pridelená na predchádzajúcich stretnutiach.
3. Každí člen tímu presunie svoje papieriky so sekcie V PLÁNE do sekcie AKTUÁLNY ŠPRINT.
4. Člen tímu s identifikovanými úlohami ich postupne odprezentuje PO.
    a. Ak bola úloha schválená PO a bola PO určená ako prioritná:
        - Člen tímu napíše názov úlohy a svoje iniciálky na papierik a prilepí ho do sekcie AKTUÁLNY ŠPRINT.
    b. Ak bola úloha schválená PO a bola PO určená ako sekundárna:
        - V prípade že odhadovaný čas vypracovania úloh pre člena tímu v sekcii AKTUÁLNY ŠPRINT plus prejednávanej úlohy je pod 8 hodín, člen tímu napíše názov úlohy a svoje iniciálky na papierik a prilepí ho do sekcie AKTUÁLNY ŠPRINT.
        - V prípade že odhadovaný čas vypracovania úloh pre člena tímu v sekcii AKTUÁLNY ŠPRINT plus prejednávanej úlohy je nad 8 hodín, člen tímu napíše názov úlohy a svoje iniciálky na papierik a prilepí ho do sekcie V PLÁNE.
5. Zapisovateľ zapíše krstné meno člena tímu prezentujúceho svoje úlohy a do stĺpca opis úlohy zapíše jeho úlohy podľa tabule. V stĺpci „ID“ dá inkrementovanú hodnotu z predchádzajúceho riadku (v prvom údajovom riadku napíše číslo 1).
6. Ak člen tímu odprezentoval všetky svoje identifikované úlohy, pokračuje ďalší člen tímu od bodu 4
7. Ak všetci členovia tímu odprezentovali svoje identifikované úlohy, zapisovateľ zapíše do posledného riadku do stĺpca „Pridelené členovi“ text „všetci členovia“ a úlohu počas diskusie identifikovanú ako všeobecnú a potrebnú pre diskusiu na ďalšom stretnutí alebo inak potrebnú pre daľšiu prácu niektorého člena.

Na konci procesu by mal zapisovateľ mať tabuľku podobnú *Tabuľke 1*, ale bez stĺpca „Stav“.

4.7 Metodika prehliadky kódov vo dvojiciach

Autor: Marianna Mušinská

4.7.1 Úvod

Účelom tejto metodiky je definovanie procesu prehliadky kódov pri vytváraní aplikácií. Za úlohu má taktiež predstaviť postup prehliadky kódov pomocou nástroja Redmine. Metodika je zameraná pre malé a stredné tímy pracujúce v jazyku C++.
Prehliadka kódov prispieva k vytváraniu kvalitných kódov. Keď od začiatku používame prehliadky kódov, tak zabezpečujeme aj nižšie náklady. Pomáha predchádzať, odhaľovať, naprávať a predchádzať chybám, zachováva konzistentnosť, zvyšuje kvalitu kódu.
K prvým krokom k dobrej prehliadky kódov je potrebné zadefinovať dobrý nástroj, ktorý pomáha pri prehliadkach kódov.

Táto kapitola sa zaoberá opisom jednotlivých rolí a opisom procesov prehliadok kódov.

Roly a zodpovednosti

Manažér kvality

- kontroluje kvalitu kódov
- určuje prehliadky v jednotlivých častiach zdrojových kódov
- režíruje časové obmedzenia na prehliadky
- je zodpovedný za to, aby zdrojové kódy boli čo najkvalitnejšie

Vývojár

- vytvára zdrojový kód
- vysvetluje jednotlivé časti kódu, odôvodňuje prečo boli zvolené jednotlivé štýly programovania
- značkuje jednotlivé časti kódu

v druhej fáze:

- číta jedotlivé značkovania
- vytvára opravy v zdojovom kóde
- upozorňuje a opravuje chyby v kóde
- pomáha v spísaní zápisnice

Dokumentarista

- Je zodpovedný za vytváranie zápisnice počas stretnutia
- Je zodpovedný za výstupnú zápisnicu prehliadky

Administrátor

- príprava a inštalácia vyžadovaných nástrojov

4.1.2 Procesy

Proces plánovanie prehliadky

Vstup:

- Vývojári
- Zdrojové kódy

Výstup:

- časový postup plánovania kontrôl
- plán predstavenia zdrojového kódu

Rola :

- vývojári, dokumentarista, manažér kvality

Plánovanie prehliadky určuje, aké zdrojové kódy má vytvárať vývojár v akom časovom rozsahu a s kým bude spolupracovať v párovej prehliadke.

Cieľom tohoto procesu je naplánovať prehliadky kódov, rozdeliť členov tímu do jednotlivých párov, ktorí sa budú navzájom kontrolovať. Taktiež sa musí naplánovať časový postup prehliadok, aby nevznikli prípady kolízie, kde jeden z dvojice vývojárov bude musieť čakať na druhého, ktorý sa nezúčastňuje dostatočne aktívne procesu.

Proces značkovanie kódu

Vstup:

- zdrojový kód

Výstup:

- značkovaná zdrojový kód

Rola:

- vývojári

Vývojár za cieľom dobrej orientácie v zdrojovom kóde označkuje svoje svoje dielo. Značkovanie sa odohráva počas príslušného šprintu jedným z vývojárom. Cieľom značiek je uľahčiť spoluprácu pri párovom programovaní.

Proces vývoja zdrojového kódu

Vstup:

- počiatočná/aktuálna časť zdrojového kódusz dokumentáciou a so značkovaním
- požiadavky na zdrojový kód

Výstup:

- zdrojový kód s možnými chybami

Rola :

- vývojári

Vývojár vytvára zdrojový kód, s dokumentáciou a značkovaním počas šprintu. Prípadne keď už má k dispozícií počiatočný zdrojový kód, postupne pokračuje v vývoji.

Proces prehliadky

Vstup:

- počiatočná/aktuálna časť zdrojového kódu s dokumentáciou a so značkovaním
- požiadavky na zdrojový kód
- plán vykonávania prehliadky

Výstup:

- počiatočná/aktuálna časť zdrojového kódu s dokumentáciou a so značkovaním a identifikovanými chybami a anomáliami

Rola:

- vývojári

Podľa vopred určeného harmonogramu jeden z dvojice, ktorý sa nezúčastnil na vývoji zdrojového kódu skontroluje kód podľa dokumentácie a s pomocou značkovania. Pri nájdnení chyby označuje a upozorňuje na chyby.

Proces opravy chýb

Vstup:

- zoznam chýb a anomálií identifikovanými počas prehliadky kódu
- zdrojové kódy so súvisiacou dokumentáciou a značkovaním

Výstup:

- opravené a funkčné zdrojové kódy
- odstránené anomálie

Rola:

- *vývojári*

4.8 Metodika - Manažment verzií zdrojového kódu (nástroj Git)

Autor: Juraj Jarábek

4.8.1 Úvod

Táto metodika má na starosť opísať postupy spojené s procesmi manažmentu verzií zdrojového kódu. Je určená pre menší vývojový team, konkrétne team s počtom členov 8. Metodika sa zameriava na nižšiu úroveň manažmentu verzii zdrojového kódu. Táto úroveň opisuje jednotlivé kroky, resp. konkrétnu činnosť pri používaní nástroju Git, ktorý používa spomenutý tým pri verziovaní.

4.8.2. Zoznam pojmov

- **Git** – Git je distribuovaný systém riadenia revízií, ktorý vytvoril Linus Torvalds. Nemal by sa zamieňať s programom GIT (GNU Interactive Tools), správcom súborov na spôsob programu Norton Commander, ktorý vytvorili Tudor Hulubei a Andrei Pitis.
- **Git Extension** -Git Extension je jediné grafické používateľské prostredie pre Git, ktoré umožňuje kontrolovať Git bez použitia príkazovej riadky. Samotný program obsahuje aj manuál, video tutorály pre rýchlu orientáciu.
- **Repozitár** - Z angl. "repository", ktoré znamená: schránka, úschovňa, úložisko, skladisko, zdroj. Pre repozitár sa tiež používá označenie "installation source". Jedná sa o miesto (napr. server alebo lokálny počítač to znamená vzdialené a lokálne úložiská), kde sú uložené zdrojové kódy projektu.
- **Vetva** - Predstavuje odklonenie vo vývoji od hlavnej, tzv. master vetvy. Hlavná vetva spravidla obsahuje len otestovaný, funkčný a stabilný kód a kaţdá nová funkcionalita sa vyvíja a testuje v samostatnej vetve.
- **Commit** – Odoslanie zmien z lokálneho úložiska na server
- **Merge** - Po slovensky: zlúčiť, splynúť, spojiť. Konkrétne pri používaní v nástroji Git sa jedná o zlúčenie rôznych vývojových vetiev, zahŕňa riešenie konfliktov medzi rôznymi verziami toho istého súboru.
- **Konflikt**: Vzniká, ak bol rovnaký súbor modifikovaný dvoma rôznymi užívateľmi.
- Rebase: Jedná sa o process získania zmien, ktoré boli vykonané na inej vetve, za účelom aktualizácie aktuálnej vetvy.
- **Revision**: Predstavuje aktuálnej verziu projektu v repozitári, alebo tiež počet komitov do repozitára. Pri vytvorení je nastavené na 0, každý komit ho inkrementuje o 1.

4.8.3. Nadväzujúce metodiky

- ➔ Metodika: Prehliadka a kontrola zdrojového kódu
- ➔ Metodika zdrojových kódov (všeobecne)
- ➔ Metodika: Manažment zmien a požiadaviek na zmenu aplikácie
- ➔ Metodika: Manažment chýb

4.8.4. Role

Zoznam rolí pre ktoré je vhodná táto metodika.

Rola	Úloha
Manažér kvality	Jeho úlohou je vytvorenie / inštalácia repozitára (úložiska). Ďalej konfigurácia úložiska a jeho úložiska
Manažér podpory vývoja	Jeho úlohou je najmä testovanie a hodnotenie kvality,
Manžér rizík	Má na starosti riešenie konfliktov
Programátor	Má na starosti vývoj nových funkcionalít na svojom lokálnom úložisku , opravu chýb a komitovanie zmien na server

*4.8.*5. Procesy

4.8.5.1 Inštalácia nástroja Git Extensions

Vstup: Nakonfigurované vývojové prostredie na serveri
Výstup: Nainštalovaný nástroj na manažment verzií – Git Extensions
Zodpovednosť: Manažér kvality, vývojár

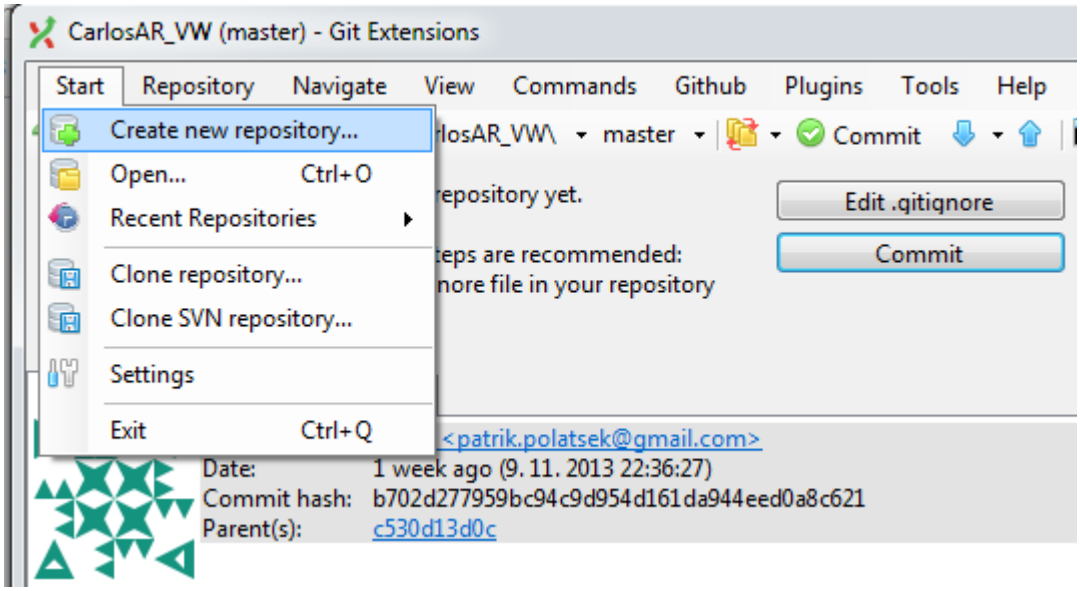
Primárnu zodpovednosť za proces inštalácie má manažér podpory vývoja, ktorý inštaluje nástroj na manažment verzií na serveri. Vývojári zodpovedajú za inštaláciu nástroja na ich lokálnom počítači, splnenie úlohy kontroluje opäť manažér kvality.

4.2 Konfigurácia nástroja Git Extensions

Vstup: Nainštalovaný nástroj na manažment verzií (Git Extensions)
Výstup: Nakonfigurované centrálne úložisko na server
Zodpovednosť: manažér kvality, programátor

Primárnu zodpovednosť za proces konfigurácie má manažér kvality, ktorý konfiguruje centrálne úložisko a prístup k nemu. Vývojár konfiguruje lokálne úložisko.

Preto môžme tento proces rozdeliť do štyroch hlavných krokov:
1. Vytvorenie centrálneho úložiska
2. Vytvorenie prípravnej (develop) vetvy
3. Nastavenie prístupových práv používateľov k centrálnemu úložisku
4. Konfigurácia lokálneho úložiska



Obr1.Vytvorenie nového úložiska (Repository) v nástroji GitExtensions

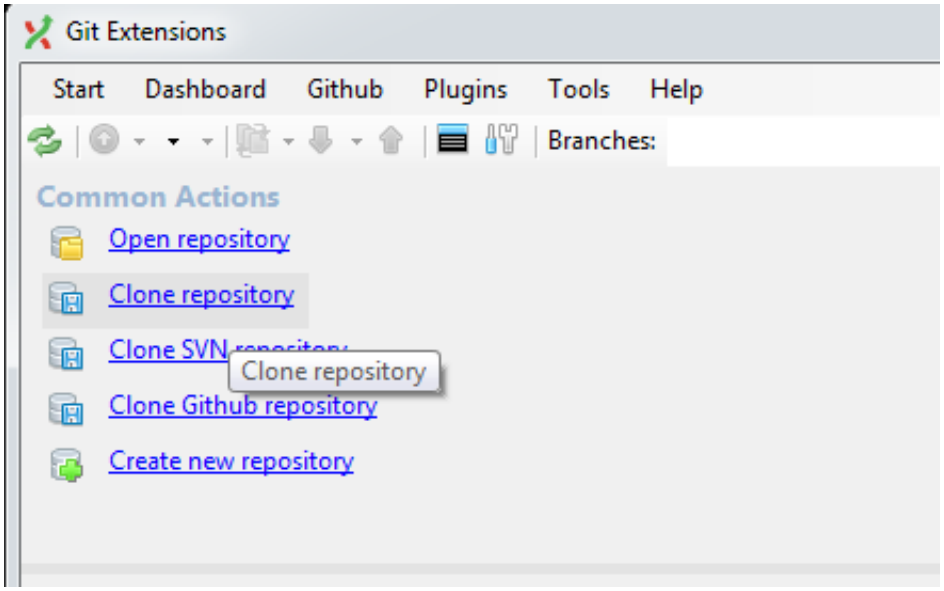
4.3 Prepojenie centrálneho úložiska s nástrojom na manažment verzií

Vstup: Nainštalovaný a nakonfigurovaný nástroj na manažment verzií (Git Extensions)
Výstup: Prístup k centrálnemu úložisku pomocou nástroja na manažment verzíi (Git Extentions)
Zodpovednosť: programátor

Proces prepojenia centrálne s nástrojom na manažment verzií ma na starosti každý člen týmu.

Tento process môžme rozdeliť do nasledujúcich krokov:

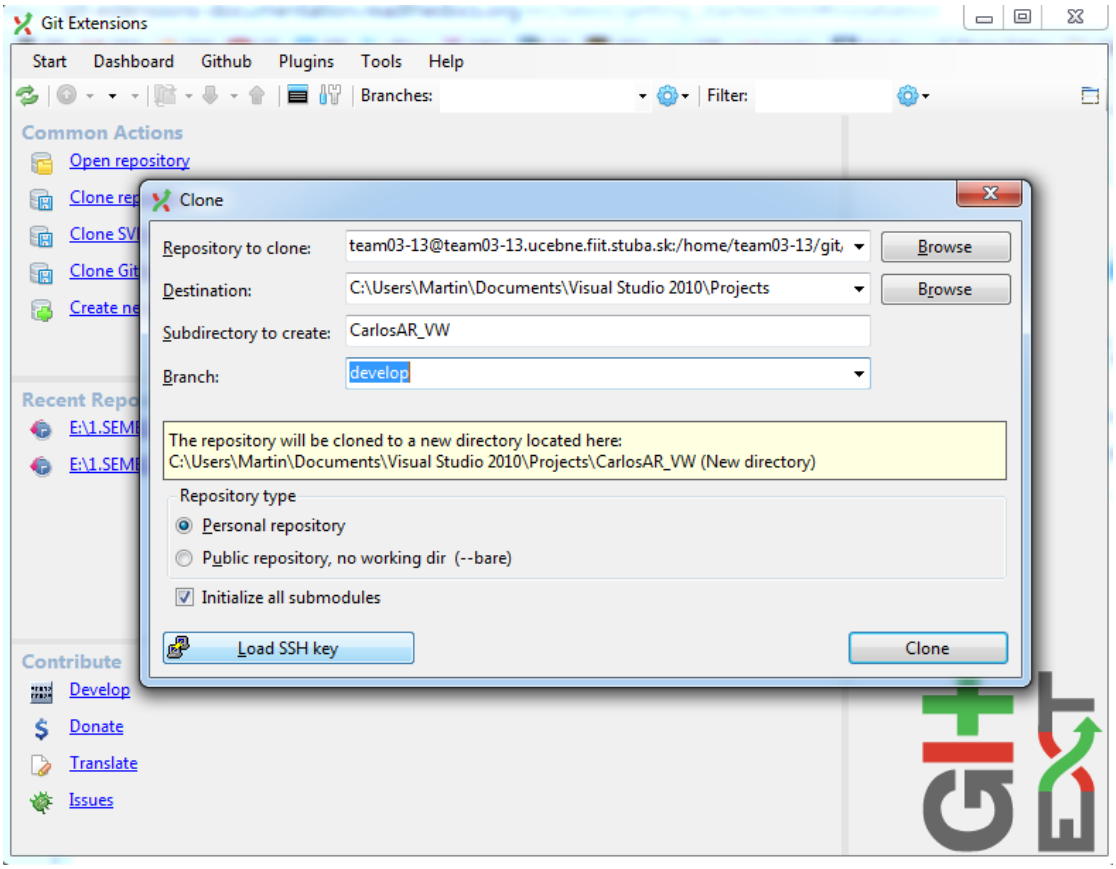
Po prvom spustení aplikácie je potrebné vybrať “Clone repository”



Obr.2 Clone repository

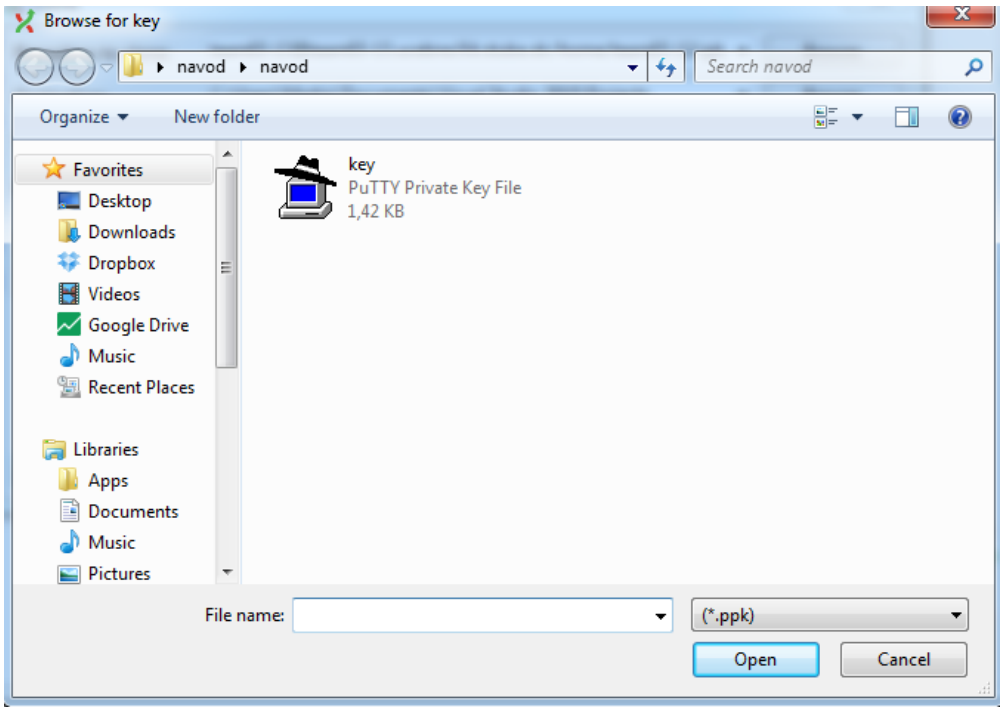
Následne vyplniť údaje podla našeho tímoveho repository:

- **Repository to clone**: eam03-13@team03-13.ucebne.fiit.stuba.sk:/home/team03-13/git/CarlosAR_VW.git
- **Destination:** Lokálna zložka napr.: C:\Users\Martin\Documents\Visual Studio 2010\Projects
- **Branch:** develop
- **Repository type:** Personal repository
- **Load SSH key:** vybrať key.ppk (Pri tom treba mať v nastavení nastavené Settings - > SSH.Putty, prípadne tí čo sú na Linuxe tak treba vložit key a mat v Settings -> SSH ... OpenSSH). V remotes treba mať nastavené Origin na team03-13@team03-13.ucebne.fiit.stuba.sk:/home/team03-13/git/CarlosAR_VW.git a správne načítaný ten SSH kľúč



Obr.3 Ukážka vyplnenia nasledujúcich údajov

Po kliknutí na SSH key vybrať key.ppk



Obr.4 Vybratie key.ppk

4.4 Proces nasadzovania

Vstup: Vývojová verzia system, Akceptačné testy
Výstup: Produkčná verzia systému, Zápis z akceptačného testovania
Zodpovednosť: Manažér vývoja, manažér kvality

Z pohľadu manažmentu verzií je dôležité rozlíšiť medzi rôznymi verziami systému. Preto sa rozlišujú tieto 2 vetvy v našom projekte:

- Hlavná (master)
- Vývojová (develop)

Manažment verzií zdrojového kódu pomocou Git. Prípravná verzia systému slúži na odskúšanie a integráciu otestovaných funkcionalít pred ich nasadením do produkčnej verzie, nasadzuje sa z prípravnej (develop) vetvy. Produkčná verzia systému je prezentovaná zákazníkovi, preto zahŕňa len odladené (otestované) funkcionality, ktoré prešli integračnými a akceptačnými testami. Táto verzia sa aktualizuje po ukončení každého vývojového cyklu, nasadzuje sa z hlavnej (master) vetvy. O nasadení do prípravnej verzie rozhoduje manažér vývoja, do produkčnej verzie manažér kvality. Okrem toho sa podľa potreby vytvárajú ďalšie vetvy systému, zachytávajúce ho v istom stave.

5. Záznamy zo stretnutí

V tejto kapitole sa nachádzajú týždenné zápisnice zo stretnutí nášho tímu. Každá zápisnica obsahuje zápis priebehu stretnutia, stav existujúcich pridelených úloh a zoznam nových úloh.

Zápis 1. stretnutia tímu č. 3

Autor zápisu:	Bc. Patrik Polatsek	
Dátum:	30.9.2013	
Miestnosť:	Laboratórium počítačového videnia a grafiky	
Prítomní:	Vedúci:	Ing. Vanda Benešová, PhD.
	Členovia tímu:	Bc. Peter Hamar
		Bc. Juraj Jarábek
		Bc. Jakub Mercz
		Bc. Marianna Mušinská
		Bc. Martin Petluš
		Bc. Patrik Polatsek
		Bc. Róbert Sabol
		Bc. Lukáš Sekerák

Priebeh stretnutia:

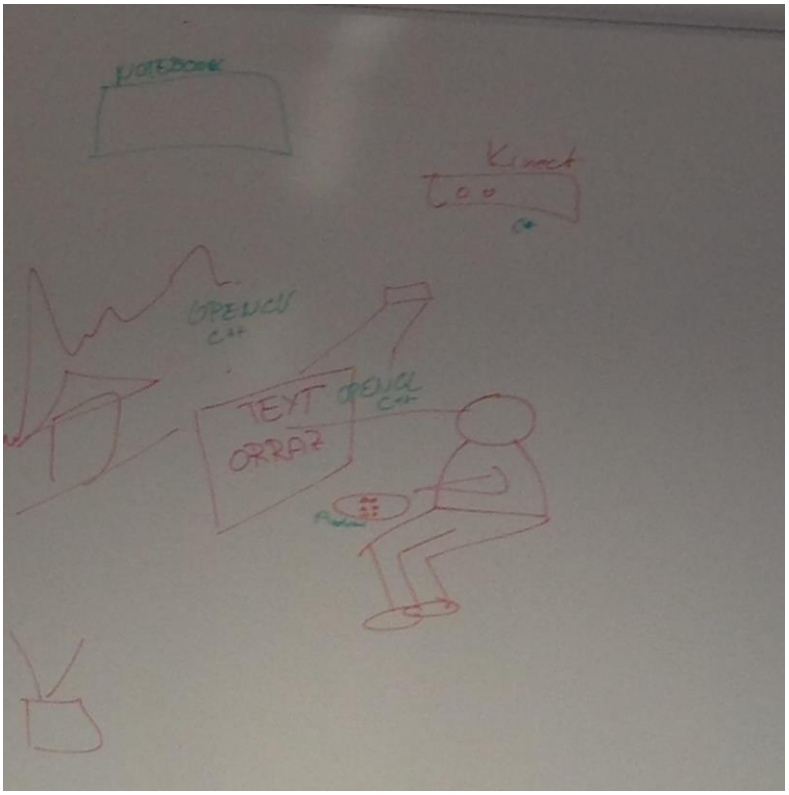
- Vedúca tímu načrtla priebeh práce na projekte
- Na vývoj aplikácie bude k dispozícii LED projektor a transparentná fólia, na ktorú sa bude vysielať obraz
- Aplikáciu, ktorú máme vytvoriť sme rozdelili do nasledovných častí (komponentov) (Príloha A):
    - riadiaca časť: notebook bude celú aplikáciu spravovať a vykonávať
    - mobilná aplikácia: využitie gps a interakcia s mobilom → technológie: Java, Android
    - detekcia objektov: z videosekvencií sa budú detekovať objekty (pomocou obrázkov uložených v databáze) → technológie: C++, OpenCV
    - sledovanie polohy hlavy: pre zobrazenie textu/obrazu v správnej pozícii voči hlave sa bude sledovať poloha hlavy → C++, KINECT
    - zobrazovacia časť: zobrazenie textu/obrazu na transparentnú fóliu → technológie: C++, OpenGL
- Predbežne sme si zadelili, ktorí členovia tímu sa budú venovať jednotlivým častiam projektu
- Na vývoj mobilnej aplikácie by sa dali využiť kódy od Richarda Sámela
- Ďalším cieľom projektu je vytvoriť hru, kt. bude reflektovať reálny svet, diskutovali sme možné námety na hru
    - hádať o akú budovu ide
    - uhádnuť dopravnú značku
- V prvej fáze projektu budeme zobrazovať len text, príp. doplnkový obraz (pre každý POI treba mať pripravené viaceré verzie a vždy zobraziť niečo iné)

Úlohy do ďalšieho stretnutia:

ID	Pridelené členovi	Opis úlohy
1	Lukáš, Jakub, Juraj	vytvoriť Android aplikáciu, ktorá bude ukladať aktuálnu gps polohu → timestamp pre videozáznam
2	Robo	kontaktovať sa s Richardom Samelom, získať kódy na gestá pre mobilnú aplikáciu
3	všetci členovia	pripraviť si nápady ako prepojiť jednotlivé komponenty (programy)

Prílohy:

Príloha A. Prvotný návrh štruktúry aplikácie



Zápis 2. stretnutia tímu č. 3

Autor zápisu: Bc. Martin Petluš
Dátum: 7.10.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní:
Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hamar
Bc. Juraj Jarábek
Bc. Jakub Mercz
Bc. Marianna Mušinská
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	Lukáš, Jakub, Juraj	vytvoriť Android aplikáciu, ktorá bude ukladať aktuálnu GPS polohu → timestamp pre videozáznam	vyriešené
2	Robo	kontaktovať sa s Richardom Samelom, získať kódy na gestá pre mobilnú aplikáciu	vyriešené
3	všetci členovia	pripraviť si nápady ako prepojiť jednotlivé komponenty (programy)	v procese riešenia

Priebeh stretnutia:

- Zhodli sme sa na tom, že najlepšie bude používať vždy tú istú kameru na snímanie - kamera bude požičaná od Andreja Fogeltona
- Všetky dáta si budeme zbierať sami (GPS a video)
- Treba porozmýšlať nad zaujímavými trasami so zaujímavými objektami (historické pamiatky, …)
- Ako vstup budeme mať dostupnú aj informáciu o pohybe auta
- Diskutovali sme o Jakubom vytvorenej aplikácie na zaznamenávanie GPS polohy
    - treba vyskúšať častejšie záznamy ako každú sekundu

    - zaznamenaná poloha a čas sa zatiaľ ukladajú do súboru, v budúcnosti bude treba vymyslieť nejaký iný spôsob, ktorý sa bude čo najbližšie približovať realtime času
- Diskutovali sme o Lukášom navrhovanej architektúre aplikácie:
    - modul s Kinnectom bude posielať polohu a uhol hlavy
    - treba si dať pozor, aby to nebolo príliš distribuované
    - systém musí byť skoro realtimeový
    - lokálna databáza na každom zariadení, každý modul bude samostatný
    - aplikácia sa najprv implementuje ako rôzne triedy a potom sa rozhodne čo ďalej (či každý modul bude ako samostatný spustiteľný súbor - program alebo jeden veľký program)
    - tiež je možnosť, že budeme mať jeden hlavný program (najmenší) a potom samostatné knižnice
    - treba sa dohodnúť na moduloch a ich interfaceoch
    - treba si definovať, aké príkazy sa budú posielať z mobilu (dotykové, rečové vstupy)
- možnou zábavnou hrou pre deti by bolo lietadielko, ktoré by sa muselo vyhýbať horizontu, alebo zbierať napríklad mince, lietadielko by sa ovládalo s pomocou mobilného zariadenia
- diskutovali sme o najlepšej polohe Kinnectu pri snímaní polohy hlavy (je treba zisťovať aj polohu očí, alebo stačí len poloha hlavy?, ...)
- vedúca pozná človeka, ktorý vie o nejakej knižnici, ktorá nám uľahčí prácu s Kinnectom
- diskutovali sme o aplikácií, ktorú by sme chceli využiť v našej aplikácií na rozpoznávanie gest:
    - mohli by sme tiež využiť rozpoznávanie hlasu - na slovenčinu to nefunguje dobre a nefunguje to bez internetu, s angličtinou to tiež nefunguje najlepšie (výslovnosť) - mohli by sme avšak použiť jednoduché povely ako one, two, stop, run, …
    - treba si rozmyslieť ako sa dá spojiť aplikácia na rozpoznávanie gest s naším projektom (cez databázu)

Úlohy do ďalšieho stretnutia:

ID	Pridelené členovi	Opis úlohy
1	Martin	študovať OpenCV, vytvoriť prvú verziu webovej stránky tímu
2	Lukáš	pozrieť sa na architektúru aplikácie podrobnejšie, nainštalovať databázu
3	Patrik	porovnávať rôzne metódy detekovania objektov, na nejakej snímke skúšať detekovanie objektov, vyskúšať či je lepšie hladať objekt nanovo na každom snímku, ale sledovať objekt

4	Robo	skúsiť ukladať gestá do lokálnej databázy a skúsiť odoslať informácie o gestách na lokálny server
5	Marianna a Peťo	preštudovať si OpenGL, pracovať na nejakej jednoduchšej úlohe (napríklad vypisovať texty na rôzne miesta na obrazovke, pracovať s nejakým 3D modelom, dopĺnať text do obrázku)
6	Jakub	nainštaluje knižnicu pre prácu s Kinnectom a spraví experimenty na snímanie hlavy z rôznych pohľadov Kinnectu
7	Juraj	doriešiť aplikáciu na záznam GPS, pozrieť sa spolu s Robom na ukladanie gest v lokálnej databáze
8	všetci členovia	každý si podrobnejšie premyslí nejaký druh hry (napríklad zábavného typu, vzdelávacieho typu, atd.), najmä také, ktoré dokážu zabaviť deti

Prílohy:

Zápis 3. stretnutia tímu č. 3

Autor zápisu: Bc. Jakub Mercz
Dátum: 14.10.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní: Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hamar
Bc. Juraj Jarábek
Bc. Jakub Mercz
Bc. Marianna Mušinská
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	Martin	študovať OpenCV, vytvoriť prvú verziu webovej stránky tímu	vyriešené
2	Lukáš	pozrieť sa na architektúru aplikácie podrobnejšie, nainštalovať databázu	v procese riešenia, nemohol nainštalovať kvôli nefunkčnosti serveru
3	Patrik	porovnávať rôzne metódy detekovania objektov, na nejakej snímke skúšať detekovanie objektov, vyskúšať či je lepšie hladať objekt nanovo na každom snímku, ale sledovať objekt	vyriešené
4	Robo	skúsiť ukladať gestá do lokálnej databázy a skúsiť odoslať informácie o gestách na lokálny server	v procese riešenia, nemal potrebný HW
5	Marianna a Peťo	preštudovať si OpenGL, pracovať na nejakej jednoduchšej úlohe (napríklad vypisovať texty na rôzne	vyriešené

		miesta na obrazovke, pracovať s nejakým 3D modelom, dopĺnať text do obrázku)	
6	Jakub	nainštaluje knižnicu pre prácu s Kinnectom a spraví experimenty na snímanie hlavy z rôznych pohľadov Kinnectu	vyriešené
7	Juraj	doriešiť aplikáciu na záznam GPS, pozrieť sa spolu s Robom na ukladanie gest v lokálnej databáze	vyriešené
8	všetci členovia	každý si podrobnejšie premyslí nejaký druh hry (napríklad zábavného typu, vzdelávacieho typu, atd.), najmä také, ktoré dokážu zabaviť deti	vyriešené

Priebeh stretnutia:

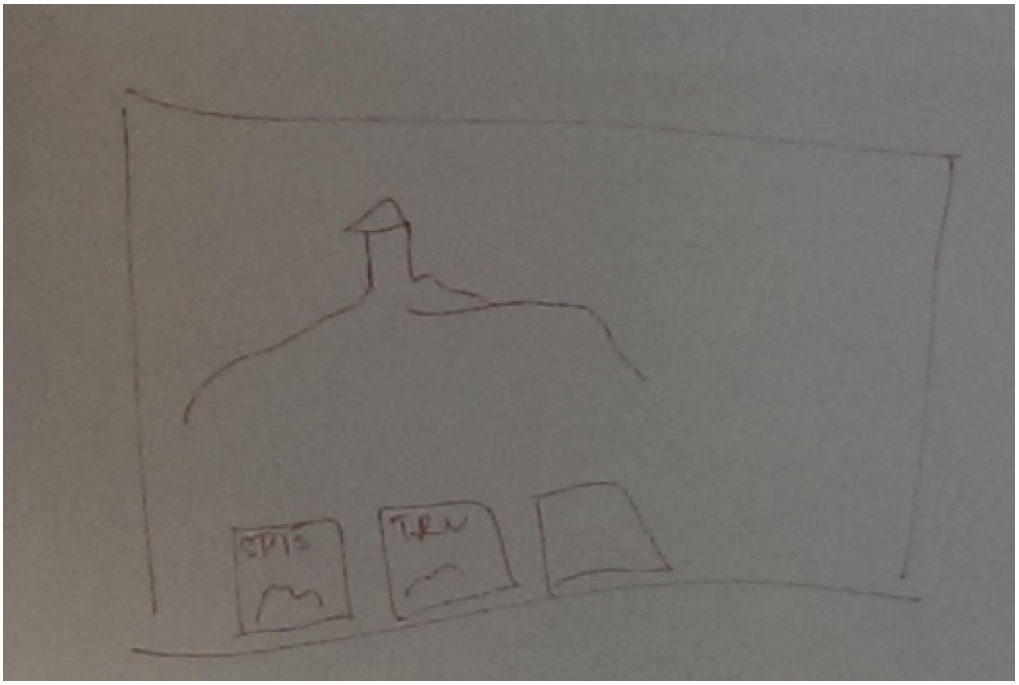
- popri kontrolovaní stavu úloh sme diskutovali o možných hrách
    - vyberanie z 3 obrázkov s názvami a hádanie čo je vidno za oknom (Príloha A)
    - rozstrielavanie zväčšeného obrazu objektu (Príloha B)
    - lietadlo, ktoré musí zostať nad horizontom a čím bližšie k horizontu letí, tým viac bodov sa pripočítava
    - hra na štýl obesenca, háda sa objekt za oknom alebo informácia k nemu
    - určiť zo zobrazených hodnôt aký vysoký alebo starý je objekt za oknom
    - hry založené na počítaní objektov za oknom, napríklad okien
- zhodli sme sa, že na stránku treba pridať potrebné logá a ochrániť emailové adresy proti spambotom
- treba aby všetci určili vstupy a výstupy svojho modulu
- pozerali sme záznamy od Juraja, ktoré natočil v Trenčíne
    - zistili sme že v noci sú kompletne nevhodné podmienky
    - v meste treba počítať s budovami cez celú zornú plochu
- Dohodli sme sa na dvojiciach pre code review:
    - Peter - Marianna
    - Juraj - Robo
    - Jakub - Lukáš
    - Martin - Patrik

Úlohy do ďalšieho stretnutia:

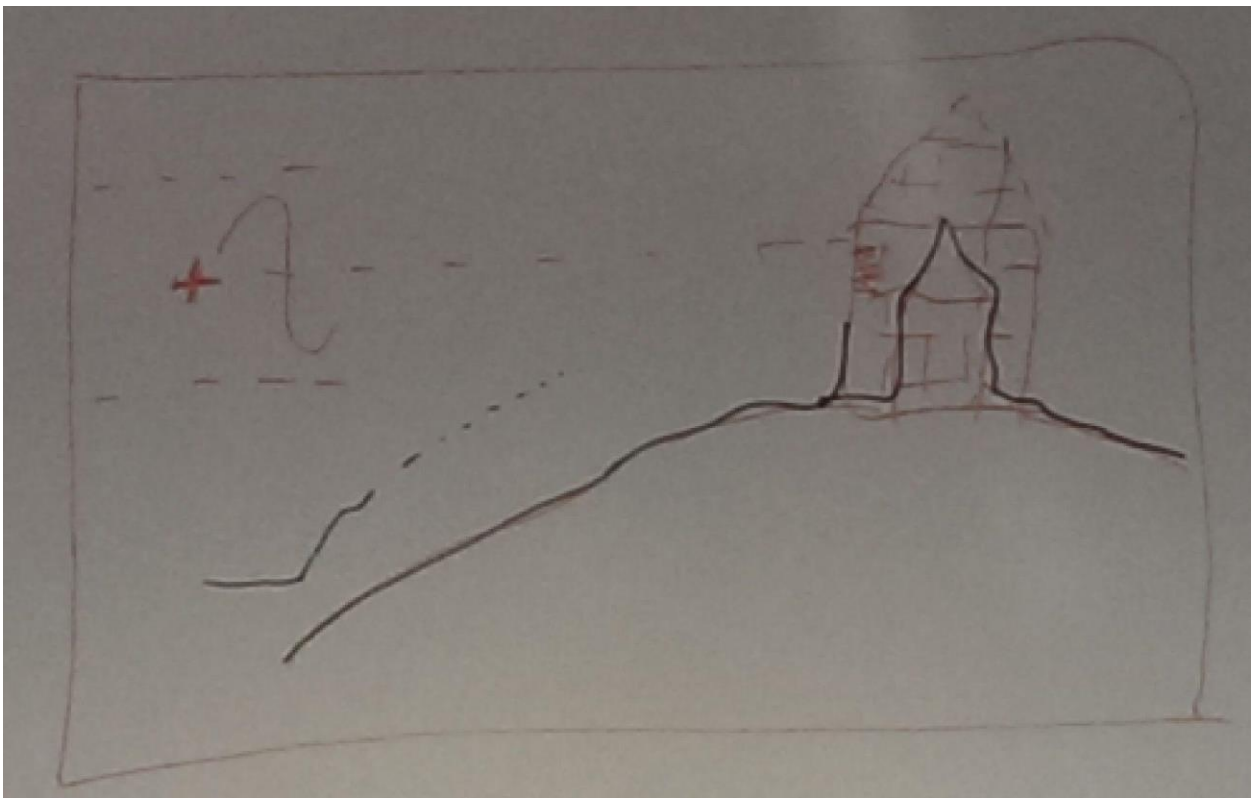
ID	Pridelené členovi	Opis úlohy
1	Juraj	natočiť ďalšie videá, aj mimo mesta, pripojiť k nim GPS informácie, sprístupniť GPS aplikáciu
2	Lukáš	dozerať na vloženie informácií pre architektúru
3	Patrik	zobrať fotky z google streetview a odskúšať detekciu
4	Martin	zaoberať sa novým modulom, ktorý ma za úlohu vypočítať polohu objektov na okne
5	Marianna a Peťo	pokračovať s OpenGL, odskúšať objavujúce sa a miznúce objekty
6	Robo	demo pre rozpoznávanie giest
7	Jakub	zaoberať sa detekciou tváre a spracovávať hĺbkovú informáciu
8	všetci členovia	definovať aké vstupy a výstupy má každý modul

Prílohy:

Príloha A. Návrh hry #1



Príloha B. Návrh hry #2



Zápis 4. stretnutia tímu č. 3

Autor zápisu: Bc. Juraj Jarábek
Dátum: 14.10.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní: Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hamar
Bc. Juraj Jarábek
Bc. Jakub Mercz
Bc. Marianna Mušinská
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	Martin	zaoberať sa novým modulom, ktorý ma za úlohu vypočítať polohu objektov na okne	Martin sa zaoberal týmto modulom a prišiel s novými nápadmi, vypočítanie vzdialenosti a daľšie srandy, in progress
2	Lukáš	dozerať na vloženie informácií pre architektúru	Bolo to rozpracované a su navrhnuté prvé templatey, menšie problémy s databázou. stále in progress.
3	Patrik	zobrať fotky z google streetview a odskúšať detekciu	Porovnanie bolo vykonané, využil existujúce kódy/metódy, stále in progress.
4	Robo	demo pre rozpoznávanie giest	stále sa pracuje, in progress

5	Marianna a Peťo	pokračovať s OpenGL, odskúšať objavujúce sa a miznúce objekty	spravili sa nejaké animácie, pridanie textu, hýbanie textu, jeho zobrazenie
6	Jakub	zaoberať sa detekciou tváre a spracovávať hĺbkovú informáciu	in progress
7	Juraj	natočiť ďalšie videá, aj mimo mesta, pripojiť k nim GPS informácie, sprístupniť GPS aplikáciu	vyriešené
8	všetci členovia	definovať aké vstupy a výstupy má každý modul	

Priebeh stretnutia:

- najskôr sme si skúsili prehliadnú fóliu s projektorom a vyhnotili, že projekcia na fóliu je príliš silná a video v pozadí nie je dobre vidieť.
- optické preloženie projektorov cez seba aby to dávalo zmysel (aby bol obraz správne preložený)
- vytvorili sme si plán / schému architektúry (viď. príloha)
    - každý modul by mohol bežať ako samostatný thread
    - Robova android aplikácia pre gestá by bola ako samostatná časť (resp. jediná externá časť) a komunikovala s programom cez TCP protkol (zatiaľ budeme)
- diskutovali sme hlbšie o celej architektúre, podrobnejšie rozvrhnutie modulov
- dohodli sme sa sme sa dlhodobejšie:
    - registrácia obrazu/kalibrácia (lokálne deskriptory), najskôr sa budeme hrať staticky akože auto stojí

- Dohodli sme sa na dvojiciach pre code review:
    - Peter - Marianna
    - Juraj - Robo
    - Jakub - Lukáš
    - Martin - Patrik

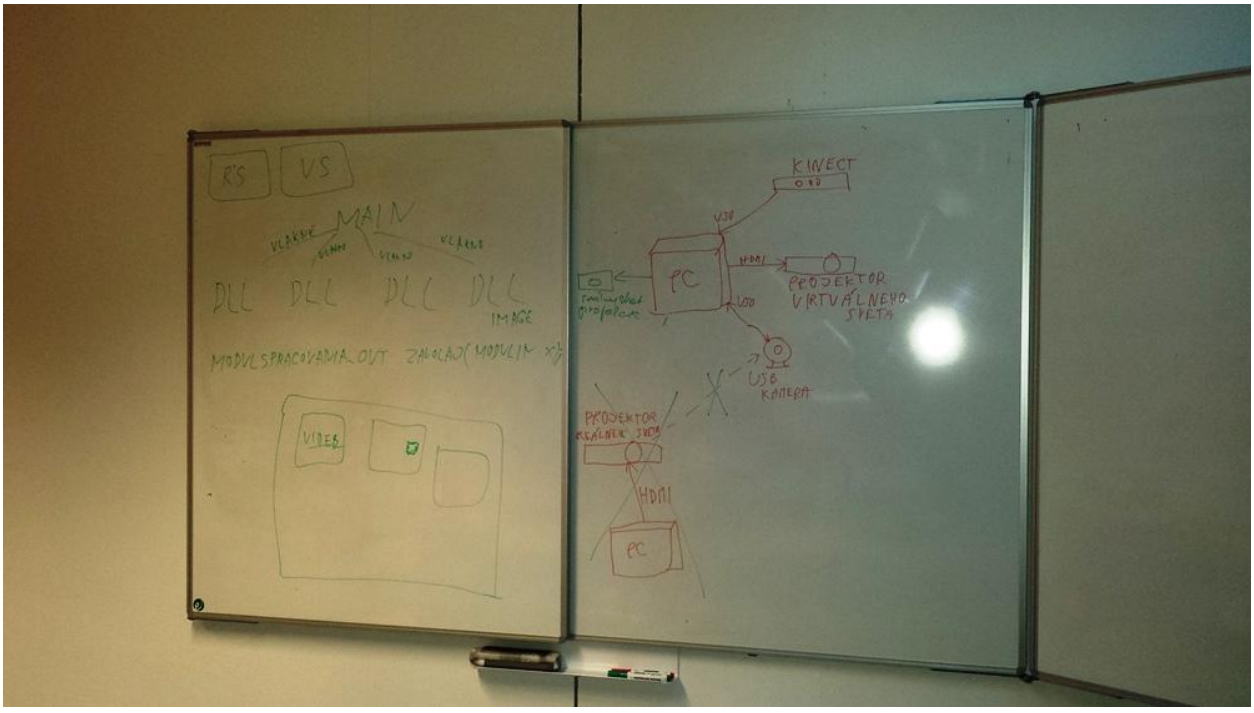
Úlohy do ďalšieho stretnutia:

ID	Pridelené členovi	Opis úlohy
1	Juraj	pozrieť si projekt s Robom

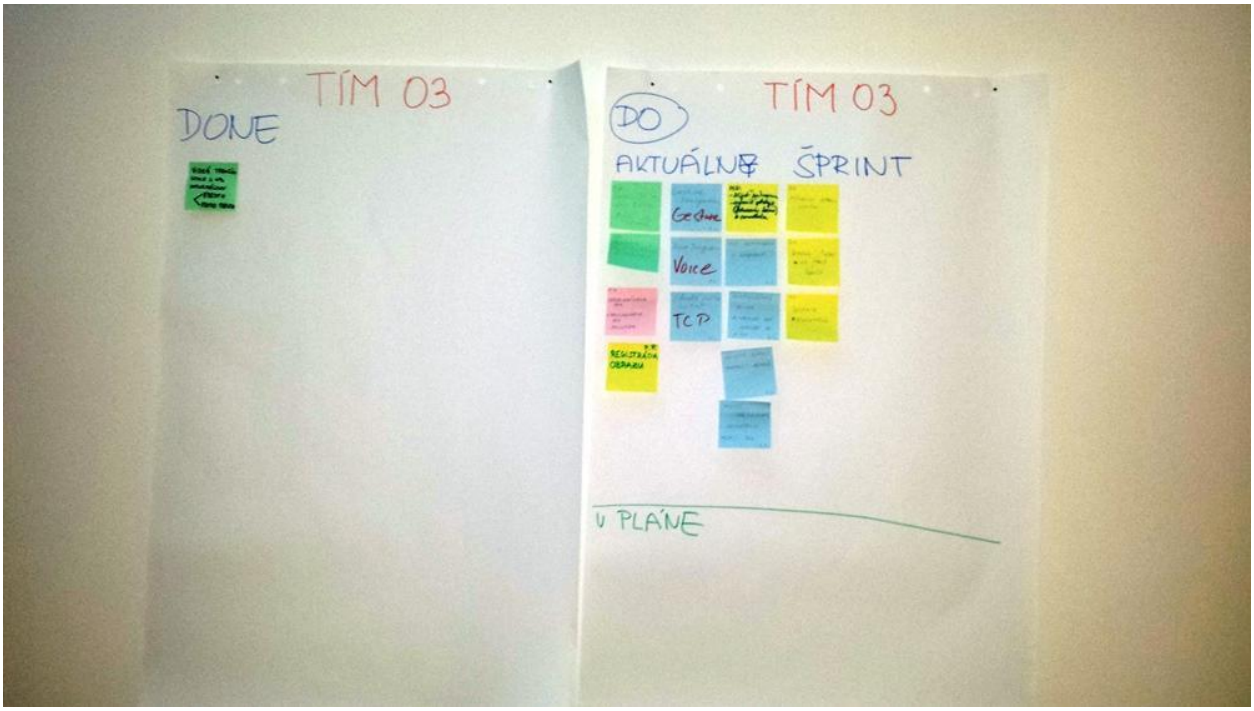
2	Lukáš	pohrať sa s databázou, sql lite
3	Patrik	registrácia obrazu/kalibrácia (lokálne deskriptory) ako priorita, neskôr detekcia
4	Martin	Zastavit za Kapcom a vytvorit prototyp na demonstrovanie vypoctov polohy hlavy v Matlabe
5	Marianna a Peťo	využíť openCV, začať nejakú najlahšiu hru
6	Robo	TCP Lukášovi, gestá a hlas
7	Jakub	zaoberať sa detekciou tváre a spracovávať hĺbkovú informáciu
8	všetci členovia	

Prílohy:

Príloha A. Návrh architektúry



Príloha B. Nová nástenka


TÍM 03
DONE
TÍM 03
DO
AKTUÁLNY ŠPRINT
Gesture
Voice
TCP
REGISTRÁCIA OBRAZU
V PLÁNE

Zápis 5. stretnutia tímu č. 3

Autor zápisu: Bc. Róbert Sabol
Dátum: 28.10.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní: Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hama
Bc. Jakub Mercz
Bc. Marianna Mušinská
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	Martin	Zastavit za Kapcom a vytvorit prototyp na demonstrovanie vypoctov polohy hlavy v Matlabe	Martin za Kapcom nebol, avsak spravil prostredie na testovanie textu + vytvoril v java scripte demo ako prvy navrh vypoctov pre poziciu hlavy
2	Lukáš	pohrať sa s databázou, sql lite	Pripravil SQL lite databazu – datovy model, nahodil do nej data
3	Patrik	registrácia obrazu/kalibrácia (lokálne deskriptory) ako priorita, neskôr detekcia	Robil detekciu objektov, porovnaval objekty . min, max vzidalenosti Spojazdnil kniznicu od Mareka Raceva
4	Robo	TCP Lukášovi, gestá a hlas	Rozpracoval TCP komunikaciu – in progress

5	Peťo	využíť openCV, začať nejakú najlahšiu hru	Spravil jednoduchu hru – kviz, obrazky otazky
6	Jakub	zaoberať sa detekciou tváre a spracovávať hĺbkovú informáciu	in progress
7	Juraj	pozrieť si projekt s Robom	vyriešené
8	Marianna	využíť openCV, začať nejakú najlahšiu hru	Neuspesne, neslo jej to s openGL

Priebeh stretnutia:

- najskôr Lukas predstavil ako spravil hlavny project, ktory bude handlovat vsetky ostatne
- ukazal nam databazu – diagram tried, a vysvetlil ako mame vyuzivat moduly z jeho projektu
- zhodli sme sa,ze budeme pouzivat doxygen
- zhodli sme sa,ze Marianna bude dokumentaristka
- povedali sme si o kalibracii hlavy a toho co vidime v Patrikovej casti projektu
- zhodli sme sa,ze Petova hra na testovanie staci – doplni do nej 3D modely
- optické preloženie projektorov cez seba aby to dávalo zmysel (aby bol obraz správne preložený)

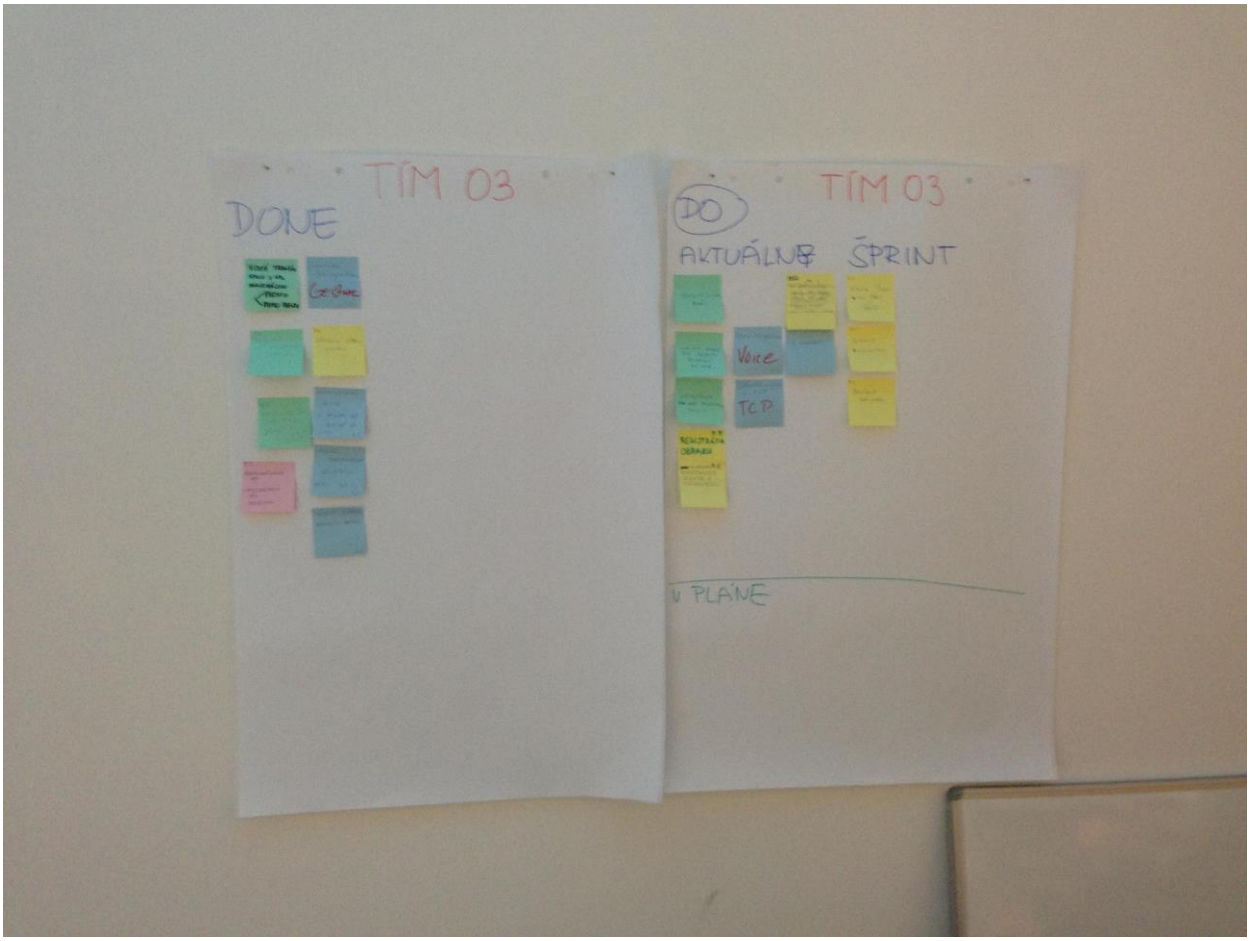
Úlohy do ďalšieho stretnutia:

ID	**Pridelené členovi**	**Opis úlohy**
1	Juraj	Doxygen – pozriet a pripravit jednoduchy dokument pokyny co mame pisat a kam aby to potom cele fungovalo
2	Lukáš	Spristupnit moduly, davat rady ako pouzivat jeho moduly, doxygen, video na snimky
3	Patrik	Finalizuje rozoznavanie objektov, registracia objektov
4	Martin	Vypocitavat poziciu objektu
5	Marianna	TCP modul do C++ projektu
6	Peťo	3D model, lietadlo

7	Robo	TCP komunikacia z Androidu
8	Jakub	Zameria tvar – nejaky bod na tvari, vzdialenost
9	všetci členovia	Najst v lukasovom projekte svoj modul

Prílohy:

Príloha A. Nová nástenka



Zápis 6. stretnutia tímu č. 3

Autor zápisu: Bc. Marianna Mušinská
Dátum: 4.11.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní: Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hamar
Bc. Juraj Jarábek
Bc. Jakub Mercz
Bc. Marianna Mušinská
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	Juraj	Doxygen – pozriet a pripravit jednoduchy dokument pokyny co mame pisat a kam aby to potom cele fungovalo	v procese riešenia
2	Lukáš	Spristupnit moduly, davat rady ako pouzivat jeho moduly, doxigen, video na snimky	vyriešené
3	Patrik	Finalizuje rozoznavanie objektov, registracia objektov	vyriešené
4	Martin	Vypocitavat poziciu objektu	vyriešené
5	Robo	TCP komunikacia z Androidu	v procese riešenia
6	Jakub	tvar – nejaky bod na tvari, vzdialenost	vyriešené
7	Peťo	3D model, lietadlo	v procese riešenia
8	Marianna	TCP modul do C++ projektu	nevyriešené, nevedela

			sa prepojiť s Androidom
9	všetci členovia	Najst v Lukasovom projekte svoj modul	vyriešené

Priebeh stretnutia:

- diskusia o pomenovaní jednotlivých modulov
- diskusia o sfunkčnení git - musí sa sfunkčniť čím skôr
- diskusia o tom, kde budú uložené GPS pozície objektu, ktoré poterbujeme na výpočet pozície
- Lukáš nám ukázal a vysvetlil na konkrétnom prípade ako fungujú moduly
- bolo nám odporučené používať namespace-y
- rozhodli sme sa, že budeme používať debuggery
- Jakubovi nastali menšie komplikácie pri spusťaní svojho programu a bolo mu odporučené používať hľbkovú mapu
- diskusia o hernej logike, a ako by mal vyzerať main
    - výsledok diskusie: main musí byť čisto riadiaci
    - a herna logika môže byť jeden modul
- zhodli sme sa, že musíme používať chybové hlášky, a musíme vedieť odchytiť výnimky
- treba vytvoriť class diagram najlepšie na papier, aby sme vedeli na tom rukou editovať

Úlohy do ďalšieho stretnutia:

1	Juraj	Doxygen - pripraviť užívateľskú príručku (dokument)
2	Lukáš	pripojiť Patrika,
3	Patrik	sfinalizovanie modulu spracovania obrazu a registracie obrazu
4	Martin	sfunkčniť git, testovanie
5	Robo	
6	Jakub	preštudovať používanie hlbkovej mapy

7	Peťo	lietadlo – pohyb pomocou klávesnice
8	Marianna	dokumentácia
9	všetci členovia	prepojiť sa s git, používať Doxygen, pomoc pri dokumentácii jednotlivych modulov, nakresliť class diagram

Prílohy:
Príloha A. Nová nástenka

Zápis 7. stretnutia tímu č. 3

Autor zápisu: Bc. Peter Hamar
Dátum: 4.11.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní: Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hamar
Bc. Juraj Jarábek
Bc. Jakub Mercz
Bc. Marianna Mušinská
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	Juraj	Doxygen - pripraviť užívateľskú príručku (dokument)	vyriešené
2	Lukáš	pripojiť Patrika, vygenerovanie class diagramu, nainštalovanie databázy	vyriešené
3	Patrik	sfinalizovanie modulu spracovania obrazu a registrácie obrazu	vyriešené
4	Martin	sfunkčniť git, testovanie	vyriešené
5	Robo	TCP komunikácia z Androidu (od minula)	v procese riešenia
6	Jakub	preštudovať používanie hĺbkovej mapy	vyriešené
7	Peťo	3D model lietadla – pohyb pomocou klávesnice	vyriešené

8	Marianna	dokumentácia	nevyriešené
9	všetci členovia	prepojiť sa s git, používať Doxygen, pomoc pri dokumentácii jednotlivých modulov, nakresliť class diagram	vyriešené

Priebeh stretnutia:

- Definovali sme presný zoznam features
  -presné rozpoznávanie objektov
  -vykonávanie aplikácie v reálnom čase
  -vypočítanie polohy pre zobrazenie doplňujúcich informácií
  -generovanie grafických prvkov AR
  -ovládanie systému pomocou Android aplikácie
  -ovládanie hry pomocou Android aplikácie
  -poskytovanie 2 aplikácií (hier), ktoré využívajú obohatenú
  -získavanie GPS a časových značiek v reálnom čase pomocou Android aplikácie
  -použitie databázy
  -vypočítanie polohy hlavy z Kinectu
  -integrácia komunikácie a celého systému

- Mariana nevedela napísať dokumentáciu. Keďže tento problém je potrebné urýchlene vyriešiť, tak sme sa dohodli, že dokumentáciu spojí a pripraví na odovzdanie Juraj.
- Git – problém s veľkými knižnicami OpenCV a Boost, vyrieši sa to pomocou premenných ciest.
- Diskusia týkajúca sa výpočtu polohy hlavy voči oknu a kalibrácia.
- Diskusia týkajúca sa umiestnenia Kinectu v aute – hĺbka hlavy.
- Diskusia týkajúca sa hry, kto určí horizont. Je potrebná spolupráca s Patrikom, ktorý určí horizont. Poskytne modulu hry 2 obrázky, jeden s horizontom a jeden klasický. Modul hry ich spojí a účelom bude aby lietadlo letelo nad horizontom. Čím bližšie bude letieť tým viac bodov hráč získa.

Úlohy do ďalšieho stretnutia:

1	Juraj	Dokumentácia – zintegrovať jednotlivé časti a odovzdanie
2	Lukáš	TCP server, kontrola integrácie, uploadnuť na git projekt
3	Patrik	Detekcia horizontu a vylepšovanie spracovania obrazu
4	Martin	Pripraví knižnice pre Git, nainštaluje team foundation server, testovanie svojho modulu
5	Robo	Prerobiť serverovú časť aby nepadala

6	Jakub	Detekcia hlavy cez hĺbkovú informáciu , kalibrácia na reálny svet
7	Peťo	Pripraviť scénu za objektom (video) a skúsiť lietadlo držať nad horizontom
8	Marianna	
9	všetci členovia	Prečítať inštrukcie od Juraja k Doxygenu

Prílohy:

Príloha A. Aktuálna nástenka


TÍM 03
DO
AKTUÁLNY ŠPRINT
ODSTRÁNIŤ
SERVER
VOICE
DOKUMENTÁCIA
DETEKCIA HORIZONTY
U PLÁNE

Zápis 8. stretnutia tímu č. 3

Autor zápisu: Bc. Patrik Polatsek
Dátum: 21.11.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní:	Vedúci:	Ing. Vanda Benešová, PhD.
	Členovia tímu:	Bc. Peter Hamar
		Bc. Jakub Mercz
		Bc. Martin Petluš
		Bc. Patrik Polatsek
		Bc. Róbert Sabol
		Bc. Lukáš Sekerák

Retrospektíva šprintu:

Start:
- ešte lepšia granularita úloh v Redmine
- lepší popis úloh v Redmine
- systematická aktualizácia stavu v Redmine

Keep:
- udržať pracovné nasadenie
- udržať motiváciu členov
- zachovať a vylepšiť teambuilding

Stop:
- zastaviť zmenšovanie tímu

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	Juraj	Dokumentácia – zintegrovať jednotlivé časti a odovzdanie	nevyriešené, delegované na Patrika

2	Lukáš	TCP server, kontrola integrácie, uploadnuť na git projekt	vyriešené
3	Patrik	Detekcia horizontu a vylepšovanie spracovania obrazu	v procese riešenia
4	Martin	Pripraví knižnice pre Git, nainštaluje team foundation server, testovanie svojho modulu	v procese riešenia, ako code review tool zvolený Gerrit
5	Robo	Prerobiť serverovú časť aby nepadala	v procese riešenia
6	Jakub	Detekcia hlavy cez hĺbkovú informáciu, kalibrácia na reálny svet	v procese riešenia
7	Peťo	Pripraviť scénu za objektom (video) a skúsiť lietadlo držať nad horizontom	vyriešené
8	Marianna		
9	všetci členovia	Prečítať inštrukcie od Juraja k Doxygenu	v procese riešenia

Priebeh stretnutia:

- na stretnutí bola riešená kalibrácia s Jakubom pri práci s Kinectom pri zisťovaní polohy hlavy
    - transformovať sústavu kinectu na inú sústavu - získať kalibračné informácie
    - Jakub navrhoval použiť špagátiky na určenie vzdialeností
    - vedúca tímu načrtla použiť 3d target (4 body) - nastaviť si hlavu na začiatku aby používateľ videl body v zástupe
- ďalej sme riešili hru - lietadlo
    - lietadlo bude padať, pokým ho používateľ bude “zdvíhať” do hora
    - jediný ovládaný smer lietadla bude nahor
    - lietadlo nesmie padnúť pod horizont
    - čím bližšie bude k horizontu, tým získa hráč viac bodov
    - hra nebude určená v meste, nakoľko v meste nemusíme vidieť horizont
    - ďalší návrh je zastavenie hry pokiaľ nebude vidieť horizont
    - dohodli sme sa, akým spôsobom bude predávaná inf. o horizonte - bin. obrázok, kt. ukáže regióny oblohy
- semestrový prototyp bude mať obmedzenú funkčnosť, ale hlaná funkcionalita by mala byť hotová
- na konci semestra by mala byť hotová hra - lietadlo a zobrazovanie zákl. turistických inf.

Úlohy do ďalšieho stretnutia:

ID	Pridelené členovi	Opis úlohy
1	všetci	každý by mal vypracovať dummy model (prototyp) svojho modulu
2	Patrik	detekcia horizontu (fake trieda s interface)
3	Martin	dokončiť gerrit, počítač v škole sprevádzkovať, git premenné
4	Jakub	rozpoznanie hlavy ako najbližšieho objektu a detekcia cez plochu tohto objektu, výpočet transf. osí z bodov
5	Robo	pridať gps do hlavnej časti android, voice recognition
6	Lukáš	definovať konštanty, vytvoriť konf. xml, načítať, poslať ho ďalej
7	Peťo	dokončiť zobrazovanie videa, jednoduchý horizont na testovanie, herná logika, zobrazovanie textu a skore

Prílohy:

Príloha A. Aktuálna nástenka

Zápis 9. stretnutia tímu č. 3

Autor zápisu: Bc. Martin Petluš
Dátum: 28.11.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní: Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hamar
Bc. Jakub Mercz
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	Stav
1	všetci	každý by mal vypracovať dummy model (prototyp) svojho modulu	vyriešené
2	Patrik	detekcia horizontu (fake trieda s interface)	vyriešené
3	Martin	dokončiť gerrit, počítač v škole sprevádzkovať, git premenné	vyriešené, premenné sú ešte v procese riešnia (čakáme sa na integráciu modulov)
4	Jakub	rozpoznanie hlavy ako najbližšieho objektu a detekcia cez plochu tohto objektu, výpočet transf. osí z bodov	Jakub navrhuje dokončiť konfiguráciu bodov a detekciu tváre v priestore odložiť na ďalší semester, zisťoval transformáciu sústav
5	Robo	pridať gps do hlavnej časti android, voice recognition	prdanie gps do hlavne časti je vyriešené, voice recognition je v procese riešenia

6	Lukáš	definovať konštanty, vytvoriť conf.xml, načítať, poslať ho ďalej	vytvoriť conf.xml, vyriešené, spravil fake android gps modul, implementoval triedu na čas, spravil návod na integráciu modulu do aplikácie
7	Peťo	dokončiť zobrazovanie videa, jednoduchý horizont na testovanie, herná logika, zobrazovanie textu a skore	mal problém s OpenCV v inom vlákne pri zobrazovaní videa na pozadí - v procese riešenia, spravil úvodnú obrazovku hry, ukončovaciu obrazovku, hry, zapracovanie fyziky, jednoduchý horizont na testovanie v procese riešenia, zobrazovanie textu a skóre vyriešené

Priebeh stretnutia:

- na konci semestra bude aplikácia v jednoduchšej forme (bude fungovať na jednom videu)
- diskutovali sme, kedy je vhodné spúštať kalibráciu modulov (vždy sa spýtať pri spúštaní aplikácie alebo externe a nikdy sa na to pri súštaní aplikácie nepýtať)
- bavili sme sa o probléme s textúrou lietadla
- treba, aby všetci používali rovnakú verziu OpenCV a iných knižníc

Úlohy do ďalšieho stretnutia:

ID	Pridelené členovi	Opis úlohy
1	všetci	integrácia modulov, pridať do gdocu aké knižnice používa každého modul
2	Patrik	ukončiť celú funkcionalitu detekcie horizontu, otestovať prípadne upraviť modul spracovania na videu
3	Martin	študovať android, git dokončiť premenné
4	Jakub	nájdenie bodov pre transformáciu medzi súradnícovými sústavami, celková integrácia inicializácie modulu
5	Robo	prispôsobenie vzhľadu gps modulu (logger) ku aplikácií, vytvoriť

		záložné ovládanie
6	Lukáš	celková integrácia modulov
7	Peťo	vytvoriť počítanie skóre, ukladanie skóre do databázay, zobrazenie reálneho horizontu a upravenie hry, aby brala v úvahu horizont

Prílohy:

Príloha A. Aktuálna nástenka

Zápis 10. stretnutia tímu č. 3

Autor zápisu: Bc. Jakub Mercz
Dátum: 05.12.2013
Miestnosť: Laboratórium počítačového videnia a grafiky

Prítomní: Vedúci: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Peter Hamar
Bc. Jakub Mercz
Bc. Martin Petluš
Bc. Patrik Polatsek
Bc. Róbert Sabol
Bc. Lukáš Sekerák

Stav zadaných úloh z minulého stretnutia:

ID	Pridelené členovi	Opis úlohy	stav
1	všetci	integrácia modulov, pridať do gdocu aké knižnice používa každého modul	V procese riešenia
2	Patrik	ukončiť celú funkcionalitu detekcie horizontu, otestovať prípadne upraviť modul spracovania na videu	Detekcia horizontu – vyriešené, Modul spracovania neotestovaný
3	Martin	študovať android, git dokončiť premenné	V procese riešenia
4	Jakub	nájdenie bodov pre transformáciu medzi súradnícovými sústavami, celková integrácia inicializácie modulu	Detekcia bodov z pohľadu kinectu – vyriešené, v reálnom svete – v procese riešenia
5	Robo	prispôsobenie vzhľadu gps modulu (logger) ku aplikácií, vytvoriť záložné ovládanie	Vyriešené
6	Lukáš	celková integrácia modulov	vyriešené
7	Peťo	vytvoriť počítanie skóre, ukladanie skóre do databázy, zobrazenie reálneho horizontu a	Úlohy ohľadom skóre odložené, zvyšok vyriešený

		upravenie hry, aby brala v úvahu horizont	

Priebeh stretnutia:

- Preberali sme časy a povinnosti na prezentáciu manažmentu v tímovom projekte

Úlohy do ďalšieho stretnutia:

ID	Pridelené členovi	Opis úlohy
1	všetci	Príprava prezentácie, finálne review kódu
2	Patrik	Príprava dokumentácie, otestovanie modulu spracovania
3	Martin	Inštalácia prostriedkov na počítači v učebni, vytvorenie repozitára pre android modul
4	Jakub	Integrácia modulu, zisťovanie bodov z reálneho sveta
5	Robo	Testovanie ovládania, pripravenie inštalácie modulu
6	Lukáš	Otestovanie modulov
7	Peťo	Ovládanie hry

6. Manažment v tímovom projekte

Táto kapitola obsahuje stručný popis jednotlivých manažérskych oblastí v tímovom projekte Carlos.

6.1 Manažment komunikácie

Komunikácia v tíme prebieha hlavne počas spoločných stretnutí aj s vedúcou celého projektu. Tie prebiehajú spravidla podľa nasledovného plánu:
1. Prebratie výsledkov úloh posledného týždňa
2. Diskusia o ďalšom smerovaní projektu a vzniknutých problémoch
3. Určenie úloh na ďalší ťýždeň

Neformálna komunikácia prebieha na stránke facebook.com, kde bola vytvorená skupina na účely tohto projektu. V rámci tejto skupiny prebieha všetká potrebná komunikácia medzi členmi mimo stretnutí tímu. Výhodou tohoto komunikačného kanálu je rýchly prístup ku všetkým členom tímu, keďže túto sociálnu sieť denne navštevuje každý člen tímu. Taktiež umožňuje viditeľnosť ktorí členovia daný príspevok už videli a v prípade neprečítania si tohto príspevku viac ako 24 niktorým členom, bol autor príspevku povinný oznámiť danú skutočnosť a prostredníctvom emailu. Emailove adresy všetkých členov tímu boli zozbierané hneď na začiatku projektu a spočiatku boli využívané ako primárny komunikačný kanál, neskôr sa stali sekundárnym.

Pre komunikáciu s vedúcou projektu mimo stretnutí tímu bolo založené emailové konto s adresou team03.1314@gmail.com, ktorého adresa bola poskytnuté vedúcej projektu. Manažér komunikácie sa stará o okamžité posunutie akýchkoľvek správ na toto konto do skupiny na stránke facebook.com slúžiacej ako primárny informačný kanál. V prípade potreby odpovede alebo správy inicializovanej tímom, mimo stretnutí tímu, je touto úlohou poverený vedúci tímu, ktorý mal na starosti formálnu komunikáciu s vedúcou projektu.

6.2 Manažment plánovania

Keďže bol celý projekt hneď na začiatku rozdelený na jednotlivé moduly, plánovanie prebieha jednak pre každý modul zvlášť ako aj pre celý projekt ako celok. Počas prvých dvoch stretnutí bol vytvorený globálny cieľ a cieľ na zimný semester, na základe ktorého každý z členov tímu vytvoril cieľ na zimný semester pre svoj konkrétny, jemu pridelený modul.

Plánovanie počas semestra prebieha počas spoločných stretnutí tímu, opísaných v kapitole venovanej manažmentu komunikácie. Tu sa na základe diskusie o problémoch a ďalšom smerovaní projektu vytvoria úlohy. V rámci prideľovania úloh na konci stretnutí sú potom úlohy, buď prenesené z predchádzajúceho šprintu, novo vytvorené na stretnutí alebo už v product backlogu, po konzultácii s vedúcou projektu zaradené buď do product backlogu na neskoršie riešenie alebo pridelené členovi so zodpovednosťou za daný modul alebo manažérsku oblasť. V prípade že úloha nespadala priamo pod žiadny modul ani manažérsku oblasť, bola pridelená členovi s dostatočnými kompetenciami a s najnižším časovým plánom v najbližšom šprinte.

Samotný product backlog nášho tímu je z dôvodu krátkych šprintov založených na iteratívnom vývoji väčšinou prázdny a zapĺňa sa len zriedkavo. Jeho napĺňanie nastáva hlavne pri nesplnení úloh kvôli neočakávaným problémom alebo pri odsúvaní úloh na neskoršie obdobie kvôli identifikácii nových úloh s vyššou prioritou.

6.3 Manažment rizík

V manažmente rizík sme riadili tieto riziká:

- Odchod členov z tímu – Toto riziko sme sa snažili minimalizovať tak, že všetky dôležité úlohy sme nedávali na starosti jednotlivcovi, ale dvojici. Aj napriek tomu, že na úlohe priamo pracovala len jedna osoba, druhá ho mala kontrolovať a vypomáhať mu. Mala byť zainteresovaná v rámci úlohy tak, že v každom momente ho mohla nahradiť. Toto riziko je v prostredí školy veľmi vysoké, najmä keď 2 členovia tímu sú prvýkrát v novom prostredí.
- Nedostupnosť objednaného hardwaru – Náš projekt Carlos, vyžaduje pre fungovanie špeciálny hardvér. Tento hardware bol objednaný na začiatku projektu a počas neho mal byť dodaný. Riziko mohlo nastať v prípade nedodania tohto hardwaru. Toto riziko sme odstránili sekundárnym hardvérom, ktorý sme provizórne poskladali a zabezpečili z vlastných zdrojov.
- Chyba pri integrácii modulov – Každý člen tímu pracoval na určitom module, v rámci prototypu malo vzniknúť 8 modulov. V každom module mohla vzniknúť chyba v zdrojovom kóde. Toto riziko je ošetrené v samostatnom bode. Zároveň pri integrácii mohol byť niektorý modul dodaný neskôr. Riziku sme predchádzali tým, že manažér rizík mal na starosti pravidelne kontrolovať dodanie modulov a ich častí.
- Vznik chyby v zdrojovom kóde – Toto riziko sme čiastočne chránili review kódom.

Na manažovanie rizík v našom projekte nevyužívame žiadne nástroje. Riziká máme formálne spísané v metodike rizík. Manažment rizík chápeme ako proces, možným rizikám sa venujeme aj pri každom stretnutí, kde ich prípadne riešime.

6.4 Manažment kvality

Počas prvej fázy projektu sme testovali zdrojový kód individuálne. Každý testoval svoj modul vo vlastnej réžii a mohol si určiť aké testy chce vytvárať, avšak iba pre jeho vlastné potreby. V druhej fáze tímového projektu (po prepojení modulov) plánujeme vytvárať unit testy v prostredí Visual Studio. Tieto testy budeme písať podľa metodiky písania jednotkových (unit) testov.

Na prehliadky kódov používame systém Gerrit. Tieto prehliadky už budú prebiehať pravidelne, raz na konci šprintu. V rámci tímu sme si jasne definovali dvojice, ktoré medzi sebou budú vykonávať prehliadky kódu. Konkrétne:

Patrik Polatcek - Martin Petluš
Jakub Mercz - Lukáš Sekerák
Peter Hamar - Róbert Sabol

Refaktorizáciu kódu vykonávame v tom momente, keď je ďalšie rozšírenie problematické, alebo neefektívne. Každý člen tímu je zodpovedný za určity modul. Čiže v prípade, ak je potrebná refaktorizácia jeho modulu, informuje tím na stretnutí. Následne si túto refaktorizáciu naplánuje do ďalšieho šprintu.

Upravovať kód iných členov tímu môžme iba po vzájomnej dohode - nie svojvoľne, alebo bez ich vedomia. Keďže však máme každý vlastný modul, k takýmto problémom nedochádza pravidelne. Doposiaľ sa takéto problémy vyriešili vždy kolaboratívne a bez väčších problémov.

6.5 Manažment podpory vývoja a integrácie

Na správu zdrojových kódov využívame nástroj Git. Git je nainštalovaný na serveri a všetci členovia svoje zdrojové kódy vždy aktualizujú tak, aby bola na serveri vždy aktuálna verzia kódu. Každý člen tímu má správne nastavené svoje konto, aby sa vždy dali idenifikovať jeho commity.
Na prístup k Gitu na serveri využívame privátny kľúč. Každý člen tímu je povinný k svojmu commitu pridať aj krátku správu s popisom zmeny.

Na manažovanie commitov ostatnými členmi tímu používame nástroj Gerrit. Členovia tímu pristupujú ku commitom cez webové rozhranie, kde majú možnosť okomentovať commit a

ohodnotiť ho. Minimálne dvaja členovia tímu musia ohodnotiť commit kladne, tak aby sa vykonaná zmena commitom uložila priamo do repozitára projektu.

Pri manažmente podory vývoja a integrácie je potrebné sa vždy starať aj o server, na ktorom je uložený repozitár pre projekt a kde beží aj webová stránka tímu, Gerrit a Redmine. Server pravidelne aktualizujeme, tak isto sa vždy staráme aj o aktuálnosť stránky. Na stránke sú vždy aktualizované zápisnice zo stretnutí, projektové dokumentácie a iné veci priamo súvisiace s tímom alebo projektom.

6.6 Manažment monitorovania

Na monitorovanie práce na projekte je v našom tíme využívaný webový nástroj na manažment projektov - Redmine. Pri začatí riešenia úlohy označí svoju úlohu člen tímu za rozpracovanú. Pri každom postupe v riešení úlohy, člen tímu zapíše pomocou tohto nástroja k svojej riešenej úlohe percentuálny stav vyriešenia. Rovnako každý člen tímu zaznamená cez Redmine počet hodín, ktoré strávil nad riešením úlohy a uvedie stručný popis aktivity. Po vyriešení úlohy člen tímu označí úlohu v Redmine ako vyriešenú. Takouto evidenciou môže manažér monitorovania sledovať priebeh všetkých pridelených úloh.

Manažér monitorovania je zodpovedný za evidenciu a uzatvorenie úloh v Redmine. Okrem toho kontroluje správnosť reportovania priebehu úloh. Stav priebehu riešenia úloh reprezentuje Ganttov graf v Redmine. Na základe porovnania doteraz stráveného času nad vykonávanou úlohou s odhadnutým časom potrebným na dokončenie úlohy je možné vyhodnotiť, v akom stave sa úloha nachádza. Priebeh a stav riešenia úloh je podrobne kontrolovaný a zhodnotený každý týždeň na tímových stretnutiach, kde každý člen odprezentuje aktuálny stav v riešení úloh a prípadné problémy, ktoré vznikli pri riešení úlohy.

6.7 Manažment dokumentácie

Na tvorbu dokumentácie, zápisníc a iných podobných textových dokumentov sa v našom tíme využívajú google dokumenty. Tvorbu zápisníc ma na starosti vždy iný člen tímu, ktorý štruktúrovane zapisuje stav úloh z minulého stretnutia, priebeh stretnutia a v závere prehľadne zaznačí nové úlohy pre všetkých členov. Projektová dokumentácia sa tvorí priebežne, kde opis modulov systému má na starosti člen tímu, ktorému je tento modul pridelený.

Zdrojový kód je dokumentovaný pomocou nástroja Doxygen a komentovaný podľa predpísaných pravidiel. Webové sídlo je aktualizované pravidelne (minimálne raz týždenne), keď sa na stránku pridávajú zápisnice zo stretnutí poprípade iné dôležité dokumenty.

7. Retrospektíva úloh

V letnom semestri tímového projektu boli záznamy zo stretnutí nahradené pravidelným exportom úloh zo systému Redmine, ktoré sú dostupné v elektronickej podobe. V tlačenej podobe uvádzame len štruktúrované retrospektívy šprintov, ktoré vznikli na konci každého šprintu.

Retrospektíva 7. šprintu

Keep:

- pracovne tempo
- kolektivny duch

Stop:

- ukoncit pridavanie novej funkcionality

Start:

- intenzivne testovanie
- experimenty

Retrospektíva 8. šprintu

Keep:

- testovanie
- experimenty

Retrospektíva 9. šprintu

Keep:

- ukončiť projekt

Retrospektíva 10. šprintu

Keep:

- ukončiť projekt

Start:

- testovanie 3.stranou
- pripraviť prezentáciu a záverečné dokumentácie k projektu

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
Fakulta informatiky a informačných technológií
Softvérové inžinierstvo a Informačné systémy

PREBERACÍ PROTOKOL

Názov tímu: Tím č. 3 (Carlos)
Tímový projekt: Zábavný systém pre spolucestujúcich v automobile (AUTO)

Vedúci tímu: Ing. Vanda Benešová, PhD.
Členovia tímu: Bc. Patrik Polatsek
Bc. Martin Petluš
Bc. Peter Hamar
Bc. Róbert Sabol
Bc. Jakub Mercz
Bc. Lukáš Sekerák

Počet strán: .....

Ing. Vanda Benešová, PhD., týmto potvrdzuje prevzatie Projektovej dokumentácie a Dokumentácie riadenia projektu.

Podpis: .....

Bratislava, dňa …….