

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta informatiky a informačných technológií

VIZUALIZÁCIA INFORMÁCIÍ V OBOHATENEJ REALITE

Dokumentácia k inžinierskemu dielu

Členovia tímu č.4:

BC. MATÚŠ CIMMERMAN

BC. IRINA DYOMINA

BC. MICHAL FAŠÁNEK

BC. JAROSLAV GAZDÍK

BC. DENIS ILLÉS

BC. FILIP JURČACKO

BC. DALIBOR MÉSZÁROS

Predmet:

Tímový projekt I

Vedúci:

Ing. Peter Kapec, PhD.

Akademický rok:

2015/2016

Obsah

Úvod.....	3
1 Globálne ciele	4
1.1 Globálne ciele pre zimný semester.....	4
1.2 Globálne ciele pre letný semester	5
2 Celkový pohľad na systém.....	7
2.1 Architektúra.....	7
2.2 Dátový model.....	8
2.3 Diagram tried	9
3 UML diagramy	10
4 Prehľad modulov	21
4.1 Implementované moduly	21
4.2 Podporované nové zariadenia.....	22
5 Moduly a funkcionality systému.....	23
5.1 Mouse3d.....	23
5.1.1 Analýza modulu Mouse3d	23
5.1.2 Návrh modulu Mouse3d.....	23
5.1.3 Implementácia.....	25
5.2 Stereoskopické 3D s AR okuliarmi Vuzix STAR 1200XL	25
5.2.1 Analýza.....	25
5.2.2 Návrh	26
5.2.3 Implementácia	27
5.3 Stereoskopické 3D s okuliarmi Nvidia 3D Vision Pro	28
5.3.1 Analýza.....	28
5.3.2 Návrh	29
5.3.3 Implementácia.....	29
5.4 Multiview „Wall“	29
5.4.1 Analýza.....	29
5.4.2 Návrh	30
5.4.3 Implementácia.....	31
5.5 Refaktorovanie modulu Leap	32
5.6 Dynamická zmena pozadia	33

Úvod

Inžinierske dielo slúži ako technická dokumentácia k predmetu Tímový projekt na fakulte informatika a informačných technológií Slovenskej technickej univerzity v Bratislave. Opisuje štruktúru a postup rozširovania existujúceho projektu na tému Vizualizácia informácií v obohatenej realite.

V prvej kapitole sa opisujú globálne ciele pre zimný a letný semester, ktoré sa budeme za nasledovné mesiace snažiť dosiahnuť. Okrem zakomponovania nových zariadení sme medzi ciele zaradili aj údržbu a opravy chýb v projekte. Na projekte robilo súčasne viacero študentov a pri spájaní došlo k menším nedostatkom, ktoré spôsobili nefunkčnosť istých funkcií. Na úvod sme si zaumienili riešiť tieto nedostatky.

V druhej kapitole je umiestnený aj celkový pohľad na súčasný systém, z ktorého vychádzame.

V tretej kapitole sú zverejnené zoznamy modulov implementovaných a plánovaných. K modulom prislúcha aj krátky stručný opis.

V štvrtej kapitole sa nachádzajú opisy modulov, ktoré implementujeme do systému. Podrobný priebeh analýzy, návrhu, implementácie a testovania.

V piatej kapitole je podrobný inštalačný manuál. Základ manuálu bol prebratý z predošlých rokov, avšak postupným aktualizovaním a rozširovaním projektu musel byť manuál tiež upravený. Inštalačný návod bol preformátovaný, kvôli prehľadnosti a rozšírené a viacero krokov, ktoré sme spozorovali pri inštalovaní softvéru jednotlivými členmi. Návod je doplnený o obrázkové postupy pre ľahšiu orientáciu.

1 Globálne ciele

1.1 Globálne ciele pre zimný semester

V prvom semestri sa budeme zaoberať analýzou problémovej oblasti. Jednotliví členovia sa musia zoznámiť s problematikou vizualizácie informácií. Na predmete tímový projekt sa bude vylepšovať a rozširovať existujúci rozsiahli projekt, ktorý má implementovanú funkcionálnosť niekoľko hardvérových zariadení. V projekte je pre každé zariadenie vytvorený modul. Naším cieľom bude zozname modulov rozšíriť.

Zimný semester máme na pláne prevažne úpravu zdrojového kódu. Z dôvodu spájania bakalárskych a diplomových prác došlo k výskytu závažných chýb a nedostatkov. Našou prioritou je dostať projekt do stabilného stavu, aby sme mohli neskôr pracovať na implementovaní nových modulov do funkčného systému. Pre náš projekt je dôležité pridanie hardvérových zariadení na efektívnejšie ovládanie grafov vo virtuálnom svete – 3D myš. Hlavnou výzvou je projekcia grafov na okuliare určené pre zobrazenie rozšírenej reality (AR okuliare).

Po odladení projektu sa začne pracovať na implementácii 3D myšky. Špeciálna myš poskytuje používateľovi 6 úrovní voľnosti (pohybom od seba/k sebe simuluje zoom, pohyb do strán znamená posuv doľava/doprava, stlačenie alebo vytiahnutie spôsobuje posuv hore/dole a otáčanie a naklápanie spôsobuje rotáciu). Ovládanie nahradí potrebu náročnej manipulácie s grafmi prostredníctvom klávesnice a klasickej myšky.

Medzi ďalšie dostupné zariadenia, ktoré máme na pláne implementovať, sú stereoskopické okuliare. Použitím okuliarov a vhodného 3D monitora budeme schopní vnímať grafy, ktoré boli načítané zo súborov alebo vytvorené analýzou volaných funkcií skriptovacieho jazyka Lua ako trojdimenzionálne objekty.

Okrem pripájania nových komponentov máme na pláne rozšíriť aj prezentačnú časť projektu. Prvotnou výzvou je zobrazenie softvéru klasicky na viacerých monitoroch v režime „extended“. Následne vyriešiť paralelné spracovanie dát pre individuálne scény a tak vytvoriť „virtuálnu jaskyňu“ (angl. CAVE).

Najnáročnejšou úlohou bude správne nasadiť funkcionality AR okuliarov. Naša vízia je zobrazenie grafu na objekt, ktorý drží používateľ v ruke a je označený špeciálnou značkou. Otáčaním objektu sa má otáčať aj samotný graf. Špeciálna značka sa nasníma pomocou modulu Aruco, ktorý je už súčasťou projektu a plne funkčný.

Po úspešnom aplikovaní predošlých modulov sa budeme zaoberať ďalšej problematike vizualizácie štruktúry softvéru. Podobne ako pri zobrazovaní grafu volaní, ktorý čerpá dáta zo zdrojových kódov jazyka Lua, sa môže vizualizovať aj štruktúra softvéru na základe modulov. Cieľom je zobraziť jednotlivé moduly ako grafické objekty, ktoré sú vhodným spôsobom prepojené. Uvedomujúc si, že jednotlivé moduly môžu v sebe zahŕňať ďalšie „podmoduly“, treba navrhnúť vnorené objekty, ktoré majú jednoznačne odzrkadľovať štruktúru softvéru.

Vyriešením predošlej problematiky sa môže vyskúšať prepojenie spomenutých geografických objektov a grafom volaní, ktorý sa môže vytvoriť z podobných modulov. Výsledkom má byť komplexná grafová štruktúra znázorňujúca jednotlivé moduly, graf volaní v moduloch a prepojenia medzi modulmi. Komplikované abstraktné dátové štruktúry používané v komplexných heterogénnych architektúrach zvyčajne nie sú prehľadné a so zvyšovaním komplexnosti sa prehľad iba zhoršuje. Touto metódou by bolo možné zobraziť vnútornú štruktúru softvéru a odhaliť relevantné dáta, prípadne nekorektné operácie s nimi.

1.2 Globálne ciele pre letný semester

Zimný semester slúžil skôr na analýzu súčasného stavu a na korekcie zdrojových kódov. Stihli sme nemalú časť projektu refaktorovať a opraviť vyskytnuté chyby. V letnom semestri sa budeme venovať implementácii nových modulov a funkcionalít, ktoré zvýšia existujúcemu projektu 3DSoftViz hodnotu.

Naším cieľom je stihnúť naprogramovať rozrobenú 3D myš na najpoužívanejšie platformy. Myška má byť naimplementovaná ako samostatný modul systému, čo najmenej závislý od hlavnej časti programu.

Medzi ďalšie prvky, ktoré sa pokúsime nasadiť patria okuliare rozšírenej reality Vuzix. Okuliare slúžia ako zobrazovacie zariadenie s fixovanou kamerou, ktoré sú schopné sledovať pohyby používateľa. Našou úlohou je nájsť ideálne riešenie ako zakomponovať Vuzix okuliare

do projektu. Popri rozšírenej reality sa budeme venovať aj 3D zobrazeniu. Máme k dispozícii Nvidia 3D Vision Pro okuliare, ktoré ponúkajú 3D stereoskopické zobrazenie. Pomocou tohto zariadenia budeme môcť vidieť vizualizované grafy v 3D.

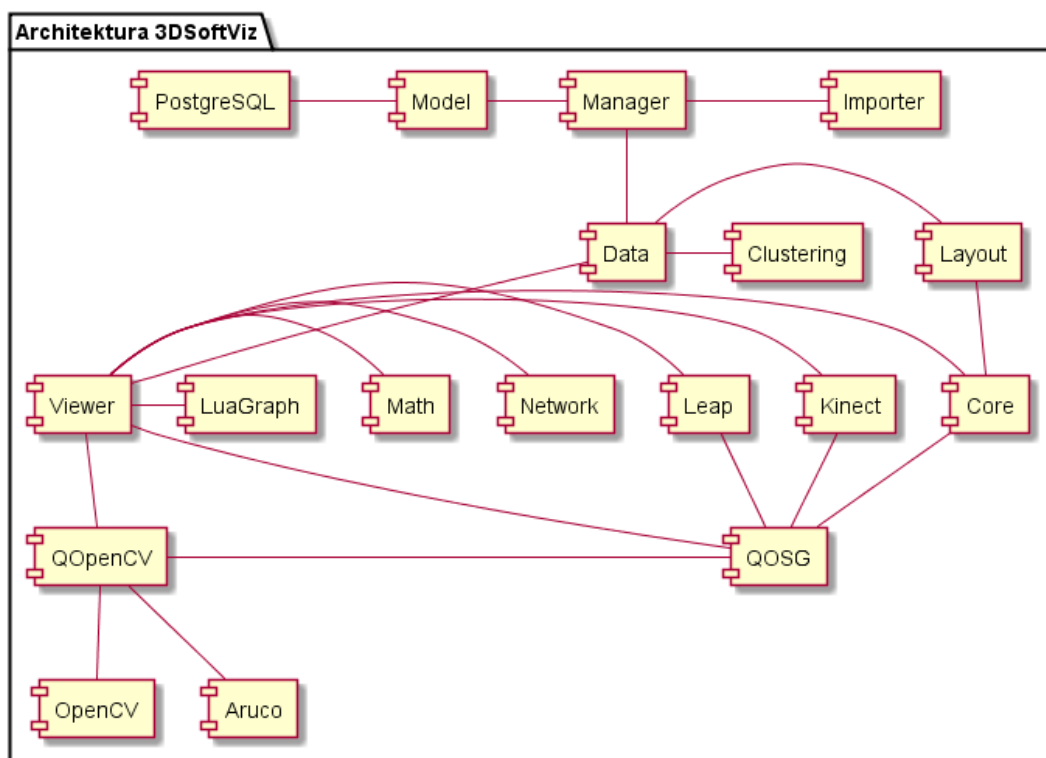
Počas semestra sa bude pracovať aj na rozšírenom zobrazení typu WALL, keďže konfigurácia pre typ CAVE je momentálne nemožná. Ideou je zobrazit' projekt na viacerých monitoroch/projektoroch. A v poslednom rade sa pozrieme na leap senzor. Okrem refaktorovania existujúceho modulu sa budeme snažiť doplniť ďalšie funkcionality použitím práve tohto snímača.

2 Celkový pohľad na systém

V tejto kapitole sa nachádza celkový pohľad na systém z hľadiska architektúry, dátového modelu, diagramu tried a jednotlivých modulov projektu.

2.1 Architektúra

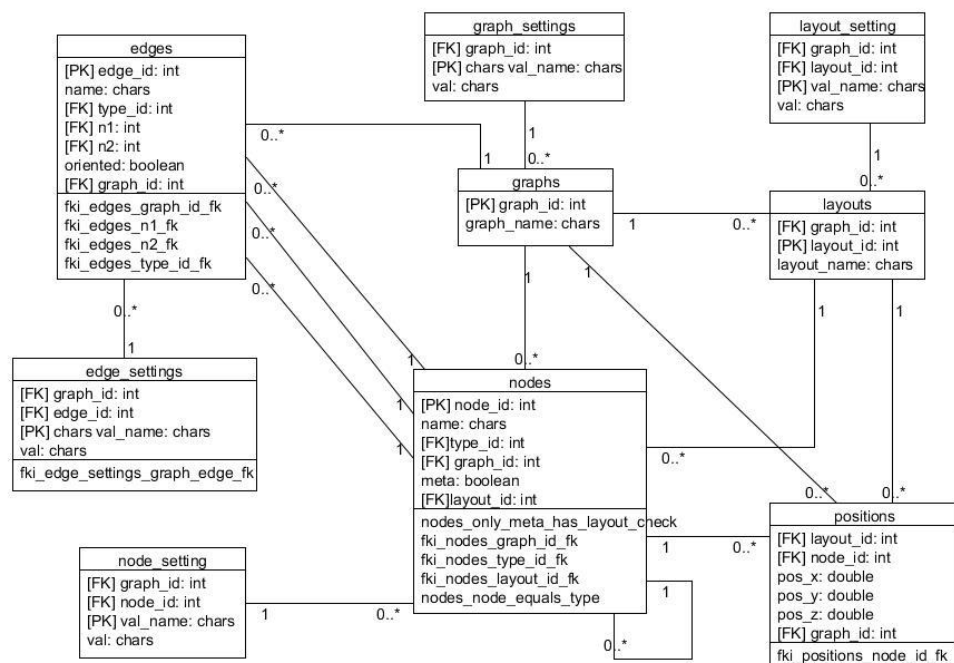
Na Obrázku 1 je zobrazená štruktúra projektu, ktorá bola štartovacím bodom pre náš projekt. Pracujeme na rozšírení projektu o nové moduly ako je 3D myš, pomocou ktorej sa zefektívni ovládanie programu, teda manipulácia s grafmi vo virtuálnom priestore. Následne sa pracovalo na úpravách existujúcich komponentoch, aby sme pripravili softvér na implementáciu nových častí. Vyriešili sme chyby, ktoré nedovoľovali využívanie celého potenciálu softvéru 3DSoftViz.



Obrázok 1 - architektúra softvéru

2.2 Dátový model

Dátový model databázy sme prebrali od predchádzajúcich prác, zobrazený je na Obrázku 2.

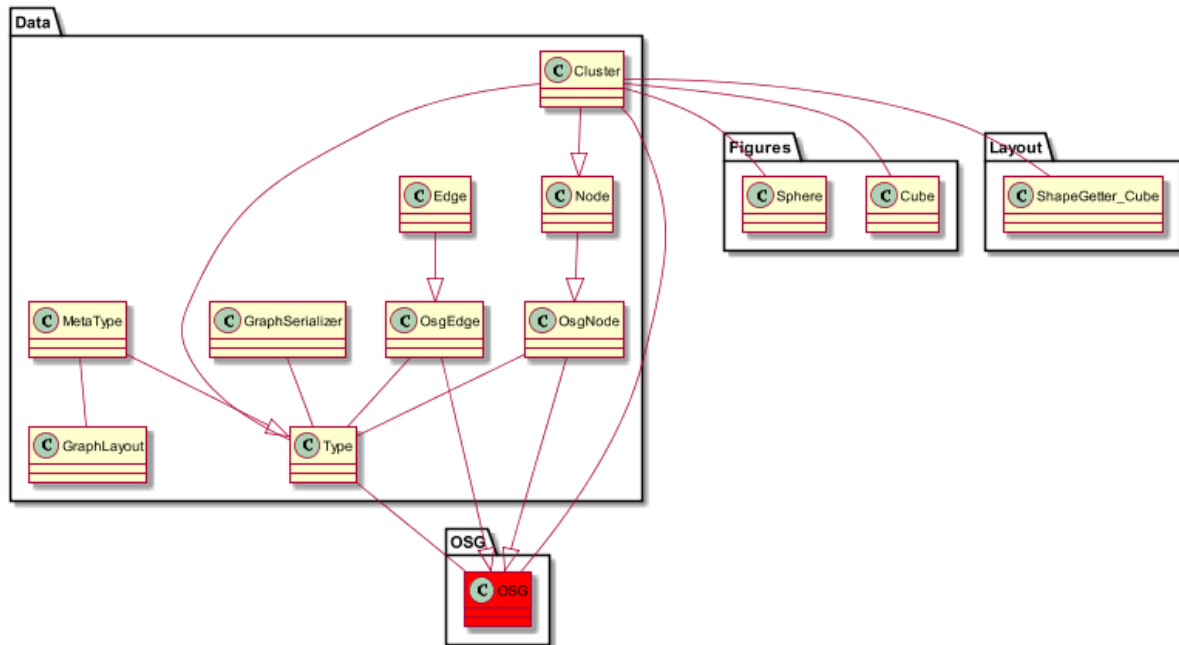


Obrázok 2 - dátový model

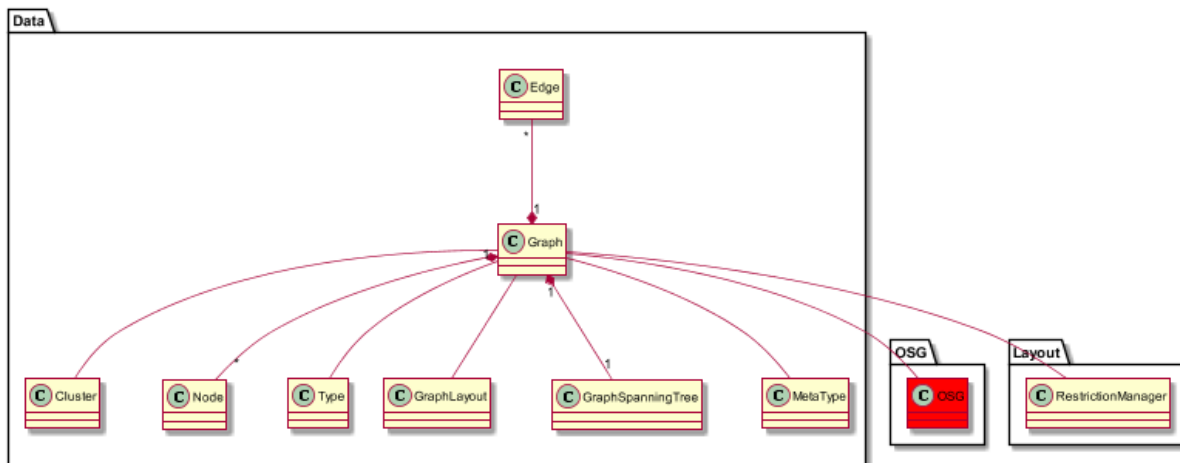
Stručný opis vybraných entít:

- **Graphs** – predstavuje jednotlivé záznamy grafov
- **Layouts**– obsahuje layout pre daný graf. Layout predstavuje konkrétne rozloženie množiny uzlov a hrán grafu v priestore.
- **Nodes**– obsahuje uzly a typy v grafe. Graf je vytvorený z množiny uzlov poprepájaných hranami, pričom každý uzol má svoj typ.
- **Positions** – obsahuje súradnice uzlov v priestore, pričom pozostáva z troch súradníc, ktoré sú naviazané na konkrétny uzol a layout.
- **Edges** – má uložené hrany spájajúce uzly v grafe. Hrana môže byť orientovaná neorientovaná a má začiatkový a koncový uzol a tiež typ.

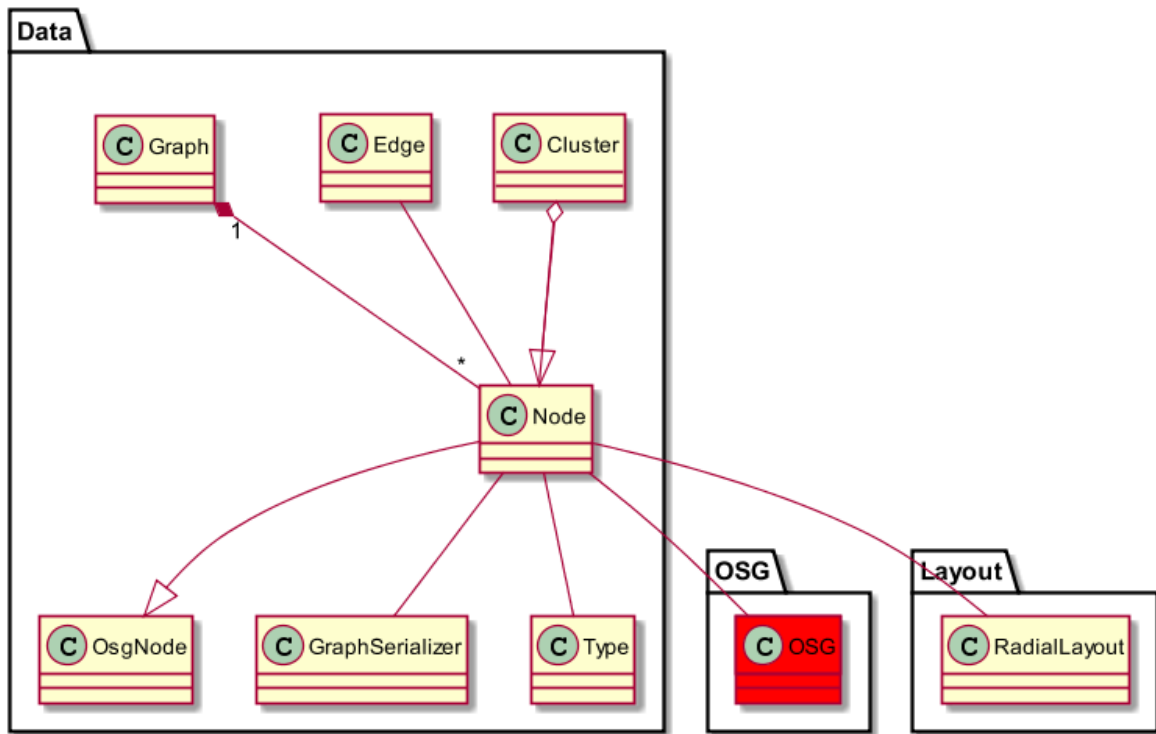
3 UML diagramy



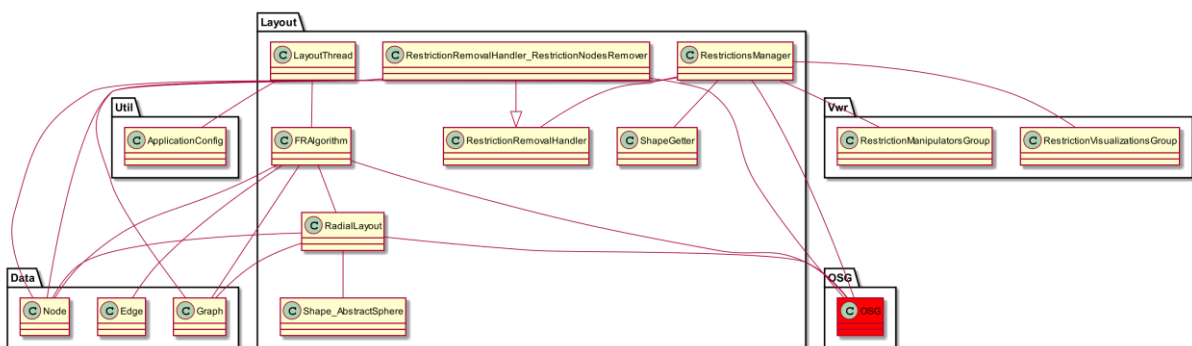
Obrázok 4 – Cluster



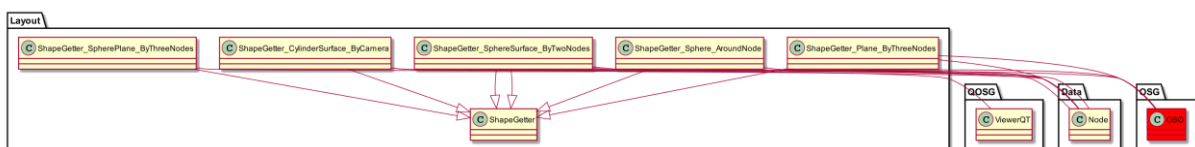
Obrázok 5 – Edge



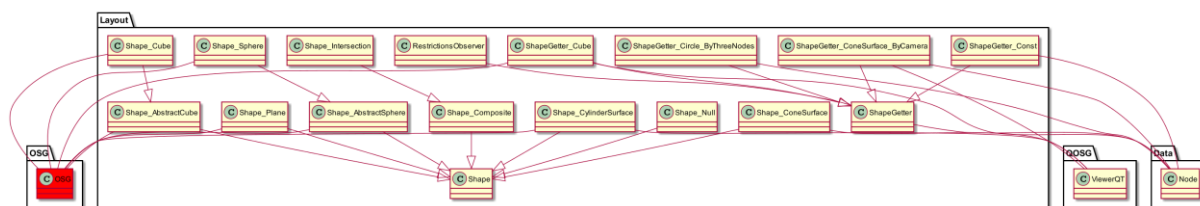
Obrázok 6 - Node



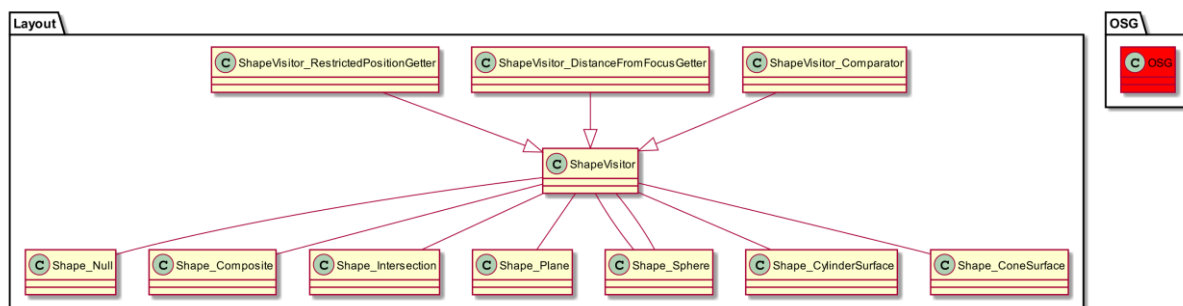
Obrázok 7 - Layout 1



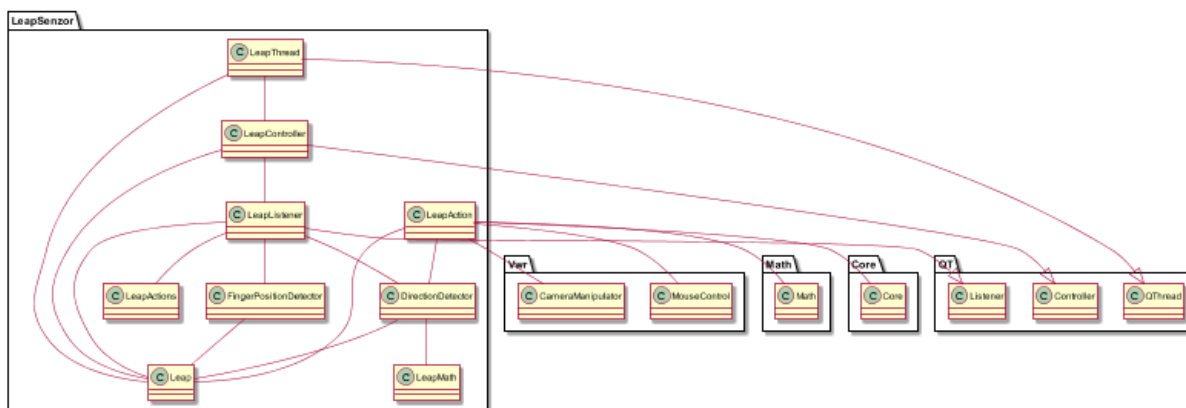
Obrázok 8 - Layout 2



Obrázok 9 - Layout 3



Obrázok 10 - Layout visitor



Obrázok 11 - Leap senzor

LeapThread dedí od QThread - include Leap, LeapController - vytvára samostatný thread pre leap senzor

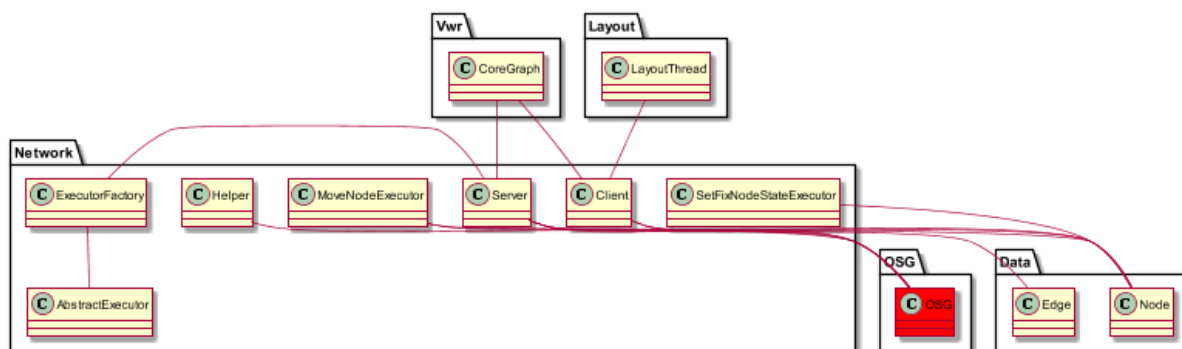
LeapListener dedí od Listener - include Leap, LeapActions, DirectionDetector, FingerPositionDetector – prekonáva funkcie Listenera, ktoré sa starajú o zmenu stavu leap senzora

LeapController dedí od Controller - include Leap, LeapListener - spúšťa a zastavuje počúvanie leap senzora

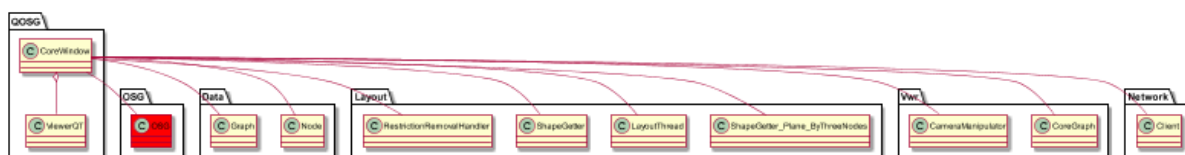
LeapAction - include DirectionDetector, MouseControl, CameraManipulator, Leap, Math, Core - obsahuje funkcie, ktoré sa vykonávajú na základe detegovaného gesta

FingerPositionDetector - include Leap - obsahuje funkcie na prácu s pozíciou prstov

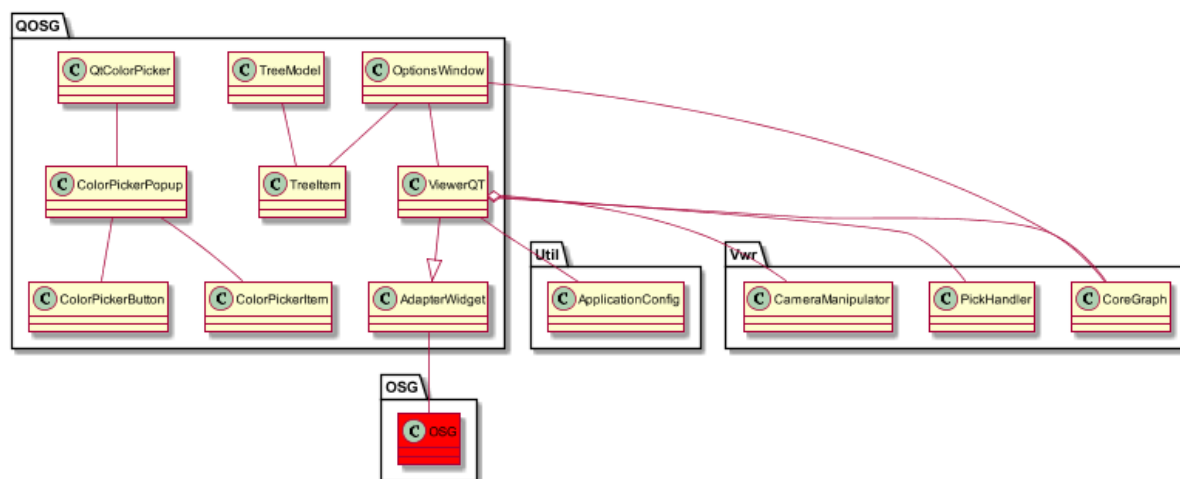
DirectionDetector - include Leap, LeapMath - obsahuje funkcie na prácu s natočením rôznych častí ruky



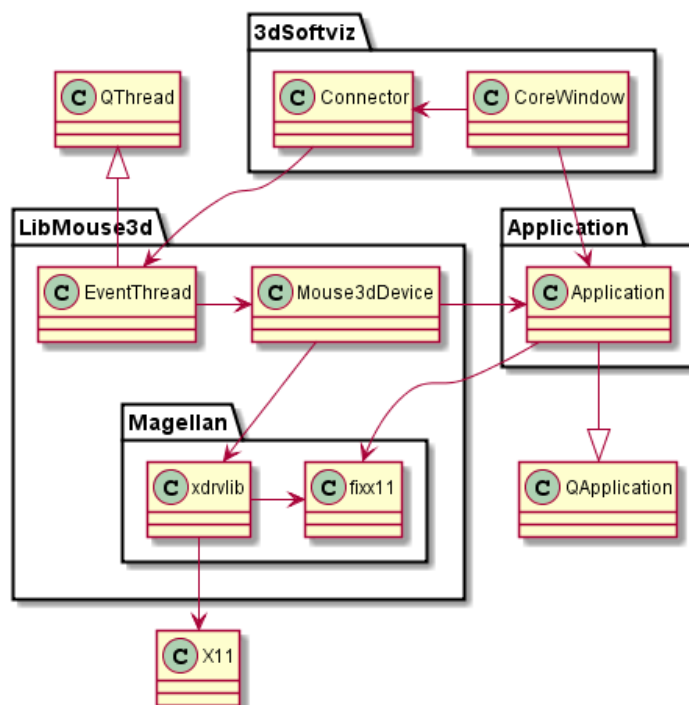
Obrázok 12 – Network



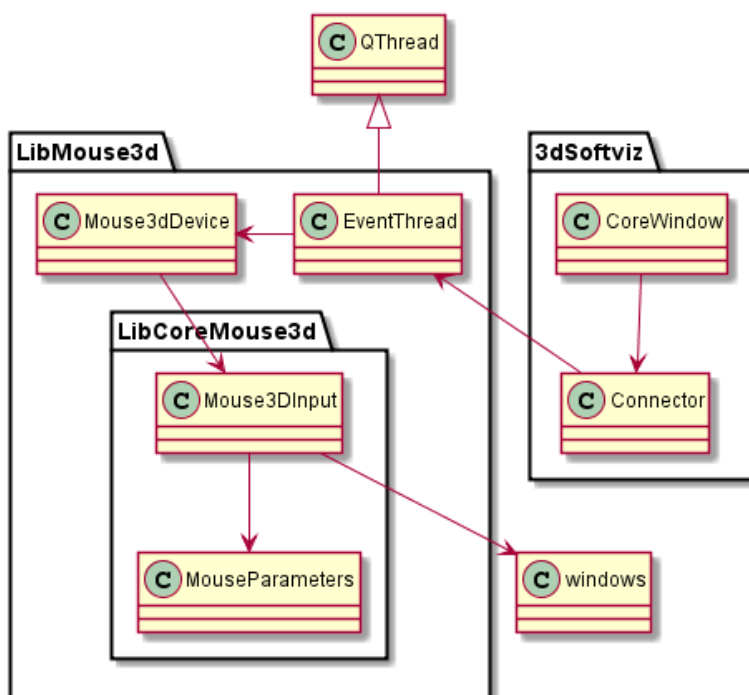
Obrázok 13 – CoreWindow



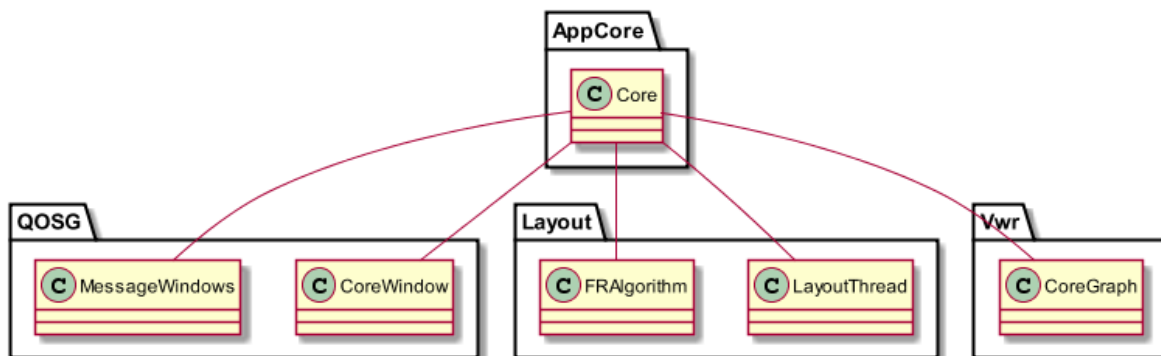
Obrázok 14 - QOSG



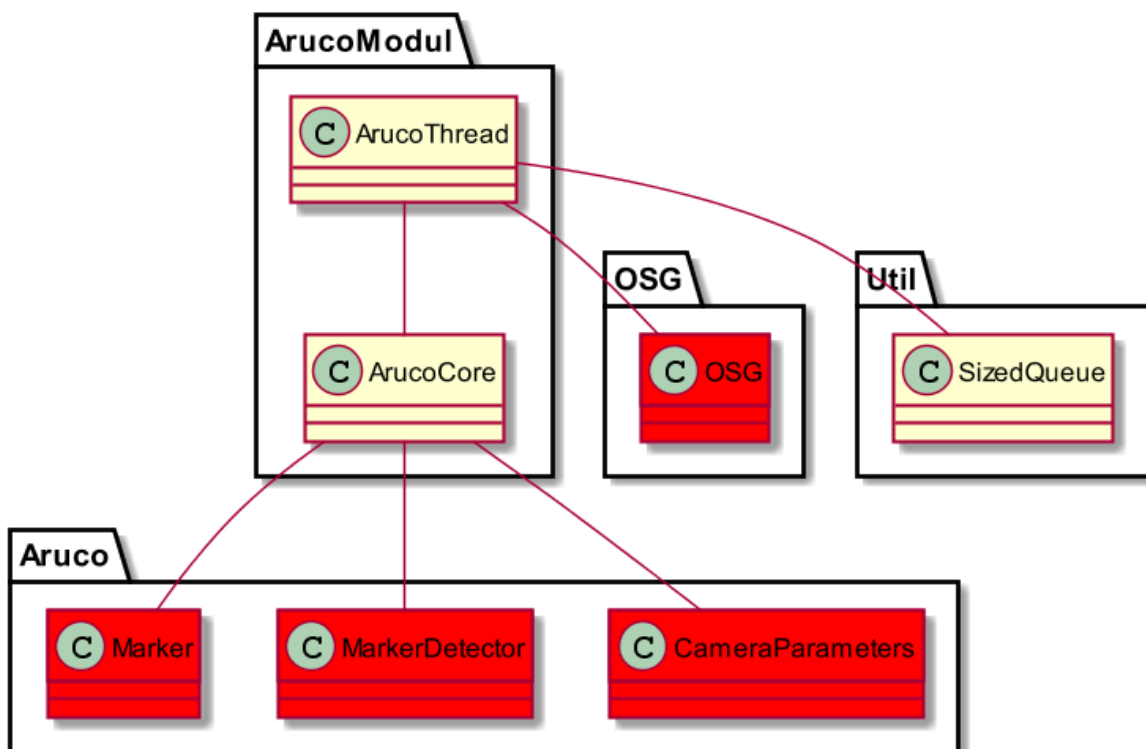
Obrázok 18 - 3D mouse Linux



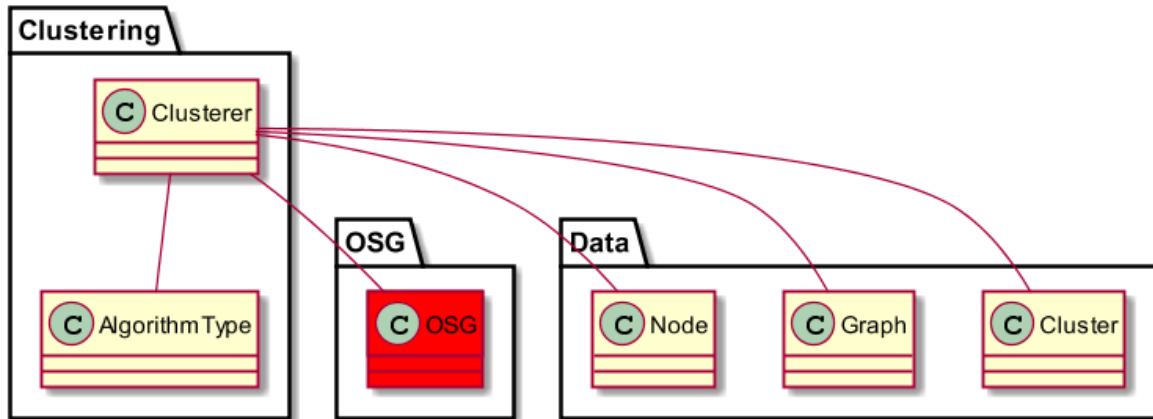
Obrázok 19 - 3D mouse Windows



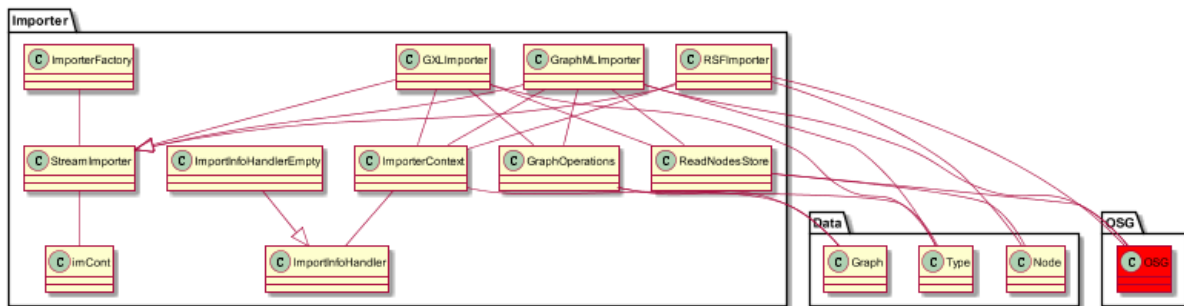
Obrázok 20 – AppCore



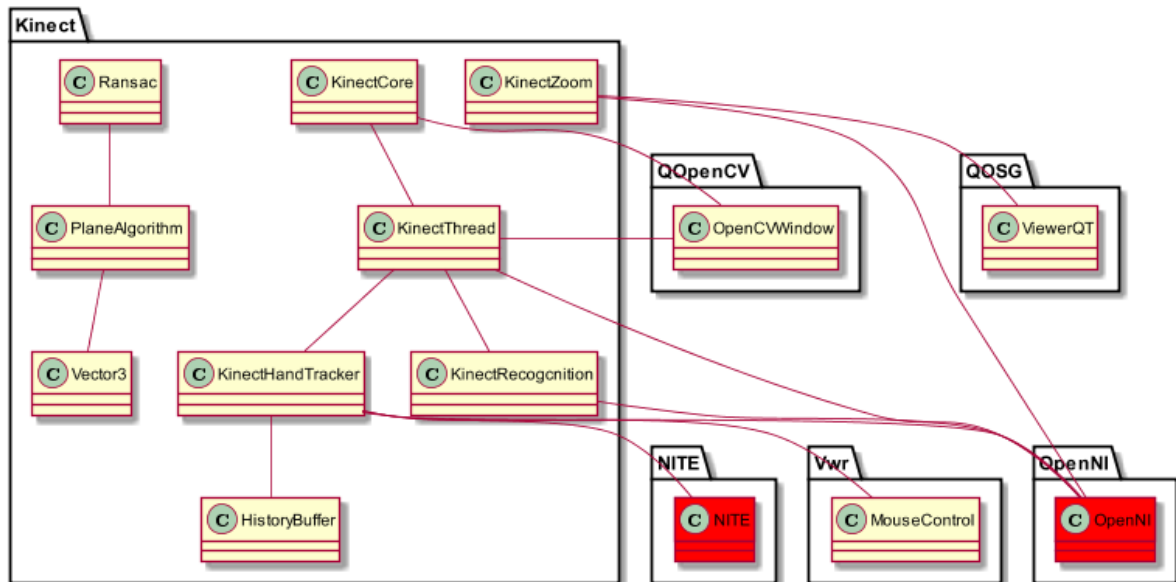
Obrázok 21 – ArucoModul



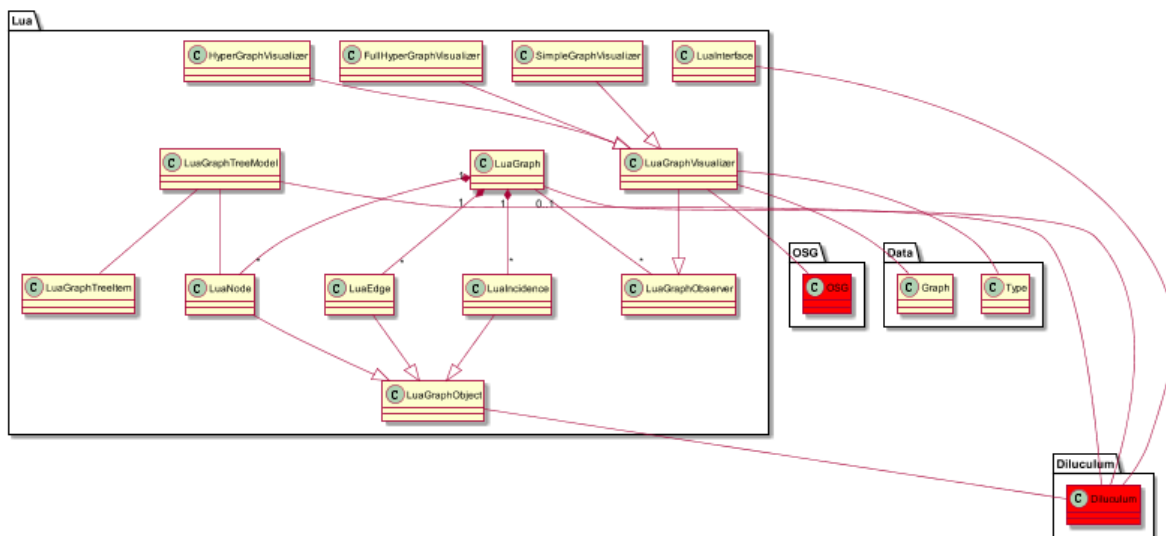
Obrázok 22 – Clustering



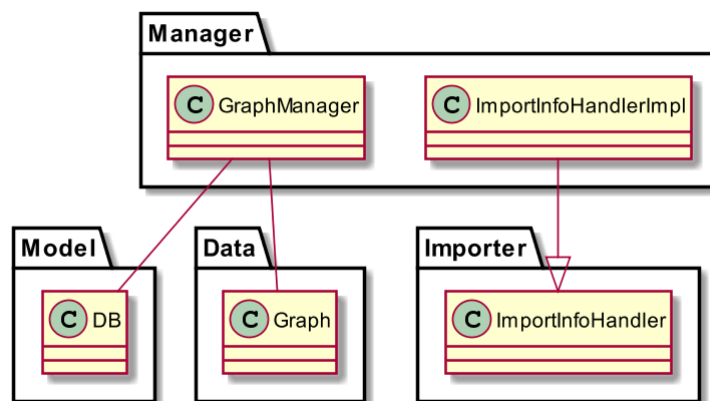
Obrázok 23 – Importer



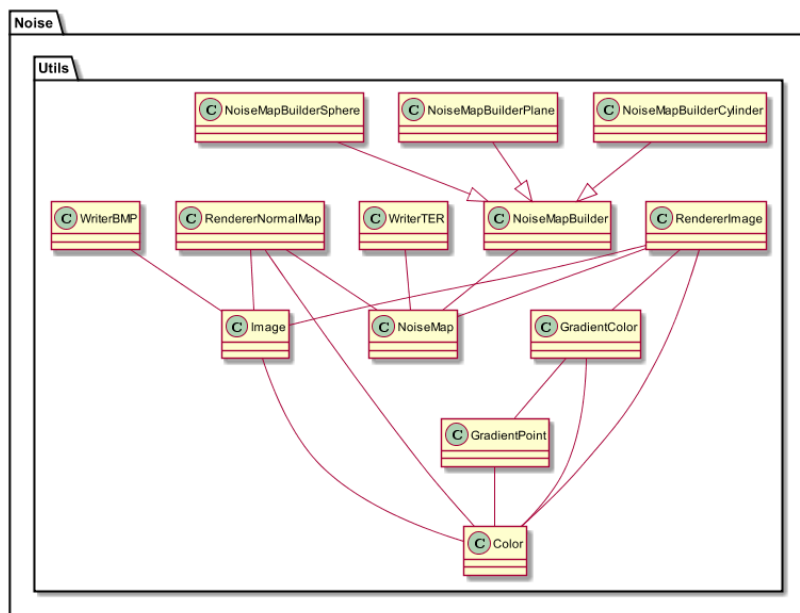
Obrázok 24 – Kinect



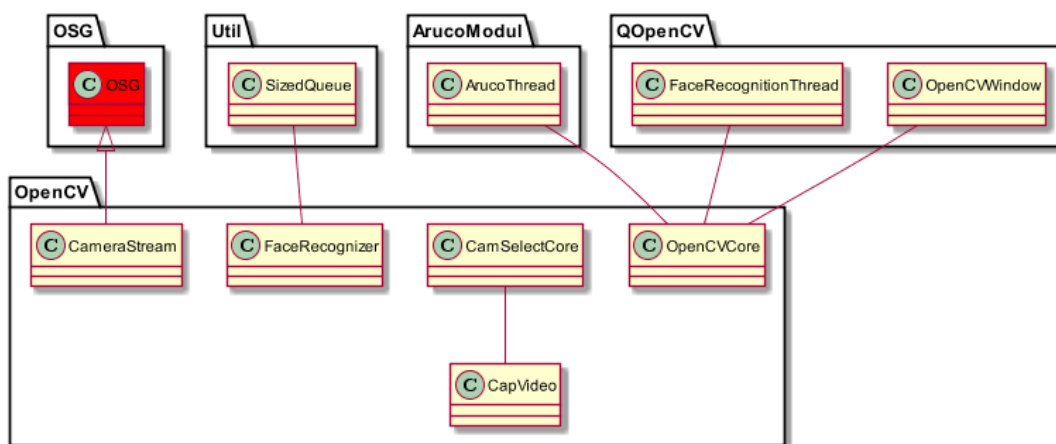
Obrázok 25 – Lua



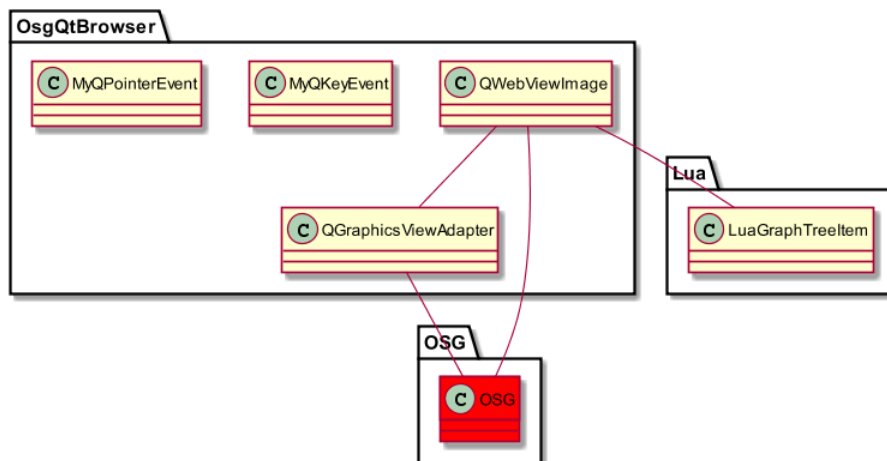
Obrázok 26 – Manager



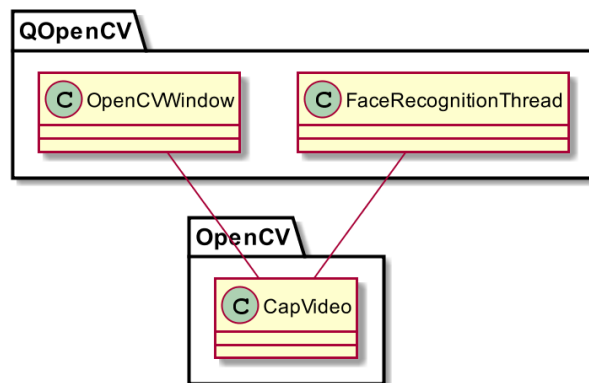
Obrázok 27 – Noise



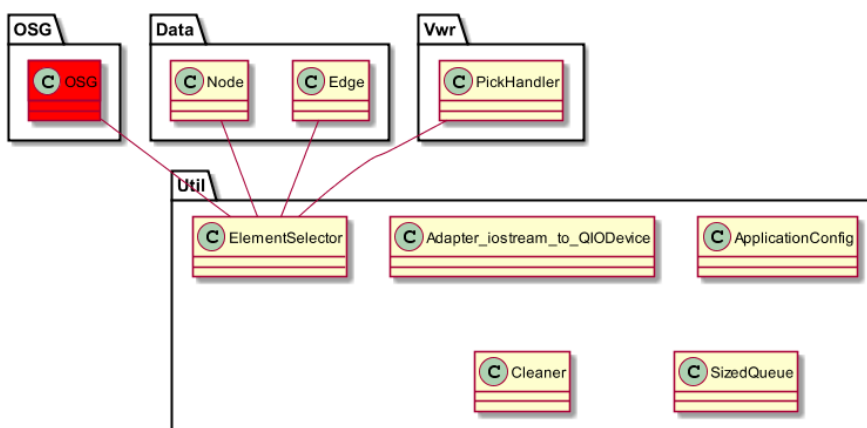
Obrázok 28 – OpenCV



Obrázok 29 – OSGQtBrowser



Obrázok 30 – QOpenCV



Obrázok 31 - Util

4 Prehľad modulov

4.1 Implementované moduly

Core – obsahuje jadro systému, inicializuje základné časti systému.

Data - dátový modul pre opis štruktúry grafu, obsahujúci triedy reprezentujúce jednotlivé prvky grafu (graph, node, edge, type, layout, ...).

Importer - modul pre parsovanie vstupných súborov vo formátoch GraphML, RSF a GXL.

Layout – modul, ktorý má na starosti rozmiestňovanie uzlov v 3D priestore, taktiež obsahuje implementácie layout algoritmu a triedy pre pridávanie ohraničení rozmiestnenia.

Manager - modul pre prácu s grafom.

Math - model pre rozšírenie práce s kamerou.

Model – modul pre komunikáciu systému s databázou. Funkcionalitou je mapovanie objektov do databázy, vytvorenie spojenia s databázou a základne funkcie výberu a uloženia grafu. Taktiež umožňuje uloženie uzlov aj s ich atribútmi a viacero rozmiestnení pre 1 graf.

Network - modul pre podporu kolaboratívnej práce nad grafom. Poskytuje klient/server funkcionality.

Noise - modul pre vytvorenie generovaného 3D priestoru pre pozadie.

OsgBrowser - modul zahŕňa viazanie udalostí pre jednotlivé klávesy a akcie myši medzi rozhraniami Qt a OpenSceneGraph a vizualizáciu načítaných grafov.

QOSG – modul pre prácu s grafickými prvkami softvéru. Má na starosti vytvorenie hlavného okna a prácu s pomocnými oknami a widgetami.

Util - zabezpečuje konfiguráciu nastavení aplikácie a funkcie pre vyčistenie pamäte.

Viewer - modul zabezpečuje pohyb v 3D priestore a prácu s kamerou. V module sa tiež pripravuje graf a jeho pre zobrazenie a vytvorenie 3D kocky pre pozadie.

Kinect – modul pre komunikáciu a ovládanie zariadenia Kinect. Medzi jeho funkcionality patrí získavanie informácií a ich nasledovné spracovanie. Obsahuje detegovanie gest, ktoré nahrádzajú ovládanie myšou, otáčanie a pohyb grafu a gestá pre ďalšie ovládanie.

Speech - implementuje funkcionality rozpoznávania hlasu.

OpenCV - zabezpečuje rozpoznávanie tváre na obraze z kamery a poskytuje funkcionality pre správu kamier.

QOpenCV – obsahuje okno pre ovládanie rozpoznávania tváre, značky a ovládanie video pozadia

Aruco – obsahuje funkcionality, ktorá vie rozpoznávať značky z kamery použitím knižnice Aruco.

5DTGloves – zabezpečuje detegovanie gesta ruky a vykonávanie korešpondujúcich akcií.

Leap senzor – deteguje dve ruky používateľa v 3D priestore a sleduje pohyby rúk až na úrovni článkov prstov.

3D myš – poskytuje ovládanie kamery pomocou tohto zariadenia

4.2 Podporované nové zariadenia

Nvidia 3D Vision PRO – slúžia na zobrazenie scény v 3D pomocou špeciálnych okuliarov

Vuzix STAR 1200XL – umožňuje zobrazit' scénu s grafmi do rozšírenej reality

5 Moduly a funkcionality systému

5.1 *Mouse3d*

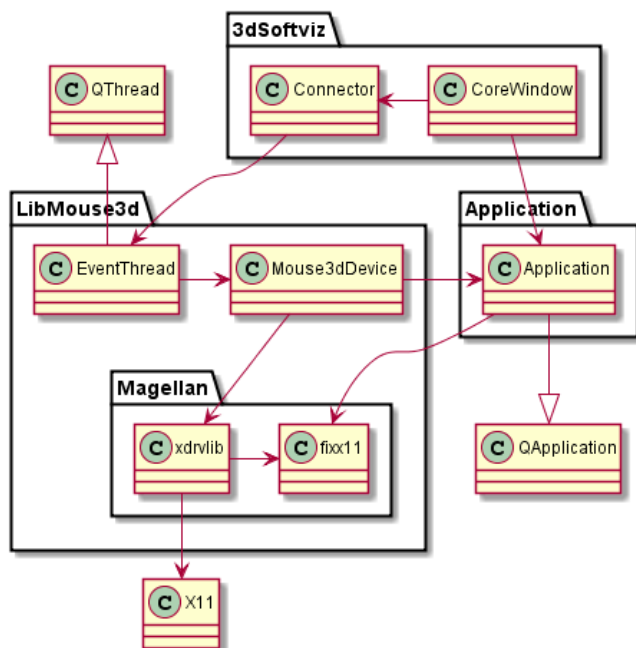
K dispozícii máme dva modely počítačových 3D myší. Kompaktný Space Navigator for notebooks je pre svoje malé rozmery ľahko prenosný. Na druhej strane, Space Pilot Pro je mohutnejšia myš obohatená o niekoľko tlačidiel navyše. Pre obe zariadenia slúži rovnaké SDK.

5.1.1 *Analýza modulu Mouse3d*

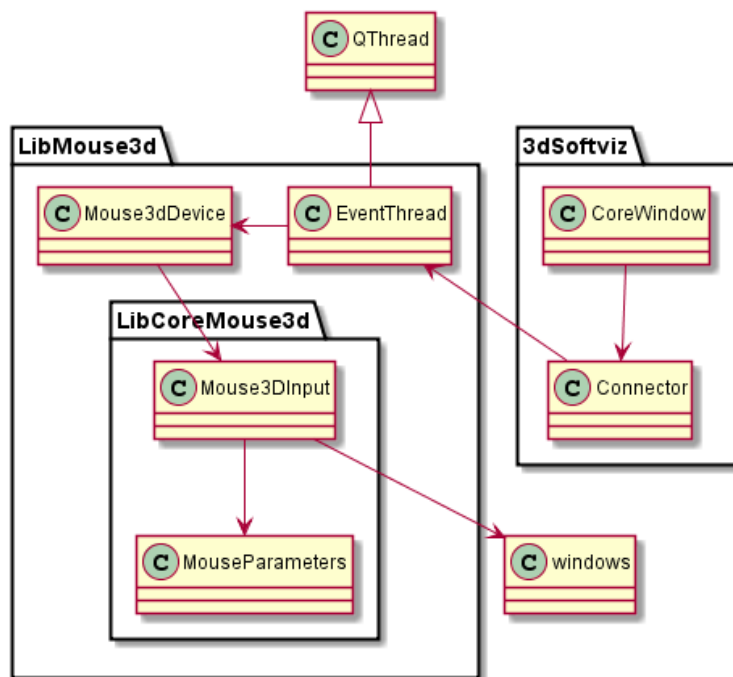
Mouse3d je samostatný implementačný modul v projekte 3dSoftviz zodpovedný za pripojenie funkcionality 3d myši. Pri využívaní priestorových údajov je podpora efektívnej manipulácie s 3d modelmi nevyhnutná pre projekt. Modul tvorí rozhranie pre pripojenie myši nezávislé na konkrétnom produkte. Jeho účelom je zachytávanie a spracúvanie signálov z 3d myši a ich interpretácia pre 3dSoftviz. Spolu s modulom je vytváraná knižnica pre prácu so SpaceNavigator for Notebooks a Space Pilot Pro myšou od spoločnosti 3dConnexion. Knižnica LibMouse3d slúži ako prostriedok na komunikáciu modulu 3dMouse s funkcionalitou SpaceNavigator. V rámci vývoja nám spoločnosť na požiadavku poskytla SDK pre tento hardvér.

5.1.2 *Návrh modulu Mouse3d*

Pri návrhu sme sa inšpirovali predošlými funkčnými modelmi. Identifikovali sme skupinu tried Mouse3d, ktoré budú zabezpečovať rozhranie medzi projektom 3dSoftviz a súbormi 3d myši. Toto je nevyhnutné kvôli tomu, aby mohla byť zo súborov 3d myši vytvorená samostatná knižnica. Skupina tried v balíku Mouse3d slúži na spracovanie signálu získaného zo špeciálnej myšky do použiteľnej podoby pre náš projekt. LibMouse3d obsahuje rozhrania, ktoré oddeľujú aplikačnú logiku myšky od samotného softvéru. V balíku sú umiestnené aj triedy slúžiace na namapovanie ovládania pomocou 3D myši – budovanie aplikácie však kvôli multipatformovému návrhu však pracuje vždy len s jednou skupinou súborov mimo štandardného rozhrania. Usporiadanie takýmto spôsobom je prospešné pre projekt z hľadiska modularity. Takto sa vytvorí modul Mouse3d, ktorý môže byť zaradený medzi ostatné. Z adresára LibMouse3d sa v projekte vytvára samostatne spustiteľná knižnica.



Obrázok 32 - triedy modulu 3dMouse pre Linux



Obrázok 33 - triedy modulu 3dMouse pre Windows

5.1.3 Implementácia

Implementovali sme moduly systému z diagramov na obrázku 32 a 33 a integrovali ich do projektu. Jednotlivé diagramy reprezentujú platformovo nezávislý návrh budovania aplikácie v závislosti na definovanej platforme.

Pri implementácií sme narazili na nedostatok vo vývojárskom balíčku produktu, ktorý znemožňoval správnu detekciu produktu v priamej spolupráci s Qt systémom pre budovanie aplikácie. Po negatívnej skúsenosti s podporným tímom pre 3D myš sme sa rozhodli preskúmať a implementovať alternatívne riešenia pre integráciu do Qt aplikácie. Všetky ostatné moduly boli napísané a obsahujú základnú funkcionality. Architektúra je však navrhnutá tak, aby bolo jednoduché ich rozšíriť a zmeniť.

Mouse3DDevice predstavuje triedu obaľujúcu multiplatformové riešenie. Obsahuje tvorbu entity podmienenej detegovanou platformou. Tá následne využíva pre prácu s driverom programové riešenie pre túto platformu.

Windows implementácia pozostáva z open source MVC reimplementácie SDK pre prácu s Qt. Originálne riešenie pochádza od Davida Dibbena a je odporúčané aj na oficiálnom podpornom fóre zariadenia. (<http://www.codegardening.com/2011/02/using-3dconnexion-mouse-with-qt.html>). Komunikácia s aplikáciou je riešená pomocou Qt funkcionality signal – slot.

Linux implementácia využíva natívny X11 driver pre správu okien, ktorý je schopný zachytávať signály pomocou správ knižnice Magelan. Riešenie si vyžaduje reimplementovanie metódy z triedy QApplication nazvanej X11EventFilter. Tá pasívne zachytáva tieto správy, ktoré sa následne preposielajú na spracovanie implementovanému modulu.

5.2 *Stereoskopické 3D s AR okuliarmi Vuzix STAR 1200XL*

5.2.1 Analýza

Vuzix okuliare sú zariadenie pre rozšírenú realitu. Pozostávajú z dvoch displejov ktoré zobrazujú obraz pred očami z pripojeného počítača. Podporujú tri režimy zobrazovania. Prvým

je normálne 2D zobrazenie, ktoré zobrazuje rovnaký obraz na oba displeje. Ďalšie dva režimy sú 3D režimy. Jeden je top-bottom režim ktorý zobrazuje vrhnú polovicu obrazovky počítača do pravého displeja a spodnú polovicu do ľavého displeja. Druhý je side-by-side režim, ktorý zobrazuje pravú polovicu obrazovky počítača na pravý displej a ľavú polovicu na pravý displej. Vhodným zobrazením stereo obrazu na počítači docielime 3D efekt. Okrem displejov je súčasťou kamera, ktorá sníma obraz pred používateľom a modul ktorý sleduje aktuálnu pozíciu okuliarov v priestore.

5.2.2 Návrh

Pre 3D zobrazenie grafov v 3DSoftviz navrhujeme využiť 3D režimy Vuzix okuliarov. Okuliare sa pre pripojený počítač javia ako zobrazovacie zariadenie, ktoré zobrazuje aktuálny obraz, je teda potrebné na strane 3DSoftviz zabezpečiť správne rozdelenie obrazu na polovice ktoré sú vzájomne posunuté a tým docielime efekt stereoskopického 3D. Pre takéto rozdelenie existuje priamo podpora v OSG, kde je potrebné nastaviť len správnu konfiguráciu, čo znamená typ 3D – horizontálne alebo vertikálne (čo sa mapuje na Vuzix ako top-bottom a side-by-side), a hodnoty pre správny 3D efekt – vzdialenosť od obrazovky (v prípade Vuzix okuliarov to sú 3 metre), a fyzická veľkosť obrazovky (Vuzix vytvára virtuálny obraz s uhlopriečkou 190cm s pomerom strán 16:9). Kameru okuliarov je možné využívať v iných moduloch 3DSoftvizu, pretože z pohľadu pripojených zariadení sa javí ako normálna USB kamera. Modul pre sledovanie pozície okuliarov momentálne nevyužijeme.

5.2.3 Implementácia

Zobrazenie scény v režime 3D je priamo podporované v OSG. V našom projekte sme upravili triedu `src/Viewer/PickHandler.cpp`, kde sme pridali jednotlivé 3D módy (`osg::DisplaySettings::StereoMode::"xxx"`). Stereo mód sa nastavuje nasledovne:

```

//split stereo 3D
else if ( ea.getKey() == osgGA::GUIEventAdapter::KEY_G ) {
    if ( splitviewMode == 0 ) {
        //turn on
        osg::DisplaySettings::instance()->setStereoMode (
        osg::DisplaySettings::StereoMode::VERTICAL_SPLIT );
        osg::DisplaySettings::instance()->setScreenDistance (
        Util::ApplicationConfig::get()->getValue( "Display.Settings.Vuzix.Distance" ).toFloat() );
        osg::DisplaySettings::instance()->setScreenHeight (
        Util::ApplicationConfig::get()->getValue( "Display.Settings.Vuzix.Height" ).toFloat() );
        osg::DisplaySettings::instance()->setScreenWidth (
        Util::ApplicationConfig::get()->getValue( "Display.Settings.Vuzix.Width" ).toFloat() );

        qDebug() << "Turned on split stereo 3D - vertical split";
    }
    else if ( splitviewMode == 1 ) {
        osg::DisplaySettings::instance()->setStereoMode (
        osg::DisplaySettings::StereoMode::HORIZONTAL_SPLIT );
        qDebug() << "Turned on split stereo 3D - horizontal split";
    }
    else {
        //turn off
        osg::DisplaySettings::instance()->setStereo( FALSE );
        //reset to default config
        osg::DisplaySettings::instance()->setScreenDistance (
        Util::ApplicationConfig::get()->getValue( "Display.Settings.Default.Distance" ).toFloat()
        );
        osg::DisplaySettings::instance()->setScreenHeight (
        Util::ApplicationConfig::get()->getValue( "Display.Settings.Default.Height" ).toFloat() );
        osg::DisplaySettings::instance()->setScreenWidth (
        Util::ApplicationConfig::get()->getValue( "Display.Settings.Default.Width" ).toFloat() );
        osg::DisplaySettings::instance()->setEyeSeparation (
        Util::ApplicationConfig::get()->getValue( "Display.Settings.Default.EyeSeparation"
        ).toFloat() );

        qDebug() << "Turned off split stereo 3D";
    }
    splitviewMode = ( splitviewMode + 1 ) % 3;    //rotate modes : vertical /
horizontal / off
}

```

Prepínanie medzi 3D režimami je namapované na klávesu ‚G‘. Zobrazenie sa prepína medzi vertikálnym, horizontálnym rozdelením a vypnutím.

Ďalšou zmenou je možnosť nastavenia vzdialenosti očí. Okuliare treba v niektorých prípadoch kalibrovať a o to sa postará nasledovný kód:

```

//adjust eye distance, 0.001m change
else if ( ea.getKey() == osgGA::GUIEventAdapter::KEY_H && ( splitviewMode != 0 ) ) {
    //-
    float distance = osg::DisplaySettings::instance()->getEyeSeparation();
    distance = distance - 0.001f;
    osg::DisplaySettings::instance()->setEyeSeparation( distance );
    qDebug() << "Eye distance : " << distance;
}
else if ( ea.getKey() == osgGA::GUIEventAdapter::KEY_J && ( splitviewMode != 0 ) ) {
    //+
    float distance = osg::DisplaySettings::instance()->getEyeSeparation();
    distance = distance + 0.001f;
    osg::DisplaySettings::instance()->setEyeSeparation( distance );
    qDebug() << "Eye distance : " << distance;
}
}

```

Klávesom H sa znižuje aktuálna hodnota vzdialenosti, ktorá je v metroch, o 0.001m, a klávesom J sa zvyšuje o rovnakú hodnotu. Potrebne údaje sú získavané z konfiguračného súboru (*resources/config/config*), do ktoré sme pridali nasledovné riadky pre OSG zobrazovanie:

```

Display.Settings.Default.Distance=0.5
Display.Settings.Default.Height=0.26
Display.Settings.Default.Width=0.325
Display.Settings.Default.EyeSeparation=0.06

```

Tieto nastavenia sú použité aj pri vypnutí 3D režimu, aby sa zobrazenie obnovilo do pôvodného stavu. Rovnako sa v súbore nachádzajú aj nastavenia pre okuliare Vuzix:

```

Display.Settings.Vuzix.Distance=3.048
Display.Settings.Vuzix.Height=0.93
Display.Settings.Vuzix.Width=1.66

```

Tieto hodnoty predstavujú veľkosť zobrazovacieho zariadenia a sú dôležité pre docielenie dobrého 3D efektu.

5.3 *Stereoskopické 3D s okuliarmi Nvidia 3D Vision Pro*

5.3.1 *Analýza*

Okuliare 3D Vision využívajú quad buffer. Jedná sa o technológiu používanú v počítačovej grafike pre implementácie stereoskopického renderovania. Pre stereoskopické renderovanie musí každé oko získať samostatný obraz. Quad buffer využíva double buffering s predným a zadným buffrom synchronizovane pre každé oko. Takto dostávame 4 buffre.

5.3.2 Návrh

OpenSceneGraph podporuje mnohé stereoskopické režimy, medzi ktoré patrí aj quad buffer (*osg::DisplaySettings::StereoMode::QUAD_BUFFER*). Tento režim funguje iba s kompatibilnými zariadeniami. Naša zostava pozostáva medzi inými aj z grafickej karty Nvidia Quadro K5000 a monitoru BenQ XL2720Z (144Hz). Okrem prepnutia režimu sme museli upraviť parametre zobrazovača – *QGLWidget*.

5.3.3 Implementácia

Prepnutie na stereoskopické režim – quad buffer sme spojili s už existujúcim riešením pre AR okuliare. Pridali sme ďalší stav, do ktorého sa vieme dostať cez klávesu ,G‘. Bola upravená trieda *src/Viewer/PickHandler.cpp* nasledovne:

```
//split stereo 3D
else if ( ea.getKey() == osgGA::GUIEventAdapter::KEY_G ) {
    if ( splitviewMode == 0 ) {
        osg::DisplaySettings::instance()->setStereoMode(
osg::DisplaySettings::StereoMode::QUAD_BUFFER );
        osg::DisplaySettings::instance()->setStereo( TRUE );
        qDebug() << "Turned on quad buffer stereo 3D";
    }
    ...
}
```

Museli sme vykonať zmenu aj pri vytváraní okna *AdapterWidget*, keďže pôvodné nastavenia nepodporovali tento režim. Trieda *src/QOSG/AdapterWidget.cpp*:

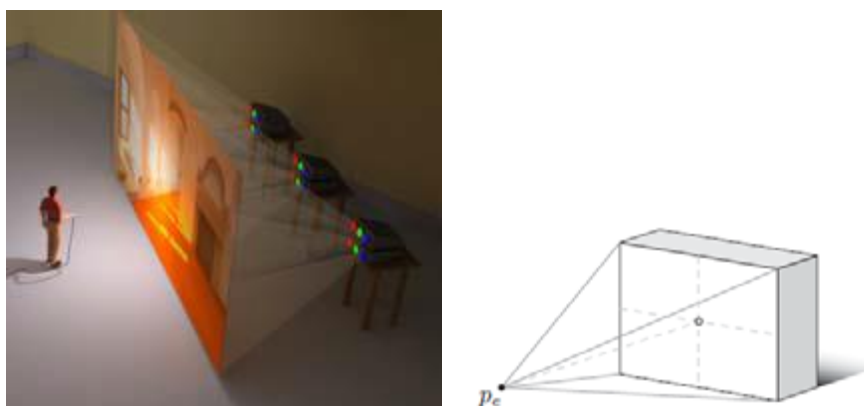
```
AdapterWidget::AdapterWidget( QWidget* parent, const char* name , const QGLWidget*
shareWidget, WindowFlags f ) :
#if QT_VERSION > 0x040000
    QGLWidget( QGLFormat( QGL::DoubleBuffer | QGL::DepthBuffer | QGL::Rgba |
QGL::StencilBuffer | QGL::AlphaChannel | QGL::StereoBuffers ), parent, shareWidget, f )
#else
    QGLWidget( parent, shareWidget, f )
#endif
{
    _gw = new osgViewer::GraphicsWindowEmbedded( 0,0,width(),height() );
    setFocusPolicy( Qt::StrongFocus );
}
```

5.4 Multiview „Wall“

5.4.1 Analýza

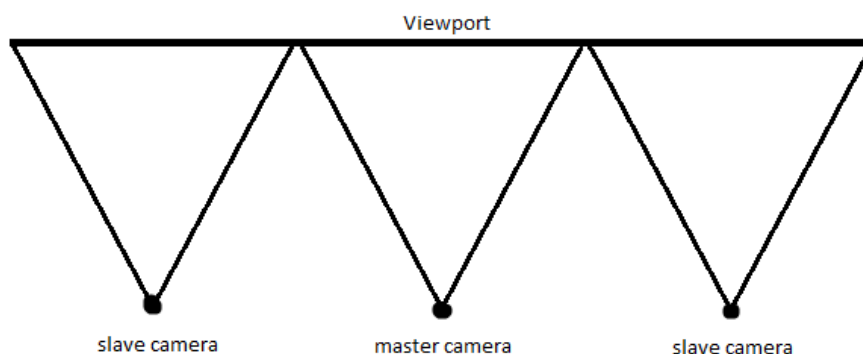
Zobrazenie pomocou viacerých monitorov, resp. projektorov sa dá docieľiť pomocou rozloženia „CAVE“ alebo „WALL“. V našom prípade sa budeme zaoberať typom WALL. Pri

tomto rozložení sa používa symetrické perspektívne zobrazenie (symmetric frustum). Na správne zobrazenie potrebujeme nepárne množstvo zobrazovacích zariadení. Najideálnejšie riešenie by bolo vytvorenie x ďalších okien, v ktorých by sa kamery správne nakonfigurovali. Bohužiaľ, náš aktuálny systém túto opciu neumožňuje, kvôli hlbokjej integrácii OSQ do Qt používateľského rozhrania. Jediným riešením je rozťahnutie jedného okna na potrebnú veľkosť. V existujúcom kóde sme našli triedu `src/Viewer/ViewerQt.cpp`, ktorá sa aktuálne zaoberá sa nastavovaním kamery a viewportov.



5.4.2 Návrh

Naše riešenie spočíva v rozdelení viewportu na toľko ekvivalentných častí, s koľkými monitormi/projektormi pracujeme. Následne vytvoríme určitý počet kamier, ktoré sa budú správať ako slave kamery relatívne od hlavnej kamery. Slave kamery sú posunuté do jedného a druhého smeru, aby sa neprekrývali. Ilustrácia výslednej zostavy kamier je znázornená na tomto obrázku (pre 3 kamery):



5.4.3 Implementácia

Na dosiahnutie požadovaného cieľa sme museli modifikovať triedu *src/Viewer/ViewerQt.cpp*. Nastavenie potrebných premenných:

```
double fovy = 60.0;
double nearClippingPlane = 0.01;
double farClippingPlane = appConf->getValue( "Viewer.Display.ViewDistance" ).toFloat();
double aspectRatio = static_cast<double>( width() )/static_cast<double>( height() );
```

```
if ( appConf->getValue( "Viewer.Display.Multiview" ).toInt() ) {
    int screenNum = appConf->getValue( "Viewer.Display.ScreenNum" ).toInt();
    int divisionError = width() % screenNum;

    //LEFT CAMERAS
    //0, 1 = i
    for ( int i=0; i<screenNum/2; i++ ) {
        osg::ref_ptr<osg::Camera> leftCam = new osg::Camera;
        leftCam->setViewport( new osg::Viewport( ( width()/screenNum +
divisionError ) * i, 0, width()/screenNum + divisionError, height() ) );
        leftCam->setGraphicsContext( getGraphicsWindow() );
        leftCam->setProjectionMatrixAsPerspective( fovy, aspectRatio,
nearClippingPlane, farClippingPlane );
        leftCam->setViewMatrix( osg::Matrix::lookAt( osg::Vec3d( -10, 0, 0 ),
osg::Vec3d( 0, 0, 0 ), osg::Vec3d( 0, 1, 0 ) ) );
        osgViewer::Viewer::addSlave( leftCam.get(), osg::Matrix::translate( 2.0
* ( screenNum/2-i ), 0.0, 0.0 ), osg::Matrix() );
    }

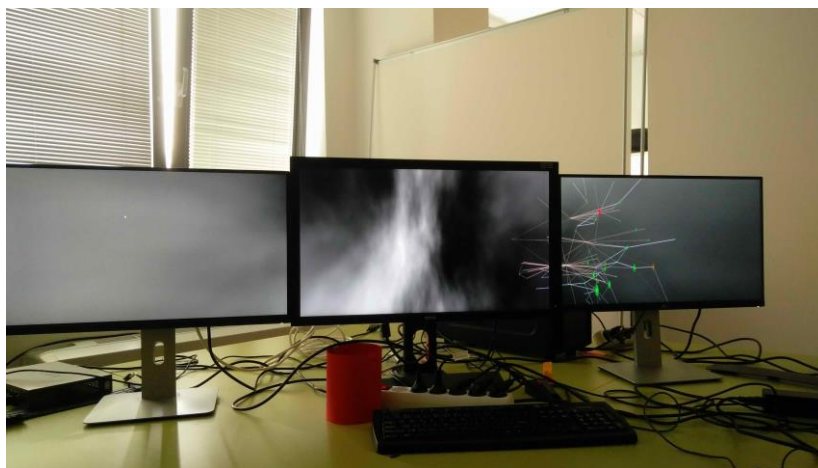
    //MIDDLE (MASTER) CAMERA
    //2 = screenNum/2
    getCamera()->setViewport( new osg::Viewport( ( width()/screenNum + divisionError
) * ( screenNum/2 ), 0, width()/screenNum + divisionError, height() ) );
    getCamera()->setGraphicsContext( getGraphicsWindow() );
    getCamera()->setProjectionMatrixAsPerspective( fovy, aspectRatio,
nearClippingPlane, farClippingPlane );
    getCamera()->setViewMatrix( osg::Matrix::lookAt( osg::Vec3d( -10, 0, 0 ),
osg::Vec3d( 0, 0, 0 ), osg::Vec3d( 0, 1, 0 ) ) );

    //RIGHT CAMERAS
    //3, 4 = screenNum/2 + 1 + i
    for ( int i=0; i<screenNum/2; i++ ) {
        osg::ref_ptr<osg::Camera> rightCam = new osg::Camera;
        rightCam->setViewport( new osg::Viewport( ( width()/screenNum +
divisionError ) * ( screenNum/2 + 1 + i ), 0, width()/screenNum + divisionError, height()
) );
        rightCam->setGraphicsContext( getGraphicsWindow() );
        rightCam->setProjectionMatrixAsPerspective( fovy, aspectRatio,
nearClippingPlane, farClippingPlane );
        rightCam->setViewMatrix( osg::Matrix::lookAt( osg::Vec3d( -10, 0, 0 ),
osg::Vec3d( 0, 0, 0 ), osg::Vec3d( 0, 1, 0 ) ) );
        osgViewer::Viewer::addSlave( rightCam.get(), osg::Matrix::translate( -
2.0 * ( i+1 ), 0.0, 0.0 ), osg::Matrix() );
    }
}
```

V implementácii sa ráta aj s prípadnou chybou pri delení nepárnym číslom a následným zaokrúhľením na celé čísla (*divisionError*). Master kamera je vždy stredná, ďalšie sa posúvajú

o hodnotu 2.0 * poradie kamery. Zapnutie/vypnutie režimu multiview, počet kamier a rozlíšenie použitých monitorov/projektorov sa nastavuje v konfiguračnom súbore *resources/config/config*.

```
Viewer.Display.Multiview=0
Viewer.Display.ScreenNum=3
Viewer.Display.MaxScreenWidth=1920
Viewer.Display.MaxScreenHeight=1080
```



5.5 Refaktorovanie modulu Leap

Hlavným cieľom refaktorovania modulu Leap je oddelenie do samostatnej knižnice ktorá nemá žiadne závislosti na zvyšok 3Dsoftviz. Druhým cieľom bola optimalizácia kódu na efektivitu a rýchlosť.

Oddelenie závislostí bolo dosiahnuté vytvorením nového abstraktného rozhrania, ktorého metódy sú volané v kóde tam, kde boli závislosti (napr. manipuláciu s kamerou). Závislosti sú až v samotnej implementácii rozhrania, ktoré je súčasťou 3Dsoftviz.

Viacere metódy v triedach Leap modulu boli upravené a optimalizované. Zjednodušili sme podmienky a zefektívnil algoritmy so zachovaním rovnakej funkcionality. Pre multiplatformovú podporu sme tiež refaktorovali použitie WIN32 knižnice pre uspávanie vlákna. Uspávanie je nutné pre lepšiu detekciu pohybu ruky SDK Leap-u. Použili sme chránané funkcie QThread, ktoré sme obalili do vlastnej triedy:

Pre kompiláciu do samostatnej knižnice bol upravený súbor CMakeLists.txt

```
#ifndef LEAPSLEEPER_H
#define LEAPSLEEPER_H

#endif // LEAPSLEEPER_H

#include <QThread>

class LeapSleeper : public QThread
{
public:
    static void usleep(unsigned long usecs) {QThread::usleep(usecs); }
    static void msleep(unsigned long msec) {QThread::msleep(msec); }
    static void sleep(unsigned long secs) {QThread::sleep(secs); }
};
```

5.6 Dynamická zmena pozadia

Cieľom je možnosť zmeny pozadia na napríklad čiernu farbu počas behu programu. Doteraz bolo toto možné len pomocou zmeny konfigurácie programu a jeho zapnutím/vypnutím.

Implementovali sme teda úplne nové textúry resp. SkyBox, ktorý je len čierny alebo biely, podľa toho aká ma byť zmena pozadia. Pri požiadavke na zmenu pozadia sa zastaví layoutovací algoritmus pre rozmiestnenie vrcholov grafu v scéne. Následne sa odstráni z grafu scény posledný uzol, ktorý vždy reprezentuje aktuálne pozadie scény. Vytvorí sa nový SkyBox - pozadie s požadovanou textúrou, tento SkyBox sa pridá na koniec grafu scény. Následne sa obnoví layoutovací algoritmus, zatiaľ čo bolo pozadie zmenené.

```
void CoreWindow::switchBackgroundSkyBox() {
    LOG(INFO) << "CoreWindow::switchBackgroundSkyBox switching to SkyBox bg";
    Data::Graph* currentGraph = Manager::GraphManager::getInstance() -
>getActiveGraph();

    int flagPlay = 0;
    if (this->isPlaying) {
        flagPlay = 1;
        pauseLayout();
    }
    if (coreGraph->updateBackground(0, currentGraph) == 0) {
        LOG(INFO) << "Background successfully updated";
    }
    else {
        LOG(ERROR) << "Background bg update failed";
    }
    if (flagPlay == 1) playLayout();
}
```

Pre každé pozadie bola vytvorená separátna funkcia, ktorá zabezpečuje logovanie a vyššie spomenuté operácie potrebné pre korektnú dynamickú zmenu pozadia. Tieto funkcie, rovnako ako aj pridanie menu pre ovládanie zmeny pozadia v aplikácií, bolo pridané do triedy

src/QOSG/CoreWindow.cpp. V triede *src/Viewer/CoreGraph.cpp* bola zmenená metóda *updateBackground* tak aby pracovala s premennou *bgVal*, ktorá reprezentuje číselnú hodnotu naviazanú na pozadie, na ktoré má nastať zmena.

```
int CoreGraph::updateBackground(int bgVal, Data::Graph* currentGraph) {
    LOG(INFO) << "CoreGraph::updateBackground - updating background";

    osg::Group* root = this->getScene();
    if (root->removeChild(root->getNumChildren()-1) == true) {

        if (bgVal == 0) { // default skybox
            SkyBox* skyBox = new SkyBox;
            root->addChild(skyBox->createSkyBox(0));
        }
        else if (bgVal == 1) { // noise skybox
            root->addChild(createSkyNoiseBox());
        }
#ifdef OPENCV_FOUND
        else if (bgVal == 2) {
            root->addChild(createTextureBackground());
        }
        else if (bgVal == 3) {
            root->addChild(createOrtho2dBackground());
        }
#endif
        else if (bgVal == -1) {
            SkyBox* skyBox = new SkyBox;
            root->addChild(skyBox->createSkyBox(-1)); // black skybox
        }
        else if (bgVal == -2) {
            SkyBox* skyBox = new SkyBox;
            root->addChild(skyBox->createSkyBox(-2)); // white skybox
        }

        reload(currentGraph);
        return 0;
    }

    return 1;
}
```

PRE POUŽÍVATEL'OV

1.1 Inštalačný návod (VC10)

v1.5 (17. 11. 2015)

1.1.1 Changelog

v1.6 - 13.03.2016

- návod pre MS VisualStudio 2013
- aktualizovaný Qt (5.5.1)

v1.5 – 17.11.2015

- aktualizované Aruco využíva FreeGlut
- upravené nastavenie pre build v QtCreator

v1.4 - 27.10.2015

- aktualizovaná knižnica OSG (v3.4) – nahradené zdroje
- zmazaný import chýbajúcich knižníc

v1.3 - 11.10.2015

- pridaný krok vytvorenia build konfigurácie

v1.2 - 06.10.2015

- pridaný postup pre buildovanie OSG knižníc
- pripojený zdroj s uploadnutými OSG knižnicami
- zvýraznené dôležité poznámky

v1.1 - 28.09.2015

- doplnené obrázky
- aktualizované zdroje
- vyriešené chyby
- zmenené poradie krokov

1.1.2 Časté problémy

Inštalácia softvéru 3DSoftViz pozostáva z mnohých krokov. Pozorné postupovanie podľa inštrukcií je nevyhnutné, inak sa môžu vyskytnúť problémy. Nasledujúce problémy sme identifikovali.

Ponuka generátorov je prázdna

Ak sa generátor nedá vybrať, je dôležité zmazať súbor CMakeLists.txt.user.2.5pre1 z *%3DSoftViz%* a zopakovať predošlý krok.

CMake - chýbajúce moduly

1. cez git shell prejsť do *%3DSoftViz%*
2. uistiť sa, že je nastavený na master vetvu
 - na prepnutie: git checkout master
3. zadať príkaz: git submodule update --init

CMake - cesta nebola nájdená

Ak nebola nájdená cesta k niektorému modulu, treba skontrolovať nastavenia premenných v RapidEE.

Chyba spojená s glut.h, glut.lib, glut.dll

Pokiaľ sa vyskytnú problémy typu:

- nevie nájsť glut.h
- hlási chyby priamo v glut.h
- niečo iné s glut.h, glut.lib alebo glut.dll

Tak najjednoduchšie riešenie je premenovať:

```
- %OSG_DIR%/ThirdParty/VC10/x86/include/GL/glut.h > ../glut.h.bak  
- %OSG_DIR%/ThirdParty/VC10/x86/lib/glut32.lib > ../glut32.lib.bak  
- %OSG_DIR%/ThirdParty/VC10/x86/lib/glut32D.lib > ../glut32D.lib.bak
```

Pri spustení crash programu s nullpoint exception

Pokiaľ sa projekt spustí iba cez argument: `-DCMAKE_BUILD_TYPE=Release`, ale cez `-DCMAKE_BUILD_TYPE=Debug` hlási nullpoint exception pri načítavaní obrázkov na pozadie a program crashne. Tak riešením je znovu stiahnuť/nakonfigurovať OSG podľa návodu.

Zm### pri buildovaní

Ak došlo k erroru pri buildovaní: `-Zm###`, treba vykonať úpravu v CMakeLists.txt > riadok 128 zmeniť /Zm216 na /Zm240 a spustiť CMake odznovu.

1.1.3 Návod pre Windows

Tento návod bol úspešne otestovaný na operačných systémoch Windows 7, 8, 8.1, 10.

Potrebný softvér

Na inštaláciu potrebujeme klonovať projekt 3DSoftViz (Tím č.4 klonuje z /cimox/3dsoftviz) z Githubu a stiahnuť nasledovné:

- CMake (v3.5.0)
- OpenSceneGraph (*jeden z nasledujúcich*)
 - OpenSceneGraph ([source](#)) - iba zdrojové súbory -> treba buildnúť (cca 40-50min)
 - OpenSceneGraph (v3.4.0) - buildnuté (17.10.2015)
 - OpenSceneGraph (v3.2.1)
- Kinect for Windows SDK 1.8
- Microsoft VisualStudio 2010 SP1 (**okrem Express edition!**)
- Qt (v4.8.5)
- QtCreator (v3.6.1)
- OpenCV (v2.4.10)
- Boost (v1.57.0)
- RapidEE - program na prácu s premennými prostredia
- Inštalácia knižnice 3rd party dependencies VC10
- OpenNI2
- NiTE2
- Debugging Tools for Windows
 - WinDbg - Win 8.1
 - WinDbg - Win 10
- FreeGlut

Postup inštalácie

1. Nainštalovať CMake. (Cesta je v dokumente označená ako %CMAKE_DIR%)
2. Nainštalovať Qt (%QT_DIR%)
3. Nainštalovať QtCreator do zložky Qt
4. Vytvoriť zložku OpenSceneGraph (%OSG_DIR%) a %OSG_DIR%/ThirdParty
5. Rozbaliť 3rd Party Knižnice (VC10) do %OSG_DIR%/ThirdParty/
6. Rozbaliť FreeGlut do %OSG_DIR%/ThirdParty/VC10/x86
7. V prípade stiahnutia zbuildovaných súborov OSG (mega.nz)
 - (a) Rozbaliť zložky build a install do %OSG_DIR%
 - (b) Vynechať nasledujúci krok.

8. V prípade stiahnutia iba zdrojových súborov OSG (oficiálna stránka)

- (a) Rozbalit' OSG_3.4 do *%OSG_DIR%*
- (b) Vytvorit' zložku build a install v *%OSG_DIR%*
- (c) Premenovať súbory:

```
%OSG_DIR%/ThirdParty/VC10/x86/include/GL/glut.h > ../glut.h.bak  
%OSG_DIR%/ThirdParty/VC10/x86/lib/glut32.lib > ../glut32.lib.bak  
%OSG_DIR%/ThirdParty/VC10/x86/lib/glut32D.lib > ../glut32D.lib.bak
```

- (d) Spustiť CMake (cmake-gui.exe)
 - i. source code > *%OSG_DIR%*/OSG_3.4
 - ii. binaries > *%OSG_DIR%*/build
 - iii. stlačiť Configure (VS2010 kompilátor)
 - iv. stlačiť Generate
 - ak došlo k erroru: File > Delete Cache a skúsiť znovu
- (e) Nájsť súbor OpenSceneGraph.sln v *%OSG_DIR%*/build
- (f) Otvoriť súbor vo VS2010
- (g) Nastaviť Solution Configuration na Debug
- (h) Nájsť projekt ALL_BUILD > pravý klik > build
- (i) Po skončení nájsť projekt INSTALL > pravý klik > build
- (j) Nastaviť Solution Configuration na Release
- (k) Nájsť projekt ALL_BUILD > pravý klik > build
- (l) Po skončení nájsť projekt INSTALL > pravý klik > build
- (m) Presunúť nainštalované súbory (default c:/Program Files (x86)/OpenSceneGraph) do *%OSG_DIR%*/install

9. Rozbalit' FreeGlut do *%OSG_DIR%*/ThirdParty/VC10/x86/ (prepísať súbory)

10. Nainštalovať OpenCV (*%OPENCV_DIR%*)

11. Rozbalit' Boost (*%BOOST_DIR%*)

Ideálne je mať všetko na spoločnom mieste kvôli prehľadnosti, napr.

12. Nainštalovať a otvoriť RapidEE, v ktorom sa vykonajú tieto zmeny:

- (a) do PATH pridať premenné:
 - i. *%CMAKE_DIR%*/bin
 - ii. *%QT_DIR%*/bin
 - iii. *%QT_DIR%*/Qtcreator/bin
 - iv. *%OSG_DIR%*/build/bin
 - v. *%OSG_DIR%*/ThirdParty/VC10/x86/bin
 - vi. *%OPENCV_DIR%*/build/x86/vc10/bin
- (b) Vytvorit' premennú CMAKE_INCLUDE_PATH a pridať:

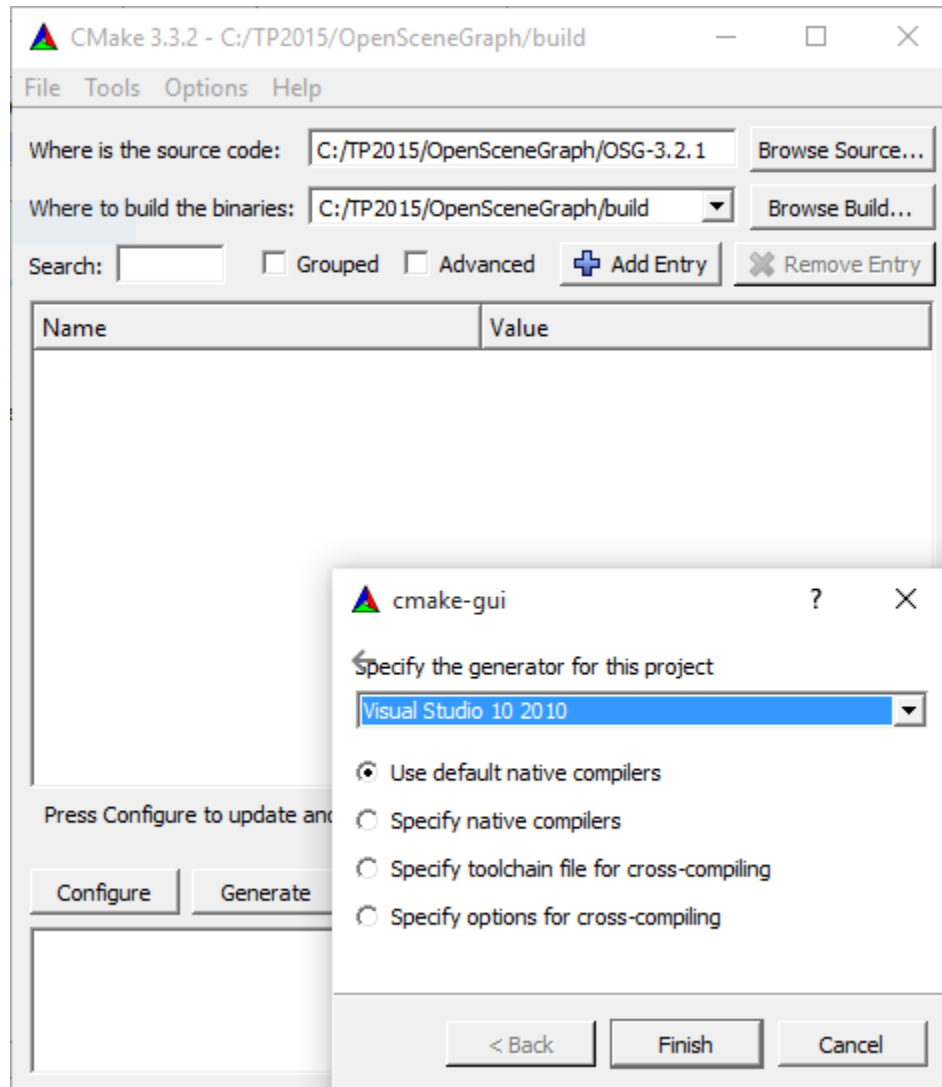


Fig. 1.1: CMake pre OSG

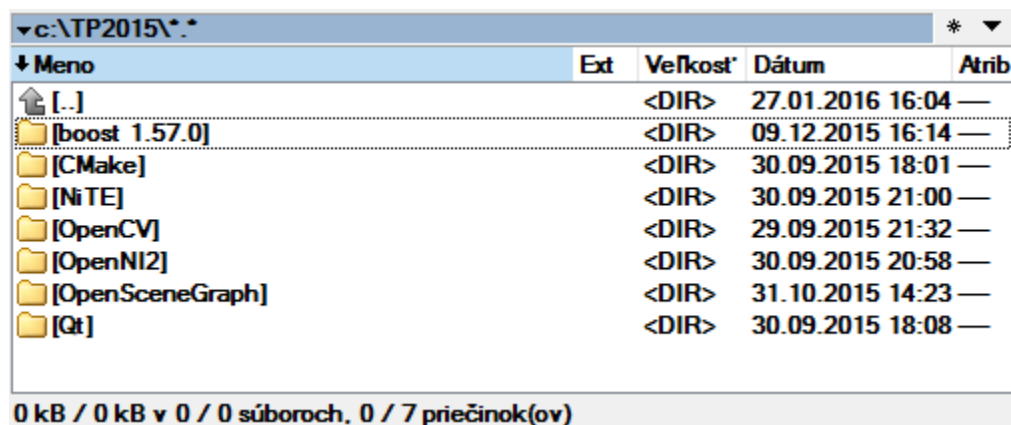


Fig. 1.2: Nainštalovaný SW

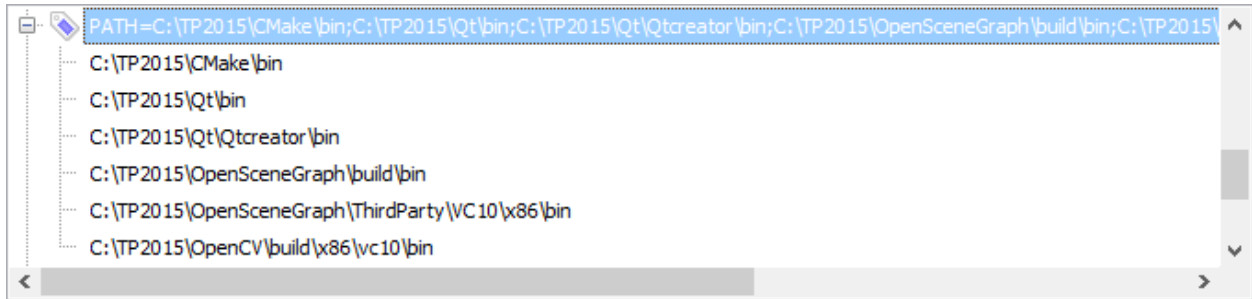


Fig. 1.3: PATH premenná

- i. *%OSG_DIR%/install/include*
- ii. *%OSG_DIR%/ThirdParty/VC10/x86/include*
- iii. *%OPENCV_DIR%/build/include*

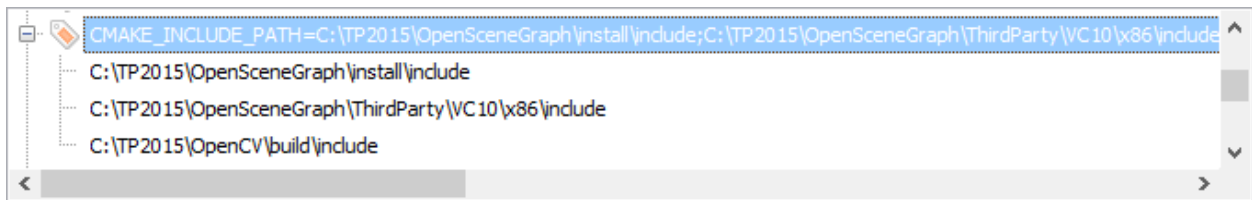


Fig. 1.4: CMAKE_INCLUDE_PATH premenná

(c) Vytvorit' premennú CMAKE_LIBRARY_PATH a pridať:

- i. *%OSG_DIR%/build/lib*
- ii. *%OSG_DIR%/install/lib*
- iii. *%OSG_DIR%/ThirdParty/VC10/x86/lib*
- iv. *%OPENCV_DIR%/build/x86/vc10/lib*

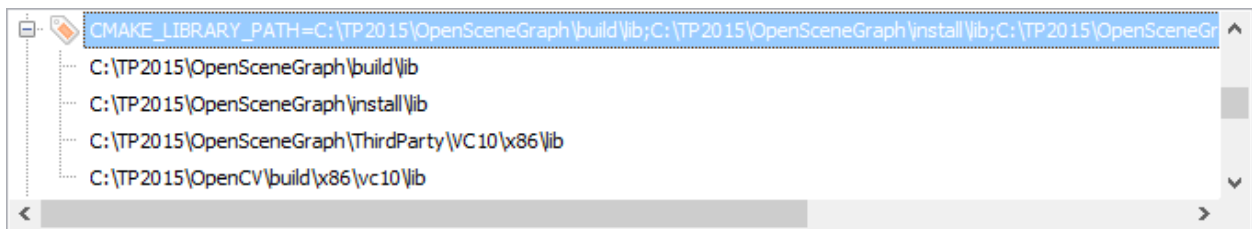


Fig. 1.5: CMAKE_LIBRARY_PATH premenná

(d) Vytvorit' premennú BOOST_INCLUDEDIR a pridať: *%BOOST_DIR%/boost*

(e) Vytvorit' premennú BOOST_LIBRARYDIR a pridať: *%BOOST_DIR%/libs*

(f) Vytvorit' premennú BOOST_ROOT a pridať: *%BOOST_DIR%*

(g) Vytvorit' premennú OPENCV_DIR a pridať: *%OPENCV_DIR%/build*

13. Naklónovať projekt 3DSoftViz cez git shell (*%3DSoftViz%*)

14. Vytvorit' v priečinku *%3DSoftViz%* priečinky *_build* a *_install*

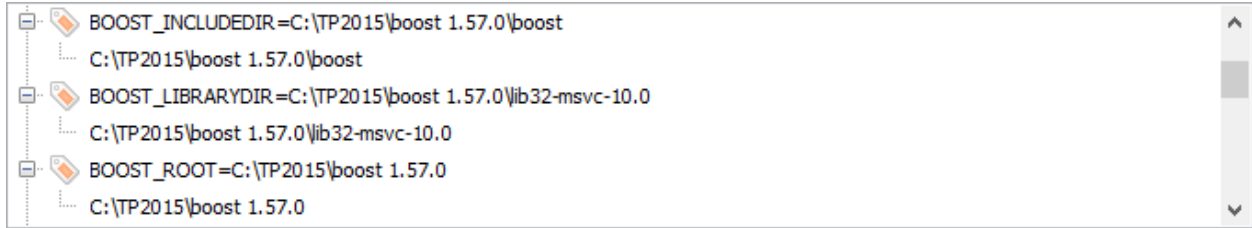


Fig. 1.6: BOOST premenné

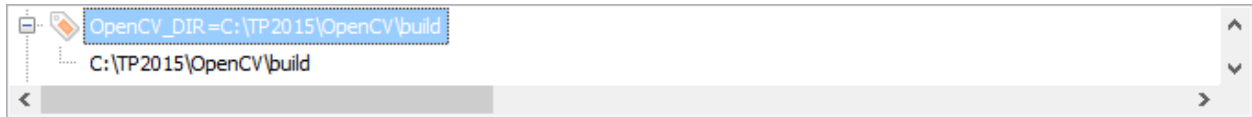


Fig. 1.7: OPENCV_DIR premenná

15. Spustiť QtCreator. Tools > Options... > Build and Run:

- (a) záložka CMake – zadať cestu `%CMAKE_DIR%/bin/cmake.exe`
- (b) záložka Compilers – ak existuje VS2010 tak sú autodetected
- (c) záložka Qt Versions – zadať cestu `%QT_DIR%/bin/qmake.exe`
- (d) záložka Kits – vytvoriť nový a vybrať hodnoty nasledovne:

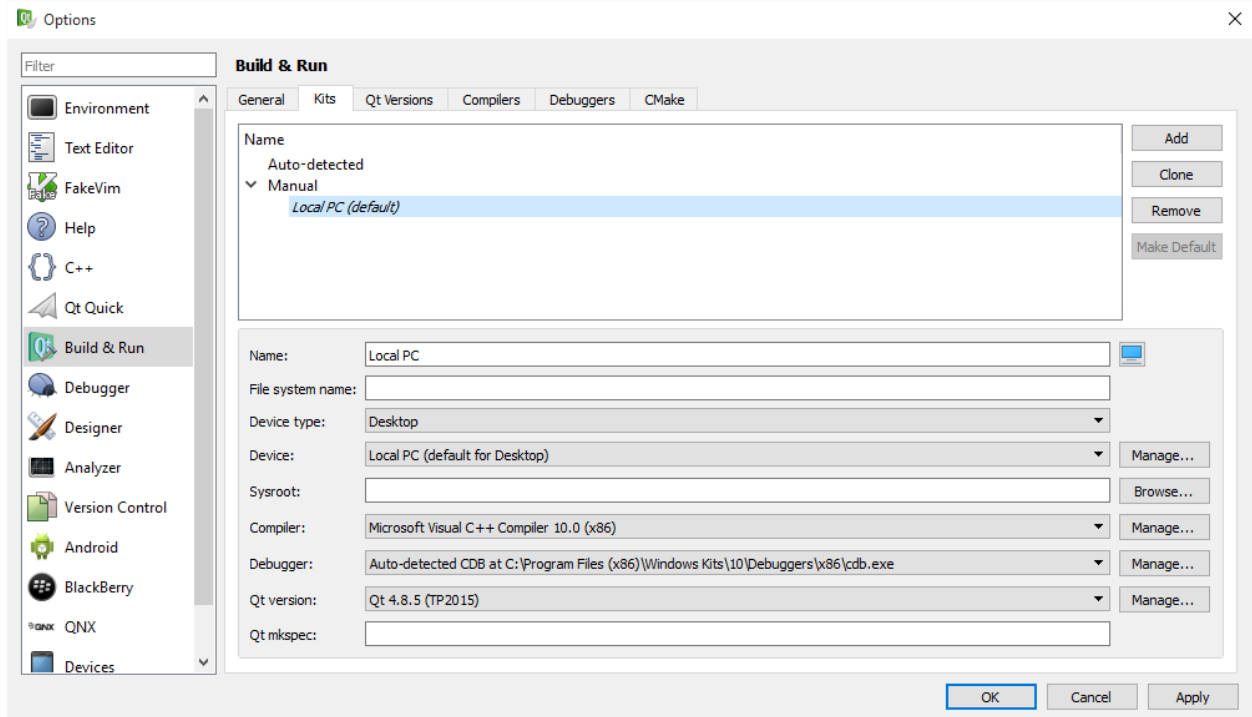


Fig. 1.8: QtC Kits nastavenia

- (e) záložka General – nastaviť default build directory: `%3DSoftViz%/_build`
- (f) Potvrdiť – OK

16. File > Open File or Project... > vybrať CMakeLists.txt z %3DSoftViz%
17. Zadať do poľa Arguments jeden z nasledujúcich prepínačov:
 - -DCMAKE_BUILD_TYPE=Debug
 - -DCMAKE_BUILD_TYPE=Release
18. Vybrať z nastavený generátor – NMake Generator (názov kitu)
(Chyba: Generátor nebol nájdený <riešenie>)
19. Stlačiť Run CMake

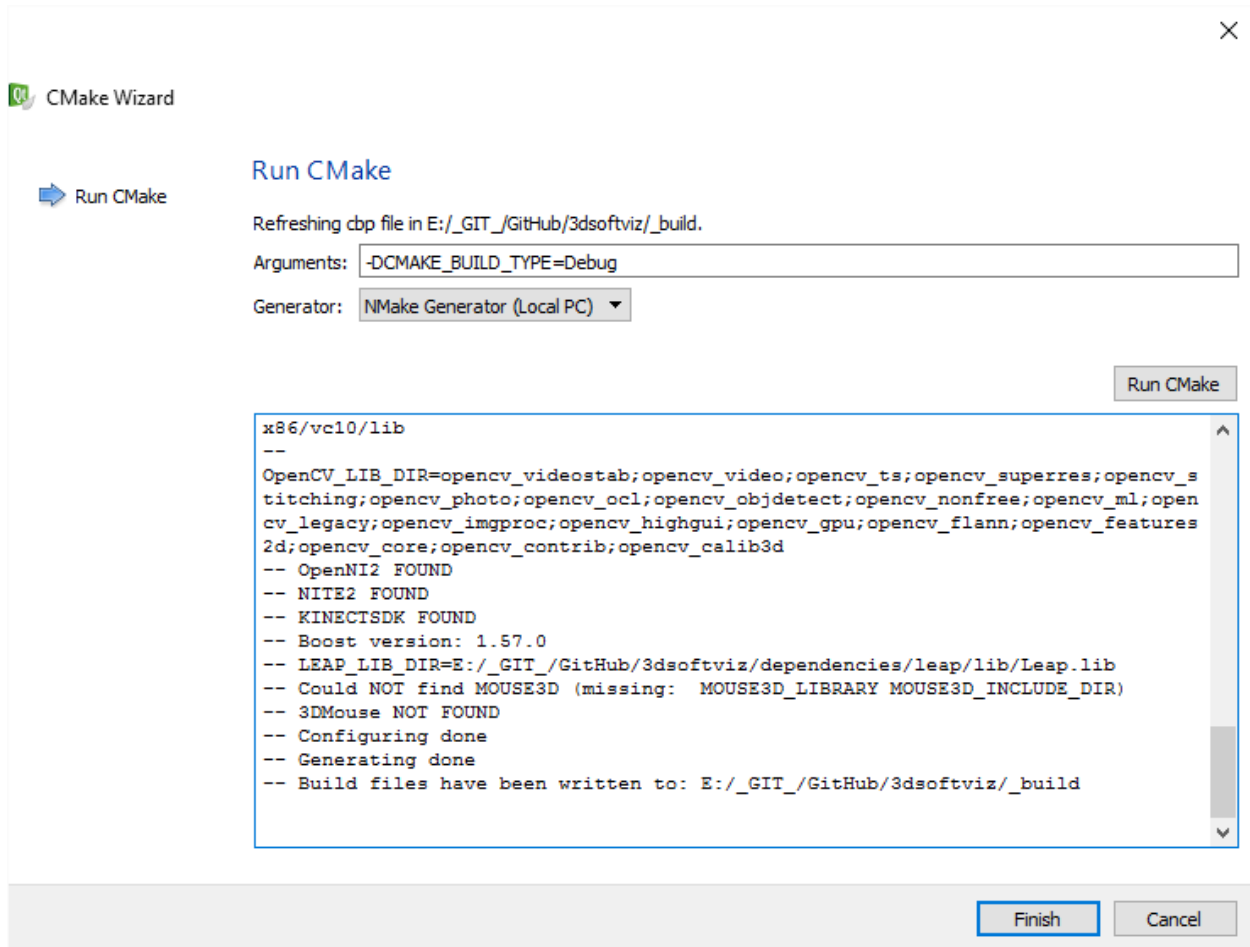


Fig. 1.9: QtC CMake wizard

(Chyba: Chýbajúce moduly <riešenie>)

(Chyba: Cesta nebola nájdená <riešenie>)

20. Ukončiť – Finish
21. Vybrať Projects > Build & Run > Build, v časti Edit build configuration kliknúť na Add > Clone selected, nazvať napr. „unity“
22. Prejsť na vytvorený build config. „unity“, v časti Build Steps otvoriť Details a zaškrtnúť pri build step *jom.exe* možnosť *install_unity*

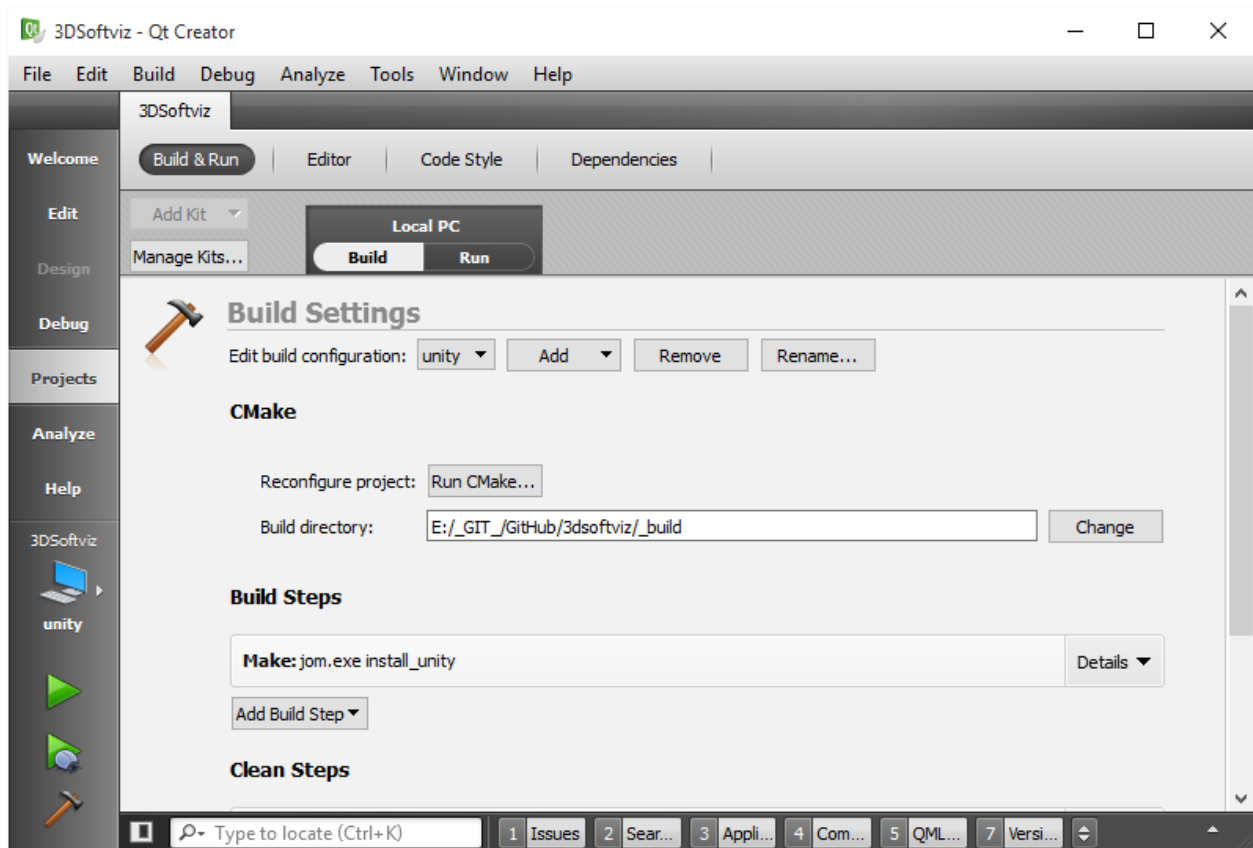


Fig. 1.10: QtC build project

23. Skontrolovať nastavenie build config – unity

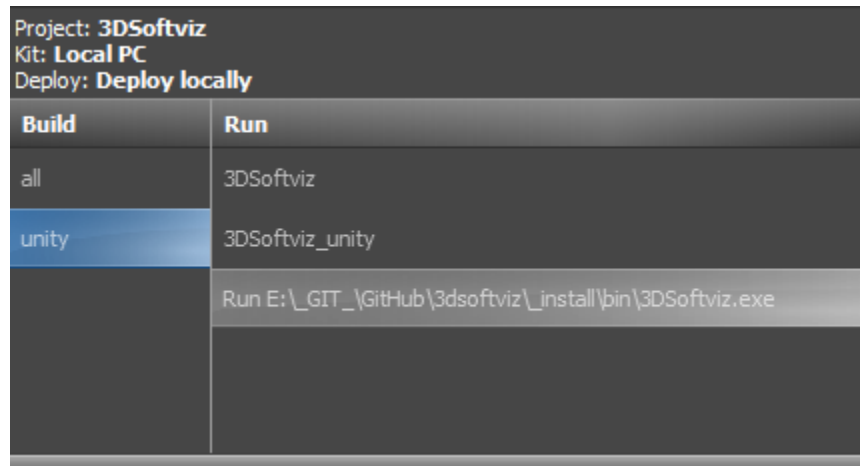


Fig. 1.11: QtC build config

24. Stačiť Build (kladivko vľavo dole)

25. Po úspešnom zbuildovaní vybrať Projects > Build & Run > Run, v časti Run pridať Add > Custom Executable a nastaviť:

(a) executable: `%3DSoftViz%/_install/bin/3DSoftviz.exe`

(b) working directory: `%3DSoftViz%/_install/bin/`

26. Skontrolovať nastavenie run config – zadaná cesta

27. Spustiť program pomocou zeleného tlačidla Run

1.1.4 Rozšírenie 3DSoftviz o Kinect

1. Nainštalovať Kinect for Windows

2. Skontrolovať v RapidEE či sa vytvorila premenná `%KINECTSDK10_DIR%`

3. Nainštalovať OpenNI2 (OpenNI-Windows-x86-2.2.msi)

- **x86! – inak sa môžu vyskytnúť problémy s linkovaním**

4. Skontrolovať v RapidEE či sa vytvorili premenné:

(a) `%OPENNI2_INCLUDE%`

(b) `%OPENNI2_LIB%`

(c) `%OPENNI2_REDIST%`

(d) `%OPENNI2_ROOT%`

5. Nainštalovať NiTE2 (NiTE-Windows-x86-2.2.msi)

- **x86! – inak sa môžu vyskytnúť problémy s linkovaním**

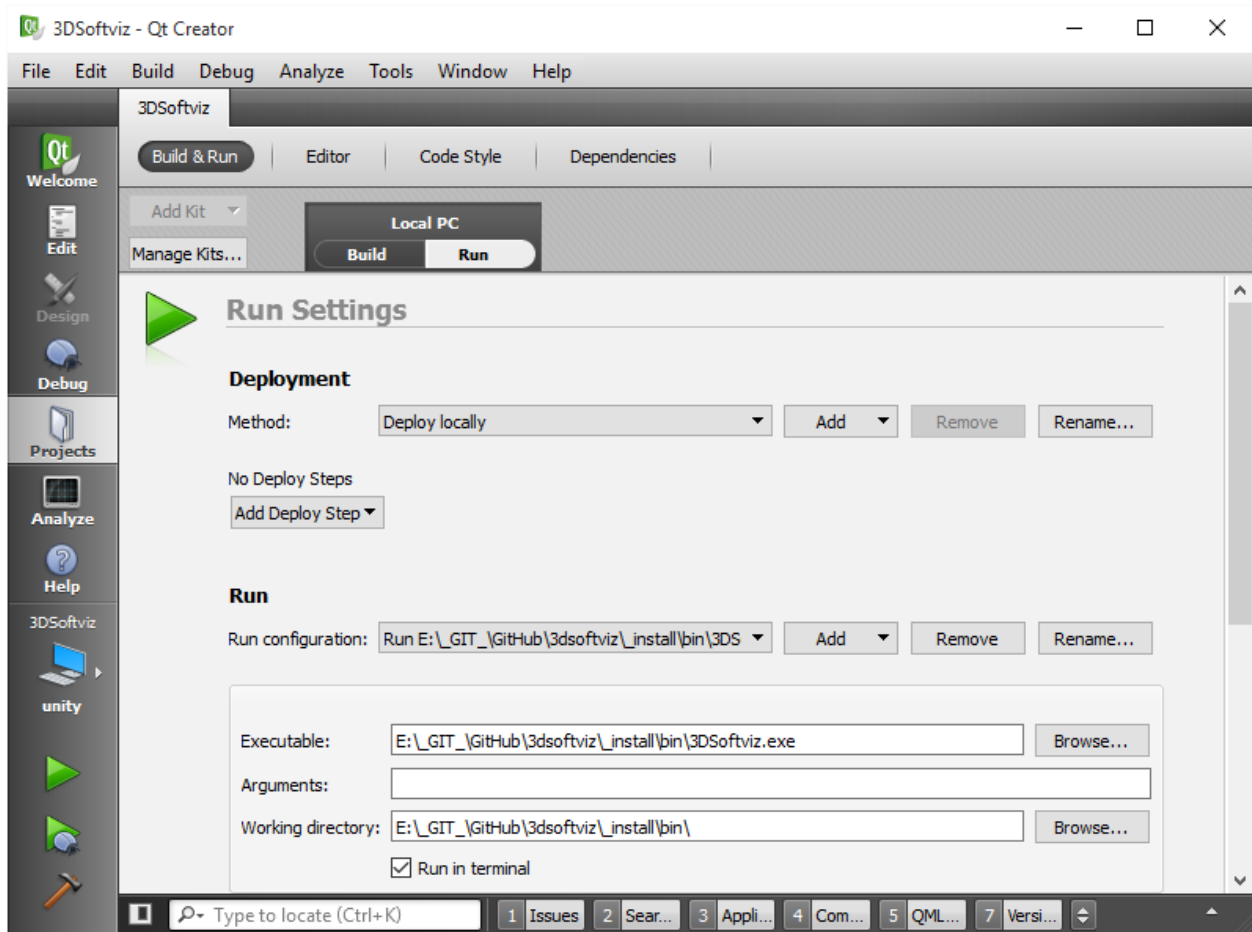


Fig. 1.12: QtC run project

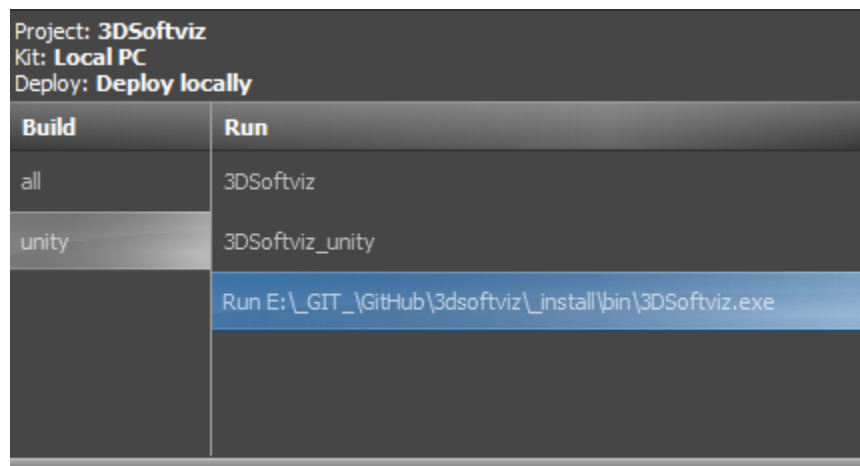


Fig. 1.13: QtC run config

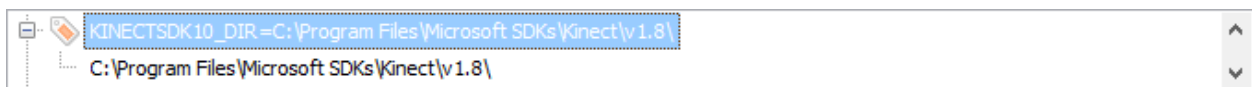


Fig. 1.14: KINECTSDK10_DIR premenná

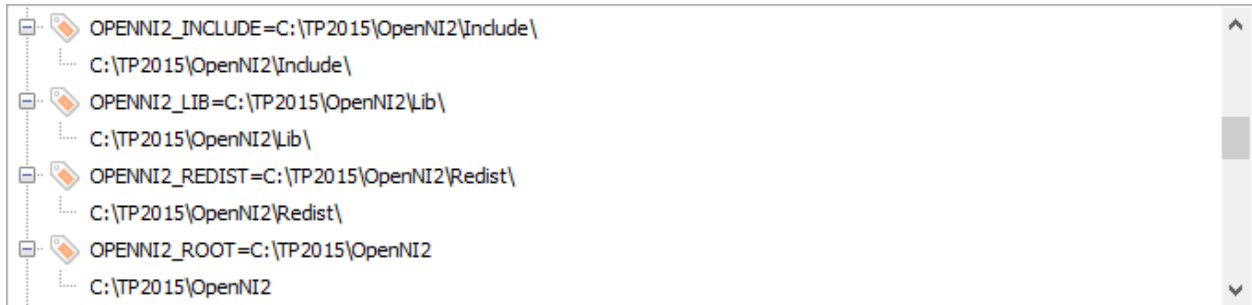


Fig. 1.15: NITE2 premenné

6. Skontrolovať v RapidEE či sa vytvorili premenné:

- (a) `%NITE2_INCLUDE%`
- (b) `%NITE2_LIB%`
- (c) `%NITE2_REDIST%`
- (d) `%NITE2_ROOT%`

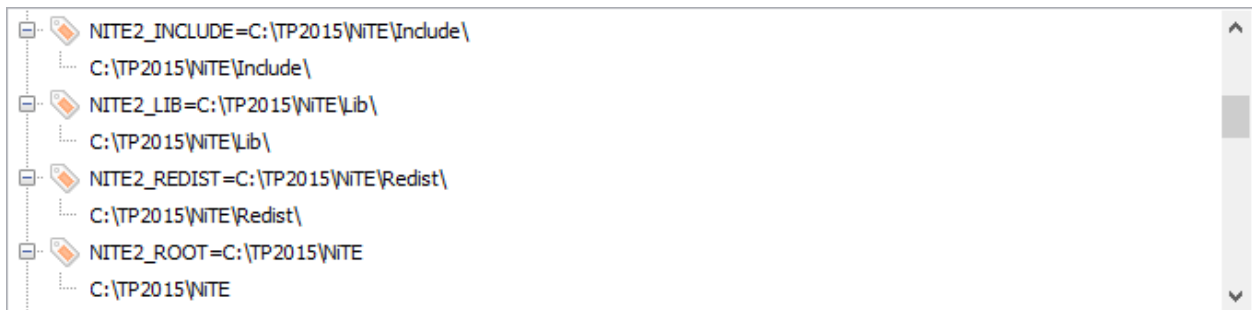


Fig. 1.16: OPENNI2 premenné

7. Pridať do premennej CMAKE_INCLUDE_PATH:

- (a) `%OPENNI2_INCLUDE%`
- (b) `%NITE2_INCLUDE%`

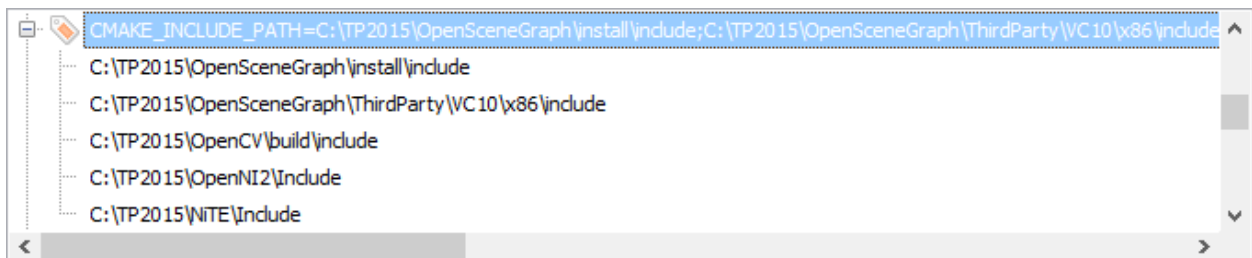


Fig. 1.17: OPENNI2 premenné

8. Pridať do premennej CMAKE_LIBRARY_PATH:

- (a) `%OPENNI2_ROOT%/Driver`

- (b) `%OPENNI2_REDIST%`
- (c) `%OPENNI2_REDIST%/OpenNI2/Drivers`
- (d) `%OPENNI2_LIB%`
- (e) `%NITE2_ROOT%/Samples/Bin/OpenNI2/Drivers`
- (f) `%NITE2_LIB%`

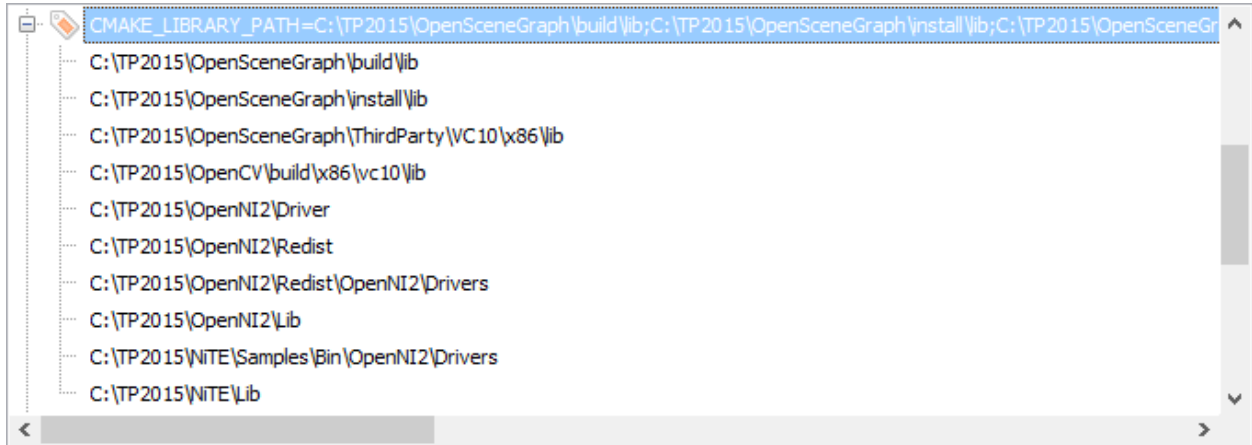


Fig. 1.18: OPENNI2 premenné

9. Pridať do premennej PATH:

- (a) `%OPENNI2_REDIST%/OpenNI2/Drivers`
- (b) `%OPENNI2_REDIST%`
- (c) `%NITE2_REDIST%`
- (d) `%NITE2_ROOT%/Samples/Bin`

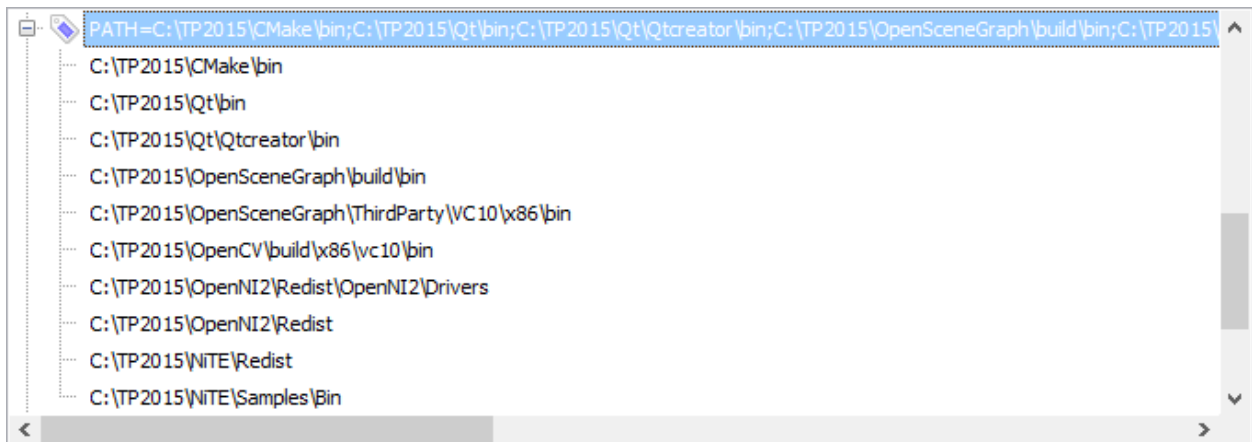


Fig. 1.19: OPENNI2 premenné

10. Spustiť CMake a skontrolovať vo výpise:

- (a) OpenNI2 FOUND

- (b) NITE2 FOUND
- (c) KINECTSDK FOUND

1.1.5 Nastavenie debugera v QtCreator

1. Nainštalovať WinDbg
2. Skontrolovať v QtCreator Tools > Options > Build & Run > záložka Debuggers či sú autodetected

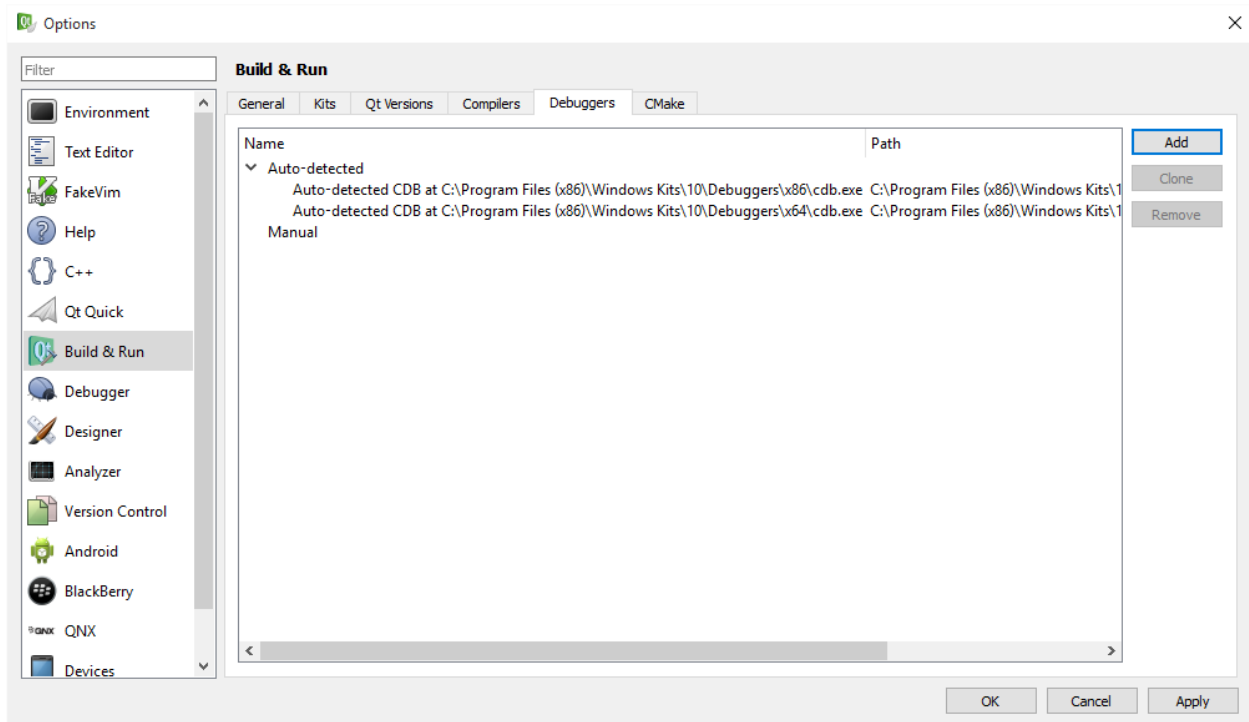


Fig. 1.20: QtC debugger nastavenia

3. Pridať do QtCreator Tools > Options > Build & Run > záložka Kits pre vytvorený profil položku Debugger (x86)
4. Spustiť CMake (-DCMAKE_BUILD_TYPE=Debug)
5. Zvoliť možnosť Debug (vľavo dole medzi Run a Build)

1.2 Inštačný návod (VC12)

v1.6 (13. 03. 2016)

1.2.1 Návod pre Windows

Tento návod bol úspešne otestovaný na operačných systémoch Windows 7, 8, 8.1, 10.

Potrebný softvér

Na inštaláciu potrebujeme klonovať projekt 3DSoftViz (Tím č.4 klonuje z /cimox/3dsoftviz) z Githubu a stiahnuť nasledovné:

- CMake (v3.5.0)
- OpenSceneGraph (*jeden z nasledujúcich*)
 - OpenSceneGraph ([source](#)) - iba zdrojové súbory -> treba buildnúť (cca 40-60min)
 - stable release: OpenSceneGraph (v3.4.0) - buildnuté (13.03.2016)
 - developer release: OpenSceneGraph (v3.5.1) - buildnuté (13.03.2016)
- Kinect for Windows SDK 1.8
- Microsoft VisualStudio 2013 (**okrem Express edition!**)
- Qt (v4.8.6)
- QtCreator (v3.6.1)
- OpenCV (v2.4.10)
- Boost (v1.57.0)
- RapidEE - program na prácu s premennými prostredia
- Inštalácia knižnice 3rd party dependencies VC12
- OpenNI2
- NiTE2
- Debugging Tools for Windows
 - WinDbg - Win 8.1
 - WinDbg - Win 10

Postup inštalácie

1. Nainštalovať CMake. (Cesta je v dokumente označená ako %CMAKE_DIR%)
2. Nainštalovať Qt (%QT_DIR%)
3. Nainštalovať QtCreator do zložky Qt
4. Vytvoriť zložku OpenSceneGraph (%OSG_DIR%)
5. V prípade stiahnutia zbuildovaných súborov OSG (mega.nz)
 - (a) Rozbalit' zložky build a install do %OSG_DIR%
 - (b) Vynechať nasledujúci krok.
6. V prípade stiahnutia iba zdrojových súborov OSG (oficiálna stránka)
 - (a) Rozbalit' OSG do %OSG_DIR%/source
 - (b) Vytvoriť zložku build a install v %OSG_DIR%
 - (c) Rozbalit' 3rd Party Knižnice (VC12) do %OSG_DIR%/ThirdParty/
 - (d) Spustiť CMake (cmake-gui.exe)
 - i. source code > %OSG_DIR%/source

- ii. binaries > %OSG_DIR%/build
- iii. stlačiť Configure (VS2013 kompilátor)
- iv. nastaviť 3rdParty na %OSG_DIR%/ThirdParty/VC12/X86
- v. stlačiť Generate
 - ak došlo k erroru: File > Delete Cache a skúsiť znovu

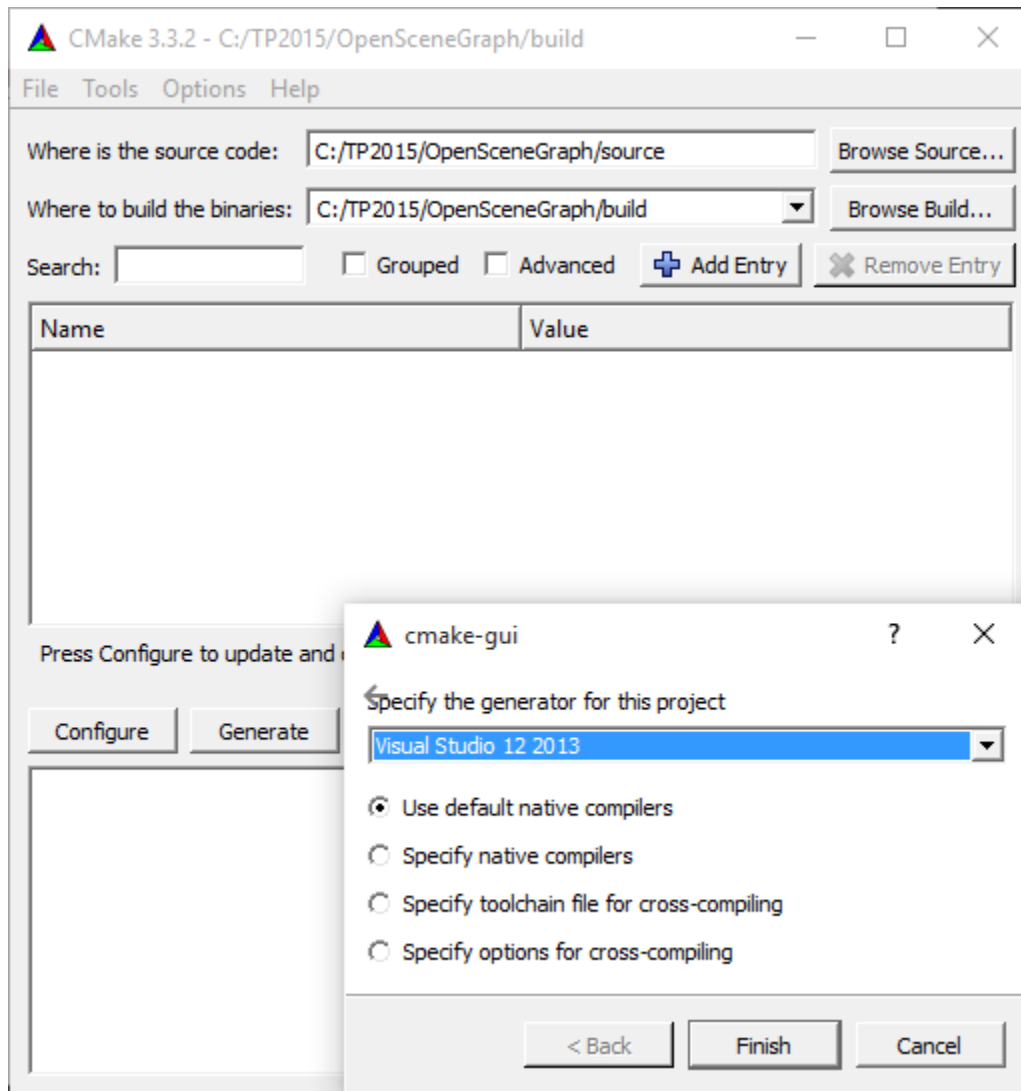


Fig. 1.21: CMake pre OSG

- (e) Nájst' súbor OpenSceneGraph.sln v %OSG_DIR%/build
- (f) Otvoriť súbor vo VS2013 (**ako správca!**)
- (g) Nastaviť Solution Configuration na Debug
- (h) Nájst' projekt ALL_BUILD > pravý klik > build
- (i) Po skončení nájsť projekt INSTALL > pravý klik > build
- (j) Nastaviť Solution Configuration na Release

- (k) Nájst' projekt ALL_BUILD > pravý klik > build
 - (l) Po skončení nájsť projekt INSTALL > pravý klik > build
 - (m) Presunúť nainštalované súbory (default c:Program Files (x86)OpenSceneGraph) do *%OSG_DIR%/install*
7. Nainštalovať OpenCV (*%OPENCV_DIR%*)
 8. Rozbalit' Boost (*%BOOST_DIR%*)

Ideálne je mať všetko na spoločnom mieste kvôli prehľadnosti, napr.

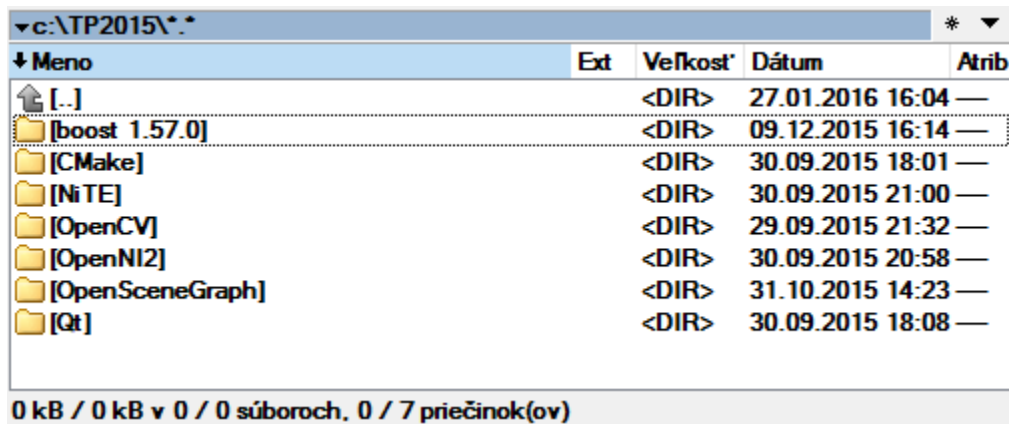


Fig. 1.22: Nainštalovaný SW

9. Nainštalovať a otvoriť RapidEE, v ktorom sa vykonajú tieto zmeny:
 - (a) do PATH pridať premenné:
 - i. *%CMAKE_DIR%/bin*
 - ii. *%QT_DIR%/bin*
 - iii. *%QT_DIR%/Qtcreator/bin*
 - iv. *%OSG_DIR%/build/bin*
 - v. *%OSG_DIR%/ThirdParty/VC12/x86/bin*
 - vi. *%OPENCV_DIR%/build/x86/vc12/bin*

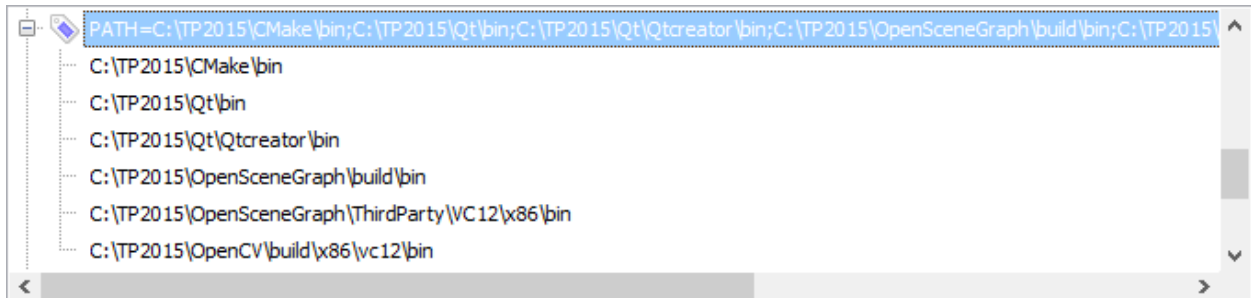


Fig. 1.23: PATH premenná

- (b) Vytvoriť premennú CMAKE_INCLUDE_PATH a pridať:
 - i. *%OSG_DIR%/install/include*

- ii. `%OSG_DIR%/ThirdParty/VC12/x86/include`
- iii. `%OPENCV_DIR%/build/include`

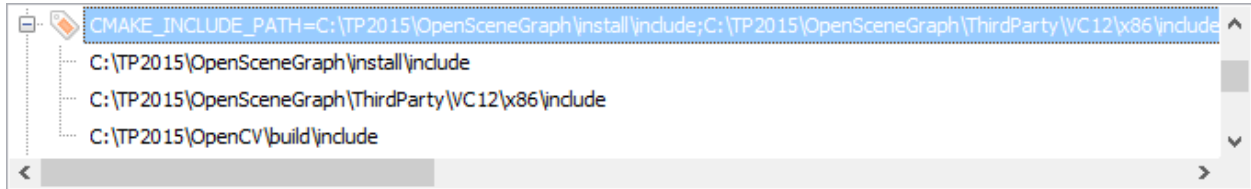


Fig. 1.24: CMAKE_INCLUDE_PATH premenná

(c) Vytvorit' premennú CMAKE_LIBRARY_PATH a pridať:

- i. `%OSG_DIR%/build/lib`
- ii. `%OSG_DIR%/install/lib`
- iii. `%OSG_DIR%/ThirdParty/VC12/x86/lib`
- iv. `%OPENCV_DIR%/build/x86/vc12/lib`

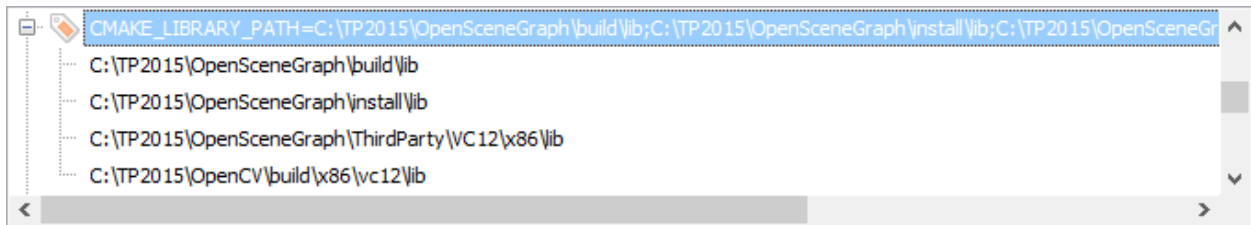


Fig. 1.25: CMAKE_LIBRARY_PATH premenná

(d) Vytvorit' premennú BOOST_INCLUDEDIR a pridať: `%BOOST_DIR%/boost`

(e) Vytvorit' premennú BOOST_LIBRARYDIR a pridať: `%BOOST_DIR%/libs`

(f) Vytvorit' premennú BOOST_ROOT a pridať: `%BOOST_DIR%`

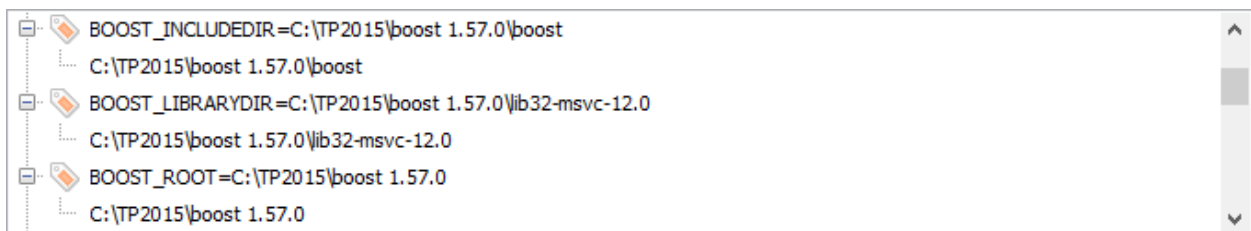


Fig. 1.26: BOOST premenne

(g) Vytvorit' premennú OPENCV_DIR a pridať: `%OPENCV_DIR%/build`

10. Naklónovať projekt 3DSoftViz cez git shell (`%3DSoftViz%`)

11. Vytvorit' v priečinku `%3DSoftViz%` priečinky `_build` a `_install`

12. Spustiť QtCreator. Tools > Options... > Build and Run:

- (a) záložka CMake – zadať cestu `%CMAKE_DIR%/bin/cmake.exe`
- (b) záložka Compilers – ak existuje VS2013 tak sú autodetected

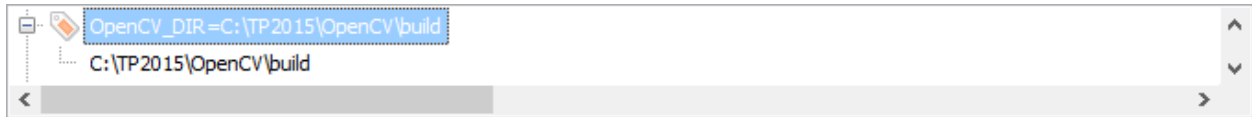


Fig. 1.27: OPENCV_DIR premenná

- (c) záložka Qt Versions – zadať cestu `%QT_DIR%/bin/qmake.exe`
 (d) záložka Kits – vytvoriť nový a vybrať hodnoty nasledovne:

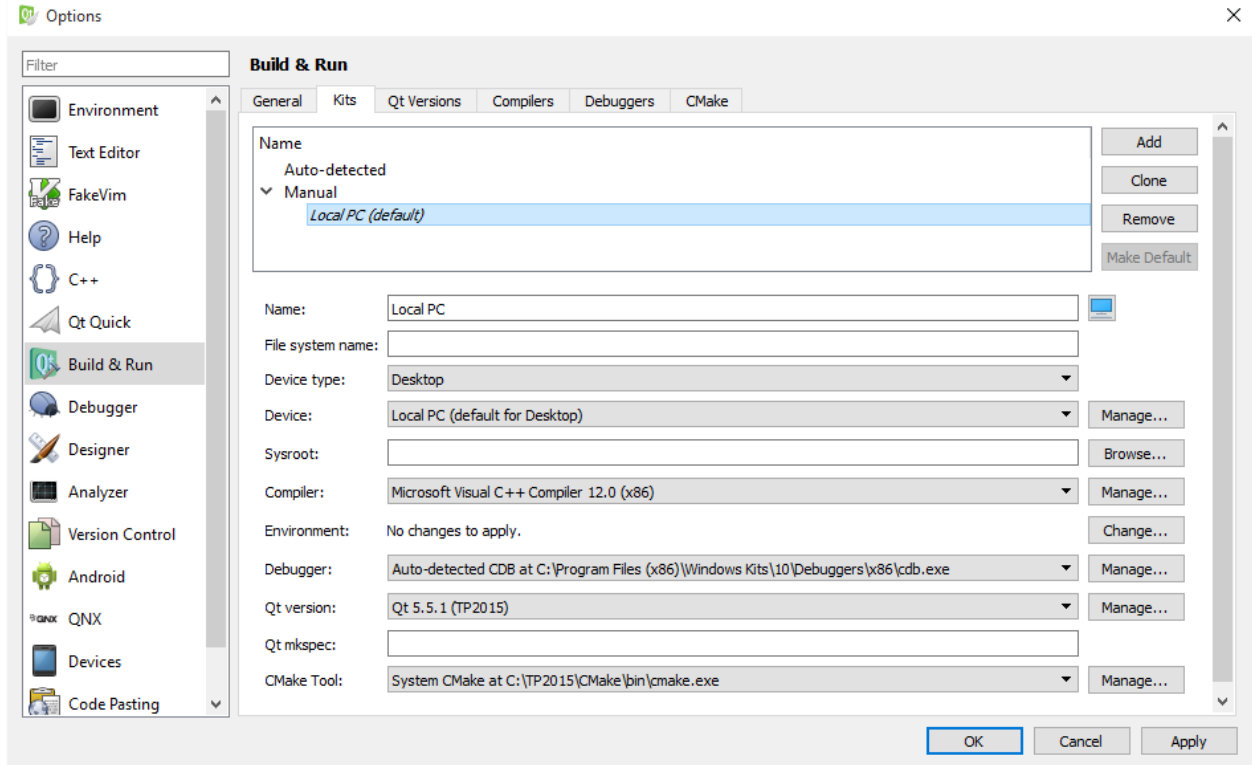


Fig. 1.28: QtC Kits nastavenia

- (e) záložka General – nastaviť default build directory: `%3DSoftViz%/_build`
 (f) Potvrdiť – OK
13. File > Open File or Project... > vybrať CMakeLists.txt z `%3DSoftViz%`
 14. Zadať do poľa Arguments jeden z nasledujúcich prepínačov:
 - `-DCMAKE_BUILD_TYPE=Debug`
 - `-DCMAKE_BUILD_TYPE=Release`
 15. Vybrať z nastavený generátor – NMake Generator (názov kitu)
 (Chyba: Generátor nebol nájdený <riešenie>)
 16. Stlačiť Run CMake
 (Chyba: Chýbajúce moduly <riešenie>)
 (Chyba: Cesta nebola nájdená <riešenie>)

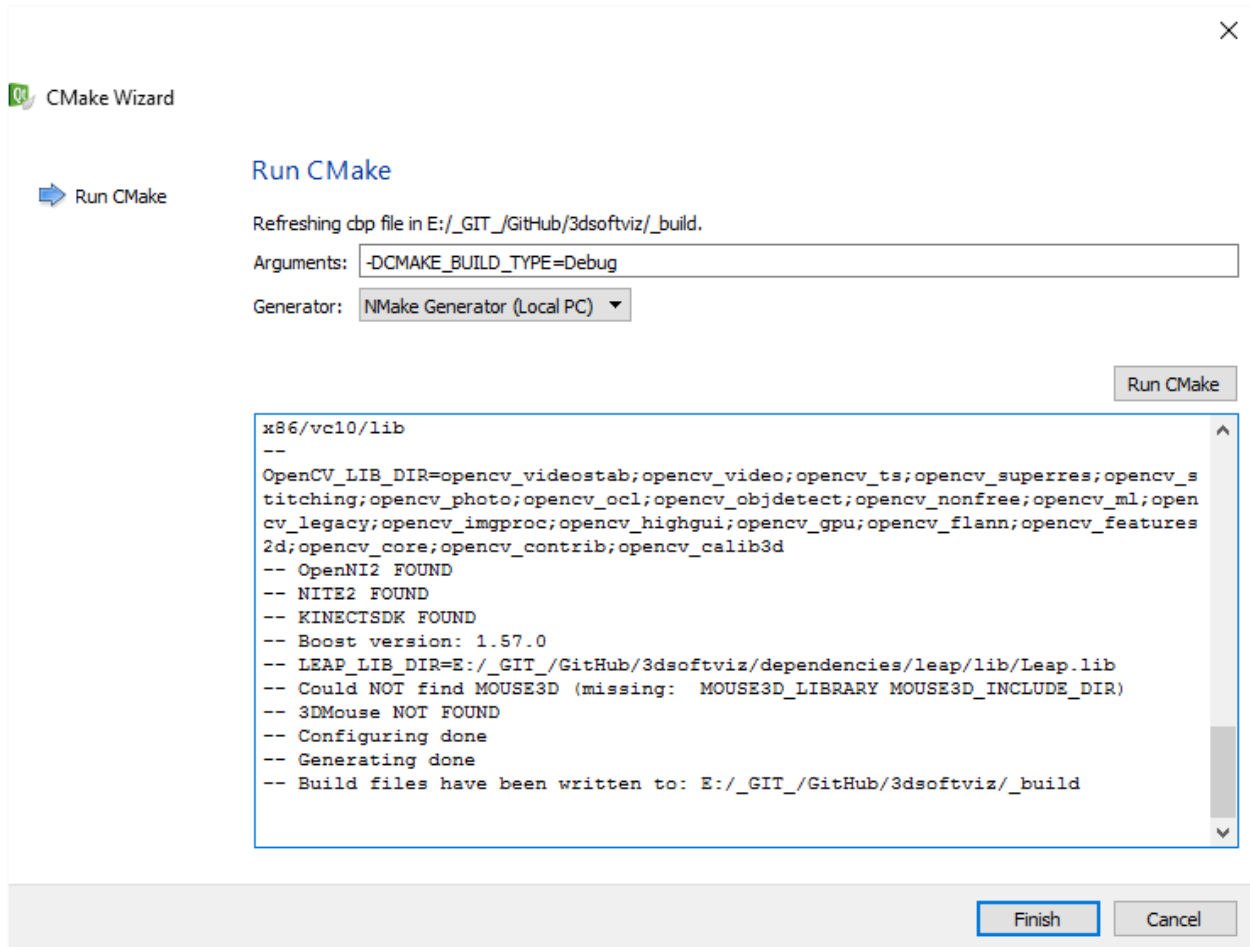


Fig. 1.29: QtCMake wizard

17. Ukončiť – Finish
18. Vybrať Projects > Build & Run > Build, v časti Edit build configuration kliknúť na Add > Clone selected, nazvať napr. „unity“
19. Prejsť na vytvorený build config. „unity“, v časti Build Steps otvoriť Details a zaškrtnúť pri build step *jom.exe* možnosť *install_unity*

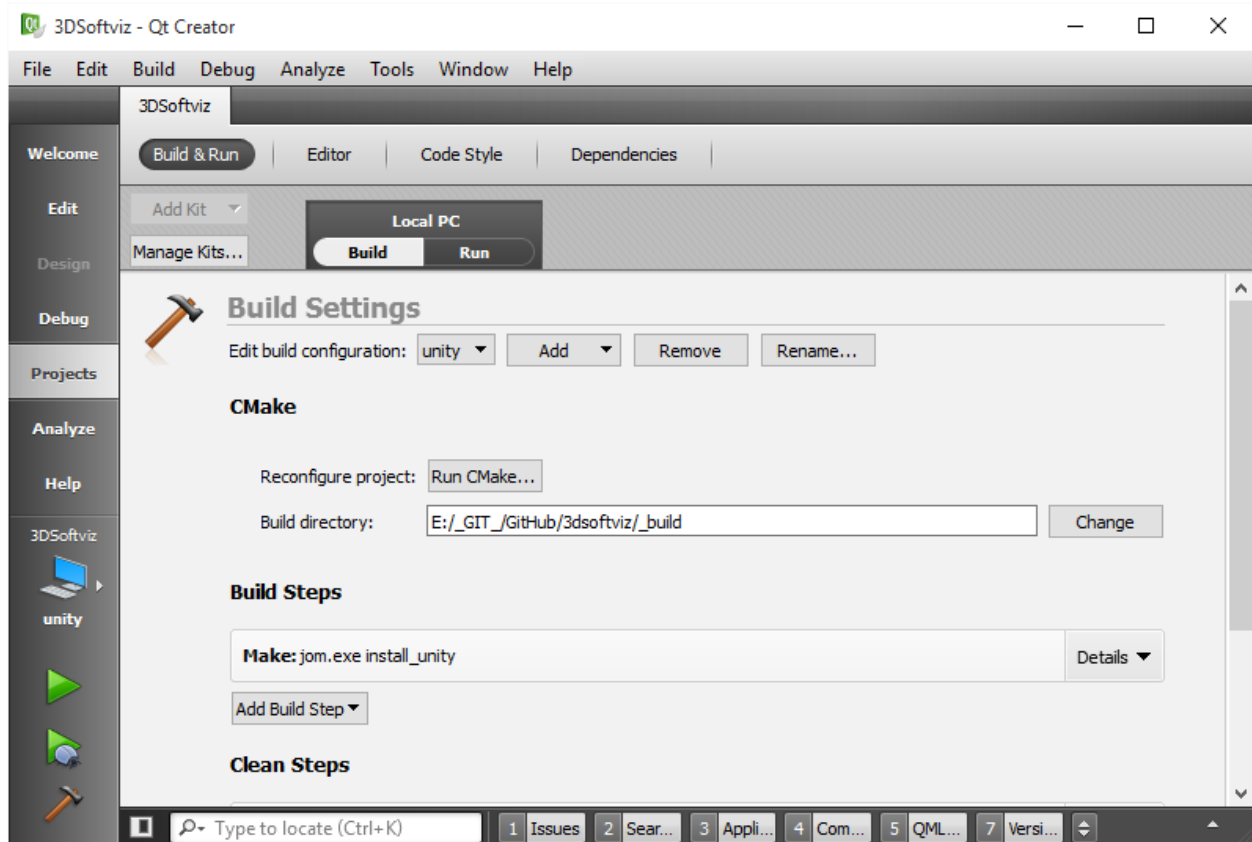


Fig. 1.30: QtC build project

20. Skontrolovať nastavenie build config – unity
21. Stačiť Build (kladivko vľavo dole)
22. Po úspešnom zbuildovaní vybrať Projects > Build & Run > Run, v časti Run pridať Add > Custom Executable a nastaviť:
 - (a) executable: `%3DSoftViz%/_install/bin/3DSoftviz.exe`
 - (b) working directory: `%3DSoftViz%/_install/bin/`
23. Skontrolovať nastavenie run config – zadaná cesta
24. Spustiť program pomocou zeleného tlačidla Run

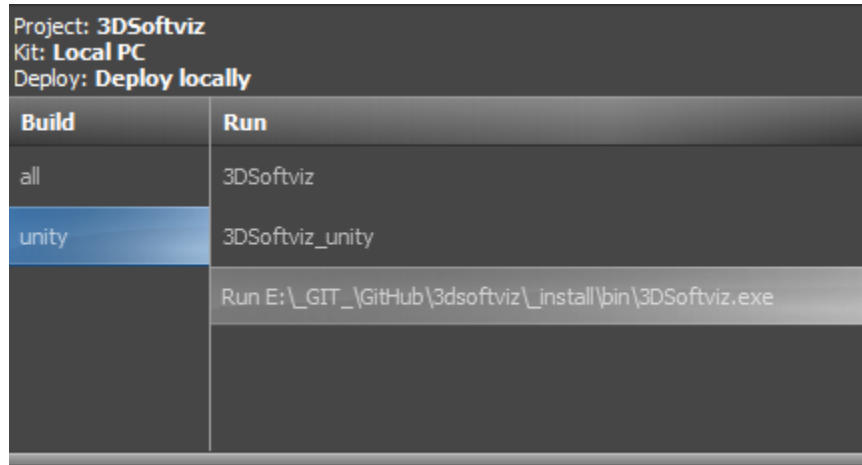


Fig. 1.31: QtC build config

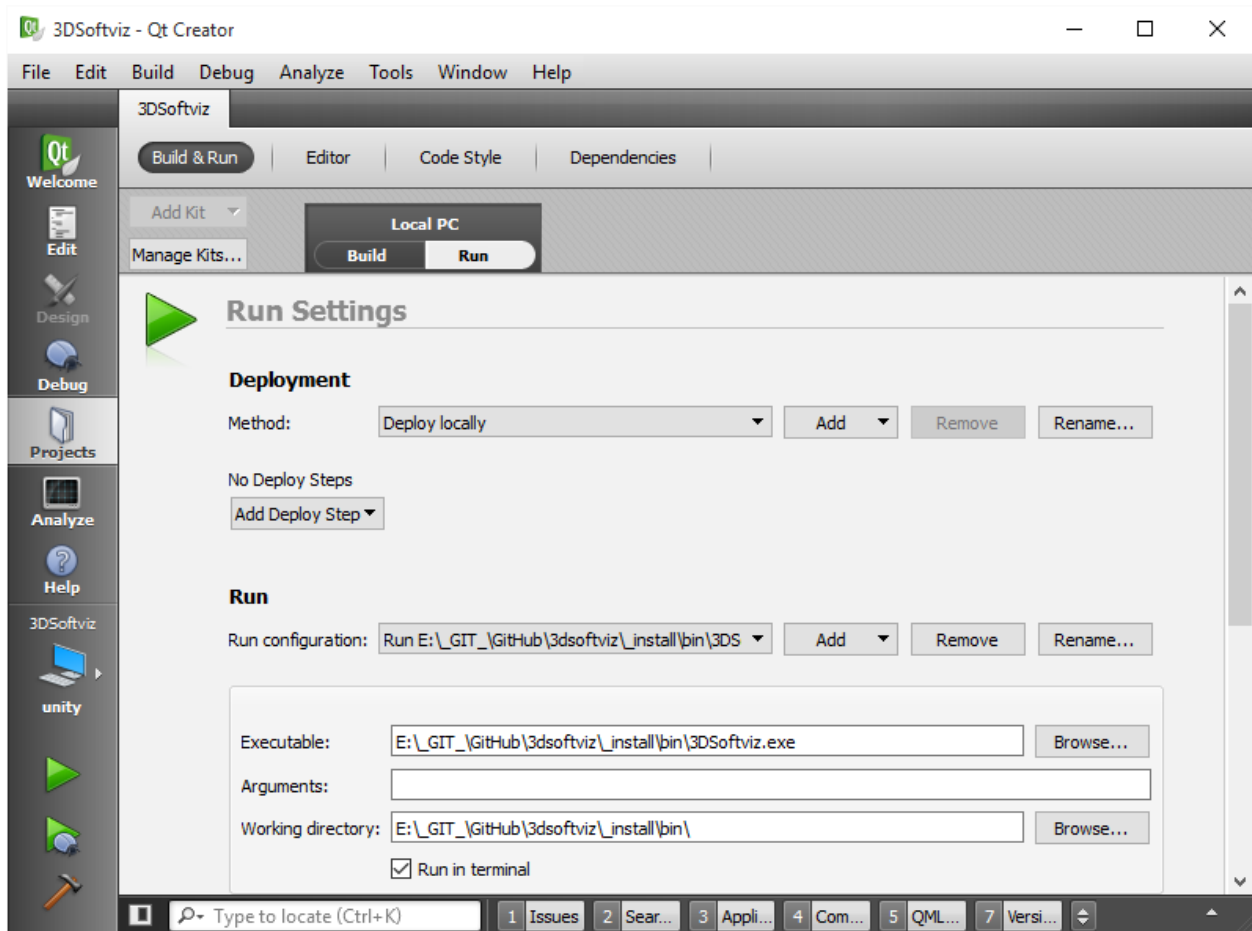


Fig. 1.32: QtC run project

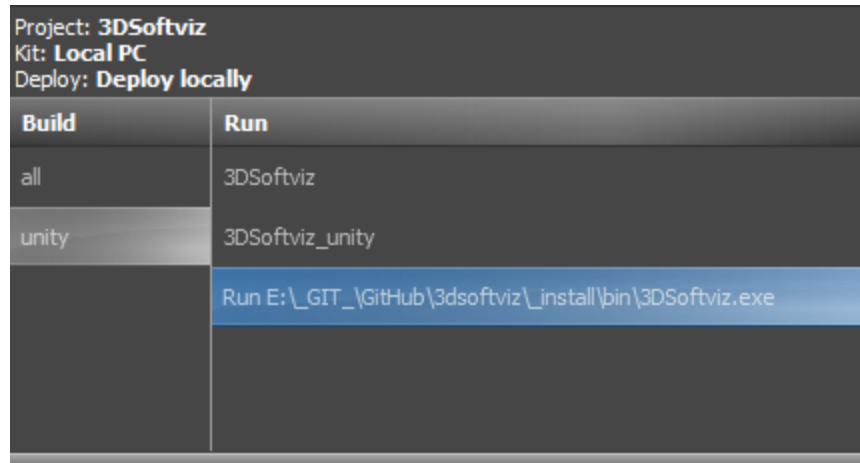


Fig. 1.33: QtC run config

1.2.2 Rozšírenie 3DSoftviz o Kinect

1. Nainštalovať Kinect for Windows
2. Skontrolovať v RapidEE či sa vytvorila premenná `%KINECTSDK10_DIR%`

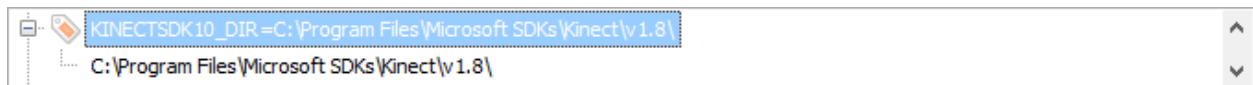


Fig. 1.34: KINECTSDK10_DIR premenná

3. Nainštalovať OpenNI2 (OpenNI-Windows-x86-2.2.msi)
 - **x86! – inak sa môžu vyskytnúť problémy s linkovaním**
4. Skontrolovať v RapidEE či sa vytvorili premenné:
 - (a) `%OPENNI2_INCLUDE%`
 - (b) `%OPENNI2_LIB%`
 - (c) `%OPENNI2_REDIST%`
 - (d) `%OPENNI2_ROOT%`

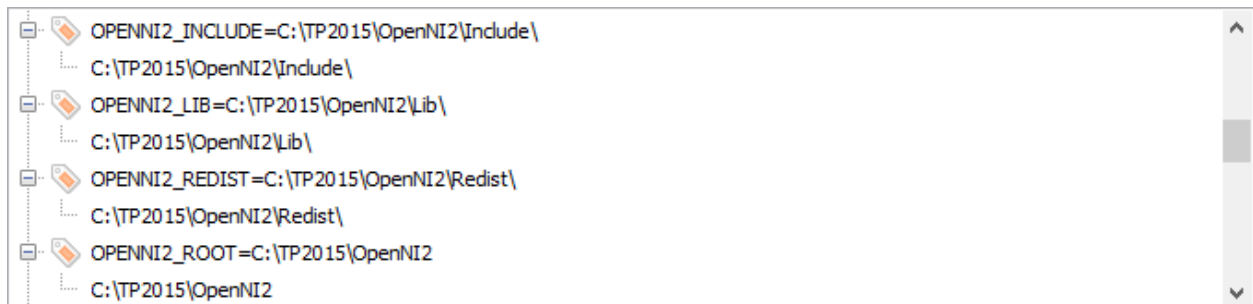


Fig. 1.35: NITE2 premenné

5. Nainštalovať NiTE2 (NiTE-Windows-x86-2.2.msi)
 - **x86!** – inak sa môžu vyskytnúť problémy s linkovaním
6. Skontrolovať v RapidEE či sa vytvorili premenné:
 - (a) `%NITE2_INCLUDE%`
 - (b) `%NITE2_LIB%`
 - (c) `%NITE2_REDIST%`
 - (d) `%NITE2_ROOT%`

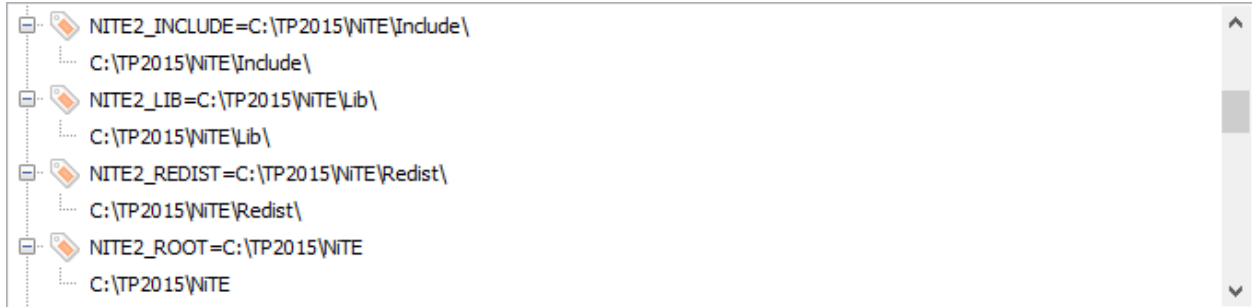


Fig. 1.36: OPENNI2 premenné

7. Pridať do premennej CMAKE_INCLUDE_PATH:
 - (a) `%OPENNI2_INCLUDE%`
 - (b) `%NITE2_INCLUDE%`

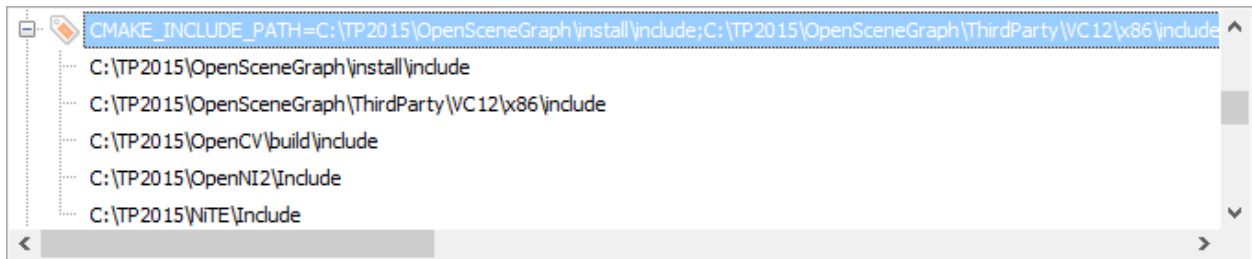


Fig. 1.37: OPENNI2 premenné

8. Pridať do premennej CMAKE_LIBRARY_PATH:
 - (a) `%OPENNI2_ROOT%/Driver`
 - (b) `%OPENNI2_REDIST%`
 - (c) `%OPENNI2_REDIST%/OpenNI2/Drivers`
 - (d) `%OPENNI2_LIB%`
 - (e) `%NITE2_ROOT%/Samples/Bin/OpenNI2/Drivers`
 - (f) `%NITE2_LIB%`
9. Pridať do premennej PATH:
 - (a) `%OPENNI2_REDIST%/OpenNI2/Drivers`

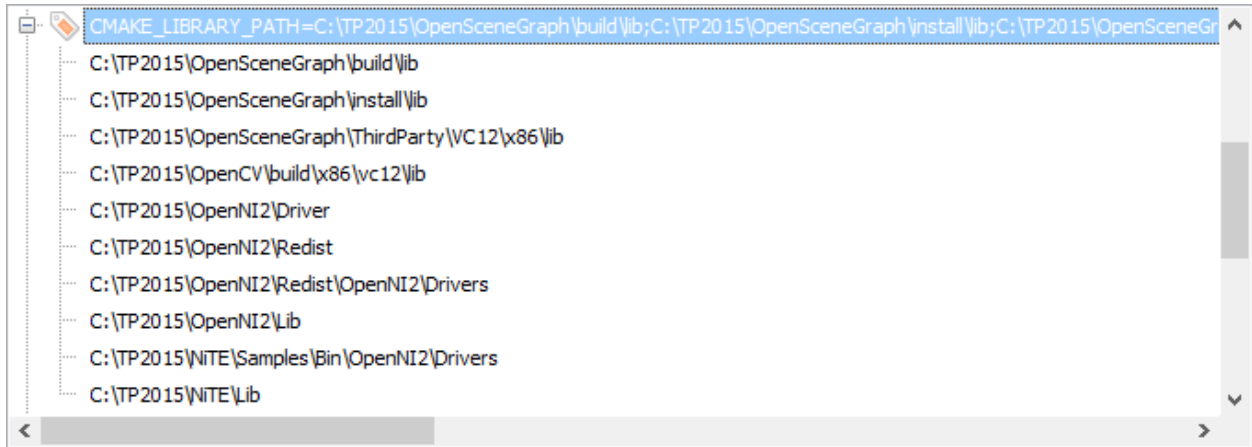


Fig. 1.38: OPENNI2 premenné

- (b) `%OPENNI2_REDIST%`
- (c) `%NITE2_REDIST%`
- (d) `%NITE2_ROOT%/Samples/Bin`

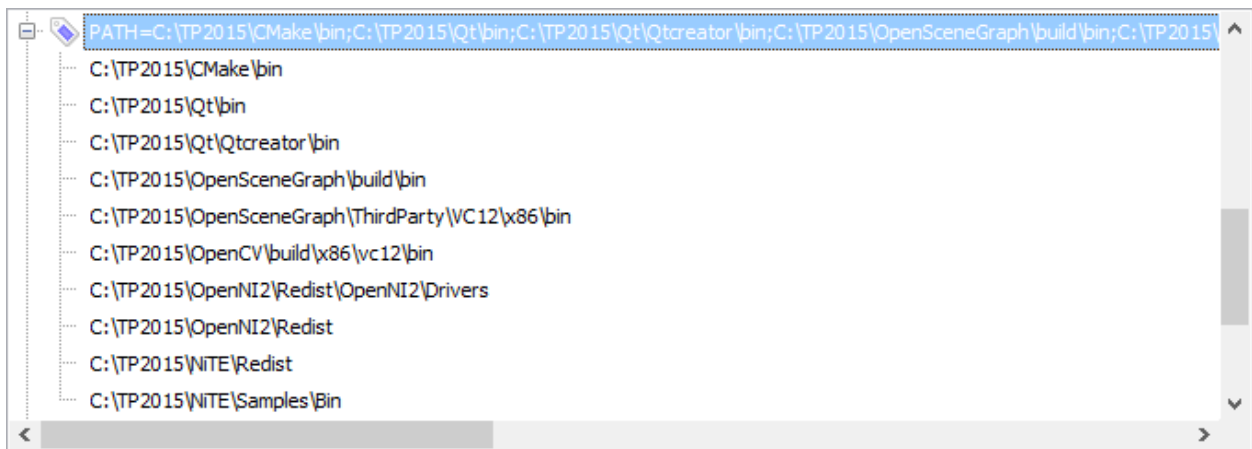


Fig. 1.39: OPENNI2 premenné

10. Spustiť CMake a skontrolovať vo výpise:

- (a) OpenNI2 FOUND
- (b) NITE2 FOUND
- (c) KINECTSDK FOUND

1.2.3 Nastavenie debuggera v QtCreator

1. Nainštalovať WinDbg
2. Skontrolovať v QtCreator Tools > Options > Build & Run > záložka Debuggers či sú autodetected

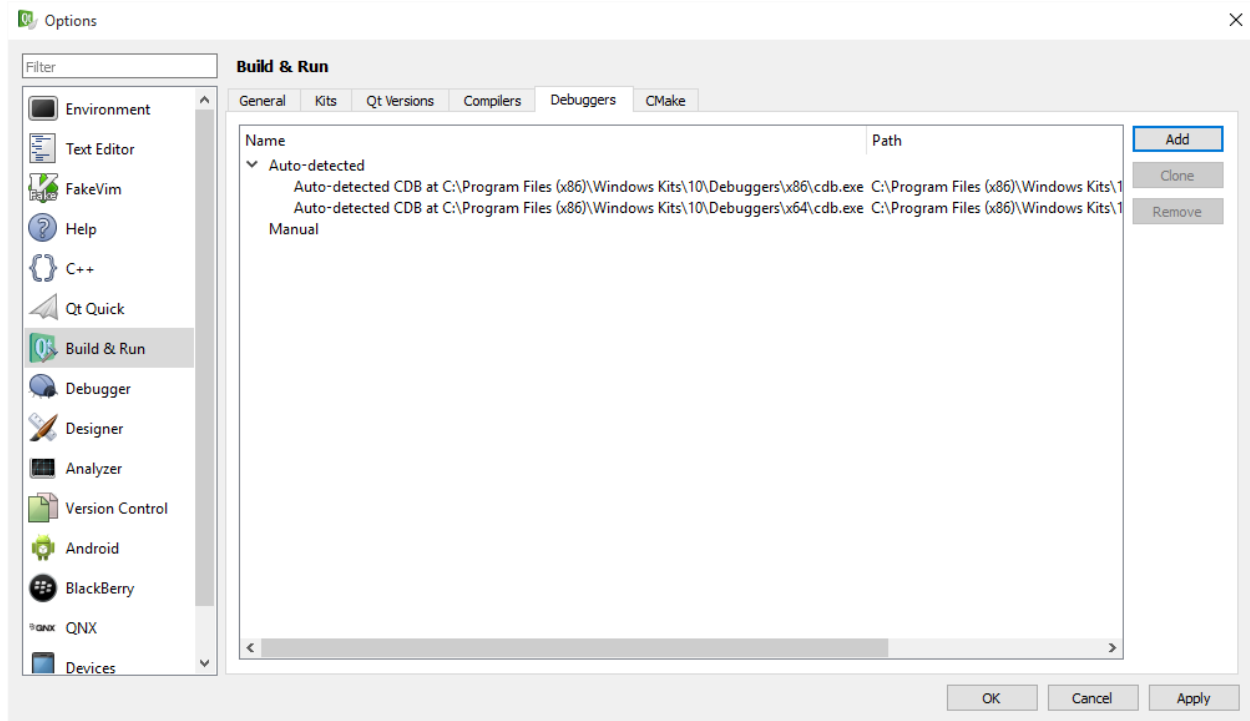
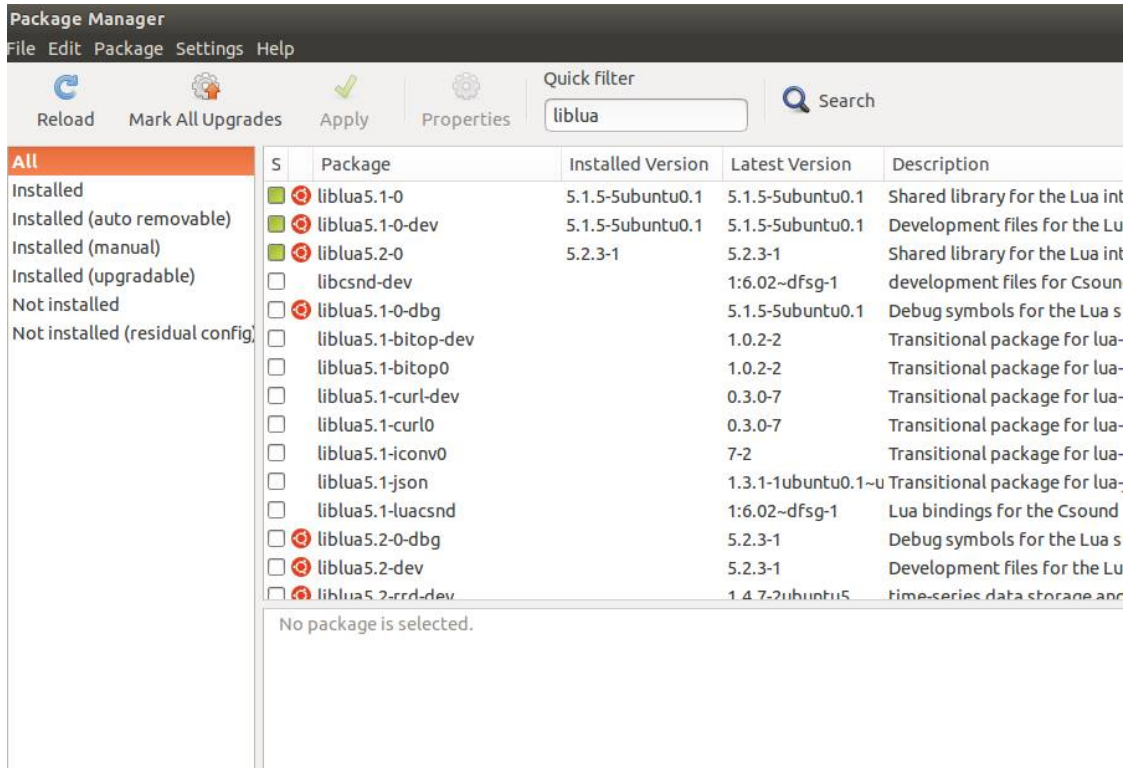


Fig. 1.40: QtC debugger nastavenia

3. Pridať do QtCreator Tools > Options > Build & Run > záložka Kits pre vytvorený profil položku Debugger (x86)
4. Spustiť CMake (-DCMAKE_BUILD_TYPE=Debug)
5. Zvoliť možnosť Debug (vľavo dole medzi Run a Build)

1.3 Návod pre Linux

1. Nainštalovať Synaptic Package Manager



2. Nainštalovať prostredníctvom Synaptic nasledujúce programy

- **Git** - git
- **OpenSceneGraph** - libopenscenegraph-dev
- **Qt4** - libqt4-dev
- **QtCreator** - qtcreator
- **Boost** - libboost-all-dev
- **Lua** - liblua5.1-0-dev
- **OpenCV** - libopencv-dev
- **FreeGlut3**
 - freeglut3
 - freeglut3-dev
- **CMake 3.5.0**
 - inštalovať zo source files, nie Synaptic

Note: Názvy knižníc sú ukázané pre Ubuntu

3. Klonovať cez Git projekt 3dsoftviz (AlphaReach klonuje z repozitára Cimox)

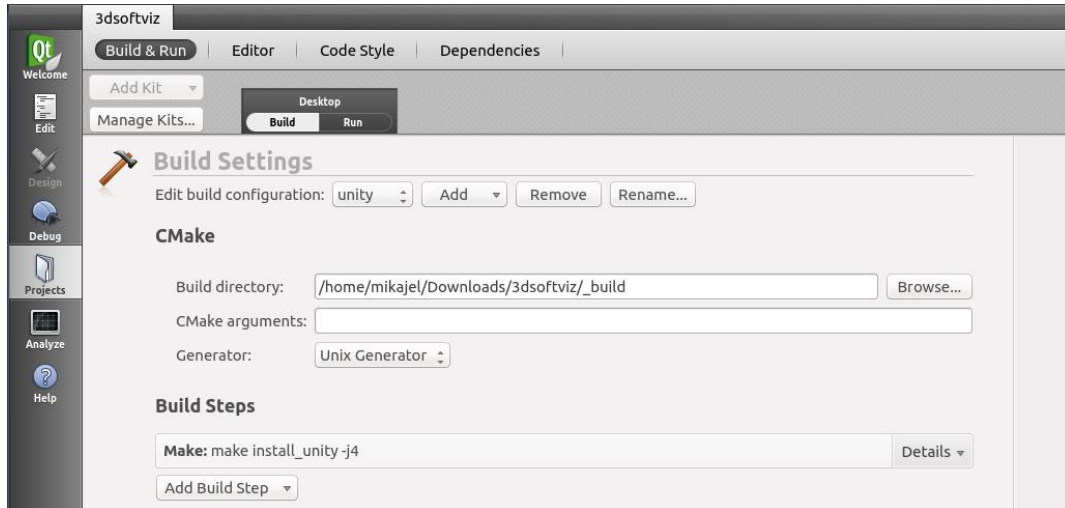
- git clone ** **url repozitára** **
- git submodule update –init –recursive

4. Zbuildovanie projektu

- Otvoriť QtCreator
- File >> Open file or project >> Otvoriť CmakeList.txt v zložke 3dsoftviz
- Vytvoriť projekt
 - Ak úspešne (dá sa kliknúť na FINISH) >> Kliknúť FINISH
 - Ak neúspešne – Nájsť chybu vo výpise

5. Nastaviť build v QtCreator-i

Projects >> Build and Run >> Build Build Settings >> Add >> Clone Selected >> pomenovať “unity”
- automaticky prepne na unity build mode Build Steps >> Details >> zaškrtnúť install_unity



Nastavenie počtu jadier na buildovanie projektu... Details >> Additional arguments “-jN”, kde N reprezentuje počet VIRTUÁLNYCH jadier

6. Pridanie Kinectu do projectu

- Nainštalovať **NiTE** – pridať systémové premenné
- Nainštalovať **OpenNI** – pridať systémové premenné

Note: Pre použitie je nutný Kinect for XBOX

Attention: Na OSX, ak je found OpenNI2 a NiTE2, aplikacia crashne pri spustaní s chybovou hlaskou:

- *not found libNiTE2.dylib*
- *not found libOpenNI2.dylib*

Treba tieto knižnice skopirovať z ich domovských priečinkov (z lib alebo redistrib).

7. Inštalácia drivera pre 3dmyš

- Nainštalovať Motif3 pomocou Synaptic Package Manager
- **Pre uistenie, že 3dmyš funguje, skúsiť xapp aplikáciu z drivera**
 - Pre spustenie drivera (ktory musí bežať pre používanie myši) spustíte príkaz

```
sudo /etc/3DxWare/daemon/3dxsrv -d usb
```

- Pre podrobnejšie inštrukcie ohľadne instalácie a používania drivera otvoriť návod priložený v suboroch oficiálneho drivera pre Linux “**InstallationInstructions_Linux.txt**“

1.4 Používateľská príručka pre 3D Softviz

Okno s aplikáciou je rozdelené na tri základné časti:

- menu
 - File – načítanie grafu zo súboru, z databázy, uloženie grafu, uloženie layoutu, ukončenie aplikácie
 - Settings – nastavenia aplikácie; konfiguračný súbor používa bodkovú notáciu, ktorá umožňuje identifikovať význam konfiguračnej premennej
 - Help
 - Test - pušť a základné grafy pre rýchle testovanie (100-uzlový, 500-uzlový, Veolia, Lua-graph)
- hlavné okno - zobrazuje graf a umožňuje s ním používateľovi interagovať
- ovládací panel – nástroje pre prácu s grafom

1.4.1 Ovládacie prvky

Ovládanie kamery:

- Vyber prvok grafu - ľavé tlačidlo myši + pohyb myšou
- Otáčanie kamery okolo grafu - pravé tlačidlo myši + pohyb myšou
- Ovládanie priblíženia obrazovky - koliesko na myši
- Ovládanie pomocou klávesnici:
 - Hore - PgUp
 - Dole - PgDn
 - Vľavo - šípka vľavo
 - Vpravo - šípka vpravo
 - Dopredu - šípka hore
 - Dozadu - šípka dole

Inicializácia automatického pohybu začne po stlačení kláves Alt + Shift a kliknutím myši na zvolenú hranu, či uzol. V závislosti od nastavenia aplikácie sú pred inicializovaním pohybu ešte automaticky vybrané body záujmu. Pokiaľ je automatický výber uzlov vypnutý, body záujmu je možné zvoliť manuálne myšou alebo stlačením klávesy Q (pre náhodný výber uzlov alebo pre výber uzlov pomocou metrick). Automatické použitie metrick je možné vypnúť v nastavení aplikácie pomocou parametra „Viewer.PickHandler.SelectInterestPoints“ nastaveného na hodnotu 1.

Iné ovládacie prvky:

- Kláves “T” – skrytie všetkých ovládacích prvkov
- Kláves “S” - štatistiky vykresľovania
- Kláves “Shift” - pridávanie ďalších objektov do výberu
- Kláves “Ctrl” - odstránenie objektov z výberu

1.4.2 Záložka GRAPH



- manipulácia s prvkami grafu (no-select mód), pohyb vybraných uzlov v priestore



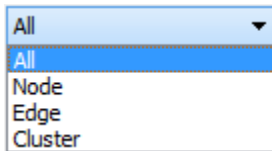
- výber jedného prvku grafu (single-select mód)

Umožňuje sústredenie sa na práve jeden objekt – môže to byť hrana aj uzol.



- výber viacerých prvkov grafu (multi-select mód)

Umožňuje vybrať v trojrozmernom zobrazení viacero objektov naraz.



- typ výberu: všetko, iba uzly, iba hrany, klastre



- centrovanie pohľadu vzhľadom na vybraný prvok grafu

V prípade, že nie je označený žiadny element, kamera bude vycentrovaná na ťažisko grafu



- pridanie meta uzla do grafu



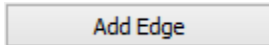
- odstránenie vybraných meta uzlov z grafu



- ukotvenie vybraných uzlov na aktuálnej pozícii, t. j. nebudú sa pohybovať v závislosti od pôsobenia síl ostatných uzlov

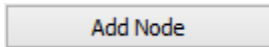


- uvoľnenie ukotvených uzlov

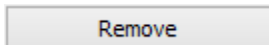


- pridanie hrany medzi dvomi vybranými uzlami

Umožňuje pridať hranu medzi dvoma vybranými uzlami, kde ešte nie je hrana, inak končí akcia chybovou hláškou. Ideálne je čiernou šípkou vybrať jeden uzol a bielou šípkou ho nastaviť na také miesto, kde sa ho dá spojiť s druhým uzlom - je potrebné mať nastavenie Node v takomto prípade spolu s Multi-select mode.

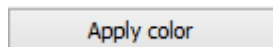
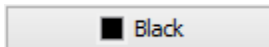


- pridanie uzla do stredu pohľadu

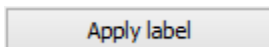


- odstránenie vybraných elementov (uzly alebo hrany)

Ak sa rozhodneme pre zmazanie hrany, uzly prepojené s touto hranou v grafe zostávajú.



, - zafarbenie zvolených uzlov a hrán farbou vybranou z palety nad tlačidlom



- aplikovanie textového označenia na vybrané uzly podľa textu z poľa nad tlačidlom



- zapnutie/vypnutie zobrazovania popisov uzlov a hrán



- spustenie/zastavenie rozmiestňovania (animovania) uzlov grafu



- zmena odpudivých síl pôsobiacich medzi uzlami



- výber vizuálnej reprezentácie uzla (square, sphere) a výber vizuálnej reprezentácie hrany (quad, cylinder, line)

1.4.3 Záložka CONSTRAINTS



- aplikovanie priestorového ohraničenia: povrch gule



- aplikovanie priestorového ohraničenia: obsah gule



- aplikovanie priestorového ohraničenia: rovina



- aplikovanie priestorového ohraničenia: zjednotenie gule a roviny

Zjednotenie gule a roviny je vhodné pre zobrazenie grafov s hustým stredom, alebo na veľké grafy.



- aplikovanie priestorového ohraničenia: kružnica

Aplikovanie obmedzenia na kružnicu na uzly v celom grafe je vhodné pre veľmi riedke grafy alebo na grafy s pravidelnou štruktúrou. Pri hustých grafoch sa hrany medzi uzlami prekrývajú



- aplikovanie priestorového ohraničenia: kužeľ

Obmedzenie na kužeľ je vhodným riešením v prípadoch, kedy má jeden uzol výrazne vyšší počet hrán ako ostatné uzly.



- aplikovanie priestorového ohraničenia: kužeľový strom

Po aplikácii sa uzly rozdelia do skupín podľa spoločného rodiča. Na tieto skupiny sa aplikujú obmedzenia na kužeľ, ktoré sú následne obmedzené na roviny v závislosti od hĺbky uzlov v strome. Kužeľový strom sa aplikuje automaticky na celý graf na základe používateľom vybraného koreňového uzla. Jedine v prípade, že graf nie je spojitý, tak sa aplikuje iba na komponent, ktorý obsahuje koreňový uzol.



- odstránenie vybraných priestorových ohraničení



25



- aplikovanie priestorového ohraničenia: povrch valca

Vloží do scény tzv. bod záujmu, od ktorého sú uzly zobrazovaného grafu odtláčané do tvaru valca. Polomer valca sa dá nastaviť pomocou číselníka.



25



- aplikovanie priestorového ohraničenia: povrch kužeľa

Vloží do scény tzv. bod záujmu, od ktorého sú uzly zobrazovaného grafu odtláčané do tvaru kužeľa. Polomer kužeľa sa dá nastaviť pomocou číselníka. Veľkosť kužeľa sa nastavuje automaticky podľa toho, kam sa používateľ prostredníctvom kamery pozerá.



- aplikovanie radiálneho rozmiestnenia na označené uzly

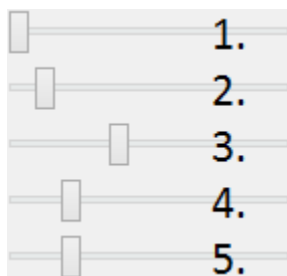
Odporúča sa používať pri stromovom type grafu. Použitie rozmiestnenia na označené uzly dáva používateľovi nové možnosti ako zväčšenie priestoru označením uzlom, alebo manuálne zhlukovanie uzlov.



- výber módu vykreslenia radiálneho rozmiestnenia (drôtený, plný)



- nastavenie módu 2D/3D radiálneho rozmiestnenia



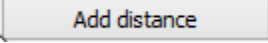
1. nastavenie veľkosti rozmiestnenia
2. nastavenie priehľadnosti rozmiestnenia
3. nastavenie počtu zobrazených gúľ
4. nastavenie faktora zosilnenia odpudivých síl v radiálnom rozmiestnení pre uzly, ktoré nie sú na rovnakej vrstve
5. nastavenie faktora zosilnenia odpudivých síl v radiálnom rozmiestnení pre uzly, ktoré sú na rovnakej vrstve

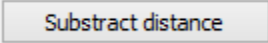
Veľkosť radiálneho zobrazenia sa dá nastaviť v rozmedzí 0 – 300, parameter priesvitnosti 0 - 100 %, veľkosť faktora zosilnenia odpudivých síl sa nastavuje medzi hodnotami 1 - 5000.

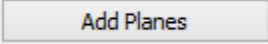
Vertigo zoom - prepínač medzi normálnou a vertigo kamerou

Tento mód kamery je vhodné použiť vtedy, keď chce používateľ meniť dva rôzne pohľady na graf: lokálny pohľad, pri ktorom môže používateľ s väčšou presnosťou skúmať jednotlivé uzly a vzťahy medzi nimi a globálny pohľad, pri

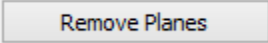
ktorom môže používateľ skúmať vzťahy medzi uzlami a rozloženie uzlov v daných hĺbkach kostry grafu v globálnom kontexte.


 - zvýšenie vzájomnej vzdialenosti medzi rovinami

 - zníženie vzájomnej vzdialenosti medzi rovinami


 - pridanie dvoch paralelných rovín

Obmedzenie na roviny sa aplikuje pri grafoch s minimálnou maximálnou hĺbkou kostry grafu hodnoty 2. Koreňový uzol v kostre grafu určí program - vyberie uzol s najväčším počtom hrán. Pri zrušení obmedzenia sa uzly „odpoja“ od roviny.

 - odobranie dvoch paralelných rovín

 - zmena násobiča odpudivých síl medzi uzlami

Násobič odpudivých síl medzi uzlami je na začiatku nastavený na 1 kvôli prvému pridaniu dvoch rovín do priestoru - nechceme, aby sa hned' zväčšili odpudivé sily.

 - vypnutie všetkých predchádzajúcich obmedzení

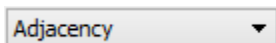
1.4.4 Záložka CLUSTERING

 - zlúčenie vybraných uzlov

Umožňuje zlúčiť vybrané uzly do jedného spoločného uzla. Takýto uzol sa bude v pokračovaní zobrazovať modrou farbou.



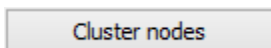
- zrušenie zlúčenia vybraných uzlov



- definovanie algoritmu, ktorým sa bude zhlukovať graf (adjacency, leafs, neighbours)

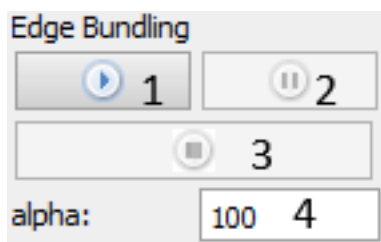
Depth:

- nastavenie počtu rekurzií pre vybraný algoritmus



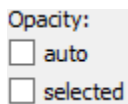
- spustenie zhlukovania nad aktívnym grafom

Ak zhlukovanie trvá viac ako 1 sekundu, objaví sa indikátor postupu.



1. spustenie algoritmu na zväzovanie hrán
2. pozastavenie algoritmu na zväzovanie hrán
3. úplne zastavenie algoritmu na zväzovanie hrán a zobrazenie pôvodného grafu
4. vstupné pole na zadanie konštanty, určujúcej silu akou sú hrany k sebe počas zväzovacieho algoritmu priťahované

Po použití funkcie zhlukovania, sa odkryjú nasledujúce možnosti:

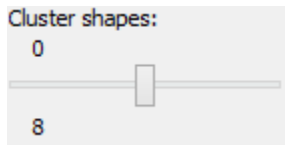


auto - automatická priehľadnosť - mení sa na základe vzdialenosti zhukov od kamery

selected - priehľadnosť označeného zhluku – pomocou posuvníka(nižšie) sa mení priehľadnosť len označených zhlukov



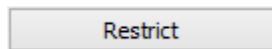
- posúvaním upravíme priehľadnosť označených zhlukov



- posúvaním sa mení prahová hodnota, pri ktorej sa menia tvary zhlukov

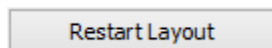
Spodné číslo udáva, koľko uzlov obsahuje daný zhluk (v tomto prípade 8).

Pri označení konkrétneho zhluku sa odkryjú nasledujúce možnosti:

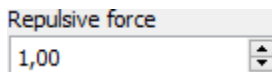


- kliknutím zmeníme označený zhluk na obmedzovač

Obmedzuje pozície uzlov tak, aby z neho nevyšli von. Keď obmedzovač posunieme dostatočne ďaleko, t.j. mimo pôvodnej pozície uzlov, uzly sa začnú lepíť na jeho stenu a posúvať spolu s ním. Ignoruje príťažlivé a odpudivé sily medzi ním a ostatnými uzlami grafu (posunutie zhluku bez obmedzovača spôsobí posun celého grafu za týmto zhlukom). Obmedzovač začína svoje pôsobenie ako kocka, je možné zmeniť jeho tvar natáhovaním a stláčaním.



- znovurozmiestnenie uzlov v priestore po tom, ako sa nalepia na hranu obmedzovača



- upravenie odpudivej sily medzi uzlami v označenom zhluku

Čím je hodnota väčšia, tým budú uzly ďalej od seba.

Ďalšie funkcie obmedzovača:

Ak na zhluk zaregistrujeme obmedzovač, môžeme s ním jednoducho pohybovať a meniť jeho tvar pomocou klávesových skratiek a myši:

- Pohyb – metóda ťahaj a pust' (drag & drop)
- Zmena veľkosti – držíme **Ctrl** a točíme kolieskom myši
- Zmena tvaru
 - na osy x – držíme **X** a **Ctrl** a točíme kolieskom myši
 - na osy y – držíme **Y** a **Ctrl** a točíme kolieskom myši
 - na osy z – držíme **Z** a **Ctrl** a točíme kolieskom myši

1.4.5 Záložka CONNECTIONS

Nick:

- napísanie mena, pod ktorým bude používateľ vystupovať v kolaborácii

- spustenie/zastavenie servera

Host:

- napísanie IP adresy servera

- pripojenie(odpojenie) ku(od) kolaborácii

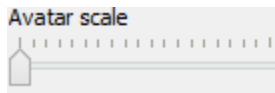
Collaborators:

- zoznam používateľov (zoradený abecedne), v ktorom je možné jedného vybrať a použiť nasledujúce funkcie:

- Spy
- Center
- Shout

- po výbere si môžeme zvolit' jednu funkciu z dvojice: *Spy* (špehovať) a *Center* (centrovať).

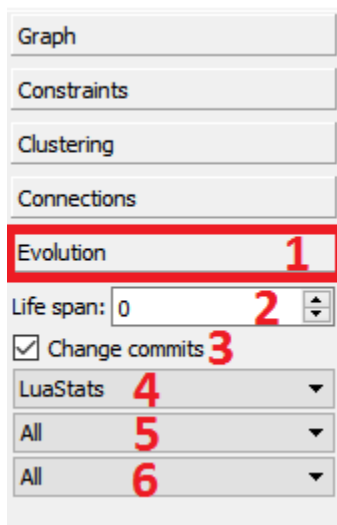
Po aktivovaní funkcie *Spy* získa používateľ pohľad iného používateľa, ktorý je priebežne aktualizovaný – znamená to, že pohybom sledovaného používateľa sa aktualizuje aj pohľad sledujúceho. Po aktivácii *Center* nasmeruje pohľad používateľa tak, aby v jeho strede bol iný používateľ. Pri centrovaní platí to isté, čo pri špehovaní – teda pri aktualizácii polohy centrovaného používateľa sa natáča aj pohľad centrujúceho používateľa. Po označení políčka *Shout* sa ostatným používateľom v scéne zobrazí pri vašom mene ikona znázorňujúca, že sa pokúšate upútať pozornosť.



- nastavenie veľkosti avatarov v scéne

Avatar je kužeľ, ktorého kruhová podstava znázorňuje smer, ktorým sa používateľ pozerá.

1.4.6 Záložka EVOLUTION



- Po rozkliknutí tabu *Evolution* (1) sa zobrazia možnosti evolúcie

2. *Lifespan* - možnosť ponechania vymazaných uzlov vo vizualizácii. Prednastavená hodnota 0 znamená, že vymazané uzly sa automaticky vymažú z grafu. V prípade hodnoty väčšej ako 0 vymazané uzly v grafe zotrývajú o verzie dlhšie podľa nastavenej hodnoty
3. *Change commits* - prepínač spracovania Git repozitáru. Ak je zaškrtnutý, inicializuje sa spracovanie na úrovni grafu volaní. V opačnom prípade - na úrovni histórie Git repozitáru
4. **Kombo box s výberom vizualizácie - prepínanie sa medzi jednotlivými možnosťami vizualizácie grafu volaní**
 - *LuaStat* - vizualizácia softvérových metrík pomocou analýzy Lua zdrojového kódu
 - *Difference* - pohľad na zmeny, ktorými softvér prešiel pri prechode na novú verziu

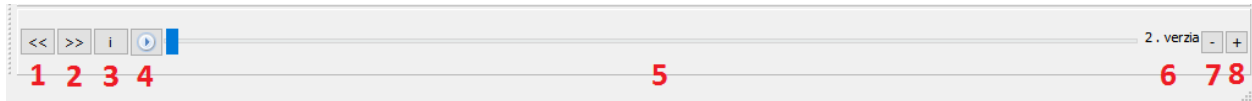
- *Changes* - aktivovanie filtrovania nad práve aktívnou vizualizáciou

5. Kombo box s výberom filtra - výber vhodnej skupiny filtra

- Prednastavená možnosť *All* - všetky prvky grafu sú zobrazené
- *Authors* - filtrovanie podľa autorov zmien v softvéri
- *Structure* - filtrovanie podľa štruktúry

6. Kombo box zoznamu možností - možnosti závisia od vybraného filtra

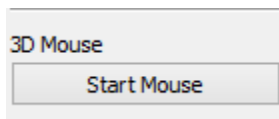
- zoznam autorov s možnosťou zobrazenia zmien všetkých autorov - *All*
- štruktúra - *Files* (zobrazí v grafe volaní len uzly reprezentujúce adresáre a súbory), *Local Functions* (zobrazí rozšírenú možnosť *Files* spolu s uzlami lokálnych funkcií), *Global Functions* (zobrazia sa uzly možnosti *Local Functions* spolu s uzlami globálnych funkcií) a *Modules* (zobrazí všetky štruktúry, ktoré sa v grafe nachádzajú)



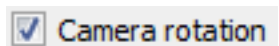
- panel ovládania evolúcie

1. Prechod na predchádzajúcu verziu - možnosť, kedy sa stav grafu vráti o jednu verziu dozadu
2. Prechod na nasledujúcu verziu - možnosť, kedy sa stav grafu posunie o jednu verziu dopredu
3. Tlačidlo informácií o verzii - zobrazí informácie o aktuálne zobrazenej verzii. Medzi zobrazené informácie patrí identifikátor, autor a dátum commitu spolu so zoznamom súborov, ktoré boli zmenené
4. Spustenie/zastavenie animácie - aktivovanie/zastavenie automatického prechodu na novú verziu
5. Posuvník - presun na konkrétnu verziu pomocou skokového prechodu medzi verziami
6. Indikátor verzie - poskytuje informáciu o aktuálne zobrazenej verzii
7. Spomalenie animácie - regulovanie rýchlosti animácie
8. Zrýchlenie animácie - regulovanie rýchlosti animácie

1.4.7 Záložka MORE FEATURES



- zapnutie 3D myšky (musí byť aktivovaný driver)



- ak je zaškrtnuté, kamera nasmerovaná na graf sa pohybuje na základe pohybu tváre, značky alebo rúk, inak sa na základe týchto akcií rotuje samotný graf

Camera enabled - povolí uje použitie kamery

Start camera - otvorenie okna pre prácu s kamerou

Start Speech - otvorenie okna pre prácu s hlasovým ovládaním

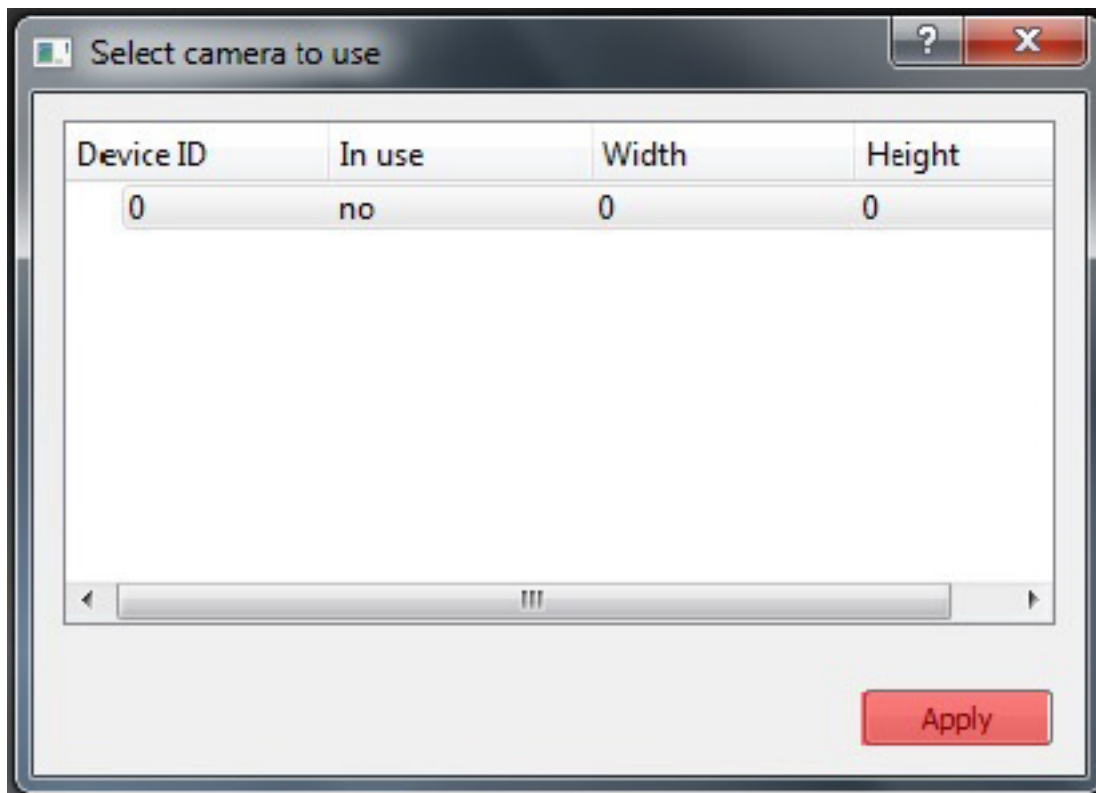
Note: Speech je momentálne vylúčený z projektu

Start Leap - zapnutie ovládania pomocou Leap Sensor-u

Okno pre prácu s kamerou

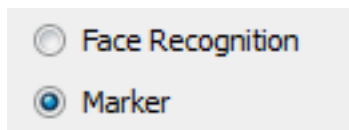
Face Recognition
 Marker - prispôbenie ľavej strany okna pre ovládanie funkcionality rozpoznávania tvare (pri zapínaní treba zaškrtnúť Camera rotation a Camera enabled)

Start Face Recognition - zvolenie kamerového zariadenia a následným potvrdením objavenie záberu z kamery
 Ukončiť túto akciu je možné tlačidlom „StopFaceRec“ (ak používateľ zatvoril okno, môže ho vrátiť na grafický interface opätovným kliknutím na „StartCamera“ a potom pozastaviť detekciu). V prípade detegovanej tváre (detekcia je reprezentovaná zeleným obdĺžnikom) sa kamera alebo graf pohybuje vď aka pohybu tváre.

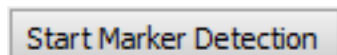


výber snímacieho zariadenia

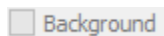
- okno pre



- prispôsobenie ľavej strany okna pre ovládanie funkcionality rozpoznávania značky



- zvolenie kamerového zariadenia a následným potvrdením objavenie záberu z kamery určenej pre rozpoznávanie značky a graf sa začne otáčať a pohybovať so značkou



- nastavenie aktuálne snímání ako pozadie pre graf

Je potrebné zmeniť parameter „Viewer.SkyBox.Noise“ v konfiguračnom súbore na hodnotu 2 alebo 3 (odporúčané je 3).

Marker is behind - prepínanie medzi pohybom podľa značky ako keby sa kamera pozerala na používateľa a naopak

Correction - zapnutie korekcie

- nastavenie korekčných parametrov

Podľa predvolených nastavení sa značka pohybuje tak, ako keby sa kamera pozerala vo vodorovnom smere. Ak by sa pozerala napr. na stôl pod miernym sklonom dole, graf by sa pri posúvaní značky po stole neposúval korektne. Preto je možné nastaviť korekčné parametre. Najskôr je potrebné nastaviť značku do polohy, kedy je detegovaná na spodnom okraji a následne stáčiť toto tlačidlo. Po nastavení sa aktivuje opcia „Correction“ (uvedené vyššie), ktorou je možné zapnúť korekciu.

- zmena spôsobu použitia značky v prípade, že používateľ má k dispozícii len jednu značku

NoVideo - vypnutie/zapnutie zobrazenia videa

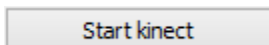
Toto prepínanie a vypnutie zobrazenia videa má vplyv len na zobrazenie v rámci tohto ovládacieho okna „Face Recognition“ and „Marker Detection“ a neovplyvňuje to ani voľbu kamery pre video pozadie.



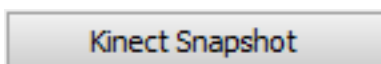
- Interakcia s vizualizáciou v obohatenej realite

- Možnosť *Custom light*, vyznačená modrou, ktorá slúži na prepínanie vlastného a základného zdroja svetla
- Možnosť *Shadow*, vyznačená žltou, ktorá slúži na zapínanie a vypínanie generovania tieňov
- Možnosť *Base*, vyznačená červenou, ktorá slúži na zobrazenie a skrytie základne
- Možnosť *Axes*, vyznačená ružovou, ktorá slúži na zobrazenie a skrytie pomocných osí
- Tlačidlo *Center graph*, vyznačené svetlo modrou, ktoré slúži na umietnetie grafu nad stred základne

Okno pre prácu s kinectom a arucom



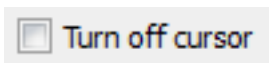
- zapnutie detekcie



- zachytenie kádra s následnou možnosťou dať ho na pozadie



- zapnutie rozpoznávania značiek



- prepínanie medzi detekovaním ruky pre manipuláciu grafu alebo kamery v podobe rotovania a medzi detekovaním ruky pre funkciu "klik" (pohyb ruky do hĺbky, nie vertikálne alebo horizontálne)

Turn off zoom

- vypne možnosť približovania

Aruco

- nastavenie práce s arucom

Start projective AR view

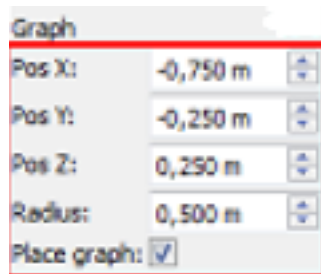
- zobrazenie okna projekčného zobrazenia

Projector		
FOV:	30,00 Å°	<input type="text"/>
Pos X:	-0,665 m	<input type="text"/>
Pos Y:	-1,345 m	<input type="text"/>
Pos Z:	0,825 m	<input type="text"/>
Dir X:	-0,085 m	<input type="text"/>
Dir Y:	1,345 m	<input type="text"/>
Dir Z:	-0,587 m	<input type="text"/>

- v projekčnom zobrazení - Projector - spinboxy na zmenu parametrov projektora (odhora) - zorné pole, pozícia (súradnice x, y, z), smer projekcie (súradnice x, y, z)

Viewer		
FOV:	90,00 Å°	<input type="text"/>
Pos X:	-1,880 m	<input type="text"/>
Pos Y:	-0,950 m	<input type="text"/>
Pos Z:	1,720 m	<input type="text"/>
Dir X:	1,130 m	<input type="text"/>

- v projekčnom zobrazení - Viewer - spinboxy na zmenu parametrov pozorovateľa (odhora) - zorné pole, pozícia (súradnice x, y, z), smer projekcie (súradnice x, y, z)



- v projekcnom zobrazeni - Graph - spinboxy na zmenu parametrov grafu (odhora)
- pozicia (súradnice x, y, z), polomer, checkbox Place graph na potvrdenie použitia parametrov grafu (štandardne označený)



- potvrdenie zadaných parametrov scény

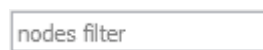
Hlasové príkazy pre Speech

- select all nodes - vybratie všetkých uzlov
- select left side - vybratie uzlov na ľavej strane
- select right side - vybratie uzlov na pravej strane
- clear screen - zrušenie vybratia uzlov
- sphere - sformovanie gule pre vybrané uzly
- unset restrictions - návrat k pôvodnému stavu - zrušenie akcie "sphere"

1.4.8 Hlavné okno



- filtrovanie hrán

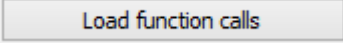


- filtrovanie uzlov


Príklady príkazov:

- "params.type like 'file' or params.type like 'directory'"
- "params.name like 'init%.lua' and params.type like 'function'"
- "params.type like 'function'"

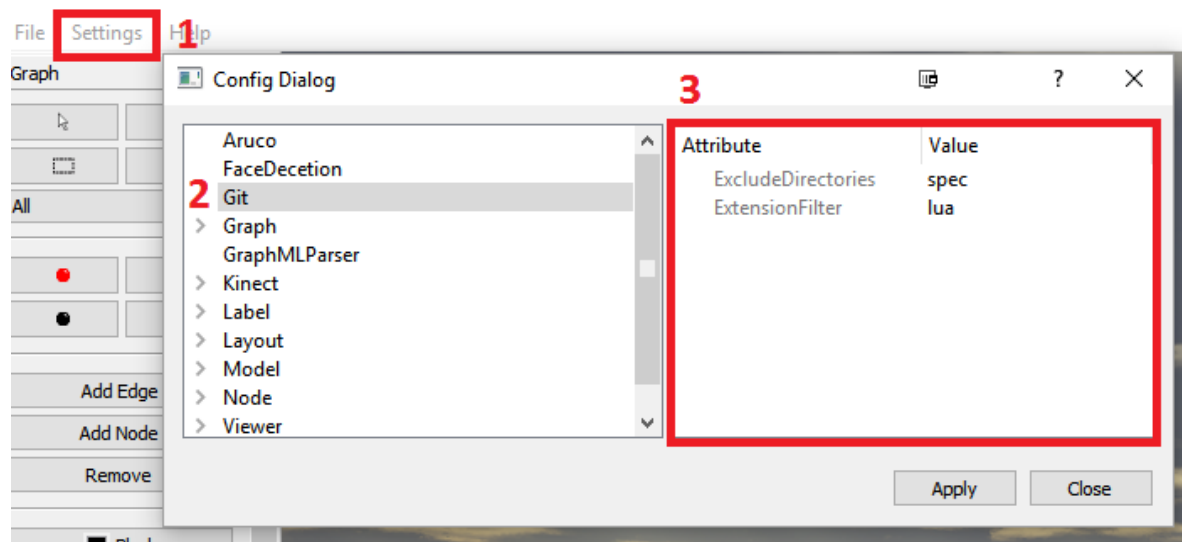
Filter je navrhnutý pre grafovú vizualizáciu softvéru s využitím softvérových metrick jazyka Lua a je vyhodnotený po stlačení klávesu „Enter“.

 - zobrazí dialóg pre výber súborov a po vybratí vykreslí do poľa pod tlačidlom graf volaní funkcií týchto súborov

Pri označení práve jedného vrcholu v poli sa zobrazí stromová štruktúra informácií o tomto vrchole.

 - prepínanie medzi zobrazovaním jedného prehliadača pre každý uzol a zobrazovaním jedného prehliadača pre všetky vyznačené uzly

1.4.9 Git repozitár



1. Settings / Options - zobrazenie dialógového okna s konfiguráciou
2. Možnosť Git - zobrazia sa možnosti konfigurácie spracovania Git repozitáru
3. Možnosti konfigurácie spracovania Git repozitáru
 - vyčlenenie adresárov (ExcludeDirectories) - ľubovoľný počet názvov adresárov oddelených znakom ”,”. Pre zadanú hodnotu sa pri spracovaní Git repozitáru odignorujú všetky súbory, ktoré vo svojej relatívnej ceste obsahujú adresár spec.
 - ExtensionFilter - funguje obrátene, pričom ponecháva len tie súbory, ktorých koncovka súboru sa zhoduje s jednou zo zadaných hodnôt. Hodnota taktiež môže obsahovať viacero koncoviek súborov, pričom musia byť oddelené znakom ”,”

1.5 Používateľská príručka pre Vuzix a Leap

1.5.1 Stereoskopické 3D s Vuzix okuliarmi

Okuliare

Ako prvé je potrebné pripojiť Vuzix okuliare k počítaču (HDMI a 2x USB)

Okuliare by mali byť rozpoznané ako zobrazovacie zariadenie, pre ktoré nastavte duplikáciu obrazu z monitora, na ktorom beží 3Dsoftviz.

Nakoniec pomocou ovládacieho zariadenia vyvolajte menu a nastavte režim zobrazovania obrazu na side-by-side 3D alebo top-bottom 3D, poprípade upravte aj jas a kontrast obrazu.

(menu sa zobrazí len priamo na displejoch okuliarov)

3Dsoftviz

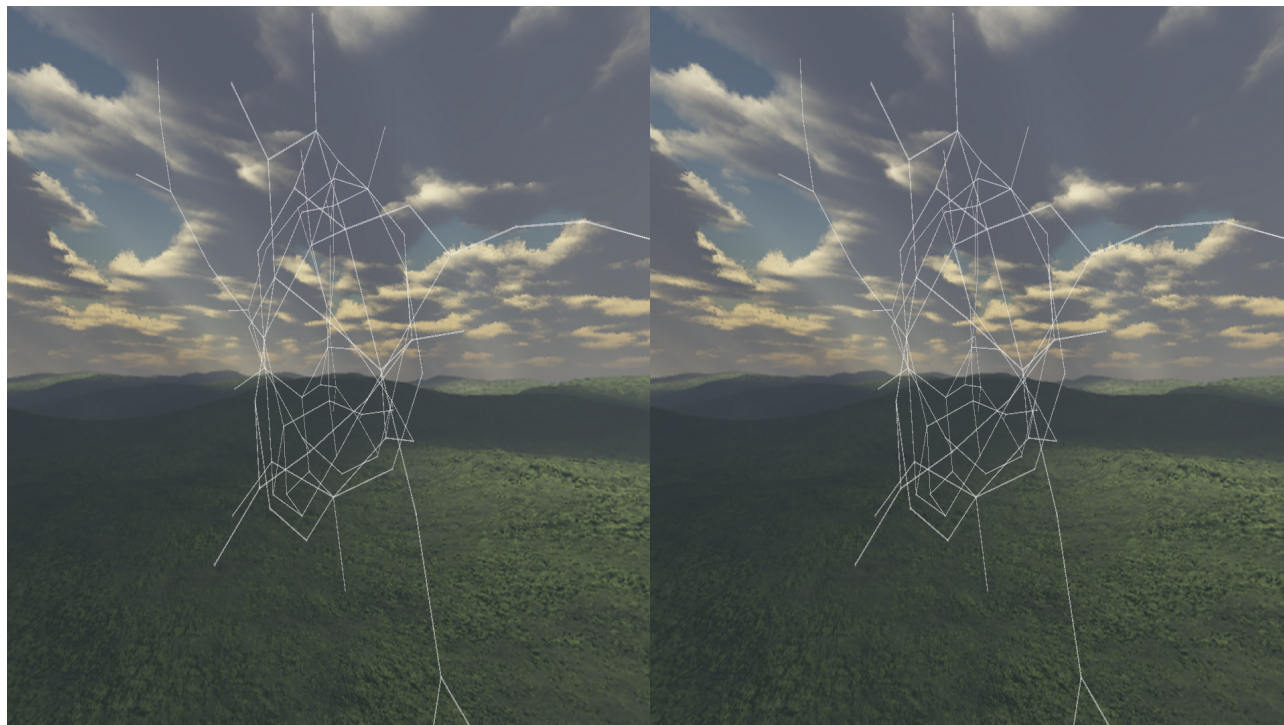
Pre správne zobrazenie je potrebné nastaviť 3Dsoftviz do režimu plnej obrazovky stlačením klávesy L a skryť panel s nástrojmi stlačením klávesy T

Následne je možné pomocou klávesy G prepínať medzi režimami:

- normálne zobrazenie
- top-bottom 3D zobrazenie
- side-by-side 3D zobrazenie

Vyberte zobrazenie korešpondujúce s nastavením okuliarov.

Klávesami H a J je možné upravovať nastavenie vzdialenosti očí, ktorá mení posun medzi obrazmi ktorým sa dosahuje 3D efekt. Úprava je po 0.01m a aktuálna hodnota sa zobrazuje v konzole.



1.5.2 Ovládanie rukami s Leap senzorem

Leap:

Ako prvé je potrebné pripojiť Leap senzor k počítaču (1x USB). Ďalej je potrebné nainštalovať oficiálny softvér k Leap senzoru z oficiálnej stránky.

3Dsoftviz:

V ľavej lište s nástrojmi je v karte *More features* možnosť *Start Leap*, po stlačení ktorej začne snímanie rúk nad senzorom.

Podporované je nasledovné ovládanie:

- Ľavá ruka s vystretými prstami – pohyb kamery dopredu
- Ľavá ruka so skrčenými prstami – zastavenie pohybu kamery
- Pravá ruka s vystretými prstami – kamera sa otáča podľa a naklonenia dlane

Snímanie je možné zastaviť opätovným stlačením rovnakého tlačidla ktoré ma počas snímania text *Stop Leap*.