

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5212-79530

ŠIMON GAŠPAR

3D MAPOVANIE A MODELOVANIE PROSTREDIA V REÁLNO M ČASE

BAKALÁRSKA PRÁCA

Vedúci práce: Ing. František Kudlačák

Máj 2019

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5212-79530

ŠIMON GAŠPAR

3D MAPOVANIE A MODELOVANIE PROSTREDIA V REÁLNO M ČASE

BAKALÁRSKA PRÁCA

Študijný program:	Informatika
Študijný odbor:	9.2.1. Informatika
Miesto vypracovania:	Ústav počítačového inžinierstva a aplikovanej informatiky, FIIT STU
Vedúci práce:	Ing. František Kudlačák
Máj 2019	

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

ZADANIE BAKALÁRSKEHO PROJEKTU

Meno študenta: **Gašpar Šimon**
Študijný odbor: Informatika
Študijný program: Informatika
Názov projektu: **3D mapovanie a modelovanie prostredia v reálnom čase**

Zadanie:

Analyzujte existujúce možnosti snímania prostredia. Analyzujte možnosti vyhodnocovania a reprezentácie nasnímaných údajov. Navrhňte informačný systém, ktorý umožňuje mapovanie priestoru a modelovanie priestoru v reálnom čase a zároveň poskytuje informácie o polohe mapovacieho zariadenia. Funkcionalitu implementujte na vhodne zvolenom systéme. Vytvorený model prostredia musí byť dostupný s minimálnou odozvou na zmenu pozície systému v čase. Podľa možnosti prakticky overte presnosť mapovania systému.

Práca musí obsahovať:

Anotáciu v slovenskom a anglickom jazyku
Analýzu problému
Opis riešenia
Zhodnotenie
Technickú dokumentáciu
Zoznam použitej literatúry
Elektronické médium obsahujúce vytvorený produkt spolu s dokumentáciou

Miesto vypracovania: Ústav počítačového inžinierstva a aplikovanej informatiky, FIIT STU, Bratislava
Vedúci projektu: Ing. František Kudlačák

Termín odovzdania práce v letnom semestri: 7. 5. 2019

**SLOVENSKÁ TECHNICKÁ UNIVERZITA
V BRATISLAVE**
Fakulta Informatiky a informačných technológií
Ilkovičova 2, 842 16 Bratislava 4
1



Bratislava 11.2.2019

prof. Ing. Pavol Návrat, PhD.
riaditeľ ÚISI

Čestne prehlasujem, že som túto prácu vypracoval samostatne, na základe konzultácií a s použitím uvedenej literatúry.

V Bratislave, 7. máj 2019

Šimon Gašpar

Pod'akovanie

Chcem sa pod'akovať vedúcemu bakalárskej práce, Ing. Františkovi Kudlačákovi, za poskytnuté konzultácie a rady k vypracovaniu tejto práce. Ďalej sa chcem pod'akovať Laure za podporu, ktorú mi poskytla počas štúdia a písania tejto práce, spolužiakom a spolubývajúcim za poskytnuté odborné informácie.

Anotácia

Slovenská technická univerzita v Bratislave

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLOGIÍ

Študijný program: Informatika

Autor: Šimon Gašpar

Bakalárska práca: 3D mapovanie a modelovanie prostredia v reálnom čase

Vedúci bakalárskej práce: Ing. František Kudlačák

Máj 2019

Práca sa zaoberá spôsobmi snímania prostredia a vytvorením 3D modelu zo získaných dát. V analýze sa popisujú spôsoby snímania prostredia a ich princípy fungovania. Následne sa popisujú kroky potrebné k vytvoreniu 3D modelu zo získaných dát z mapovaného prostredia. Bližšie sa venujeme spôsobu mapovania pomocou kamier, ktorý umožňuje spracovať získané snímky dvoma spôsobmi. Týmito dvoma spôsobmi je štruktúra z pohybu a stereo videnie. Špecifikácia, návrh a implementácia sa primárne zaoberá spomenutým spôsobom spracovania snímok ako aj získaniu snímok z kamier. Výsledkom práce je program, ktorý je schopný získavať snímky z kamier a následne ich spracovať pomocou spomenutých spôsobov a vytvoriť 3D model. V kapitole zaoberajúcej sa testovaním sme sa zamerali na rýchlosť spracovania snímok a na kvalitu výsledného 3D modelu. Zo získaných dát z testovania sme vyvodili záver a úspešnosť celej práce. V zhodnotení práce sme taktiež načrtli ďalšie smery, ktorými sa môže práca zaoberať.

Annotation

Slovak University of Technology Bratislava

FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree course: Informatic

Author: Šimon Gašpar

Bachelor's Thesis: 3D mapping and modelling of environment in realtime

Supervisor: Ing. František Kudlačák

May 2019

The thesis deals with the methods of mapping the environment and creating a 3D model from the data obtained. The analysis describes the methods of scanning the environment and their operating principles. Subsequently, the steps needed to create a 3D model from the data from the mapped environment are described. We are focusing on the method of mapping with cameras, which allows processing the acquired images in two ways. These two ways are the structure of motion and stereo vision. Specification, design and implementation primarily deal with the mentioned image processing as well as obtaining images from cameras. The result of the work is a program that is able to acquire images from cameras and then process them using the mentioned methods and create a 3D model. In the chapter dealing with testing, we are aimed at the speed of image processing and the quality of the resulting 3D model. From the data obtained from the testing, we concluded the conclusion and the success of the whole work. In the evaluation of the work we also outlined other directions that the work can deal with.

Obsah

1.	Úvod.....	1
2.	Analýza.....	2
2.1.	3D Model.....	2
2.1.1.	Point cloud.....	3
2.2.	3D Modelovanie	3
2.2.1.	Constructive solid geometry.....	4
2.2.2.	Polygonálne modelovanie	5
2.3.	Mapovanie prostredia	7
2.3.1.	Snímanie pomocou laseru	7
2.3.2.	Snímanie pomocou kamery – stereo videnie.....	9
2.3.2.1.	Fotogrametria.....	9
2.3.2.2.	Kalibrácia	11
2.3.2.3.	Rektifikácia	13
2.3.2.4.	Hĺbková mapa	14
2.3.3.	Snímanie pomocou kamery – multiview.....	16
2.3.3.1.	Feature detection	16
2.3.3.2.	Zhoda kľúčových bodov	18
2.3.3.3.	Structure from Motion	18
2.3.4.	RGB-D kamera.....	18
2.4.	Existujúce riešenia	19
2.4.1.	ZED kamera	19
2.4.2.	RealityCapture.....	20
2.4.3.	Kinect a KinectFusion	20
2.4.4.	MeshLab.....	21
2.5.	Zhodnotenie	21
3.	Špecifikácia	23
3.1.	Funkčná špecifikácia	23
3.2.	Hardware	23
3.2.1.	Výpočtové prostriedky	24
3.2.2.	Kamery	24
3.3.	Software.....	24
3.3.1.	CUDA.....	24
3.3.2.	Visualization Toolkit.....	24

3.3.3.	Open Source Computer Vision	25
3.3.4.	Emgu CV	25
3.3.5.	JSON.NET	25
3.3.6.	VisualSFM	25
3.4.	Testovacie scenáre	26
3.4.1.	Funkčné testy	26
3.4.2.	Výkonnostné testy	26
4.	Návrh	28
4.1.	Stereo kamera	28
4.2.	Vstupné a výstupné dáta	29
4.3.	Nastavenie kamery	30
4.4.	Priebeh aplikácie	31
4.4.1.	Stereo videnie	32
4.4.2.	Structure from Motion	33
4.5.	Zobrazenie	34
5.	Implementácia	36
5.1.	Vstupné a výstupné dáta	36
5.2.	Stereo kamera	37
5.3.	Grafické rozhranie	39
5.4.	Stereo videnie	40
5.4.1.	Úprava snímok (kalibrácia/rektifikácia)	40
5.4.2.	Výsledný model	41
5.5.	Štruktúra z pohybu	42
5.5.1.	Výpočet deskriptorov	42
5.5.2.	Nájdenie zhôd medzi descriptorami	43
5.5.3.	Výsledný model	44
6.	Testovanie	45
6.1.	Stereo videnie	45
6.2.	Structure from Motion	46
6.3.	Bonus	56
6.4.	Zhodnotenie	57
7.	Záver	58
	Bibliografia	59
	Príloha A – Plán práce 2017/2018	62
	Príloha B - Plán práce 2018/2019	63

Príloha C – Technická dokumentácia.....	64
Príloha D – Inšalačná príručka.....	66
Príloha E - Používateľská príručka.....	67
Príloha F – Elektronické médium.....	69

Slovník pojmov a skratiek

NURBS	<i>Non-uniform rational basis spline</i> - matematický model používaný v počítačovej grafike
CCD	<i>Charge-coupled device</i> - nábojovo viazaná súčiastka
CMOS	<i>Complementary Metal–Oxide–Semiconductor</i> - logický integrovaný obvod
LIDAR	<i>Light Detection And Ranging</i> - metóda merania vzdialeností
CPU	<i>Central processing unit</i> - centrálna procesorová jednotka
GPU	<i>Graphics processing unit</i> - grafická procesorová jednotka
FPS	<i>Frames per second</i> - počet snímok za sekundu
JPG	<i>Joint Photographic Group</i> - formát súboru
JPEG	<i>Joint Photographic Experts Group</i> - formát súboru
JSON	<i>JavaScript Object Notation</i> - formát súboru
NVM	<i>N-View Match</i> - formát súboru
CV	<i>Computer Vision</i> - počítačové videnie
RAM	<i>Random Access Memory</i> - pamäť s priamym prístupom
State-of-the-art	najmodernejšie
Framework	softwarová štruktúra, ktorá slúži ako podpora pri programovaní
Disparity	rozdielnosť
Multiview	viac násobný pohľad
Feature detection	proces nájdania kľúčových bodov pri štruktúre z pohybu

Zoznam obrázkov

Obrázok 1 - CSG objekt reprezentovaný binárnym stromom (30)	4
Obrázok 2 - Polygonálny model bizóna (31)	5
Obrázok 3 – Graficky znázornené termíny (7).....	6
Obrázok 4 - Laserová triangulácia (32).....	7
Obrázok 5 - Otáčanie hlavice (9)	8
Obrázok 6 – Point cloud (10)	9
Obrázok 7 - Princípy (11).....	10
Obrázok 8 - Skreslenie (13)	12
Obrázok 9 - Epipolárna geometria (12).....	13
Obrázok 10 - Princíp hĺbkovej mapy (13).....	14
Obrázok 11 - ZED kamera	20
Obrázok 12 - Zapojenie	28
Obrázok 13 - Dátový model pre SFMeck	29
Obrázok 14 - Dátový model pre stereo videnie.....	30
Obrázok 15 - Diagram nastavenia kamier	30
Obrázok 16 - Diagram vytvorenia 3D modelu	32
Obrázok 17 - Tok dát pri SFM	33
Obrázok 18 - Návrh programu	34
Obrázok 19 - Dátový model pre 3D model	36
Obrázok 20 - Dátový model pre SFM	37
Obrázok 21 - Stereo kamera	38
Obrázok 22 - Hlavné okno programu	39
Obrázok 23 - Nájdenie šablóny v snímkach.....	40
Obrázok 24 - Hĺbková mapa	41
Obrázok 25 - Nájdené kľúčové body	43
Obrázok 26 - Nájdené zhody medzi snímkami	44
Obrázok 27 - Mračno bodov zo SFM.....	44
Obrázok 28 - Point cloud po ohraničení kľúčových bodov	56
Obrázok 29 - Model pri znížení rozlíšenia snímok	57

Zoznam tabuliek

Tabuľka 1 - Parametre kamier (22)	21
Tabuľka 2 – Priemerný čas spracovania pri stereo videní	45
Tabuľka 3 - Čas nájdenia kľúčových bodov	47
Tabuľka 4 - Strata kľúčových bodov pri počítaní deskriptorov	48
Tabuľka 5 - Čas spracovania FAST kľúčových bodov	49
Tabuľka 6 - Čas spracovania ORB kľúčových bodov	50
Tabuľka 7 - Čas spracovania CUDA ORB kľúčových bodov	51
Tabuľka 8 - Čas párovania deskriptorov	52
Tabuľka 9 - Čas spracovania snímok	53
Tabuľka 10 - Čas spracovania a vytvorenia 3D modelu	54
Tabuľka 11 - Hodnoty výsledného 3D modelu	55
Tabuľka 12 - Čas spracovania pri ohraničení kľúčových bodov	56

1. Úvod

Táto práca sa zaoberá 3D mapovaním a modelovaním prostredia, ktoré by malo byť realizované v reálnom čase. V analýze sa skúmajú existujúce zariadenia, ktoré slúžia na získanie dát z prostredia a možnosti spracovania týchto dát. Pre pochopenie, čo všetko je zahrnuté v rekonštrukcii scény, sme analýzu rozdelili do niekoľkých častí, ktoré sa zaoberajú 3D modelmi a modelovaním 3D objektov, zariadeniami slúžiacimi na získanie dát a spracovaním získaných dát. V rámci spracovania dát získaných zo zariadení slúžiacich na snímanie prostredia opisujeme jednotlivé kroky, ktoré sú potrebné k dosiahnutiu 3D modelu. V niektorých krokoch opisujeme konkrétne algoritmy, ktoré sa snažíme podrobnejšie vysvetliť.

V ďalšej kapitole určíme špecifikácie programu, ktoré vznikli na základe dostupných poznatkov. Spomenuté sú všetky softvérové a hardvérové komponenty slúžiace na riešenie 3D rekonštrukcie scény. Na konci kapitoly určujúcej vlastnosti programu sú spísané testovacie scenáre slúžiace na kontrolu výsledného programu k overeniu vlastností.

V návrhu sa detailnejšie popisujú kroky, ktoré sa budú vykonávať počas spustenia programu. K niektorým návrhom sú k dispozícii diagramy a modely pre lepšiu predstavu návrhu. V nasledujúcej kapitole zaoberajúcej sa implementáciou sú opísané jednotlivé časti z návrhu so zdrojovými kódmi programu. V implementácii sa snažíme opísať najzaujímavejšie časti, ktoré bolo potrebné implementovať v programe pre dosiahnutie 3D modelu zo získaných 2D snímok.

Výsledný program sa podrobí testovacím scenárom, ktoré boli navrhnuté. Z týchto scenárov nameriame hodnoty, ktoré následne zobrazíme v grafoch a tabuľkách a vytvoríme zhodnotenie testovania. Na záver zhrnieme celú prácu s programom a vyvodíme úspešnosť riešenia. V prílohe budú poskytnuté informácie ohľadom programu, ako aj technická dokumentácia a používateľská príručka.

2. Analýza

V analýze sa zaoberáme existujúcimi zariadeniami a spôsobmi pre 3D rekonštrukciu scény. Rovnako sa zaoberáme najaktuálnejšími state-of-the-art riešeniami problémov, ktoré riešia problémy vyskytujúce sa pri procese vytvorenia 3D modelu z dát získaných snímaním prostredia. Pri niektorých postupoch uvádzame konkrétne algoritmy, ktoré detailnejšie vysvetľujeme. Ďalej uvádzame existujúce zariadenia slúžiace na snímanie prostredia ako aj programy na prácu a/alebo vytvorenie 3D modelu zo snímaného prostredia.

2.1. 3D Model

Model je reprezentácia skutočného objektu, ktorý je vytvorený s čo najväčším množstvom vlastností, ktoré sa dajú získať zo skutočného objektu a sú s ním totožné. Vďaka pokroku v informačných technológiách je vo vytvorených modeloch čoraz väčšie množstvo informácií. Toto množstvo získaných informácií a detailov je často krát omnoho väčšie, ako je pre následné spracovanie prípustné. Preto sa pri bežných prácach v súčasnosti viac uprednostňujú menej detailnejšie modely, ktoré neobsahujú také množstvo informácií a ich následné spracovanie nie je tak časovo náročné. Výsledná kvalita reprezentácie modelu s prebytočným množstvom informácií sa po zjednodušení vo väčšine prípadov nezmení. Existuje veľké množstvo rôznych typov modelov, ktoré sa dajú zaradiť do dvoch hlavných kategórií. Prvú kategóriu tvoria povrchové modely alebo tiež nazývané aj ako hraničné. Druhá kategória obsahuje objemové modely, ktoré predstavujú uzavreté 3D objekty. (1)

Hraničné modely sa používajú najčastejšie kvôli objektom, ktoré nie sú uzavreté. Je to z toho dôvodu, že pre tieto objekty nie je možné vytvoriť zodpovedajúci objemový model, keďže daný objekt nedokážeme pravdivo uzavrieť. Ďalší dôvod, prečo sa používa častejšie hraničný model je ten, že pri väčšine objektov nepoznáme vnútro objektu, a tým pádom dokážeme ušetriť výkon, ktorý sa využije pri spracovaní povrchu objektu. (1)

Príklady využitia 3D modelov:

- vizualizácia dizajnu (zobrazenie rôznych farieb, tvarov a motívov),
- zhodnotenie vzhľadu,
- sledovanie vzťahov (svetlo, odraz),
- určiť náklady, objem a iné vlastnosti. (2)

2.1.1. Point cloud

Point cloud je množina dátových bodov definovaná určitým systémom súradníc, ktorým môže byť dvojrozmerný alebo trojrozmerný systém. Napríklad v systéme 3D súradníc môže point cloud predstavovať obrysy určitého reálneho alebo vytvoreného fyzického systému. Point cloud sa používa najmä na vytvorenie 3D modelu, ktorý sa následne využíva v rôznych oblastiach, akými sú medicínske zobrazovanie, architektúra, 3D tlač, 3D hry, aplikácie využívajúcu virtuálnu a/alebo rozšírenú realitu a ďalšie.

V najčastejšom systéme 3D karteziánskych súradníc je bod označený tromi súradnicami, ktoré spoločne tvoria korekciu na presný bod v priestore vzhľadom na miesto pôvodu. Primárne označenie ôs sú x , y a z , ktoré sa rozprestierajú v dvoch smeroch a súradnice určujú vzdialenosť bodu od priesečníka osí a smeru divergencie vyjadrený ako $+$ alebo $-$.

Zariadenia na získanie mračna bodov, ktoré sa nazývajú aj 3D snímače, zhromažďujú bodové merania z objektov alebo fotografií v reálnom svete pre point cloud, ktorý sa môže následne pretransformovať do rôznych typov 3D modelov. Takýmito zariadeniami môžu byť senzory Microsoft Kinect, rôzne stereo kamerové systémy akým je ZED kamera alebo vysokorýchlostné laserové snímače s vysokým rozlíšením, ktoré dokážu vyprodukovať veľké množstvo bodových údajov pri vysokej frekvencií.

2.2. 3D Modelovanie

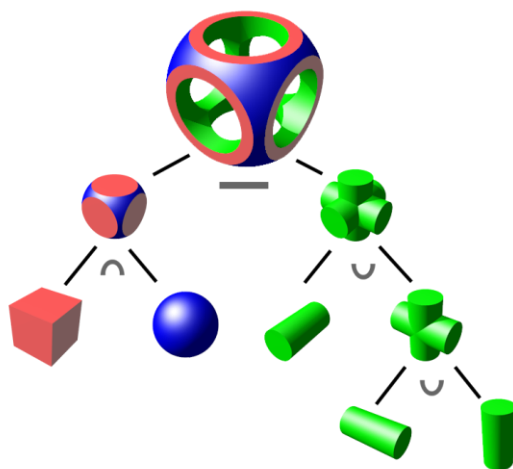
Modelovanie môžeme popísať ako proces alebo aktivita tvarovania a vytvárania jednotlivých objektov, ktoré sú neskôr použité v scéne. Pri 3D modelovaní vychádzame z geometrických primitív alebo z mračna bodov, ktorý je získaný pomocou 3D snímačov, popřípadě z iných procesov na prácu s objektmi a modelmi. Získaný point cloud sa následne pomocou rôznych modelovacích techník pretransformuje na 3D model snímaného objektu, z ktorého sme získali zodpovedajúci point cloud. Existuje mnoho modelovacích techník, z ktorých najčastejšie sa používajú:

- CSG,
- NURBS,
- polygonálne modelovanie.

Modelovacie procesy môžu zahŕňať úpravu povrchu alebo materiálových vlastností, akými sú lesk, farba a iné vlastnosti. Model objektu môže byť vybavený aj kostrou, ktorá ovplyvňuje zodpovedajúce časti modelu. Hlavné využitie kostry modelu je pri animácii objektu, kedy je potrebné pohybovať len s určitou časťou modelu. (3)

2.2.1. Constructive solid geometry

Skrátene CSG, poprípade v slovenských článkoch je možné pre tento výraz nájsť termín konštruktívna geometria telies. Tento spôsob modelovania využíva množinu geometrických primitív, akými môžu byť guľa, kocka alebo valec, ktorým je možné nastaviť parametre. Táto vlastnosť umožňuje zdefinovať základné hodnoty, akými sú polomer kruhu, dĺžky strán alebo aj umiestnenie v priestore pre dané primitívum. Tieto hodnoty parametrov sa uplatnia pri vytváraní týchto primitív. S pomocou prieniku, rozdielu a ďalších logických operácií, dokážeme vytvoriť CSG strom, ktorý si uchováva postupnosť operácií a reprezentuje výsledný model ako je to vidieť na obrázku 1. (4) (5)



Obrázok 1 - CSG objekt reprezentovaný binárnym stromom (30)

Ako je vidieť na obrázku 1, výsledný model, ktorý sa získal vďaka logickým operáciám, pozostával pôvodne z červenej kocky, modrej guli a troch zelených valcov. Niektoré modely nemusia mať len jedno jediné riešenie, ale je možné sa k výslednému modelu dopracovať aj pomocou inej sady geometrických primitív a logických operácií.

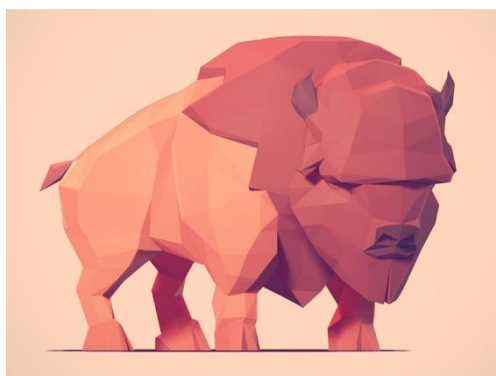
Jedna z nevýhod pri CSG modelovaní je vytvorenie modelu z voľných tvarov, ktoré sa reprezentujú v mračne bodov. Vytvorenie modelu z mračna bodov je veľmi náročné, a preto si CSG modely našli miesto v computer-aided design softvéroch často známe pod skratkou CAD. CAD software je možné v slovenských textoch nájsť pod pojmom „softvér pre automatizované projektovanie“. Jeden z najznámejších CAD programov, v ktorom sa využíva tento typ modelu je AutoCAD. Tieto softvéry sa okrem grafických činností používajú aj na výrobné alebo inžinierske výpočty, kde sa využíva jedna z mnohých výhod CSG. Táto výhoda, ktorú spomíname, umožňuje zdefinovať vlastnosti objektom z CSG stromu (napríklad vlastnosti akými sú vodotesnosť, pevnosť a ďalšie) na získanie vlastností výsledného modelu. Tie sa získajú pomocou logických operácií podľa CSG stromu. Ďalšia výhoda je, že aj pri

komplexných scénach je možné vytvoriť strom s malým počtom primitívov, ktorý bude potrebovať menej času na spracovanie, ako by to bolo pri použití iných modelovacích techník.

(6)

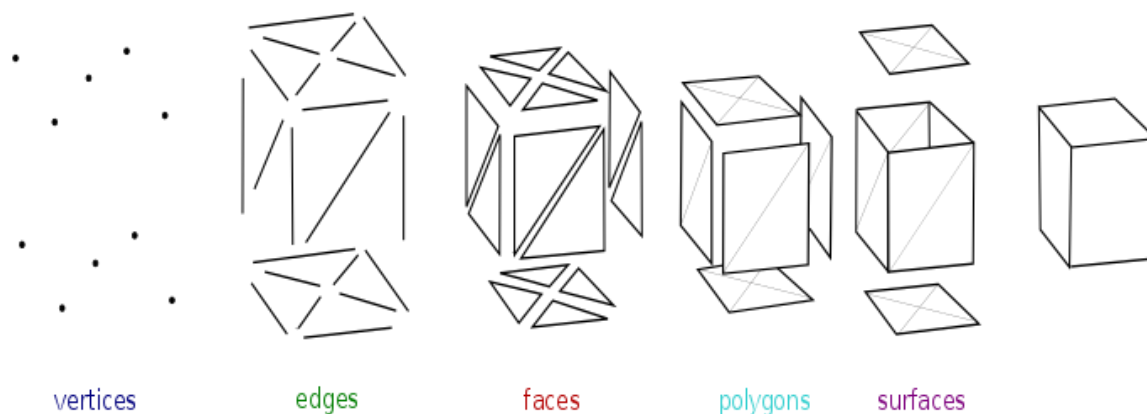
2.2.2. Polygonálne modelovanie

Bod v trojdimenzionálnom priestore je základným objektom pri modelovaní polygonálneho modelu, ktorý sa taktiež nazýva ako vrchol. Pri spojení dvoch vrcholov s priamkou získame hranu a pri troch vrcholoch spojených s tromi hranami získame najjednoduchší polygón v euklidovskom priestore, ktorý sa využíva pri polygonálnom modelovaní. Tento najjednoduchší polygón sa v bežnej komunikácii nazýva trojuholník. Zložitejšie polygóny dokážeme získať kombináciou viacerých trojuholníkov alebo trojuholníkov s objektmi, ktoré pozostávajú z viac ako troch hrán. Najbežnejšími používanými polygónmi pri vytváraní polygonálneho modelu sú trojuholník a druhý najjednoduchší polygón štvoruholník, ktorý vznikne zo štyroch hrán (štvorhranný polygón). Príklad takéhoto polygonálneho modelovania je možné vidieť na obrázku 2, ktorý predstavuje low polygonal modeling. Skupina polygónov navzájom prepojená zdieľanými hranami sa nazýva mesh. Mesh sa dá popísať aj ako sieť polygónov.



Obrázok 2 - Polygonálny model bizóna (31)

Pre lepšiu predstavu, čo vlastne znamená mesh, je veľmi dobre znázornené na obrázku bizóna, v ktorom súbor susediacich polygónov tvoriacich sieť predstavuje mesh modelu. Na obrázku číslo 3 sú znázornené jednotlivé termíny, s ktorými sa často stretáva pri polygonálnom modelovaní. Aby bol mesh atraktívny, je žiadúce, aby sa navzájom nepretínal, a aby žiadna hrana neprešla cez polygón. Mesh by nemal obsahovať zdvojené elementy (hrany, vrcholy a ďalšie), ktoré by narušali mesh a zároveň výsledný 3D polygonálny model. (3)



Obrázok 3 – Graficky znázornené termíny (7)

Zložitosť polygonálneho modelu závisí od niekoľkých faktorov, pričom jedným z najdôležitejších je množstvo polygónov a hrán, ktoré obsahuje. Poly count (počet polygónov) je termín, ktorý sa používa pri diskusii o množstve polygónov v modeli. Podľa počtu polygónov môžeme rozdeliť modely na takzvané low alebo high poly modely. Tieto termíny sú však veľmi subjektívne, pretože závisia od cieľového zariadenia, ktoré bude tieto modely vykresľovať. Moderné počítače a konzoly dokážu vykresliť milióny trojuholníkov, pričom mobilné zariadenia a vreckové konzoly sú zvyčajne obmedzené z hľadiska výpočtového výkonu. Kvôli nízkej časovej náročnosti sa low poly modely využívajú najmä pri real-time aplikáciách, v ktorých sa vyžaduje, aby snímková frekvencia bola čo najvyššia. Na druhej strane high poly modely sa uplatnili vo filmoch, kde sa očakáva od výsledného modelu, čo najrealistickejšie zobrazenie a časová náročnosť pre spracovanie nie je kľúčová. Vygenerovanie jednej snímky takéhoto high poly modelu, môže presahovať niekoľko hodín aj dní. Na dosiahnutie nižšieho času na vykreslenie, vznikli takzvané renderovacie farmy, ktoré majú za úlohu urýchliť celý proces spracovania. (8)

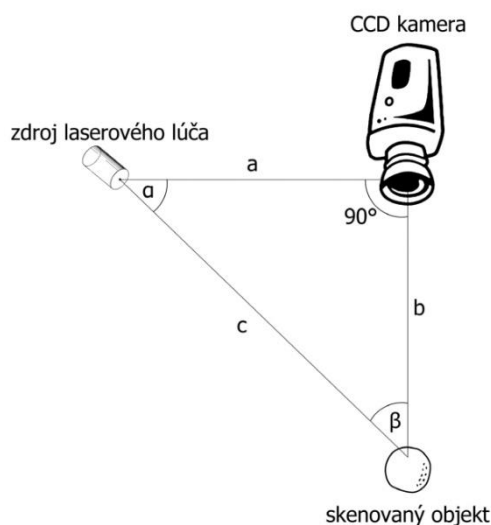
Najväčšou nevýhodou polygonálneho modelovania je tá, že polygóny nedokážu presne znázorniť zakrivenú plochu. Pre znázornenie zakrivenej plochy sa vyžaduje veľké množstvo malých polygónov, ktoré môžu extrémne zvýšiť časovú náročnosť celého modelu. Na druhej strane sú tieto modely rýchlejšie ako iné reprezentácie. Pri zobrazovaní iných typov modelov nie je možné dosiahnuť interaktívnu snímkovú frekvenciu (10 a vyššie) s podobným množstvom detailov ako pri polygonálnom modeli. (3)

2.3. Mapovanie prostredia

Základným krokom pri vytvorení 3D modelu prostredia je získanie vhodných dát tohto prostredia, ktoré bude možné pretransformovať do odpovedajúceho 3D modelu. V tejto kapitole spomenieme najčastejšie používané zariadenia na snímanie prostredia a vysvetlíme princípy, akými tieto zariadenia snímajú prostredie. V ďalších kapitolách si opíšeme základné kroky, ktoré sú potrebné k získaniu žiadaného 3D modelu. V kapitolách sa venujeme aj state-of-the-art postupom, ktoré sa v dobe písania práce používajú.

2.3.1. Snímanie pomocou laseru

Laserové skenery sú založené na vysielaní laserového lúča, ktorý je následne od objektu odrazený naspäť a je zachytený pomocou snímačov typu CCD/CMOS. Na výpočet vzdialenosti laseru od objektu sa využíva metóda triangulácie, ktorá je znázornená na obrázku 4. Presnosť takéhoto laserového zariadenia závisí hlavne od vzdialenosti laserového lúča a CCD/CMOS snímača. Táto vzdialenosť sa vo väčšine prípadov nedá meniť. (9)



Obrázok 4 - Laserová triangulácia (32)

Vzdialenosť medzi vysielateľom laserového zariadenia a objektu sa vypočíta podľa času, ktorý prejde medzi vyslaním lúča a jeho spätným získaním. Táto metóda výpočtu vzdialenosti je vhodná na stredné a väčšie vzdialenosti. Čím viac je objekt vzdialený, tým je lepšie snímať väčšie objekty. Je to z toho dôvodu, že pri malých vzdialenostiach zariadenie príjme lúč skôr ako začne merať čas a pri malých objektoch sa nemusí vypočítať dostatočné množstvo bodov na vygenerovanie modelu. Za každý jeden späť získaný lúč sa vypočíta vzdialenosť práve jedného bodu zo snímaného prostredia. Týmto spôsobom vieme získať dostatočné množstvo bodov pre vytvorenie 3D mračna bodov, z ktorého bude možné v ďalších krokoch vytvoriť

zodpovedajúci 3D model. Pre prípad, že chceme mať vo výslednom modeli aj textúru, je potrebné pridať k laserovému zariadeniu ďalší snímač, ktorý dokáže zachytiť farbu snímaného objektu. (9)

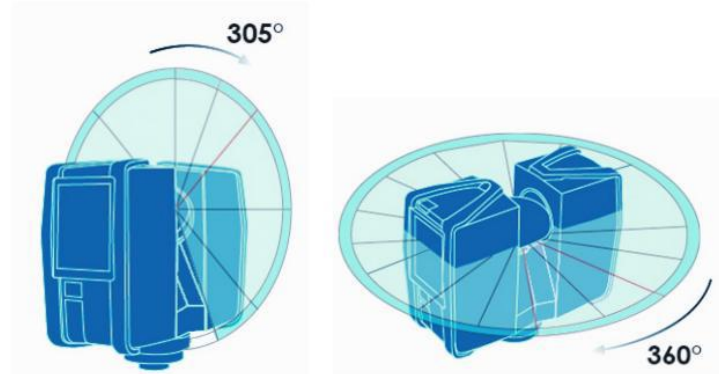
Ako už bolo spomenuté, pri vyslaní jedného lúča vieme získať vzdialenosť jedného bodu. Preto je potrebné vyslať, čo najväčšie množstvo lúčov, aby sme získali dostatočne veľký point cloud na vytvorenie 3D modelu. Čím väčší point cloud, tým bude výsledný model presnejšou a dôveryhodnejšou kópiou snímaného prostredia. Existujú tri spôsoby vysielania lúča, ktorými vieme docieľiť potrebný point cloud na vygenerovanie modelu:

a) Otáčanie vysieláča a prijímača laserového lúča okolo vodorovnej osi

Vysielač vyšle prvý lúč a po jeho zachytení sa otočí na ďalšiu polohu a vyšle ďalší lúč. Uhol medzi polohami je spravidla vždy menší ako 1° . Celý tento proces sa opakuje pokiaľ sa vysielač neotočí o 360° . Z dôvodu konštrukcie, pri ktorom býva laserové zariadenie často postavené na trojnožke kvôli stabilite, je tento rozsah zväčša menší ako 360° . Doba tohto procesu snímania trvá menej ako 1 sekundu. Týmto získame body kružnice okolo vysielača, ktorý je vidieť na obrázku 5.

b) Otáčanie vysieláča a prijímača laserového lúča okolo zvislej osi

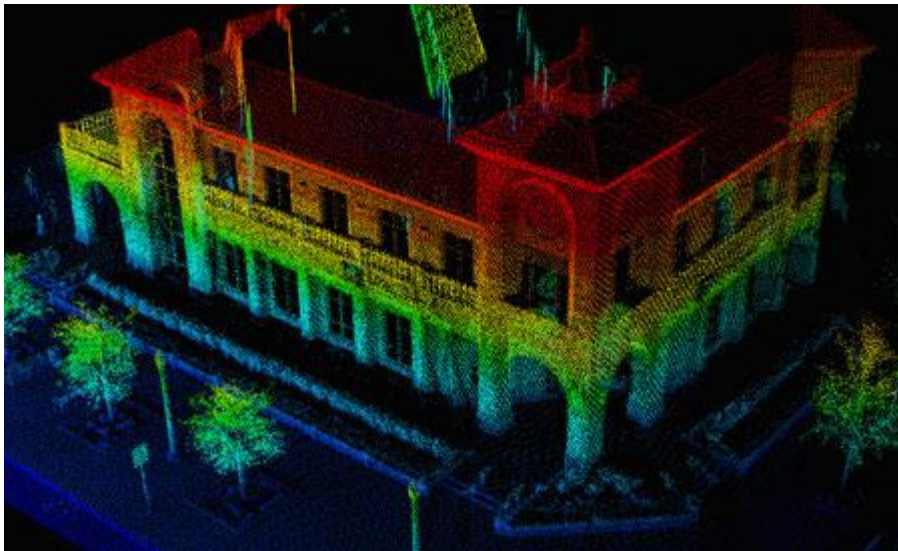
Spôsob je rovnaký ako pri otáčaní okolo vodorovnej osi. V tomto prípade sa zariadenie otočí vždy o 360° , keďže mu z konštrukčného hľadiska nič nebráni pri otočení. Spôsob snímania je možné vidieť dole na obrázku 5.



Obrázok 5 - Otáčanie hlavice (9)

c) Skenovanie objektu z viacerých pozícií

Pri jednorazovom skenovaní objektu nedokážeme získať potrebné množstvo údajov na vytvorenie 3D modelu. Preto je nutné získať nasledujúce údaje z iných uhl'ov pohľadov. Každé ďalšie snímání objektu je preto získavané z inej pozície. Po získaní údajov z viacerých miest, dostaneme mračno bodov, ktoré je možné pretransformovať pomocou rôznych softvérových programov do 3D modelu snímaného objektu. (9)



Obrázok 6 – Point cloud (10)

Na obrázku vyššie je zobrazený výsledný point cloud, ktorý bol získaný pomocou mobilného laserového zariadenia, taktiež nazývaného ako mobilný LIDAR. Na získanie 3D modelu sa obdržaný point cloud následne spracuje pomocou grafického programu. Medzi najznámejšie značky patria programy ako Blender, 3Ds Max, Maya a iné. Takýto grafický program z načítaného mračna bodov vytvorí žiadaný 3D model pomocou techník a nástrojov, ktoré z bodov vytvoria spojnice a vytvorený priestor medzi spojniciami vyplní. V prípade, že máme k dispozícii zo snímania prostredia aj textúru, je možné túto textúru na model premietnuť.

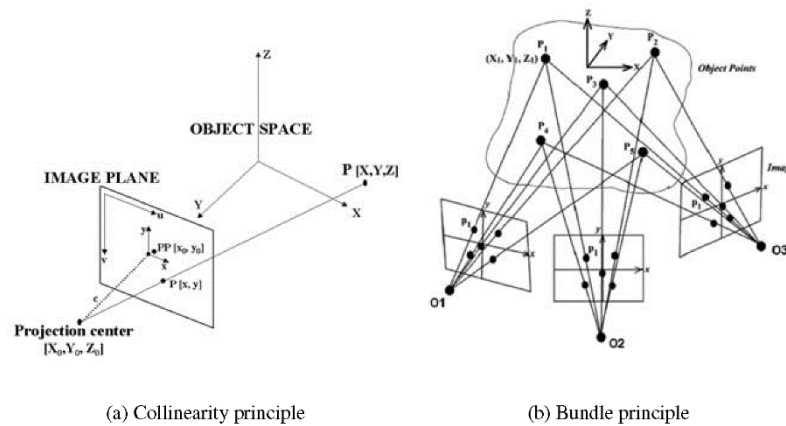
2.3.2. Snímanie pomocou kamery – stereo videnie

Pri tomto snímaní pomocou dvoch kamier, ktoré vytvárajú stereo kameru, sa riešenie inšpiruje ľudským videním a základnou myšlienkou triangulácie. Takáto stereo kamera snímajúca objekt umožňuje zo získanej stereo snímky pomocou fotogrametrií určiť vzdialenosť snímaného objektu od kamier. Dôležitosť a podstatu fotogrametrie ako aj ďalších potrebných krokov k získaniu mračna bodov (kalibrácia, rektifikácia a ďalšie) sú podrobnejšie opísané v jednotlivých podkapitolách.

2.3.2.1. Fotogrametria

Fotogrametria je jedna z najznámejších a najdôležitejších obrazových techník, ktorá umožňuje odvodenie presných, metrických a sémantických informácií z fotografií. Táto obrazová technika premieňa 2D snímky na 3D dáta, čím sa získava geometrický vzťah medzi získanými snímkami a scénou počas mapovania prostredia. Fotogrametria všeobecne využíva

minimálne dva obrazy tej istej scény alebo objektu z rôznych uhlov na získanie 3D dát. Objekt alebo scéna, ktorá je zachytená z rôznych polôh, umožňuje stereoskopický pohľad a odvodenie 3D informácií o pozorovanom objekte v prekrývajúcich sa oblastiach obrazov. (11)



Obrázok 7 - Princípy (11)

Základným princípom fotogrametrického spracovania je použitie viacerých snímok (minimálne dvoch) a princíp kolinearnosti (obrázok 7a). Takýto princíp vytvára vzťah medzi snímkom a objektovým priestorom, ktorý definuje priamku medzi perspektívnym stredom kamery, obrazovým bodom $P[x, y]$ a objektovým bodom $P[X, Y, Z]$. Kolineárny model je formulovaný ako (11):

$$x = -f \frac{r_{11}(X-X_0)+r_{21}(Y-Y_0)+r_{31}(Z-Z_0)}{r_{13}(X-X_0)+r_{23}(Y-Y_0)+r_{33}(Z-Z_0)} + x_0 \quad [1]$$

$$y = -f \frac{r_{12}(X-X_0)+r_{22}(Y-Y_0)+r_{32}(Z-Z_0)}{r_{13}(X-X_0)+r_{23}(Y-Y_0)+r_{33}(Z-Z_0)} + y_0 \quad [2]$$

Kde vnútorné parametre orientácie predstavujú

f konštanta kamery alebo ohnisková vzdialenosť
 x_0, y_0 hlavný bod

a vonkajšie parametre orientácie

X_0, Y_0, Z_0 pozícia perspektívneho centra
 $r_{11}, r_{12}, \dots, r_{33}$ elementy rotačnej matice

x, y súradnice 2D obrázku
 X, Y, Z súradnice 3D objektu

Všetky merania vykonané na digitálnych obrazoch odkazujú na súradnicový systém pixelov, zatiaľ čo rovnica kolinearnosti odkazuje na metrický súradnicový systém obrazu.

Konverzia pixelov na súradnice obrazu sa uskutočňuje pomocou premeny homogénnej deformácie, ktorá pozná rozmery senzoru a veľkosť pixelu. (11)

Pre každý obrazový bod zachytený aspoň dvoma snímkami sa nazýva aj ako spojovací bod, je vytvorená kolineárna rovnica. Výsledok systému rovníc je získaný pomocou iteratívnej metódy, ktorá vyžaduje dobrú inicializačnú aproximáciu neznámych parametrov. Systém rovníc je iteratívne riešený pomocou metódy najmenších štvorcov a po linearizácii a zavedení chybového vektora e , sa môže vyjadriť ako (11):

$$-e = A \cdot x \cdot l \quad [3]$$

kde:

e	chybový vektor
A	matica $n \times m$
x	neznámy vektor
l	pozorovací vektor

Výhoda fotogrametrie spočíva v tom, že obrázky obsahujú všetky potrebné informácie, ktoré sú pre tvorbu 3D modelu dôležité, ako napríklad geometria, textúra a iné. Zariadenia na získanie obrazov prostredia (fotoaparáty, kamery, mobily a ďalšie) sú vo všeobecnosti lacné, jednoduché na premiestnenie, ľahko použiteľné a majú potenciál na vysokú presnosť. Najväčšia nevýhoda spočíva práve v zložitosti získania 3D súradníc zo snímaného objektu. (11)

2.3.2.2. Kalibrácia

Aby bolo možné získať čo najlepší výsledok z mapovania prostredia pomocou kamier, je potrebné najprv vykonať kalibráciu pre použité kamery. Je to dôležitý prvok, ktorý nám umožňuje, aby sme v ďalších fázach boli schopní porovnávať reálne predmety zo získaných snímok. Reálny predmet predstavuje objekt zo snímaného prostredia. Z dôvodu, že kamery majú od výroby určité skreslenie kvôli šošovke, môže sa stať, že vlastnosti snímaného objektu nie sú totožné s tými, ktoré sú na získaných snímkach. Takýmito vlastnosťami sú najčastejšie rovné, rovnobežné alebo kolmé čiary. Ak sa na snímkach nezobrazuje rovnaký objekt s rovnakými vlastnosťami, je v ďalších fázach takmer nemožné porovnávať snímky.

Ku kalibrácii kamery je potrebné získať informácie ohľadom vnútorných nastavení kamery, ktorými sú ohnisková vzdialenosť a stred kamery. Získané údaje k danej kamere je možné aplikovať aj v neskorších prípadoch. Je to z toho dôvodu, že tieto informácie vychádzajú z konštrukcie kamery a behom používania kamery sa nemenia. V prípade zmeny rozlíšenia

kamery je možné tieto údaje použiť, ak sa dodatočne prispôbia k novému rozlíšeniu. Tieto hodnoty sa prezentujú v takzvanej matici kamery, ktorá je znázornená nižšie. (12)

$$\mathbf{A} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad [4]$$

Ako už bolo spomenuté, hodnoty f_x a f_y predstavujú ohniskovú vzdialenosť, ktorá sa taktiež nazýva aj ako konštanta kamery a hodnoty c_x a c_y predstavujú vektor posunutia stredu kamery. Pomocou tejto matice je možné vypočítať súradnice priestorového bodu zo súradníc snímky. Dôvod, prečo je potrebná kalibrácia kamier je ten, že pri prechode obrazu cez šošovku dochádza k skresleniu obrazu. Tento jav nám prekáža v porovnávaní reálneho objektu z viacerých snímok, a preto je nutná kalibrácia, ktorá nám tento jav zo snímok odstráni. Existujú dva typy skreslenia, ktoré môžu nastať pri snímaní. Prvým typom skreslenia je radiálne skreslenie, ktoré je znázornené na obrázku 8 a druhým typom je tangenciálne skreslenie. (12)



Obrázok 8 - Skreslenie (13)

Radiálne skreslenie je známe aj ako rybie oko. Toto skreslenie spôsobuje, že sa rovné čiary zobrazujú ako krivky. Radiálne skreslenie sa dá zapísať pomocou rovníc.

$$x_d = x(1 + k_1 r^2 + k_2 r^4 + \dots) \quad [5]$$

$$y_d = y(1 + k_1 r^2 + k_2 r^4 + \dots) \quad [6]$$

Hodnota r predstavuje vzdialenosť bodu od stredu snímky a hodnota k_n predstavuje koeficient skreslenia. Rovnako sa dá zapísať aj druhý typ skreslenia.

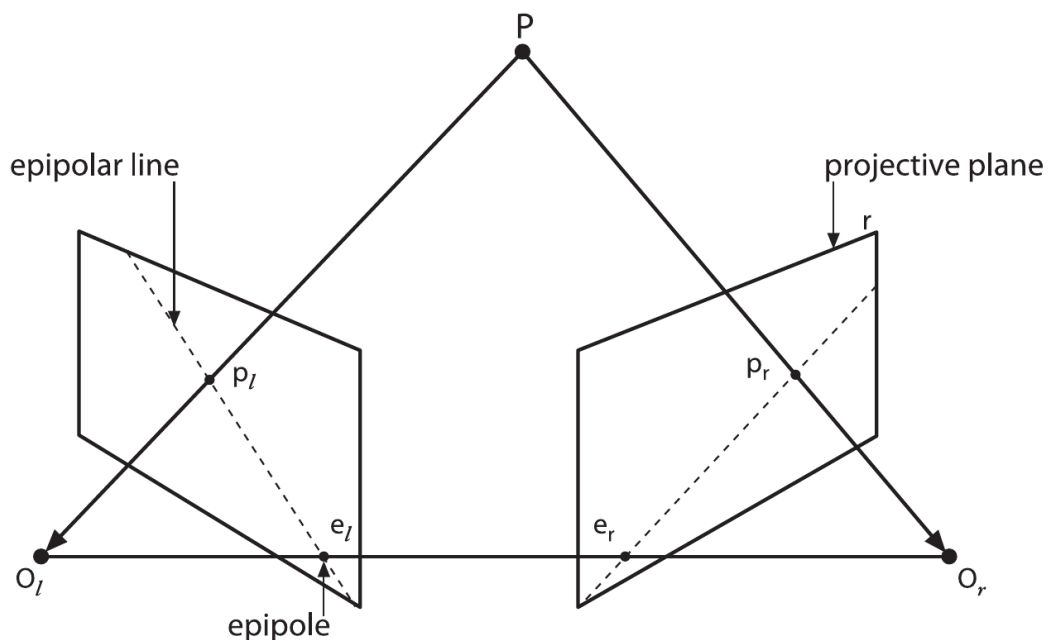
$$x_d = x + (2p_1 xy + p_2(r^2 + 2x^2)) \quad [7]$$

$$y_d = y + (p_1(r^2 + 2y^2) + 2p_2 xy) \quad [8]$$

V oboch prípadoch sa dajú rovnice rozšíriť o ďalšie stupne k alebo p . Parametre skreslenia sú po celý čas rovnaké, ako to bolo v prípade vnútorných vlastností kamery. Preto si môžeme tieto parametre uchovať a uplatniť ich v neskorších použitíach kamery. (12)

2.3.2.3. Rektifikácia

Po úspešnej kalibrácii kamier, ktorou sme sa zbavili nežiadúceho javu skreslenia obrazu, je možné pokračovať k ďalšiemu kroku rektifikácii. Tento krok nám urýchli hľadanie rovnakých bodov z dvoch snímok. Výsledok rektifikácie je horizontálne zarovnanie snímok, vďaka ktorému stačí hľadať páry bodov na rovnakej x-ovej čiare v oboch snímkach. Tento stav bude dosiahnutý pomocou transformácii oboch snímok, v ktorých bude stred kamier v rovnakej výške a ich osi rovnobežné. Rektifikácia funguje na princípe epipolárnej geometrie, ktorá je znázornená na obrázku 9. (12)



Obrázok 9 - Epipolárna geometria (12)

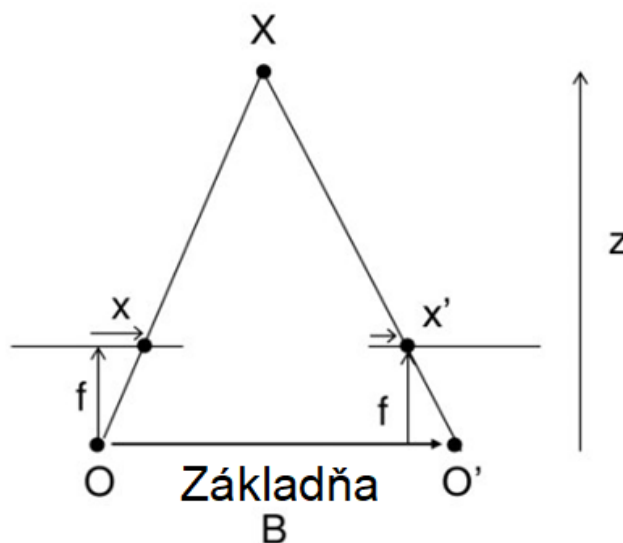
Máme snímky z dvoch kamier, ktorá každá má vlastné ohnisko (O_l a O_r). Bod P nám predstavuje reálny objekt v priestore, ktorý bol snímaný kamerami. Ten sa v jednotlivých snímkach z kamier premietne ako bod p_l a p_r . Body e_l a e_r nám predstavujú ohnisko druhej kamery. e_r predstavuje premietnutie bodu O_l pravou kamerou a to isté sa spraví s druhým bodom O_r v ľavej kamere, ktorý sa označí ako e_l . Body O_l , O_r a P nám určia epipolárnu rovinu, v ktorej body e_l a e_r predstavujú epipóly. Epipolárne priamky nám na obrázku predstavujú priamky e_l-p_l a e_r-p_r . Pre vysvetlenie zmyslu epipól je dôležité si všimnúť, že všetky objekty v priestore nachádzajúce sa na priamke O_l-P sa zobrazia v ľavej kamere v bode p_l . Rovnako to

platí aj pre pravú kameru. Z toho nám vychádza, že sa priamka O_r-P zobrazí v ľavej kamere presne na priamke e_l-p_l .

Po tejto operácii sa nám značne urýchlí hľadanie párov bodov medzi snímkami, keďže zobrazenie objektu v priestore sa bude nachádzať v druhej kamere na epipolárnej priamke, ktorú stačí prehľadať. Týmto spôsobom sa nám zníži čas potrebný na nájdenie páru a aj pravdepodobnosť vzniku chybného páru. Ďalšou výhodou je zachovanie vodorovného poradia, čo znamená, že ak sa body X a Y nachádzajú v oboch snímkach a bod X je zobrazený vyššie ako bod Y v rámci ľavej kamery, tak to platí aj pre pravú kameru. Teraz, keď sme bližšie popísali epipolárnu geometriu, nám zostáva už len vodorovné zarovnanie epipolárnych priamok. Toho dosiahneme pomocou esenciálnej a fundamentálnej matici. Tie sa vo väčšine prípadov označujú ako E a F . Matica E obsahuje údaje o vzájomnej polohe oboch kamier a matica F obsahuje rovnaké údaje ako matica E , pričom je obohatená o údaje vnútorných vlastnostiach kamery. (12)

2.3.2.4. Hĺbková mapa

Po kalibrácii a rektifikácii máme obe kamery pripravené na párovanie bodov pre nasledujúce získané snímky. Výsledky ďalšieho procesu predstavujú informácie o rozdiel pozícií bodov z jednej snímky od zodpovedajúcich bodov z druhej snímky. Tento proces je veľmi kritický a výsledok nie je vždy ideálny. Pravdepodobnosť výskytu chýb v tomto procese sa výrazne zníži po úspešnej a správnej kalibrácii kamier a vďaka rektifikácii sa celý proces výpočtu urýchlí. (12)



Obrázok 10 - Princíp hĺbkovej mapy (13)

Keď sa nám podarí priradiť ku každému bodu z jednej snímky vzdialenosť zodpovedajúceho bodu z druhej snímky, získame tým mapu disparity. Hodnotu disparity je možné opísať pomocou nasledujúceho vzťahu. (12)

$$\text{disparity} = x - x' = \frac{Bf}{z} \quad [9]$$

Túto mapu je možné jednoducho vizualizovať. Vo väčšine prípadov sa na vzdialenosti aplikujú odtiene šedej, vďaka čomu získame obrázok s vizuálnou informáciou, ktorá nám poskytuje údaje o priestorovom rozmiestnení objektov v scéne. Získaný obrázok sa nazýva hĺbková mapa. Po získaní máp rozdielnosti pre obe snímky je možné spraviť kontrolu konzistencie a zistiť tak oblasti, v ktorých nebolo možné vykonať párovanie. Pixel v hĺbkovej mape je platný v prípade, že rozdiel hodnoty disparity bodu v ľavej snímke a hodnoty disparity zodpovedajúcemu bodu v pravej snímke, je menší ako určená hranica pre akceptáciu. V prípade, že je rozdiel vyšší ako určená hranica je skúmaný pixel neplatný. (12)

Jeden zo základných algoritmov, ktorý rieši výpočet hĺbkovej mapy, je stereo block matching algoritmus, taktiež nazývaný ako StereoBM. Ako už z názvu vyplýva, tento algoritmus využíva stereo obrázky, ktoré pozostávajú z dvoch snímok zachytených v rovnakom čase, z rôznych pozícií, snímajúce rovnakú scénu. Problém, ktorý treba riešiť pri porovnávaní snímok, je v spárovaní každého bodu prvej snímky so zodpovedajúcim bodom v druhej snímke. K riešeniu tohto problému sa využíva jednoduchá operácia súčtu absolútnych rozdielov, v angličtine sum of absolute differences skrátene SAD. Pred samotným výpočtom hĺbkovej mapy je potrebné previesť oba snímky na snímku využívajúcu len odtiene šedej, ktorej pixely majú hodnoty od 0 do 255. Následne vyberieme bloky z oboch snímok, v ktorých hodnoty pixelov od seba odčítame a absolútnu hodnotu každého pixelu sčítame. Týmto procesom dostaneme pre každý pixel SAD hodnotu porovnaných snímok. Menšia hodnota znamená, že porovnávané bloky sú si viac podobné. Porovnanie, ktoré má najmenšiu hodnotu by malo predstavovať identické bloky. Tento algoritmus primárne porovnáva horizontálne, a preto je potrebné použiť rektifikáciu, ktorá nám zarovná snímky do rovnakej výšky. Existuje verzia algoritmu nazývaná semi global block matching, skrátene StereoSGBM, ktorá porovnáva hodnoty aj v iných smeroch.

Nevýhoda tohto postupu porovnávania blokov je tá, že zo snímok nevieme získať pozíciu zariadenia, ktoré dané snímky zachytilo. Pre vytvorenie rozsiahleho mračna bodov je potrebné k zariadeniu pripojiť inerciálnu meraciu jednotku (ďalej ako IMU), z ktorej vieme získať potrebné údaje o polohe zariadenia. Získané hodnoty z IMU sa následne aplikujú na

každý point cloud vypočítaný z páru snímok. Aplikáciou daných hodnôt z IMU docielime správne umiestnenie bodov v 3D priestore vzhľadom na už vypočítane body. Ďalšou možnosťou je získať informácie o zmene snímky oproti predchádzajúcej, ktorá sa využíva v ďalšom spôsobe mapovania prostredia pomocou kamery - multiview.

2.3.3. Snímanie pomocou kamery – multiview

Ďalší spôsob rekonštrukcie scény využívajúci fotoaparát alebo kameru je pomocou získania snímok z rôznych uhlov a pozícií snímaného prostredia. Následne sa zo snímok snažíme odhadnúť vzťah medzi jednotlivými snímkami a vygenerovať mračno bodov. Tento spôsob snímania pri rekonštrukcii scény má určité pozitíva a negatíva oproti snímaniu pomocou stereo kamery. Oba spôsoby však využívajú techniky fotogrametrie. V tejto kapitole si opíšeme základné kroky potrebné k dosiahnutiu mračna bodov zo série snímok, ktoré sú získané prevažne z jedného zariadenia, poprípade môžu byť z viacerých odlišných fotoaparátov/kamier.

2.3.3.1. Feature detection

Základným problémom pri porovnávaní snímok zo série je extrahovanie popisu lokálnych bodov tak, aby to umožňovalo správnu identifikáciu zhody medzi bodmi. Dané body by nemali brať ohľad na zmenu orientácie, veľkosti, svetla alebo 3D pozície. Prvým krokom pri porovnávaní zahŕňa identifikáciu spoločných bodov v sérii snímok. Práve tieto body umožňujú nájsť zhodu v rôznych snímkach a následnú rekonštrukciu scény. (14)

Prvé algoritmy sa zakladali na štatistike porovnávania alebo hľadania rohov. Tieto postupy sa používali najmä v aplikáciách, ktoré sa zaoberali sledovaním krátkych pohybov alebo stereo snímok. Tieto postupy boli aplikovateľné len v prípade, kedy snímky mali rovnakú veľkosť a boli snímané z podobného uhla. Feature point (ďalej ako kľúčový bod) je bod/pixel, ktorý je invariantný na zmeny veľkosti, orientácie, osvetlenia, ktorý je potrebný na širšie porovnávanie. Oblasť, ktorá sa porovnáva zo snímky, musí byť schopná sa adaptovať na zmenu perspektívy medzi snímkami. (14)

Najčastejšie používaním algoritmom na hľadanie feature points je scale-invariant feature transform, skrátene SIFT. Tento algoritmus umožňuje vykonať dramatické zmeny vlastností pixelu len s malými zmenami v deskriptore. Okrem toho je SIFT algoritmus robustný voči zmenám v 3D pohľade pre nerovné povrchy. SIFT algoritmus sa skladá zo štyroch hlavných bodov:

- **Detekcia priestorových extrémov**

Tento prvý krok zahŕňa efektívnu identifikáciu miesta, ktoré dokážeme opakovane nájsť v rozličných pohľadoch. Spacescale prístup sa využíva na detekciu lokácie hľadáním stabilných vlastností, ktoré sú nemenné voči zmenám vo veľkostiach. Čiernobiely obraz intenzity sa postupne konvoluje s Gaussovou funkciou v rôznych veľkostiach a rozdielom medzi po sebe nasledujúcich Gaussových snímok. Lokálne extrémny sú nájdené pomocou porovnávania každého vzorkového bodu so susednými ôsmymi v danom obraze a deviatimi susedmi nad a pod stupnicou.

- **Lokalizácia kľúčových bodov**

SIFT následne vykoná prispôsobenie 3D kvadratickej funkcie pre každého kandidáta z kľúčových bodov. Hustota kľúčových bodov identifikovaných na snímke závisí od textúry, ostroty a rozlíšenia snímky. Komplexné scény budú fungovať najlepšie, zatiaľ čo bez tvaré alebo jednotvárne povrchy, akými sú sneh, piesok a podobne, budú na lokalizáciu náročnejšie.

- **Pridelenie orientácie**

Konzistencia orientácie pre každý kľúčový bod je priradená analýzou dominantných smerov lokálnych gradientov použitím Gaussian-smooth snímky k najbližšiemu rozsahu kľúčových bodov.

- **Deskriptor kľúčových bodov**

Ďalej je potrebný deskriptor pre každý kľúčový bod, ktorý je dostatočne rozlíšiteľný a zároveň stabilný voči zmenám v 3D pohľade alebo osvetlení. Prístup SIFT k získaniu deskriptoru spočíva v tom, že zvažuje citlivosť gradientov, ale nie ich lokáciu. Na gradienty sa aplikuje Gaussová váhová funkcia, aby sa predišlo vzdialeným gradientom. Gradienty sú nahromadené do orientačných histogramov veľkosti 4x4. Deskriptor je teda pole histogramov o veľkosti 4x4 s ôsmymi orientačnými košmi, ktoré sa zapíše do 128 elementového vektora pre každý kľúčový bod. Aby sa predišlo osvetľovacím efektom, tak sa vektor normalizuje na dĺžku jednotky, čím sa upravuje voči zmenám v kontraste. (14)

2.3.3.2. Zhoda kľúčových bodov

Potom, ako sa nájdu kľúčové body, je potrebné nájsť zhodu medzi kľúčovými bodmi v rôznych snímkach. Nie je však garantované, že každý kľúčový bod bude mať partnera v inej snímke. Preto je potrebné použiť metódy na odstránenie nevhodných bodov.

Zložitosť deskriptoru kľúčových bodov a veľký počet týchto bodov znamená, že euklidovské prehľadávanie najbližšieho suseda v takomto veľkom priestore je náročné a nákladné na výpočet. Účinné riešenie tohto problému sú k-d stromy, ktoré sú typom binárneho stromu, často používaného na výpočet najbližšieho suseda. Hľadanie najbližšieho suseda, tak pracuje rekurzívne a výhoda tejto dátovej štruktúry je tá, že dokáže rýchlo eliminovať veľké oblasti prehľadávaného priestoru. (14).

2.3.3.3. Structure from Motion

Doslovný preklad Structure from Motion je štruktúra z pohybu, ktorú je možné zadefinovať ako fotogrametrickú zobrazovaciu techniku na odhad trojrozmerných štruktúr z dvojrozmerných sekvencií. (15)

Skrátene SFM je proces rekonštrukcie 3D štruktúry zo série 2D snímok prepojených pohybom nosiča. Existujú prípady, kedy sa za SFM považuje celý proces rekonštrukcie 3D scény aj s procesmi, ktoré sme si popísali vyššie, ktorými sú detekcia bodov záujmov a porovnávanie kľúčových bodov. V skutočnosti predstavuje jeden proces z celej série. Ide o proces, ktorý sa snaží súčasne rekonštruovať štruktúru 3D scény, polohu a orientáciu kamery a často krátko aj vnútorné vlastnosti kamery. Vnútorné vlastnosti kamery sú opísané horným trojuholníkom matice o veľkosti 3×3 , taktiež známej ako matica kamery (viac o matici kamery a kalibrácii je popísané v kapitole zaoberajúcej sa kalibráciou). Výsledok SFM procesu je vytvorenie mračna bodov a zrekonštruovanie pozícií kamier. K dosiahnutiu lepšieho 3D modelu sa používajú ďalšie techniky na spracovanie získaného mračna bodov zo SFM. (14)

2.3.4. RGB-D kamera

V období nástupu senzorov RGB-D, akým je aj Microsoft Kinect, bol veľký pokrok v mapovaní prostredia a v algoritmoch SLAM (súčasná lokalizácia a mapovanie). Vzhľadom na cenu týchto senzorov a dostupnosti GPU sa stávali husté rekonštrukcie čoraz populárnejšie pri riešení problematik v robotike. (16)

RGB-D kamery zachytávajú farebné obrazy RGB, ktoré sú obohatené o informáciu hĺbky pre každý jeden pixel. Takzvané „structured light RGB-D“ kamery sú založené na stereo

technológiách, a preto zdieľajú mnoho vlastností so stereo kamerami. Základné rozdiely spočívajú v rozsahu a priestorovej hustote hĺbkových dát. Structured light RGB-D kamery osvetľujú scénu štruktúrovaným svetelným vzorom, ktorým dokážu odhadnúť hĺbku, avšak sú limitované projektorom. V týchto kamerách sa využíva infračervené svetlo pri premietaní vzoru, ktoré nie je bežné vidieť v porovnaní s klasickými kamerami alebo ľudským zrakom. Získané snímky sa ďalej spracujú, hĺbka sa vypočíta vďaka perspektívnemu skresleniu vzoru vďaka hĺbke rôznych objektov. Nevýhodou tohto typu zariadenia je limitovaná vzdialenosť kamery od objektu pri snímaní. V prípade Microsoft Kinect je to pár metrov. (17)

Existujú kamery, ktoré nevyužívajú svetelné vzory, ale čas letu (Time-of-Flight, skrátené ToF), ktorý sa využíva v LIDAR skeneroch. Rozdiel medzi LIDAR zariadením a kamerou využívajúcou ToF je, že LIDAR obsahuje mechanické komponenty na realizáciu skenovania, zatiaľ čo ToF kamery vykonávajú výpočet času letu na integrovaných obvodoch CMOS alebo CCD technológií. (18)

2.4. Existujúce riešenia

V tejto kapitole opisujeme produkty, ktoré úplne alebo čiastočne riešia rekonštrukciu scény od zariadení, ktoré snímajú prostredia, cez aplikácie spracovávajúce získané dáta zo zariadení snímajúce prostredia, až po softvérové knižnice a programy na tvorbu a editovanie 3D modelu.

2.4.1. ZED kamera

ZED kamera je zariadenie, ktoré bolo vyvinuté spoločnosťou Stereolabs, snažiac sa napodobniť ľudské videnie. Tento produkt predstavuje stereo kameru, ktorá dokáže snímať hĺbku založenú na pasívnej stereo vízii. Výstupom stereo kamery je video, ktoré sa skladá z dvoch snímok vo vysokom rozlíšení.

Táto kamera je pripojená k počítaču alebo inému výpočtovému zariadeniu pomocou USB 3.0, kde sa následne získane video spracuje na grafickej karte a výsledkom je hĺbková mapa. Na dostatočnom výkonnom počítači je spracovanie snímok v reálnom čase. Hĺbkové dáta sú vo vzdialenosti jedného až pätnástich metrov od zariadenia. Hlavné využitie tohto zariadenia sa našlo v autonómnych navigáciách pri sledovaní pohybu a vo virtuálnej/rozšírenej realite. (19)



Obrázok 11 - ZED kamera

2.4.2. RealityCapture

RealityCapture je program vyvíjaný spoločnosťou Capturing Reality, ktorý vytvára 3D modely na základe snímok z rôznych zariadení a/alebo informácií z laseru. RealityCapture je zameraný na spracovanie obrovského počtu snímok získaných pomocou fotoaparátov s vysokým rozlíšením v relatívnom rýchlom procese, ktorých počet môže presiahnuť aj niekoľko pár tisíc snímok. 3D model vytvorený z 3000 snímok zvládne program za menej ako hodinu. Doba procesu vytvárania 3D modelu závisí od počtu grafických kariet a procesorov. Nároky pre tento program nie sú vysoké. Vyžaduje si však minimálne 8GB pamäte RAM a grafickú kartu značky NVIDIA s minimálnou pamäťou 1GB podporujúca platformu CUDA. (20)

2.4.3. Kinect a KinectFusion

Pre senzor Kinect od spoločnosti Microsoft vznikol rekonštrukčný systém nazývaný KinectFusion. Tento systém využíva zariadenia Kinect verzie 1 a 2, z ktorých získané údaje sa následne spracujú do výsledného 3D modelu počas niekoľkých sekúnd. To je dosiahnuté tým, že behom získavania snímok z Kinect kamery sa neustále sleduje 6DOF zariadenie a pridávajú sa hĺbkové dáta do jedného globálneho modelu. Tento model je neustále obohacovaný ďalšími dátami, ktoré sa získavajú počas celej doby snímania scény. Dokonca aj malé pohyby, akými sú otrasy kamery, vedú k novým dátam obohacujúce výsledný model. (21)

Kinect zariadenia patria do kategórie RGB-D kamier. V prvej verzii sa používa štruktúrovaný svetelný vzor a v druhej verzii sa využíva čas letu. V tabuľke sú uvedené základné parametre týchto dvoch verzií kamier.

	Kinect verzia 1	Kinect verzia 2
Rozlíšenie farebnej kamery	640 x 480 30FPS	1920 x 1080 30FPS
Rozlíšenie hĺbkovej kamery	320 x 240	512x424
Maximálna vzdialenosť	~4.5 m	~4.5 m
Minimálna vzdialenosť	40 cm	50 cm
Horizontálny uhol záberu	57°	70°
Vertikálny uhol záberu	43°	60°
Hĺbková technológia	Structured light pattern	Time-of-Flight
Komunikácia	USB 2.0	USB 3.0

Tabuľka 1 - Parametre kamier (22)

2.4.4. MeshLab

MeshLab je voľne dostupný systém na spracovanie a editovanie 3D modelov. Poskytuje sadu nástrojov na úpravu, čistenie, vykresľovanie, renderovanie a ďalšie užitočné funkcie. Ponúka funkcie na spracovanie surových dát z 3D zariadení na digitalizáciu, poprípade na prípravu modelov pre 3D tlač. (23)

2.5. Zhodnotenie

Z dostupných informácií spomenutých v analýze dokážeme vyvodit', že na snímanie prostredia sa využívajú primárne tri typy zariadení. Najdrahším, najrýchlejším a najpresnejším zariadením na snímanie je laserové zariadenie, ktorý nám po snímaní prostredia vytvorí point cloud. Najlacnejším zariadením na ľahkú prácu sú fotoaparáty, kamery, mobily a ďalšie podobné zariadenia. Tie sú jednoduché na prácu a manipuláciu. Výsledkom snímania pomocou kamier sú snímky, ktoré obsahujú všetky potrebné informácie na získanie mračna bodov s použitím fotogrametrie. Spracovanie týchto snímok pre dosiahnutie mračna bodov je z dostupných možností na rekonštrukciu scény najnáročnejšie. Ďalším riešením je hybridné zariadenie - RGB-D kamera. Táto kamera obsahuje okrem základnej kamery aj ďalší snímač, ktorý slúži na výpočet hĺbky každého získaného pixelu.

Pre rekonštrukciu scény sme sa rozhodli vytvoriť softvér využívajúci kamery na snímanie prostredia. Toto riešenie je finančne nenáročné a dostupné pre bežného človeka. Na snímanie nám postačia mobilné telefóny s kamerou, k spracovaniu použijeme počítač alebo notebook s operačným systémom Windows, ktorý je jedným z najrozšírenejších operačných systémov na svete.

Existujúcich riešení a algoritmov na rekonštrukciu scény pomocou kamery je mnoho. Vďaka rôznym algoritmom je možné poskytnúť používateľovi voľnosť k navolení vlastných špecifikácií na dosiahnutie požadovaného výsledku. Podľa použitého spôsobu a algoritmu dokážeme vytvoriť riešenie, ktoré je schopné spracovať získané snímky z kamier pod 1 sekundu, a teda môžeme hovoriť o spracovaní v reálnom čase. Poprípade dokážeme poskytnúť používateľovi možnosť spracovať snímky zo snímania tým, že sa do programu môžu nahráť neskôr a používateľ tak nepotrebuje mať počas snímania prostredia k dispozícii počítač na spracovanie snímok.

3. Špecifikácia

Cieľom našej práce je vytvorenie aplikácie, ktorá bude spracovávať 2D snímky a vytvorí z nich trojrozmerný model. Aplikácia bude umožňovať prispôsobenie významných krokov podľa potrieb užívateľa. Vďaka prispôsobeniu bude možné dosiahnuť spracovanie v rýchlejšom čase alebo získať model s hustejším mračnom bodov. Aplikácia bude schopná spracovávať údaje v reálnom čase, čo v našom prípade predstavuje vykonanie jedného kroku alebo sady krokov pod 1 sekundu. V nasledujúcich podkapitolách sa upresnia prostriedky a komponenty, ktoré naša aplikácia využíva.

3.1. Funkčná špecifikácia

Aplikácia bude schopná prijímať snímky zo stereo kamery vytvorenej z dvoch kamier, samostatnej kamery alebo nahraním snímok z disku počítača. V prípade spracovania snímok pomocou stereo videnia bude aplikácia umožňovať kalibráciu stereo kamery. Výsledok tejto kalibrácie bude možné uložiť a opätovne použiť v nasledujúcich použitíach programu. Tieto informácie z kalibrácie slúžia na úpravu získaných snímok zo stereo kamery, ktoré používame na vytvorenie hĺbkovej mapy.

Vytvorenie hĺbkovej mapy pri stereo videní bude možné dosiahnuť v čase kratšom ako 1 sekunda. Toto časové kritérium platí aj pri príprave dát pre štruktúru z pohybu, ktorá zahŕňa nájdenie kľúčových bodov, vypočítanie descriptorov a nájdenie zhôd s ostatnými snímkami, ktoré bude možné dosiahnuť pre jednotlivé spomenuté kroky, poprípade pre celú túto sadu krokov pod 1 sekundu.

Pre dosiahnutie spracovania dát v reálnom čase je užívateľovi umožnené špecifikovať a prispôsobiť dôležité kroky v procese spracovania snímok za cieľom dosiahnutia kratšieho času spracovania dát podľa výkonu zariadenia, na ktorom bude aplikácia spustená. Pre tento účel sa používa CUDA, ktorá využíva výkon grafických kariet značky NVIDIA, ktoré umožňujú paralelné spracovanie.

3.2. Hardware

V tejto podkapitole opisujeme zariadenia, ktoré sú použité v našom projekte 3D mapovanie a modelovanie prostredia v reálnom čase a ich dôležité údaje, ktoré sú kľúčové pre splnenie cieľov nášho projektu.

3.2.1. Výpočtové prostriedky

Pre projekt 3D mapovania prostredia a jeho následné modelovanie v reálnom čase používame notebook značky Dell Inspiron 15 s procesorom Intel Core i7 7700HQ architektúry Kaby Lake so štyrmi jadrami, operačnou pamäťou 16GB, 256GB SSD diskom, grafickou kartou značky NVIDIA GeForce GTX 1060 s pamäťou 6GB a s operačným systémom Windows 10 Home 64bit.

3.2.2. Kamery

K vytvoreniu stereo videnia používame dve kamery značky Kayeton so senzorom OmniVision OV4689 model KYT-U400-01ND. Tieto kamery dosahujú pri rozlíšení FullHD(1920x1080) 60FPS, HD(1280x720) 120FPS a pri rozlíšení 640x320 dosahuje 330FPS. Komunikácia s počítačom je zabezpečená pomocou USB 2.0. Taktiež podporuje UVC, čo znamená USB video device class, inak povedané, podporuje prenos videa, ako pri bežných webových kamerách pre operačné systémy Windows, Linux, Android a Mac.

3.3. Software

Pre náš projekt sa používa vývojové prostredie Microsoft Visual Studio, verzia 2017 15.9.11 a programovací jazyk C# verzia 7.3. Pre dosiahnutie nášho cieľa používame knižnice, platformy a nástroje, ktoré sú v nasledujúcich kapitolách bližšie opísané.

3.3.1. CUDA

CUDA je platforma pre paralelné programovanie vyvíjaná spoločnosťou NVIDIA Corporation, ktorá je určená na využitie grafických procesorov, ktoré obsahujú grafické karty spoločnosti NVIDIA. Procesory, ktoré obsahujú grafické karty, sú vytvorené za účelom masívneho paralelného spracovania dát. Ich výpočtová sila plynie zo schopnosti vykonávať paralelne veľké množstvo nezávislých výpočtov, ktoré sa pri spracovaní obrazu bežne vyskytujú. Rozdiel medzi CPU a GPU je taký, že GPU sa zameriava viac na výpočet ako na radiaci obvod a prácu s pamäťou. CUDA je navrhnutá pre jednoduchú spoluprácu s jazykmi C, C++ a Fortran. Svoju efektivitu má založenú na troch princípoch, ktorými sú zdieľaná pamäť, bariérová synchronizácia a hierarchia skupín vlákien. (12)

3.3.2. Visualization Toolkit

Visualization Toolkit (VTK) je voľne dostupný softvérový systém pre 3D počítačovú grafiku, spracovanie obrazu a jeho vizualizáciu. VTK podporuje veľké množstvo

vizualizačných algoritmov vrátane skalárnych, vektorových, textúrových a mnohých ďalších. Táto nástrojová sada podporuje paralelné spracovanie a je multiplatformová, takže ju dokážeme používať na operačných systémoch Linux, Windows, Mac a Unix platformách. (24)

3.3.3. Open Source Computer Vision

Open Source Computer Vision, v skratenej verzii OpenCV, je voľne dostupná a otvorená knižnica pre počítačové videnie a strojové učenie. Je distribuovaná pod BSD licenciou pre ľahšie komerčné využitie firiem a úpravu kódu pre aplikácie počítačového videnia. Obsahuje cez 2500 optimalizovaných algoritmov pre počítačové videnie a strojové učenie. Používa sa na identifikovanie objektov, detekciu tváre, sledovanie pohybujúcich sa objektov, extrahovanie 3D modelov z objektov, a iné. Obrovská výhoda tejto knižnice je v kvalite dokumentácie. OpenCV používa mnoho firiem, akými sú Google, Microsoft, IBM, Intel, rovnako ako aj rôzne startupy. Túto knižnicu je možné použiť v jazykoch C/C++, Java, Python a MATLAB pod operačnými systémami Windows, Mac, Linux a Android. V súčasnej dobe sa aktívne vyvíjajú plnohodnotné CUDA a OpenCL rozhrania. Natívny jazyk tejto knižnice je C++. (19) (25)

3.3.4. Emgu CV

Emgu CV je cross platformový .NET wrapper pre OpenCV knižnicu spracovávaní obrázkov. Tento wrapper umožňuje používanie OpenCV funkcií z .NET kompatibilných jazykov, ktorými sú C#, VB, VC++, IronPython a ďalšie. (26)

3.3.5. JSON.NET

JSON.NET je populárny vysoko výkonný JSON framework pre .NET. Tento framework poskytuje serializáciu a deserializáciu .NET objektov do/z JSON formátu. Poskytuje LINQ, čím zjednodušuje prácu s čítaním a zapisovaním do JSON. (27)

3.3.6. VisualSFM

VisualSFM je GUI aplikácia pre 3D rekonštrukciu scény využívajúca Structure from Motion (viac informácií o SFM sa nachádza v kapitole Structure from Motion). Tento rekonštrukčný systém využíva viacjadrový paralelizmus pre detekciu kľúčových bodov a ich následné porovnávanie, ako aj pri iných procesoch počas vytvárania 3D modelu. Pre hustú rekonštrukciu scény využíva tento program ďalšie nástroje, akými sú PMVS/CMVS a ďalšie. (28)

3.4. Testovacie scenáre

V tejto kapitole opisujeme testovacie scenáre pre náš výsledný program, ktoré nám overia funkčnosť a kvalitu programu. Tieto scenáre sa rozdelia do funkčných testov, ktoré nám overia, či nám program funguje podľa predpokladov a výkonnostných testov, s ktorými zistíme správnosť, rýchlosť a kvalitu riešenia.

3.4.1. Funkčné testy

Prvým funkčným testom je overenie fungovania kamier. Ide o scenár, ktorý overuje zaznamenanie kamier, ich nastavenie a ukladanie snímok pre SFM a stereo videnie. Ako doplnok k testu je overenie nastavenia rozlíšenia kamier nastaveného pre použité kamery.

Ďalší test slúži na overenie funkčnosti kalibrácie. Ide o kontrolu zobrazenia kalibračného okna slúžiaceho na nastavenie a spustenie kalibrácie. Výsledkom kalibrácie sú hodnoty vlastností kamier, ktoré je možné uložiť a znova načítať.

V ďalšej časti overíme vytváranie objektov reprezentujúce algoritmy slúžiace pre SFM a stereo videnie. Väčšina algoritmov má možnosť nastavenia parametrov, ktoré je možné nastaviť z grafického rozhrania. Toto nastavenie sa overí pre každý algoritmus slúžiaci v rámci SFM alebo stereo videnia. V rámci testovania algoritmov sa overí funkčnosť použitia grafickej karty, ktorú niektoré z použitých algoritmov využívajú.

Posledným testom bude vytvorenie 3D modelu z procesov SFM a stereo videnia. Pri SFM sa overí tok dát od získania snímok, vygenerovania kľúčových bodov, vypočítania deskriptorov, nájdenie zhôd, spracovanie dát s VisualSFM a nakoniec zobrazenie modelu v programe. Pri stereo videní získame snímky, vygenerujeme hĺbkovú mapu a vygenerujeme mračno bodov, ktoré bude taktiež možné zobrazit'.

Tieto štyri testy nám overia základnú funkčnosť programu. Sú to pomerne komplexné testy, ktoré zahŕňajú overenie aj vedľajších komponentov nachádzajúcich sa v programe, ktoré síce neboli spomenuté v scenároch, avšak sú súčasťou pre splnenie.

3.4.2. Výkonnostné testy

Program slúži na vytvorenie 3D modelu z 2D snímok v reálnom čase, a preto sa výkonnostné testy zamerajú na tieto hodnoty. Pri SFM sa bude overovať kvalita 3D modelu a rýchlosť spracovania. K dispozícii je mnoho algoritmov slúžiacich na spracovanie snímok, z ktorých väčšinu overíme. Výkonnostné testy pre SFM budú používať sadu predpripravených dát, aby boli zabezpečené rovnaké podmienky pre rôzne kombinácie.

V prvom teste zistíme množstvo kľúčových bodov pri rôznych algoritmoch. Algoritmom nastavíme základne hodnoty, keďže každý algoritmus má iné vstupné parametre, a zistíme maximálny počet nájdených kľúčových bodov. Počas merania si zaznamenáme čas potrebný na získanie všetkých kľúčových bodov zo snímok.

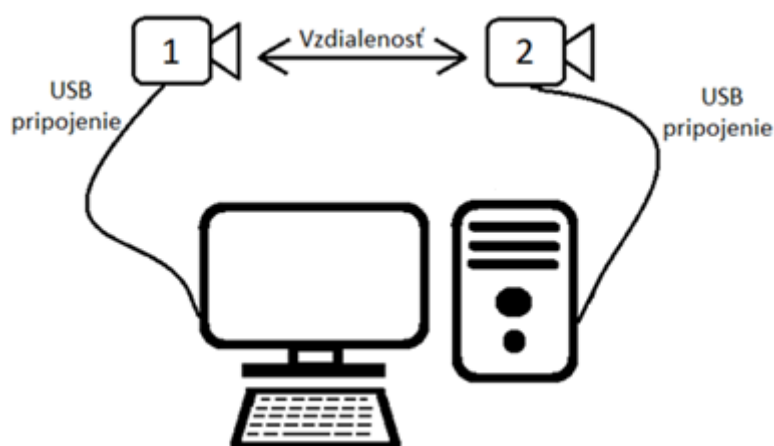
V ďalšom teste pre SFM bude zistenie času potrebného na výpočet deskriptoru pre kľúčové body z predchádzajúceho testovania. Zo získaných údajov vypočítame priemerný čas, ktorý je potrebný na získanie deskriptoru z kľúčového bodu.

Pre nájdenie zhôd sa vytvorí ďalší test, s ktorým odmeriame čas potrebný na prehľadanie všetkých dvojíc medzi dvoma deskriptormi. Posledný test bude na zistenie počtu 3D bodov vo výslednom modeli. V tomto kroku nás bude zaujímať, koľko bodov zo zistených kľúčových bodov a nájdených zhôd sa nakoniec nachádza v modeli.

Pre stereo videnie zistíme čas potrebný na vygenerovanie hĺbkovej mapy pri základných nastaveniach s použitím rôznych algoritmov a zistíme počet nájdených 3D bodov. Pre dosiahnutie lepšieho výsledku budeme prispôsobovať hodnoty v algoritmoch.

4. Návrh

Kayeton kamery, ktoré máme k dispozícii pre náš projekt, nám slúžia na vytvorenie stereo kamery. Tieto kamery majú k dispozícii kábel s USB rozhraním, ktorý sa využije pre komunikáciu. Pri riešení stereo videnia sa naše kamery umiestnia v určitej vzdialenosti od seba, čím dosiahneme priestorové videnie. USB kábel nám umožní hýbať s kamerami nezávisle od počítača, čím získame možnosť premiestňovať a otáčať kamery podľa možnosti kábla a zmapovať tým väčšiu časť prostredia. Základný návrh zapojenia komponentov je znázornený na obrázku 12. Naším cieľom je primárne vytvorenie aplikácie, ktorá spracuje vstupné dáta, ktorými sú okrem iného aj získané snímky z kamier. Významné toky, akými sú vytvorenie mračna bodov pri SFM alebo vytvorenie hĺbkovej mapy pri stereo videní sú detailnejšie popísané v jednotlivých podkapitolách.



Obrázok 12 - Zapojenie

4.1. Stereo kamera

Naša aplikácia, ktorá slúži na vytvorenie 3D modelu, využíva snímky získané snímaním prostredia v rovnakom čase z dvoch rôznych uhlov pohľadu pre vytvorenie hĺbkovej mapy pri stereo videní. Získané snímky z oboch kamier sa pri vytváraní hĺbkovej mapy porovnávajú, aby sa zistilo, ako nám vzdialenosť medzi kamerami ovplyvní pozíciu snímaného objektu v jednotlivých snímkach. Pre dosiahnutie tohto cieľa použijeme dve plne identické kamery. Dôvodom použitia takýchto kamier je, že nebudú vyžadovať toľko hardvérových a softvérových úprav, ako by to bolo pri použití neidentických kamier. Získavanie snímok je zabezpečené pripojením káblov cez USB porty, ktorých rýchlosť prenosu je dostatočná pre nastavenie 60 FPS a pre rozlíšenie 1920x1080. Alternatívnou možnosťou zapojenia pre použité kamery je ich zapojenie priamo na základnú dosku výpočtovej techniky. Pri našom návrhu

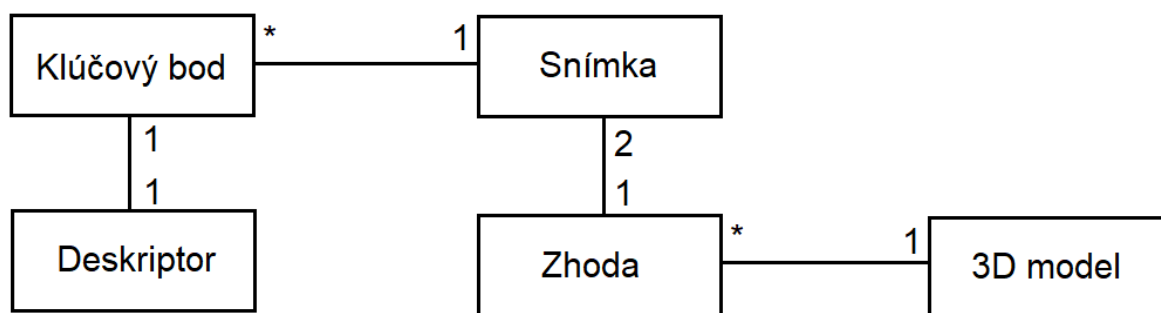
využívame USB pripojenie, ktoré nám vďaka USB káblu umožňuje umiestniť kamery v rôznych vzdialenostiach od seba. Tým získame možnosť umiestňovať kamery na rôzne miesta podľa tvaru podkladu, ktorý bude slúžiť na pripevnenie kamier.

Pri vytváraní modelu pomocou SFM, nám postačí len jedna kamera. Avšak pri použití stereo kamery máme väčšie množstvo dát, ktoré nám vytvorí hustejšie mračno bodov. V rámci SFM sa získané snímky porovnávajú medzi kamerami a predchádzajúcou dvojicou. Týmto spôsobom nevynecháme žiadnu zachytenú snímku a vytvoríme dostatočné množstvo zhôd pre vytvorenie mračna bodov.

4.2. Vstupné a výstupné dáta

Vstupnými dátami pre náš program sú primárne snímky z kamier, ktoré sú pripojené k počítaču pomocou USB. Tieto snímky z kamier sa v programe uchovávajú v maticiach. Ďalšou možnosťou vstupných dát sú obrázky. Sú to hlavne súbory s príponami JPG a JPEG. Všetky spracované snímky sa zapíšu na disk vo formáte JPEG.

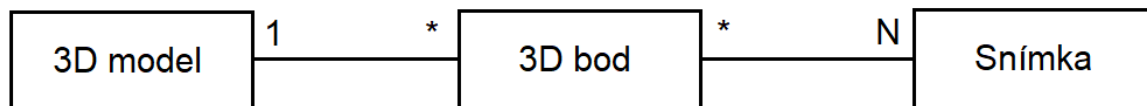
Ďalšie formáty, ktoré využíva náš program sú JSON, TXT, SIFT a NVM. JSON formát sa využíva na zápis vlastností kamier ako aj na zápis mračna bodov získaného z hĺbkovej mapy. Štruktúra JSON súborov zodpovedá k niektorému modelu v našom zdrojovom kóde. S prácou s JSON súbormi nám slúži framework JSON.NET, ktorý nám značne uľahčí prácu s JSON objektmi. SIFT formát je určený na zápis kľúčových bodov nájdených v snímkach, ktorý je potrebný pre nástroj VisualSFM. Okrem toho využívame klasický čistý text a posledný typ súboru je NVM. Výsledkom VisualSFM je súbor typu NVM, ktorý uchováva mračno bodov 3D modelu a pozíciu kamier. Tento súbor slúži ako vstupný údaj pre program, ktorý sa následne spracuje a vyobrazí 3D model.



Obrázok 13 - Dátový model pre SFMeck

Základný dátový model pre SFM je znázornený na obrázku vyššie. V SFM budeme mať 5 hlavných dátových typov. 3D model obsahuje zoznam zhôd, ktoré boli nájdené medzi snímkami. Vždy sa porovnávajú deskriptory dvoch snímok, ktoré sú vypočítané podľa

kľúčových bodov porovnávajúcich snímok. 3D model sa spracuje nástrojom VisualSFM, ktorý nám vytvorí súbor typu NVM. Na obrázku nižšie je znázornený dátový model pre spracovanie súboru programom.



Obrázok 14 - Dátový model pre stereo videnie

3D model, ktorý nám vznikne po spracovaní NVM súboru sa skladá z množiny 3D bodov. Takto vzniknutý 3D model sa následne zobrazí v programe pomocou knižnice VTK.

4.3. Nastavenie kamery

Kalibrácia a rektifikácia je pre vytvorenie hĺbkovej mapy pri stereo videní kľúčová. Algoritmy, ktoré sa používajú pre výpočet hĺbkovej mapy sa opierajú o to, že snímky zo stereo kamery sú posunuté v rámci roviny. Žiaľ, vo väčšine prípadov má kamera zakrivenie alebo iné defekty, ktoré sú nežiadúce. Aby sme ušetrili čas a nemuseli opakovať kalibráciu a rektifikáciu, si budeme môcť po úspešnom nastavení kamier uložiť výsledné údaje z týchto krokov. Tieto uložené údaje predstavujú vnútorné a vonkajšie vlastnosti kamier. Hlavný tok údajov pre nastavenie kamier je navrhnutý na obrázku 15.



Obrázok 15 - Diagram nastavenia kamier

Pre kalibráciu bude potrebné zaobstarat' vhodnú vzorku dát, ktorá bude predstavovať dvojicu snímok (aspoň 25) z ľavej a pravej kamery. Na týchto snímkach musí byť zobrazená použitá šablóna, podľa ktorej budeme kalibrovať. Šablóna predstavuje vzor, na ktorom je možné nájsť rovnobežné čiary, podľa ktorých sa dá určiť zakrivenie kamier a ohniskovú vzdialenosť.

Po úspešnom dokončení kalibrácie, ktorá nám zistí prípadné zakrivenie snímok, získame časť údajov, ktorá sa použije v rektifikácii. Tá na základe daných informácií zistí prípadné posunutie jednej kamery oproti druhej. Výsledok rektifikácie je, že nám dvojicu snímok zarovná do rovnakej výšky a podľa potrieb otočí. Týmto získame ďalšiu a poslednú časť vlastností kamier. Po získaní hodnôt z kalibrácie a rektifikácie bude možné si tieto údaje uložiť pre ďalšie použitie programu. Tento proces musíme vykonať aspoň raz pre správne zadefinovanie vnútorných a vonkajších vlastností kamier. V prípade, že boli tieto vlastnosti zadefinované nesprávne, je potrebné tento proces zopakovať s inou vzorkou dát pre dosiahnutie lepších výsledkov.

4.4. Pribeh aplikácie

Rozhodli sme sa, že aplikácia na vytvorenie 3D modelu pomocou 2D snímok bude môcť využiť dva spôsoby. Oba spôsoby vychádzajú z fotogrametrie a využívajú len snímky získané z kamier. Všetky potrebné informácie sme schopní získať zo sady snímok a nie je potrebné použiť iné zariadenie k dosiahnutiu nášho cieľa.

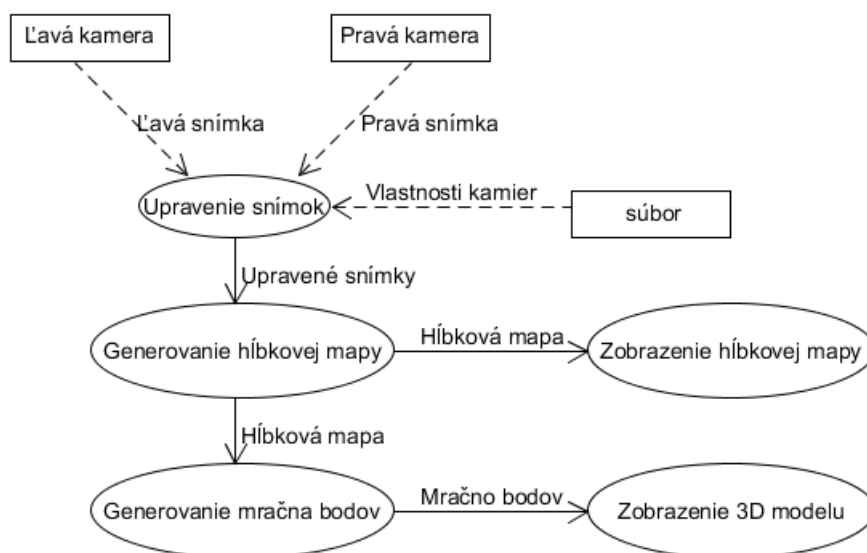
Prvým spôsobom je vytvorenie mračna bodov z páru snímok zo stereo kamery. V tomto prípade máme kamery pevne umiestnené v priestore alebo na platforme, a zmena jednej kamery sa rovnako prejaví v druhej kamere. Týmto spôsobom vieme určiť orientáciu kamier medzi sebou navzájom a prispôbiť snímky tak, aby sme prehľadávanie snímok obmedzili len na prehľadávanie v rovine (na rovnakom riadku v snímkach). Takouto úpravou snímok vieme rapídne urýchliť výpočet hĺbkovej mapy a následne pretransformovať do mračna bodov.

Druhým spôsobom je použitie metódy na vytvorenie štruktúry z pohybu. Tento spôsob nevyžaduje úpravu snímok, pretože nájdenie kľúčových bodov a vypočítanie deskriptorov nie je citlivé na prípadne defekty snímok. Nevýhoda tohto spôsobu je, že mračno bodov bude maximálne také veľké, ako je polovica počtu všetkých nájdených kľúčových bodov zo snímok. V prípade stereo videnia to môže byť pre každý pár snímok počet rovný rozlíšeniu snímok.

V nasledujúcich podkapitolách sa opíše návrh hlavného toku pre oba spôsoby riešenia v našom programe.

4.4.1. Stereo videnie

Návrh jednej z hlavných častí aplikácie je v zjednodušenej forme vyobrazený na obrázku 16. Tento návrh predstavuje hlavný tok a myšlienku našej aplikácie, ktorá vytvorí 3D model pomocou stereo videnia.



Obrázok 16 - Diagram vytvorenia 3D modelu

Na obrázku vyššie je zobrazený cyklus, ktorý by sa mal po špecifikovaní a prispôbení nášho programu vykonať v reálnom čase, poprípade aspoň niektoré časti z cyklu. Po získaní snímok z ľavej a pravej kamery cez USB pripojenie aplikujeme údaje o vnútorných a vonkajších parametroch kamier. Parametre kamier získame pomocou kalibrácie a rektifikácie. Po aplikovaní parametrov získame snímky v požadovanom stave a sú pripravené pre porovnávanie. Takto upravené snímky, ktoré zobrazujú reálne vlastnosti objektov, akými sú rovnobežné čiary bez skreslenia, používame na vygenerovanie hĺbkovej mapy. Na vygenerovanie hĺbkovej mapy sú potrebné minimálne dva snímky z rôznych uhlov pohľadu snímaného prostredia. V týchto snímkach sa hľadá rozdiel pozícií objektov z prvej snímky od druhej snímky. Rozdiel pozícií objektov predstavuje hľadanú informáciu pre vytvorenie hĺbkovej mapy a neskôr 3D modelu.

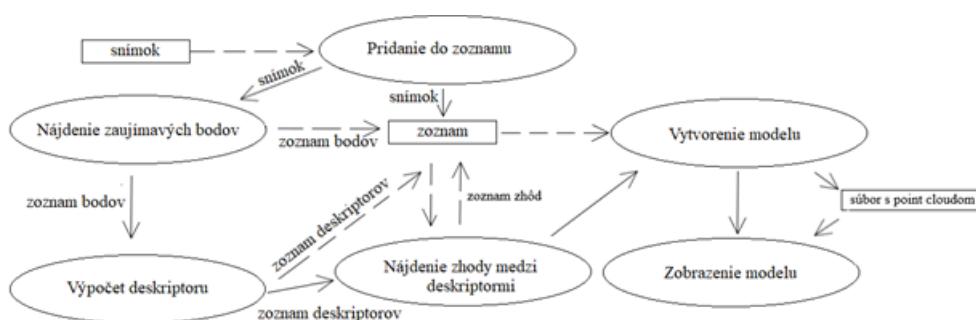
Výsledok celého generovania hĺbkovej mapy je matica s informáciami o pozícií bodu na snímke a rozdiel daného bodu oproti pozícií z druhej snímky. Pred samotným zobrazením hĺbkovej mapy sa informácia o rozdielnej pozícií pretransformuje na farbu pre lepšiu vizuálnu informáciu. Ak je kvalita hĺbkovej mapy nepostačujúca, bude to pravdepodobne kvôli zlej kalibrácii a rektifikácii. Preto je potrebné tieto kroky vykonať s lepšou vzorkou dát. Hĺbková

mapa sa taktiež používa pri vygenerovaní mračna bodov, ktoré sa používa na lepšie zobrazenie prostredia ako 3D model.

Pre dosiahnutie nášho cieľa sa pre niektoré časti programu využíva akcelerácia grafickej karty, ktorá je na výpočty niekoľkokrát rýchlejšia ako procesor. Tým je dosiahnutie výsledku v reálnom čase pravdepodobnejšie.

4.4.2. Structure from Motion

Ďalší proces nášho programu slúži na vytvorenie mračna bodov pomocou procesu Structure from Motion. Na obrázku nižšie je znázornený priebeh procesu zo snímky k vytvoreniu mračna bodov.



Obrázok 17 - Tok dát pri SFM

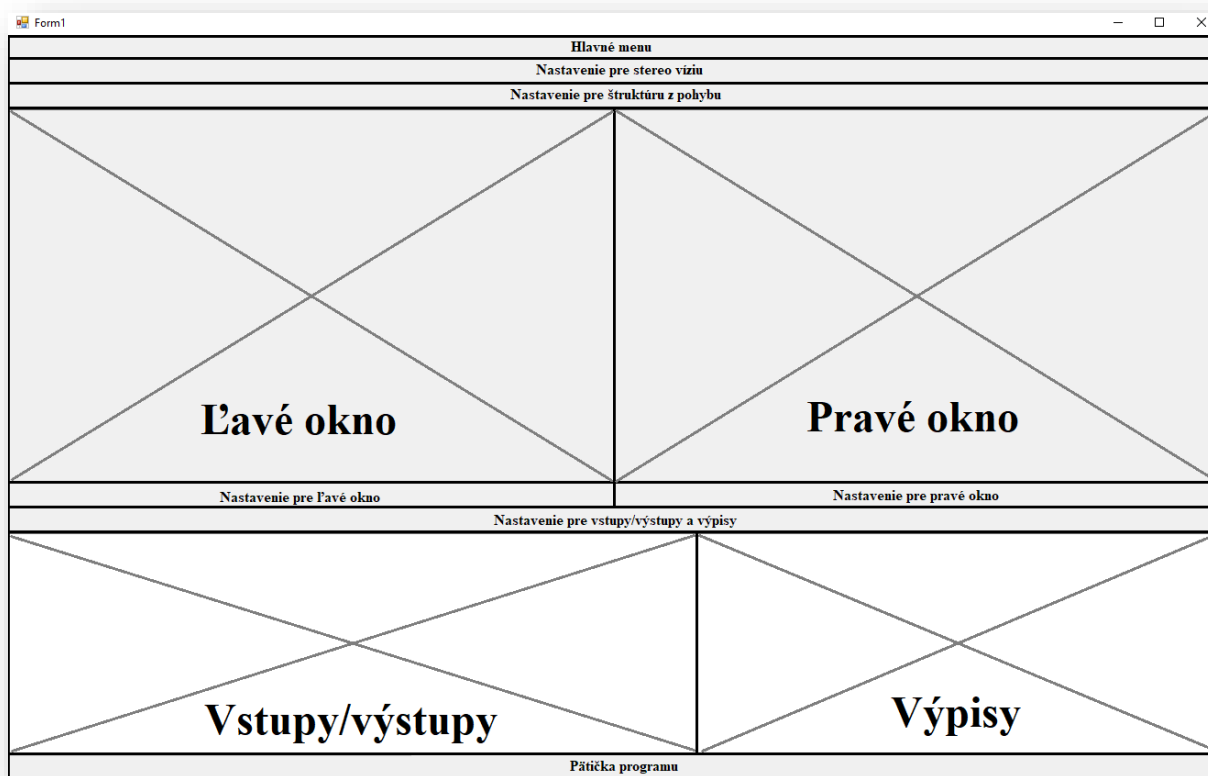
Treba podotknúť, že z každého kroku si ukladáme výsledok do globálneho zoznamu, ktorý sa v predposlednom kroku „Vytvorenie modelu“ použije. Pre každý snímok, ktorý si uložíme do globálneho zoznamu, si vypočítame kľúčové body. Tento zoznam kľúčových bodov si uchováme pre danú snímku v globálnom zozname. Následne sa pre tieto kľúčové body vypočíta deskriptor, ktorý sa použije pri nájdení zhôd v nasledujúcom kroku. Samozrejme aj v tomto kroku „Výpočet deskriptoru“ si výsledok uložíme v globálnom zozname. Podľa spôsobu hľadania zhôd bude potrebné použiť uložené deskriptory z predošlých snímok, ktoré sme už spracovali. Vďaka deskriptoru z aktuálneho snímku a z predchádzajúcich sme schopní nájsť zhody medzi snímkami. Všetky zhody si opäť uchováme v globálnom zozname. Dôvod, prečo si všetko ukladáme do jedného globálneho zoznamu je ten, že pri vytvorení modelu, je potrebné mať k dispozícii tieto uchované informácie.

Vstupnými dátami v tomto procese môžu byť jedna snímka alebo zoznam snímok. Podľa toho, čo sa zvolí, sa tok dát čiastočne upraví. V prípade, že snímky získavame z kamier, sa získavanie snímok, až po nájdenie zhody medzi deskriptormi, vloží do cyklu. Po ukončení získavania snímok z kamier, sa následne vypočíta model a v ďalšom kroku ho zobrazíme. V prípade, že aplikácií poskytneme sadu snímok, tak táto sada prejde všetkými krokmi, ako je

znázornené na obrázku. Keď poskytneme programu sadu snímok, dokážeme určiť veľkosť tejto sady a pracovať s daným číslom určujúcim veľkosť sady v paralelnom spracovaní. Paralelné spracovanie nám umožní maximalizovať výkon zariadenia, na ktorom je spustená aplikácia a dosiahnuť kratší čas pri spracovaní viacero snímok.

4.5. Zobrazenie

Na obrázku 18 je znázornené rozloženie hlavného okna nášho programu. S týmto oknom bude používateľ najviac pracovať. Pre zobrazenie našej aplikácií používame Windows Forms (WinForms), nakoľko Visual Studio poskytuje designer (grafického návrhára) a programovací jazyk tejto knižnice je C#.



Obrázok 18 - Návrh programu

Najväčšie objekty nášho hlavného okna sú dve menšie vizualizačné okná, ktoré zobrazujú potrebné informácie pre používateľa. Týmito informáciami sú získané snímky z kamier, nájdené zhody, hĺbková mapa alebo výsledné mračno bodov. Pod každým z týchto okien sa nachádza aj možnosť zobrazenia potrebnej informácií. V prípade, že užívateľ si užívateľ nestihne pozrieť výsledné snímky alebo má potrebu si pozrieť vstupy/výstupy z procesov, má možnosť si tieto dáta zobrazit' pod týmito oknami. Pre zobrazenie vstupných

alebo výstupných dát je možnosť nastaviť si typ dát na zobrazenie. Na začiatku hlavného okna sa nachádza základné menu, ktoré slúži na prvotné nastavenie aplikácie. Pod týmto menu sa nachádzajú ďalšie dve menu pre už konkrétne procesy stereo videnia a štruktúry z pohybu.

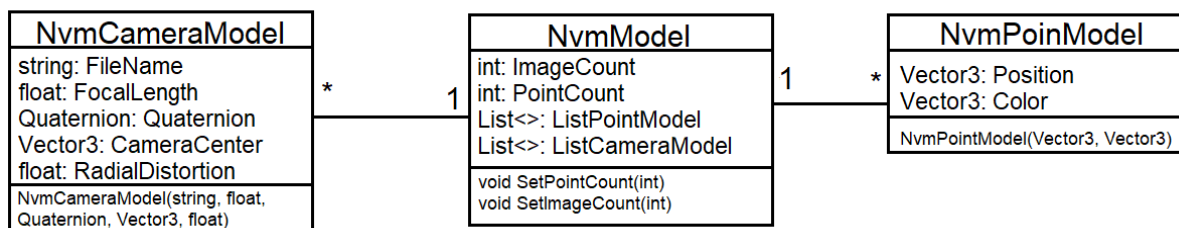
Všetky potrebné kroky sa snažíme umiestniť na hlavnom okne, aby používateľ mohol jednoducho a intuitívne použiť aplikáciu podľa svojich potrieb k vytvoreniu 3D modelu z 2D snímok. Aby používateľ mohol maximalizovať svoj úspech, má možnosť si jednotlivé kroky nastaviť podľa svojich potrieb pomocou elementov akými sú ComboBox, CheckBox a ďalšie, ktoré sú poskytnuté knižnicou WinForms.

5. Implementácia

K vytvoreniu programu na rekonštrukciu scény používame vývojové prostredie Visual Studio 2017, v ktorom sme si vytvorili 64 bitový projekt využívajúci C# jazyk verzie 7.3 a WinForms. Použili sme 64 bitovú verziu kvôli Emgu CV využívajúcu CUDA, ktorá funguje primárne v tejto verzii. Ďalej sme si v programe doinštalovali JSON.NET pre prácu s JSON objektmi, ktoré využívame v programe. K získaniu snímok slúži knižnica Emgu CV ako aj pri všetkých procesoch spracovania snímok. Verzia Emgu CV je 3.4.3.3016 a JSON.NET 12.0.1. Podrobnejší popis implementácie je rozdelený do nasledujúcich kapitol opisujúcich fungovanie procesov pri spracovaní snímok a ďalších zaujímavostí týkajúcich sa programu.

5.1. Vstupné a výstupné dáta

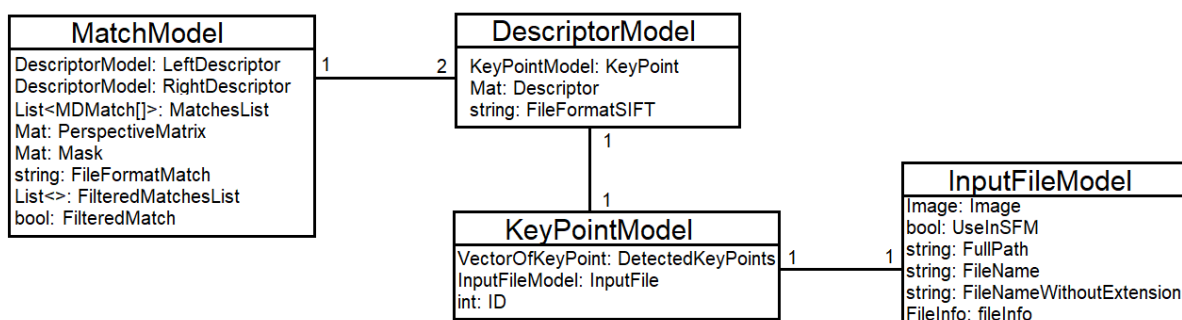
V programe sa vyskytujú dva zložitejšie dátové štruktúry, ktoré boli popísane v návrhu. Prvou štruktúrou je 3D model, ktorý sa načíta z NVM súboru z VisualSFM. Na obrázku 19 je táto štruktúra zakreslená.



Obrázok 19 - Dátový model pre 3D model

Na obrázku je vidieť, že v programe máme NvmModel, ktorý predstavuje 3D model spracovaný programom. Tento model obsahuje hodnoty o počte kamier a bodov, ako aj zoznam týchto dát. Kvôli obmedzeniam hodnôt obsahuje model funkcie, ktoré umožňujú zmeniť hodnoty v modeli. NvmPointModel obsahuje trojrozmerný vektor pre pozíciu a farbu. Posledný model v tejto štruktúre NvmCameraModel obsahuje informácie o snímkach, z ktorých máme 3D body v zozname ListPointModel v NvmModel.

K dosiahnutiu 3D modelu poskytujeme VisualSFM zoznam všetkých nájdených zhôd medzi snímkami. Štruktúru tejto zhody je znázornená na obrázku 20.



Obrázok 20 - Dátový model pre SFM

MatchModel predstavuje informácie o zhodách medzi dvoma snímkami. V tomto modeli si uchováame objekty typu DescriptorModel pre ľavú a pravú snímku, ako aj zoznam všetkých a vyfiltrovaných zhôd. Ďalšími informáciami v tomto modeli sú Mask, PerspectiveMatrix a FilteredMatch, ktoré slúžia pre vyfiltrovanie všetkých zhôd. Poslednou informáciou je text obsahujúci všetky nájdené zhody, ktorý je potrebný pre VisualSFM. DescriptorModel obsahuje objekt KeyPoint, z ktorého sme vypočítali descriptor a jeho SIFT formát sme uchovali do FileFormatSIFT. Pre každý snímok, ktorý spracujeme, vytvoríme objekt typu InputFileModel obsahujúci základné informácie o snímke. Z tohto modelu sa následne vytvorí KeyPointModel, obsahujúci zoznam kľúčových bodov a identifikátor.

Ďalšie súbory používané v programe sú identické kópie predstavujúce modely zo zdrojového kódu. Ako príklad uvediem JSON súbor predstavujúci vlastnosti kamier, ktorý je identický s modelom CalibrationModel.

5.2. Stereo kamera

Stereo kameru sme si vytvorili pripevnením dvoch kamier značky Kayeton na pevný podklad. Tento podklad ma v sebe diery o veľkosti 2 mm, čím nám poskytuje možnosť pripevnenia kamier pomocou dištančných skrutiek. Tieto skrutky nám poskytujú voľný prístup k portom na zapojenie USB káblu. Kamery sme si umiestnili čo najbližšie k sebe a zarovnali do jednej roviny. Týmto spôsobom získame veľmi podobné umiestnenie ako je u očí človeka a dosiahneme tým priestorové videnie. Treba podotknúť, že podložka na ktorej sú umiestnené kamery, je z pevného materiálu, aby sme eliminovali možnosť meniť orientáciu a pozíciu len jednej kamery. Výslednú rekonštrukciu stereo kamery je vidieť na obrázku nižšie.



Obrázok 21 - Stereo kamera

Keďže stereo kamera pozostáva z dvoch kamier, treba zabezpečiť synchronizáciu snímania prostredia, aby nedošlo k chybným párom snímok. Pre dosiahnutie najlepšej synchronizácie kamier je možné pri niektorých typoch kamier použiť hardvérovú synchronizáciu. V našom prípade túto možnosť nemáme a je potrebné ich zosynchronizovať softvérovo. Knižnica OpenCV nám poskytuje riešenie pre tento problém a to volaním dvoch metód *Grab()* a *Retrieve()*.

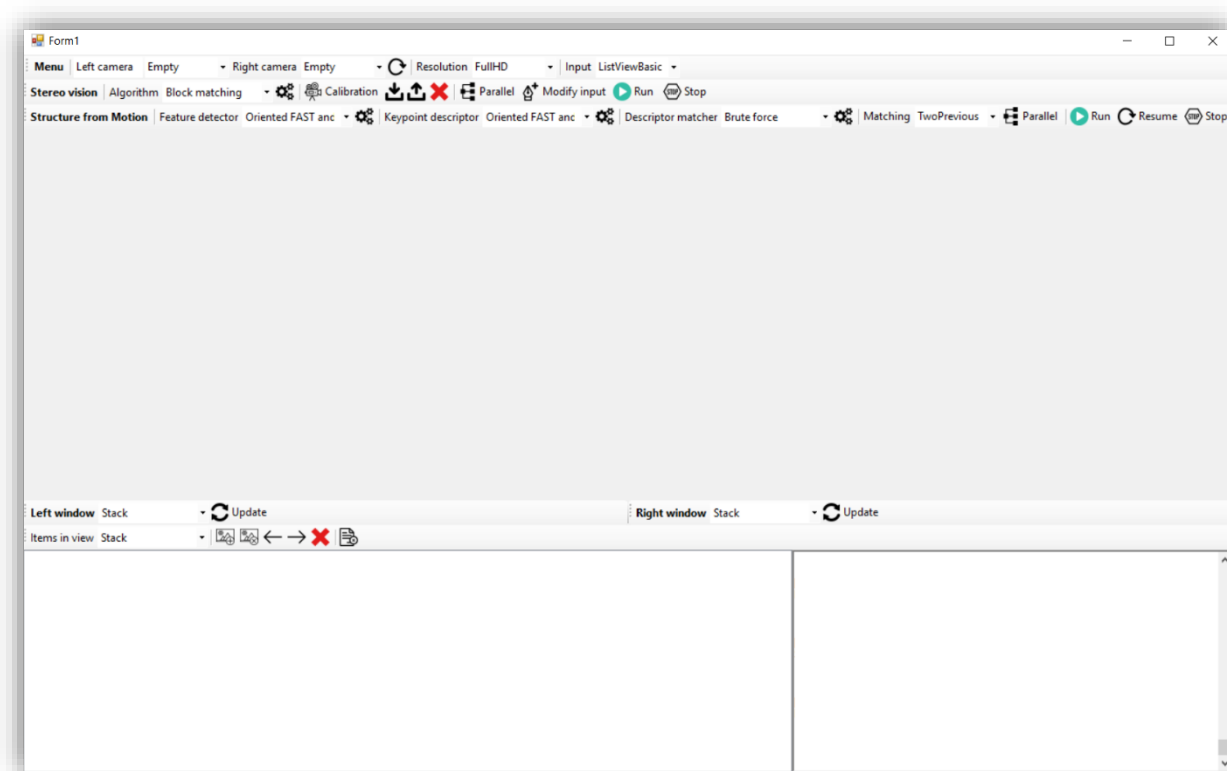
```
cap0.grab();  
cap1.grab();  
cap0.retrieve(LeftFrame);  
cap1.retrieve(RightFrame);
```

Cap0 a cap1 predstavujú kamery, ktoré tvoria stereo kameru. Metóda *Grab()* slúži práve na synchronizáciu kamier, ktoré nemožno zosynchronizovať hardvérovo. Táto funkcia je rýchlejšia ako funkcia *Retrieve()*. Celý proces získania snímok z kamier prebieha tak, že obe kamery vykonajú funkciu *Grab()* a následne sa pre obe kamery zavolá funkcia *Retrieve()*, ktorá dekoduje nasledujúci rámec.

Pred samotným získaním snímok je potrebné určiť, ktoré kamery sa majú použiť v našom programe. Zoznam z ktorého je možné určiť kamery sa získa pomocou Windows knižnice, ktorá prehľadá všetky „Plug and play“ zariadenia typu kamera. Následne podľa zvoleného zariadenia si vytvoríme objekt predstavujúci túto kameru a pomocou vyššie spomenutých metód *Grab()* a *Retrieve()* získame snímky pre ďalšie procesy.

5.3. Grafické rozhranie

Návrh hlavného okna sme premietli do implementácie, ktoré je vidieť na obrázku nižšie.



Obrázok 22 - Hlavné okno programu

V strede hlavného okna sú dva objekty, ktoré slúžia na zobrazenie snímok podľa potrieb užívateľa. Obsah týchto pohľadov je možné zmeniť pomocou elementu typu ComboBox, ktorý je pod každým pohľadom. Tlačidlo „Update“ slúži na aktualizáciu 3D modelu. Pod týmto riadkom nastavenia máme ďalší riadok slúžiaci na manipuláciu so súbormi v našom programe. Pre lepšiu prácu so súbormi máme pod týmto riadkom element typu ListView, ktorý nám zobrazuje všetky vstupujúce a vystupujúce súbory z nášho programu počas spracovania snímok. V hornej časti hlavného okna máme tri riadky slúžiace na nastavenie a prispôsobenie aplikácie podľa potrieb používateľa. Prvý riadok slúži na základné nastavenie programu. Ďalšie riadky špecifikujú konkrétny spôsob spracovania snímok. Prvý riadok je na nastavenie stereo videnia a kalibráciu kamier. Druhý riadok slúži na špecifikovanie SFM. V aplikácii používame aj ďalšie okná na nastavenie už konkrétnych zvolených algoritmov alebo dialógy pre lepšiu interakciu s používateľom.

5.4. Stereo videnie

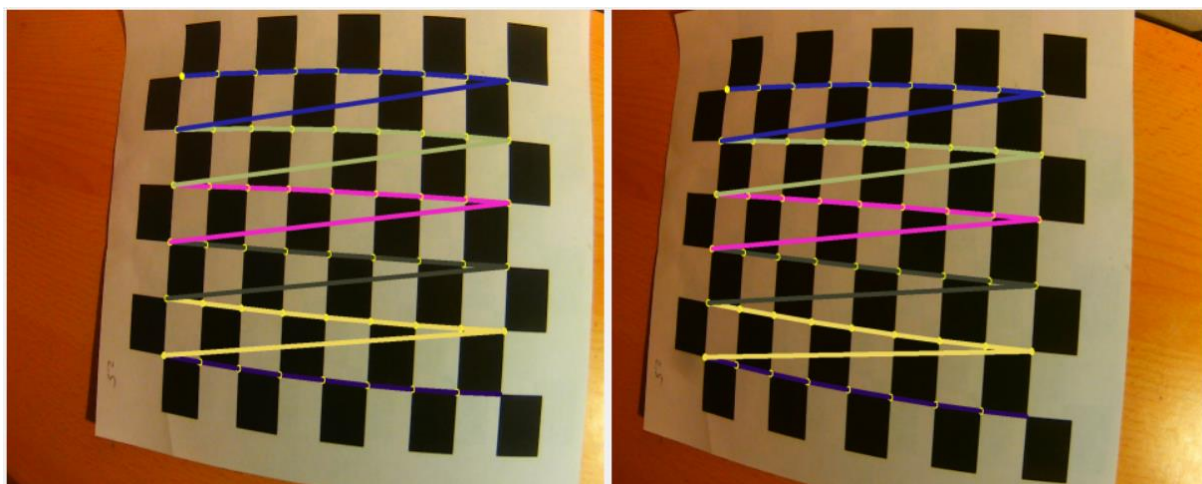
Stereo videnie vychádza zo snímok, ktoré boli získané stereo kamerou. Tieto snímky by mali zachytávať rovnaké prostredie, aby bolo možné určiť trojrozmerné body objektov. Ideálne podmienky sú, ak by sa kamery nachádzali v rovnakej výške a nemali žiadne defekty v snímkach, akými napríklad sú zakrivenie obrazu a ďalšie. K týmto podmienkam sa vieme priblížiť pomocou kalibrácie, rektifikácie a ďalšími postupmi.

V ďalšej kapitole sa zameriame na výsledok celého mapovania pomocou stereo videnia a na spôsob, akým to vieme docieľiť. Za výsledný 3D model sa už považuje matica rozdielnosti alebo vizuálna hĺbková mapa. My sme však implementovali aj zobrazenie týchto informácií ako point cloud, ktorý je možné zobrazit' v našej aplikácii.

5.4.1. Úprava snímok (kalibrácia/rektifikácia)

Pred samotným spustením kalibrácie je potrebné nastaviť, ktorá z pripojených kamier je ľavá a pravá. Ďalej je dobré nastaviť aj rozlíšenie snímok pre kamery. Nastavené rozlíšenie sa aplikuje pre obe kamery. Tieto nastavenia sa nachádzajú na prvom riadku v hlavnom okne programu.

Kalibráciu je možné spustiť z hlavného okna kliknutím na tlačidlo „Calibration“. Po kliknutí sa otvorí nové okno s dvoma pohľadmi na snímky z ľavej a pravej kamery. Tie slúžia primárne na nasmerovanie kamier, aby sme mali na snímkach použitú šablónu, podľa ktorej kalibrujeme. Ak je možné v snímkach identifikovať šablónu pre ľavú aj pravú kameru, zobrazia sa tieto snímky s nakreslenými hranami pod snímkami z kamier a uložia sa. Takáto dvojica snímok s nájdenou šablónou je znázornená na obrázku 23.



Obrázok 23 - Nájdenie šablóny v snímkach

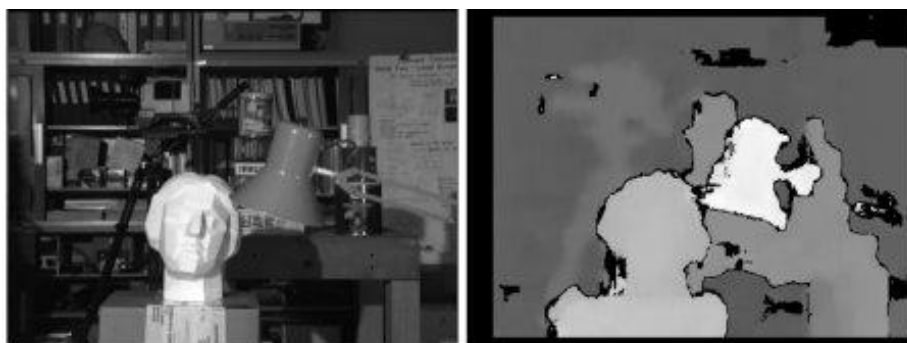
Po úspešnej kalibrácii sa následne ihneď spustí rektifikácia, ktorá nám zistí orientáciu kamier medzi sebou navzájom. Výsledkom rektifikácie sú matice, ktoré nám popisujú rotáciu, transformáciu a ďalšie údaje o kamerách. Po úspešnej kalibrácii a rektifikácii získame vnútorné a vonkajšie vlastnosti našej stereo kamery. Tieto vlastnosti sa použijú v ďalších procesoch spracovania.

Pomocou získaných informácií vieme upraviť snímky tak, aby sme celý proces počítania hĺbkovej mapy urýchlili a zlepšili presnosť. Preto je celá kalibrácia veľmi dôležitá a je potrebné ju spraviť aj niekoľko krát s inou vzorkou dát pre dosiahnutie dobrého výsledku.

V prípade, že získame z kalibrácie a rektifikácie veľmi dobré údaje, je možné si tieto informácie uložiť pre ďalšie použitie aplikácie. Všetky informácie získané z týchto procesov sú uložené v objekte typu *CalibrationModel*, z ktorého si vytvoríme JSON súbor a ten sa zapíše na disk. Na opätovné použitie kalibračných dát slúži tlačidlo v hlavnom okne, ktoré načíta z JSON súboru informácie pre objekt *CalibrationModel* v zdrojovom kóde.

5.4.2. Výsledný model

Prvý model, ktorý získame je matica rozdielnosti. Táto matica je o veľkosti rozlíšenia snímok, ktoré sme použili na výpočet. Túto maticu rozdielnosti dokážeme vizualizovať pomocou hodnoty rozdielu, ktorú obsahujú všetky body. Pre maximálnu a minimálnu hodnotu, ktorá sa v matici nachádza, si určíme farby a spektrum medzi týmito farbami sa aplikuje na hodnoty z matice rozdielnosti. Po aplikácii farieb na hodnoty získame hĺbkovú mapu. Táto mapa predstavuje obrázok, ktorý poukazuje na umiestnenie objektov v priestore pomocou farieb, ako je to vidieť na obrázku nižšie.



Obrázok 24 - Hĺbková mapa

Biela farba predstavuje objekt, prípadne objekty najbližšie ku kamerám a čím je farba tmavšia, tým sa vzdialenosť objektu od kamier zväčšuje. Čierna farba predstavuje pixely, pre ktoré nebolo možné určiť zhodný bod v druhej snímke, a preto sa tento bod označil ako neplatný.

Z matice rozdielnosti získame point cloud tým, že sa na túto maticu použijú informácie získané z kalibrácie. Týmto získame z dvojrozmerných bodov zo snímok trojrozmerné body v priestore.

Pre lepšiu manipuláciu s rôznymi algoritmi sme si v projekte vytvorili *enumerate* objekt s názvom *EStereoCorrespondenceAlgorithm*, v ktorom máme uvedené všetky spôsoby získania hĺbkovej mapy. Každý spôsob počítania má vlastnú triedu, ktorá implementuje rozhranie *IStereoSolver* a dedí z abstraktnej triedy *AbstractStereoSolver*. Toto rozhranie nám ohraničuje možnosti použitia a zjednodušuje prácu s objektmi. Pre implementovanie ďalšieho spôsobu počítania hĺbkovej mapy je potrebné vytvoriť novú triedu, ktorá dedí z abstraktnej triedy *AbstractStereoSolver* a implementuje rozhranie *IStereoSolver*. Ďalej je potrebné nový typ pridať do *EStereoCorrespondenceAlgorithm* a prepojiť s grafickým rozhraním programu.

5.5. Štruktúra z pohybu

Pre SFM využívame knižnicu Emgu CV. Tá nám poskytuje mnoho funkcií a algoritmov potrebných na vytvorenie 3D modelu zo série snímok. Aby sme celý proces urýchlili, využívame k tomu viacero vlákien, asynchrónne volania a výkon grafickej karty. Implementácia jednotlivých krokov potrebných k vytvoreniu 3D modelu sú detailnejšie popísané v podkapitolách. Pripomíname, že výsledok každého kroku si uchováme v globálnom zozname. Kvôli uchovaniu všetkých informácií je spotreba RAM pamäti priveľká, avšak získame tým čas, ktorý by bol potrebný na čítanie dát z disku.

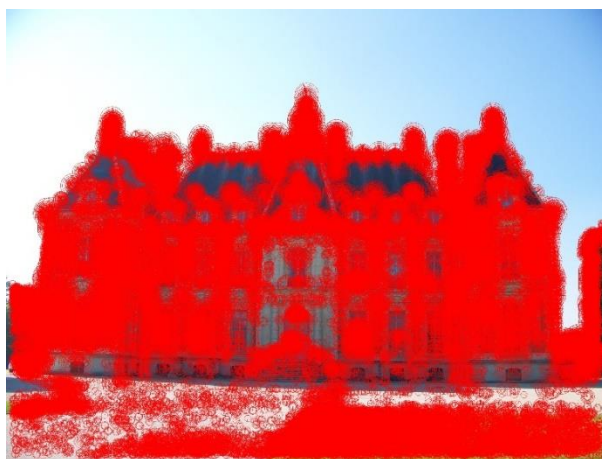
Výsledný model sa zapisuje na disk ako súbor typu NVM. Tento súbor v sebe uchováva mračno bodov a pozície kamier, z ktorých boli získané snímky spracované našou aplikáciou. V nasledujúcej kapitole bližšie opíšeme implementáciu nájdania a vypočítania deskriptoru pre kľúčové body.

5.5.1. Výpočet deskriptorov

Pred samotným výpočtom deskriptoru je potrebné nájsť kľúčové body na snímke. Na hlavnom okne máme možnosť určiť detektor, ktorý sa použije na nájdenie kľúčových bodov potrebných pre výpočet deskriptoru. Pre ľahšiu prácu sme si vytvorili objekt typu *enumerate* s názvom *EFeatureDetector*, ktorý nám poskytuje typy dostupných detektorov. Každý detektor má vlastnú triedu, ktorá implementuje rozhranie *IFeatureDetector*. Pre každú triedu treba implementovať funkciu *DetectKeyPoints()* z rozhrania *IFeatureDetector*. Táto metóda nám vráti zoznam kľúčových bodov, z ktorého sa vypočítajú deskriptory.

Podobný spôsob implementácie máme aj pre výpočet deskriptorov. Pre lepšiu manipuláciu s objektmi sme si pre výpočet deskriptorov vytvorili *enumerate EFeatureDescriptor* a rozhranie *IFeatureDescriptor*. *EFeatureDescriptor* nám poskytuje typy deskriptorov, ktoré sú použiteľné v našom programe. Rozhranie *IFeatureDescriptor* poskytuje funkciu *ComputeDescriptor()*, ktorá slúži na výpočet deskriptoru.

Pre zaujímavosť sme v našej aplikácii implementovali zobrazovanie kľúčových bodov s deskriptormi, aby bolo možné vidieť, kde a koľko týchto bodov sme dokázali identifikovať. Takýto výstup je možné vidieť na obrázku 25.



Obrázok 25 - Nájdené kľúčové body

Na obrázku sú znázornené nájdené body s červenou kružnicou. Veľkosť tejto kružnice znázorňuje hodnotu, ktorá predstavuje veľkosť oblasti použitej na výpočet deskriptoru. V rámci tejto kružnici je čiara predstavujúca orientáciu tohto bodu. Tieto informácie vychádzajú z použitého algoritmu na nájdenie kľúčových bodov. Informácie z týchto dvoch procesov sa zapisujú do konzoly, ktorá slúži pre lepší prehľad toho, čo sa deje na pozadí.

5.5.2. Nájdenie zhôd medzi deskriptormi

Tak ako v predchádzajúcich častiach, tak aj tu sa implementuje vlastné rozhranie a *enumerate* pre lepšiu prácu. K dispozícii máme viacero spôsobov porovnávania snímok, ktoré je možné zvoliť na hlavnom okne. Ide o spôsob porovnávania aktuálneho snímku s predchádzajúcimi snímkami. Naša aplikácia je schopná získať snímky zo stereo kamery, a preto sme implementovali porovnávanie aj pre stereo snímky.

Po spustení procesu pre nájdenie zhôd získame zoznam nájdených dvojíc medzi deskriptormi snímok. Aby bol výsledný model čo najpresnejší, tak sme implementovali do programu filtrovanie. Toto filtrovanie nám odstráni extrémne medzi nájdenými dvojicami. Takýto extrém sa určí po prehladaní všetkých nájdených dvojíc a uchováme si najväčšiu

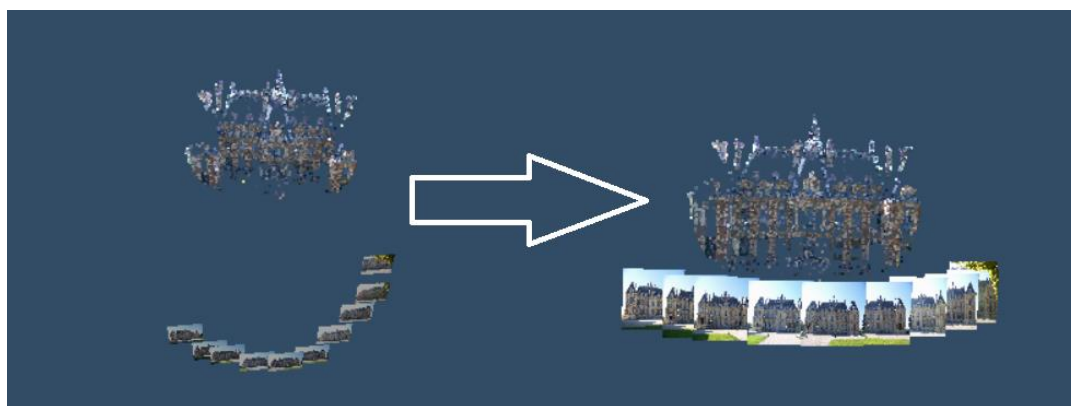
vzdialenosť medzi deskriptormi. Túto hodnotu aplikujeme do funkcie, ktorá nám určí hranicu pre akceptovanie dvojice. Ďalej sa vyfiltrované dvojice ohraničia oblasťou, ktorá by sa mala nachádzať na oboch porovnávajúcich sa snímkach. Dvojice prekračujúce ohraničenú oblasť sa odstránia. Po odstránení týchto dvojíc nám zostanú zhody, ktoré sú zobrazené na obrázku nižšie. Zelené body predstavujú nájdené kľúčové body v snímke. Červené línie predstavujú zhodu medzi kľúčovými bodmi medzi snímkami.



Obrázok 26 - Nájdené zhody medzi snímkami

5.5.3. Výsledný model

Po úspešnom spracovaní snímok máme k dispozícii všetky potrebné informácie na zostrojenie 3D modelu v našej aplikácii. Pre vygenerovanie 3D modelu používame nástroj VisualSFM. Tento nástroj sa spustí ako nový proces, ktorému sa poskytnú všetky potrebné informácie na zostrojenie modelu. Výsledok VisualSFM je súbor typu NVM, ktorý obsahuje point cloud a pozície kamier, z ktorých boli spracované snímky získané. Pre zobrazenie informácií z NVM súboru využívame knižnicu VTK, ktorá poskytuje nástroje na zobrazenie 3D modelu. Výsledný 3D model zobrazený v našom programe je na obrázku 27.



Obrázok 27 - Mračno bodov zo SFM

6. Testovanie

V prvom rade pred testovaním výkonnostných scenárov sa program otestuje funkčnými scenármi. Všetky scenáre a testy sa overujú na notebooku, ktorý bol popísaný v kapitole zaoberajúcej sa špecifikáciou. Prvým funkčným testom je overenie funkčnosti pripojených kamier. Program bol schopný rozpoznať 3 kamery. Prvou bola kamera notebooku a ďalšie dve boli kamery tvoriace stereo kameru. Nastavenie rozlíšenia kamier sa úspešne nastavili. Zachytené snímky sa zobrazujú v programe a prvý test môžeme prehlásiť za úspešný.

Ďalší test sa zaoberá kalibráciou kamier. Po nastavení základných hodnôt pre kamery sme prešli na možnosť kalibrácie. Následne sa nám otvorilo nové okno s možnosťami nastaviť parametre pre kalibráciu. Po úspešnom nasnímaní šablóny, ktorá slúži na kalibráciu, sa nám spustil proces kalibrácie. Po skončení procesu sa zobrazilo upozornenie o úspešnej kalibrácii. Následne sme otestovali uloženie a načítanie kalibračných dát.

Predposledný funkčný test overuje vytváranie a nastavenie objektov v programe. Z dostupných algoritmov je možné pre väčšinu z nich nastaviť parametre, ktoré sa úspešne aplikujú v kóde. Dôvod, prečo sa nám nezobrazilo okno pre nastavenie parametrov pri každom algoritme, je, že táto možnosť nebola umožnená v kóde.

Posledný test slúži na overenie základného toku dát. V prípade SFM sme si nastavili algoritmy využívajúce grafickú kartu ako aj procesor. V oboch prípadoch sa nám podarilo vygenerovať a zobraziť 3D model z testovacej sady, ktorá sa používa pri testovaní SFM. V prípade stereo videnia sme použili taktiež algoritmy využívajúce grafickú kartu a procesor. Aj tu sa nám podarilo vytvoriť 3D model. Tento posledný test je taktiež úspešný.

6.1. Stereo videnie

Kalibrácia kamier nám po viacerých pokusov nepriniesla vhodné dáta a preto sa testovanie zameralo len na spracovanie snímok, ktoré vedie k vypočítaniu hĺbkovej mapy.

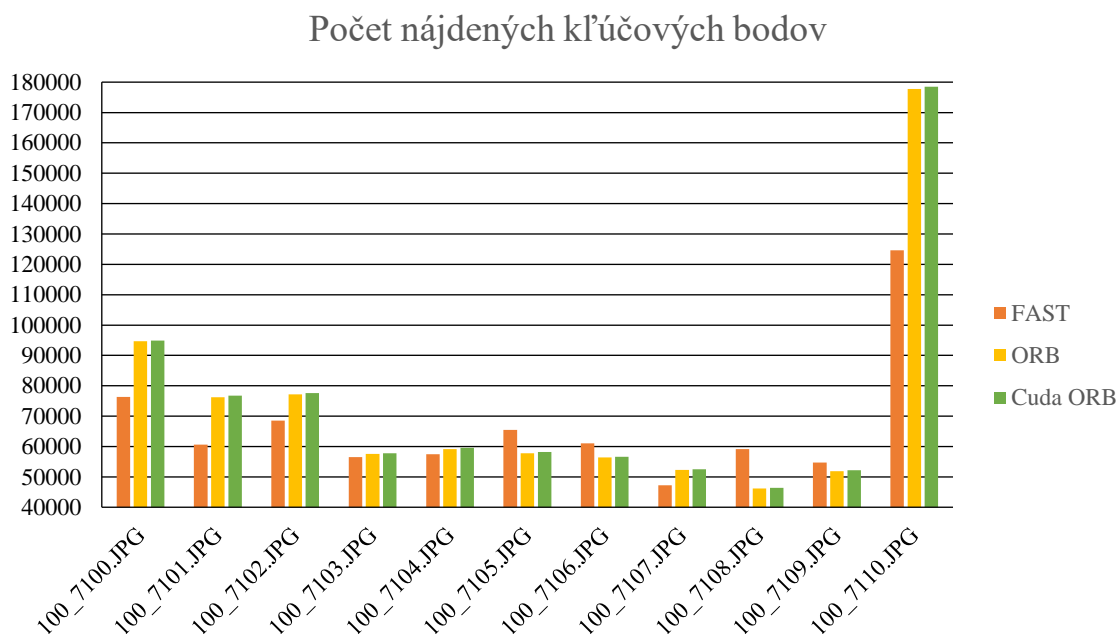
	StereoBM	CudaStereoBM	StereoSGBM	CudaConstantSpaceBP
4912x3264	263 ms	121 ms	5,71 s	2,62 s
2832x2128	91 ms	54 ms	1,69 s	99 ms
640x360	10 ms	5 ms	39 ms	49 ms

Tabuľka 2 – Priemerný čas spracovania pri stereo videní

Tabuľka obsahuje priemerné hodnoty času, ktorý bol potrebný na spracovanie dvoch snímok tvoriacich stereo snímku. Kvôli chybným dátam z kalibrácie sa nedá overiť kvalita výsledného modelu, ktorým je hĺbková mapa. Preto nie je možné overiť point cloud, ktorý sa vytvára z hĺbkovej mapy a podľa kalibračných hodnôt.

6.2. Structure from Motion

Testovacia sada sa skladá z 11 obrázkov, na ktorých sa nachádza Chateau de Sceau. Túto sadu zhotovil Pierre Moulon a používa sa ako ukážka pre knižnicu openMVG, ktorá sa zaoberá viac pohľadovou geometriou. Každý obrázok tejto sady má rozlíšenie 2832x2128, čo je približne 6 miliónov bodov.



Graf 1 - Počet kľúčových bodov

Graf číslo 1 znázorňuje počet nájdených kľúčových bodov v snímkach pomocou troch algoritmov, ktorými sú FAST, CPU verzia ORB a GPU verzia ORB. Rozdiel medzi ORB a CUDA ORB je zanedbateľný, avšak oproti FAST algoritmu tu rozdiel je. Vo väčšine snímok nám FAST algoritmus našiel menej bodov ako ostatné použité algoritmy alebo rozdiel medzi nájdenými bodmi nebol veľký. V jednom prípade však FAST algoritmus našiel o 28% viac ako v prípade ORB, avšak tento rozdiel bol ojedinelý. Celkový počet bodov pre algoritmus FAST je 731705, ORB 807125 a CUDA ORB 810884.

	FAST priemer	FAST medián	ORB priemer	ORB medián	CUDA ORB priemer	CUDA ORB medián
100_7100.JPG	150.11	148.00	467.42	312.50	222.83	180.00
100_7101.JPG	138.06	143.00	267.42	256.50	147.92	137.50
100_7102.JPG	142.00	147.00	265.75	249.50	147.54	139.50
100_7103.JPG	134.56	133.50	236.21	234.00	138.08	128.50
100_7104.JPG	130.78	127.50	231.29	214.00	131.42	129.00
100_7105.JPG	140.11	136.00	223.50	211.00	141.63	134.00
100_7106.JPG	133.39	129.00	218.58	212.50	132.75	128.50
100_7107.JPG	126.44	121.00	208.00	195.00	124.46	121.50
100_7108.JPG	130.89	125.00	210.13	201.00	126.04	121.50
100_7109.JPG	132.94	130.00	227.88	215.00	131.08	126.50
100_7110.JPG	177.89	167.00	388.58	366.00	168.50	163.50
Spolu	1 537.16	1 507.00	2 708.77	2 667.00	1 612.24	1 510.00

Tabuľka 3 - Čas nájdenia kľúčových bodov

Tabuľka vyššie zobrazuje približné hodnoty času v milisekundách potrebného na nájdenie kľúčových bodov v snímkach. Každý snímok sa pre každý algoritmus použil približne 25 krát a zo získaných hodnôt sa vypočítali priemerné hodnoty a medián pre každý snímok. Zo získaných údajov je vidieť, že najlepšie výsledky má CUDA ORB, ktorá je takmer o polovicu rýchlejšia ako CPU verzia. FAST algoritmus a CUDA ORB sú veľmi podobné, pokiaľ ide o čas, avšak ako je vidieť na grafe číslo 1, počet získaných kľúčových bodov pri CUDA ORB je väčší ako pri FAST, a preto sa dá zhodnotiť, že najlepší algoritmus pre túto testovaciu sadu na nájdenie kľúčových bodov je CUDA ORB.

	FAST detektor	FREAK deskriptor	BRIEF deskriptor	ORB deskriptor	CUDA ORB deskriptor
100_7100.JPG	76365	73390	72674	72262	-
100_7101.JPG	60663	60056	59924	59854	-
100_7102.JPG	68512	67019	66639	66434	-
100_7103.JPG	56499	54583	54131	53871	-
100_7104.JPG	57471	55655	55189	54936	-
100_7105.JPG	65467	62497	61815	61389	-
100_7106.JPG	61078	58355	57734	57365	-
100_7107.JPG	47263	45312	44888	44634	-
100_7108.JPG	59121	55868	55111	54667	-
100_7109.JPG	54691	50186	49156	48511	-
100_7110.JPG	124575	119437	118216	117481	-
Spolu	731705	702358	695477	691404	-
Rozdiel	0	29347	36228	40301	-

Tabuľka 4 - Strata kľúčových bodov pri počítaní deskriptorov

Tabuľka vyššie poukazuje na stratu kľúčových bodov, ktorým nebolo možné vypočítať deskriptor. Hodnoty v tabuľke predstavujú počet kľúčových bodov. Vo väčšine kombinácií sme stratu nenašli, len pri použití FAST algoritmu na nájdenie kľúčových bodov sa po spracovaní časť bodov stratila. V kombinácií FAST a CUDA ORB nám program zobrazoval chybovú hlášku, ktorá nás upozorňovala na zlý prístup k dátam, a preto nebolo možné otestovať túto kombináciu.

FAST detektor								
	FREAK priemer	FREAK medián	BRIEF priemer	BRIEF medián	ORB priemer	ORB medián	CUDA ORB priemer	CUDA ORB medián
100_71 00.JPG	1306.67	1386.50	401.00	397.00	263.33	256.00	-	-
100_71 01.JPG	894.50	874.00	358.00	345.00	224.67	222.50	-	-
100_71 02.JPG	921.00	787.00	370.83	356.00	245.33	242.00	-	-
100_71 03.JPG	715.33	599.00	340.50	326.00	224.17	217.00	-	-
100_71 04.JPG	794.67	689.00	319.83	316.50	225.67	220.50	-	-
100_71 05.JPG	897.50	732.50	329.83	320.50	244.83	242.50	-	-
100_71 06.JPG	865.50	718.00	323.17	317.50	219.17	217.50	-	-
100_71 07.JPG	639.50	567.00	264.67	263.00	202.83	202.00	-	-
100_71 08.JPG	740.50	605.50	324.67	334.50	223.00	224.00	-	-
100_71 09.JPG	913.00	742.50	294.00	287.50	220.67	214.50	-	-
100_71 10.JPG	2646.50	1945.00	609.00	586.00	344.33	343.50	-	-
Spolu	11334.70	9646.00	3935.50	3849.50	2638.00	2602.00	-	-

Tabuľka 5 - Čas spracovania FAST kľúčových bodov

Tabuľka 5 predstavuje spracovanie kľúčových bodov, ktoré boli nájdené pomocou FAST algoritmu. Hodnoty v tabuľke predstavujú čas v milisekundách potrebný na spracovanie kľúčových bodov. Rovnaké tabuľky sú vytvorené nižšie pre ORB a CUDA ORB detektory.

ORB detektor								
	FREAK priemer	FREAK medián	BRIEF priemer	BRIEF medián	ORB priemer	ORB medián	CUDA ORB priemer	CUDA ORB medián
100_71 00.JPG	3754.83	3792.00	944.50	937.50	341.67	341.50	251.33	215.00
100_71 01.JPG	1863.33	1874.00	803.67	796.40	307.50	307.00	149.67	143.00
100_71 02.JPG	2385.83	2407.00	787.33	782.50	297.83	309.00	151.00	145.50
100_71 03.JPG	1634.00	1653.00	659.67	662.50	270.17	263.00	142.33	138.50
100_71 04.JPG	1624.67	1633.00	663.17	665.50	266.17	272.00	143.33	137.50
100_71 05.JPG	1860.50	1861.50	655.17	647.50	259.00	266.00	148.67	148.50
100_71 06.JPG	1779.33	1780.50	637.83	645.50	262.67	274.50	139.17	137.00
100_71 07.JPG	1402.83	1400.00	599.50	601.00	249.00	253.00	135.67	132.00
100_71 08.JPG	1439.17	1436.00	577.17	577.00	238.83	246.00	140.00	134.50
100_71 09.JPG	1558.17	1555.00	559.00	549.50	259.17	256.50	142.67	140.00
100_71 10.JPG	17610.50	17811.00	1662.0	1654.0	513.67	491.00	189.83	186.00
Spolu	36913.20	37203.00	8549.00	8518.90	3265.66	3279.50	1733.66	1784.00

Tabuľka 6 - Čas spracovania ORB kľúčových bodov

CUDA ORB detektor								
	FREAK priemer	FREAK medián	BRIEF priemer	BRIEF medián	ORB priemer	ORB medián	CUDA ORB priemer	CUDA ORB medián
100_71 00.JPG	4207.33	2636.50	652.17	655.00	330.17	324.50	129.00	132.00
100_71 01.JPG	1738.50	1453.50	535.00	535.00	280.50	282.00	117.50	117.00
100_71 02.JPG	2092.00	1749.50	550.83	556.50	273.33	274.00	122.33	121.50
100_71 03.JPG	1504.67	1286.50	432.50	413.00	283.17	262.00	116.33	116.00
100_71 04.JPG	1523.50	1299.50	465.50	462.50	295.83	266.50	112.67	112.50
100_71 05.JPG	1704.83	1477.00	451.17	457.50	283.83	250.50	113.17	111.50
100_71 06.JPG	1638.33	1425.50	453.00	449.00	291.17	247.00	114.33	113.00
100_71 07.JPG	1377.50	1122.00	413.00	413.00	265.17	242.00	112.33	109.50
100_71 08.JPG	1514.00	1178.50	419.67	409.00	258.83	237.00	110.33	108.50
100_71 09.JPG	1563.17	1277.50	362.50	359.00	247.00	244.00	109.17	106.00
100_71 10.JPG	12654.50	9985.50	1047.00	1038.00	446.33	443.50	152.33	150.00
Spolu	31518.33	24891.50	5782.33	5747.50	3255.33	3073.00	1309.49	1297.50

Tabuľka 7 - Čas spracovania CUDA ORB kľúčových bodov

Zo získaných dát sa nám potvrdila rýchlosť spracovania dát pomocou grafickej karty, ktorá je neprekonateľná z dostupných algoritmov. V kombinácii CUDA ORB algoritmu pre detektor a deskriptor dokážeme získať a spracovať 810884 bodov z 11 snímok za približne 3 sekundy. Druhým najrýchlejším algoritmom na spracovanie je CPU verzia ORB a najpomalším je FREAK. Dôvodom, prečo je FREAK algoritmus pomalší, môže byť okrem

iných okolností aj fakt, že výsledný deskriptor pre daný bod obsahuje 64 elementový vektor. Ostatné algoritmy vytvárajú 32 elementový vektor.

Ďalšia tabuľka predstavuje čas potrebný na nájdenie zhôd medzi deskriptormi. V programe máme možnosť porovnávať snímky pomocou CPU a GPU verzie brute force (ďalej ako BF) algoritmu. Kvôli obmedzeniam grafickej karty sme museli počet deskriptorov znížiť na 30000.

Detektor	Deskriptor	BF	BF	CUDA BF	CUDA BF
		priemer	medián	priemer	medián
FAST	FREAK	9677.4	9464.5	622.3	511.5
FAST	BRIEF	5443.6	5283.5	590.2	479
FAST ORB	ORB	6611.5	6543	621.6	503
FAST	CUDA ORB	-	-	-	-
ORB	FREAK	9566.7	9535.5	650.4	536.5
ORB	BRIEF	5501.3	5288.5	632.6	511.5
ORB	ORB	5066.5	5055	879.6	495
ORB	CUDA ORB	6217.9	5890.5	506.8	497
CUDA ORB	FREAK	9421.5	9310.5	665.4	549.5
CUDA ORB	BRIEF	5626.2	5486.5	590.9	483
CUDA ORB	ORB	5629	5393.5	598.1	483.5
CUDA ORB	CUDA ORB	5743.7	5697.5	495.8	479.5

Tabuľka 8 - Čas párovania deskriptorov

Hodnoty v tabuľke 8 predstavujú čas v milisekundách na nájdenie zhôd medzi dvoma deskriptormi, ktorých počet každého je 30000 kľúčových bodov. Podľa získaných hodnôt by priemerný čas na spracovanie jednej snímky a nájdenie zhôd s jednou ďalšou snímku pri použití CUDA ORB a CUDA BF pri 30000 kľúčových bodov malo trvať približne 761 milisekúnd. Ďalšou tabuľkou overíme túto teóriu, kde zhodnotíme spracovanie kombinácii CUDA ORB s CUDA BF pri rôznych spôsoboch spracovania snímok. Spracovanie „1 pár“ predstavuje porovnanie aktuálneho snímku s predchádzajúcou snímku, „2 páry“ predstavuje porovnanie s dvoma predchádzajúcimi snímkami a „AllWithAll“ je porovnanie každého snímku s každým. Hodnota v zátvorkách predstavuje celkový počet porovnaní, ktoré bolo pri určenom spôsobe dosiahnuté.

CUDA BF						
	1 pár sekvenčne (10)	1 pár paralelne (10)	2 páry sekvenčne (18)	2 páry paralelne (18)	AllWithAll sekvenčne (55)	AllWithAll paralelne (55)
FAST	71	38	78	44	118	71
FREAK						
FAST	46	25	55	33	90	58
BRIEF						
FAST	51	26	60	32	238	56
ORB						
FAST	-	-	-	-	-	-
CUDA ORB						
ORB	106	49	115	57	161	174
FREAK						
ORB	29	71	81	41	171	65
BRIEF						
ORB	29	71	81	41	171	65
ORB						
ORB	57	29	67	35	335	64
CUDA ORB						
CUDA ORB	56	30	63	35	105	63
FREAK						
CUDA ORB	52	29	60	36	99	63
BRIEF						
CUDA ORB	57	29	67	35	335	64
ORB						
CUDA ORB	47	28	56	34	107	60
CUDA ORB						

Tabuľka 9 - Čas spracovania snímok

Hodnoty v tabuľke vyššie predstavujú čas v sekundách potrebný na spracovanie snímok. Kvôli zápisu na disk, filtrovaniu a uchovávaní dát sa nám nepodarilo spracovať snímky s rozlíšením 2832x2128 z použitej sady pod 1 sekundu, avšak sme sa veľmi priblížili.

Nasledujúca tabuľka predstavuje čas v sekundách potrebný na spracovanie a vytvorenie výsledného 3D modelu so sčítaním a zápisom súborov na disk.

CUDA BF						
	1 pár sekvenčne (10)	1 pár paralelne (10)	2 páry sekvenčne (18)	2 páry paralelne (18)	AllWithAll sekvenčne (55)	AllWithAll paralelne (55)
FAST	117	99	130	114	169	140
FREAK						
FAST	97	93	118	106	146	132
BRIEF						
FAST	110	90	117	93	351	135
ORB						
FAST	-	-	-	-	-	-
CUDA ORB						
ORB	166	110	174	125	241	242
FREAK						
ORB	150	100	160	110	248	144
BRIEF						
ORB	109	86	119	99	166	129
ORB						
ORB	118	92	136	105	404	132
CUDA ORB						
CUDA ORB	203	111	151	131	193	148
FREAK						
CUDA ORB	110	99	118	110	156	142
BRIEF						
CUDA ORB	102	94	121	103	149	130
ORB						
CUDA ORB	97	90	110	104	160	132
CUDA ORB						

Tabuľka 10 - Čas spracovania a vytvorenia 3D modelu

Posledná tabuľka v tejto kapitole predstavuje hodnoty výsledného 3D modelu pri použitých algoritmoch. Hodnoty v tabuľke predstavujú počty rozpoznaných kamier a spoločných bodov, ktoré sa nachádzajú vo výslednom mračne bodov.

CUDA BF						
	1 pár počet snímok	1 pár počet bodov	2 páry počet snímok	2 páry počet bodov	AllWithAll počet snímok	AllWithAll počet bodov
FAST						
FREAK	10	15293	10	17047	10	16605
FAST						
BRIEF	10	3650	11	28142	11	30235
FAST						
ORB	11	23603	10	24042	11	29034
FAST						
CUDA ORB	-	-	-	-	-	-
ORB						
FREAK	10	21649	11	12069	11	20955
ORB						
BRIEF	10	15682	11	17957	10	17850
ORB						
ORB	10	12102	10	15942	10	17125
ORB						
CUDA ORB	11	13804	10	14476	9	17239
CUDA ORB						
FREAK	11	20591	11	21710	11	25310
CUDA ORB						
BRIEF	10	13562	11	15849	10	18429
CUDA ORB						
ORB	10	13708	10	13943	10	16995
CUDA ORB						
CUDA ORB	9	10703	10	13040	10	16892

Tabuľka 11 - Hodnoty výsledného 3D modelu

Najzaujímavejšia hodnota z tabuľky vyššie je hodnota pri použití CUDA ORB pre detektor a deskriptor. Síce je tento algoritmus najrýchlejší, vo výslednom modeli má skoro najmenej bodov v porovnaní s ostatnými. Ďalou zaujímavosťou je algoritmus FREAK. Ten bol jeden z najpomalších pri získaní deskriptorov, avšak vo výslednom modeli je viac bodov v porovnaní s ostatnými.

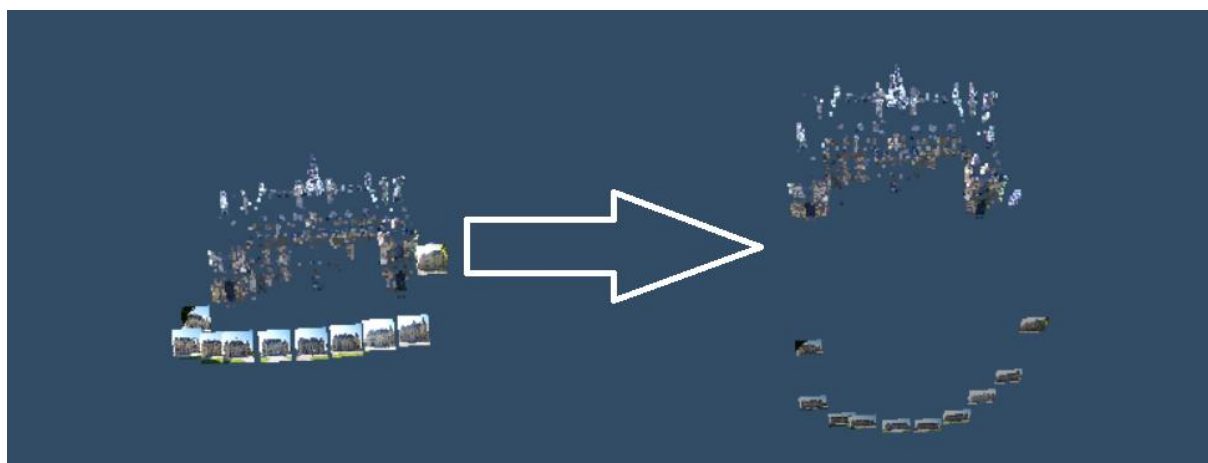
6.3. Bonus

V tabuľke nižšie sú zobrazené hodnoty času, počas ktorého sa nám podarilo spracovať snímky. Čas 11 sekúnd predstavuje spracovanie jednej snímky pod 1 sekundu z použitej sady, v ktorej sa nachádza práve 11 snímok s rozlíšením 2832x2128. Použitá kombinácia je CUDA ORB s CUDA BF, ktoré sú najrýchlejšie. Hodnoty v tabuľke predstavujú čas spracovania snímok pri určení hornej hranice nájdených kľúčových bodov. Hodnota v zátvorkách predstavuje celkový počet párov v použitom spôsobe.

	1 pár (10)	2 páry (18)	AllWithAll (55)
20000 bodov	9.96 s	10.78 s	23.26 s
15000 bodov	7.36 s	9.18 s	17.44 s
10000 bodov	5.48 s	5.76 s	10.83 s
5000 bodov	3.76 s	4.35 s	6.69 s

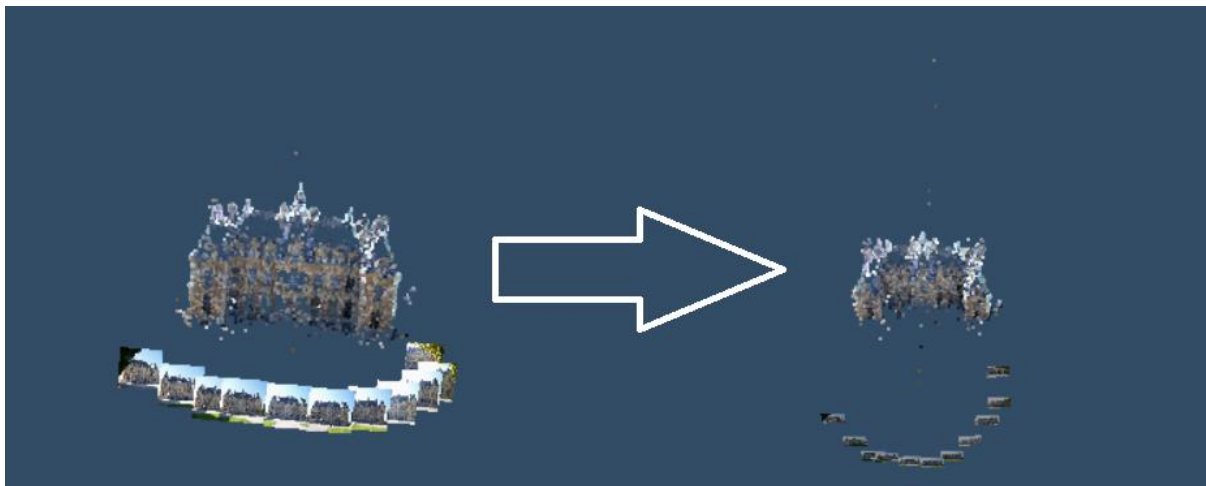
Tabuľka 12 - Čas spracovania pri ohraničení kľúčových bodov

Na obrázku nižšie je výsledný model pri použití 2 párov, pričom hranica kľúčových bodov bola 20000. Síce sme dosiahli spracovanie jednej snímky pod sekundu, avšak na úkor kvality 3D modelu.



Obrázok 28 - Point cloud po ohraničení kľúčových bodov

Pre dôkaz, že program dokáže spracovať snímky pod 1 sekundu, sme zmenili rozlíšenie snímok na štvrtinu. Z rozlíšenia 2832x2128 nám vzniklo rozlíšenie 708x532. Kombináciou algoritmov je znova CUDA ORB a CUDA BF s maximálnou hranicou kľúčových bodov 30 000 a párovanie snímok každá s každou. Spracovanie snímok sme dosiahli za 10 sekúnd a výsledný model obsahuje 11 kamier a 12 371 bodov. Tento model obsahuje menej bodov ako pri testovaní s vyšším rozlíšením a má viac chybných bodov. Model z tohto testovania je na obrázku nižšie, kde sú vidieť aj chybné body.



Obrázok 29 - Model pri znížení rozlíšenia snímok

6.4. Zhodnotenie

Pri testovaní SFM, kde sme použili snímky s rozlíšením 2832x2128 a kombináciu CUDA ORB s CUDA BF, ktoré sú najrýchlejšie z dostupných algoritmov sa nám takmer podarilo dosiahnuť čas spracovania snímok pri porovnávaní každej snímky s každou pod 1 sekundu. Tento čas sa nám podarilo dosiahnuť pri znížení počtu kľúčových bodov a výsledný model obsahoval len malú čas strechy kaštieľa.

Program dokáže spracovať snímku pod 1 sekundu po znížení hodnôt v použitých algoritmoch pri snímke s vysokým rozlíšením. V prípade, že snímky majú nízke rozlíšenie je výsledný model chudobnejší, avšak spracovanie je niekoľko násobne rýchlejšie. Testovaním sme dokázali, že program dokáže spracovať snímky pre vytvorenie 3D modelu v reálnom čase pri použití kombinácii vhodných hodnôt pre algoritmy a rozlíšenia snímky.

Pri stereo videní program spracuje snímky rýchlejšie ako pri SFM, ale výsledok veľmi závisí od kvality kalibrácie kamier. Pri chybnjej kalibrácii je hĺbková mapa a zároveň mračno bodov zo stereo snímky nepoužiteľné.

7. Záver

Práca sa zaoberá aktuálnymi spôsobmi snímania prostredia a vytvorenia 3D modelu zo získaných údajov. Pre riešenie 3D mapovania a modelovania prostredia v reálnom čase sme sa rozhodli použiť kamery pre snímanie prostredia, pretože kamery sú ľahko dostupné a práca s nimi je jednoduchá. Pri použití kamier pre snímanie prostredia existujú dva spôsoby spracovania snímok pre vytvorenie 3D modelu. Vytvorili sme preto program, ktorý umožňuje spracovať snímky oboma spôsobmi. Prvým spôsobom je využitie stereo videnia, pri ktorom je potrebné použiť stereo kameru. V našom prípade sme si stereo kameru vytvorili z dvoch samostatných kamier, ktoré boli umiestnené vedľa seba. Ďalší spôsob je vytvorenie štruktúry z pohybu, ktoré vyžaduje nájdenie kľúčových bodov v snímkach, vypočítanie deskriptorov a nájdenie zhôd medzi snímkami.

Testovanie aplikácie ukázalo, že naše riešenie dokáže spracovať snímky v reálnom čase po prispôbení procesov programu. Žiaľ, rýchlosť aplikácie je nepriamo úmerná s kvalitou výsledného modelu. Čím je spracovanie snímok detailnejšie, a tým pádom pomalšie, tým je kvalita výsledného modelu vyššia.

Ďalším možným zlepšením programu je vytvorenie vlastných algoritmov pre SFM, ktoré by využívali výkon grafickej karty alebo implementácie vlastného bundle adjustment algoritmu slúžiaceho na vytvorenie výsledného 3D modelu. Pri použití vlastného bundle adjustment algoritmu sa môže následne implementovať do programu animácia tvorenia 3D modelu, čím by sa dosiahla lepšia interakcia medzi používateľom a programom. Pre dosiahnutie rýchleho spracovania snímok sa v programe uchováva mnoho informácii, a preto je vhodné v prípadných rozšíreniach programu optimalizovať uchovávanie dát alebo znížiť počet informácii potrebných v procesoch.

Bibliografia

1. Theoharis, T., a iní. *Graphics and visualization: principles & algorithms*. Boca Raton : CRC Press, 2008.
2. Badler, Norman I. a Glassner, Andrew S. *Introduction to Computer Graphics Course Notes*. s. l. : SIGGRAPH, 1997.
3. Hess, H. *3D Computer Graphics*. Mainz : PediaPress, 2006.
4. Žára, J., a iní. *Moderní počítačová grafika*. Brno : Computer Press, 2004.
5. Ružický, Eugen a Ferko, Andrej. *Počítačová grafika a spracovanie obrazu*. Bratislava : SAMOSATO, 2012.
6. Karaffová, Markéta. *Efficient Ray Tracing of CSG Models*. Prague : s.n., 2016.
7. <https://en.wikipedia.org>. [Online] [Dátum: 28. 11 2018.] https://en.wikipedia.org/wiki/Polygon_mesh.
8. Terävä, Tapio. *Workflows for Creating 3D Game Characters*. 2017.
9. Strapek, Martin. Modelování výroby za pomoci metody 3D Laserscanningu. [Online] [Dátum: 5. 12 2017.] <http://hdl.handle.net/11025/25211>.
10. [Online] [Dátum: 8. 12 2018.] <https://www.sanborn.com/mobile-lidar/#bwg24/254>.
11. Guidi, Gabriele a Remondino, Fabio. *3D Modelling from Real Data*. s.l. : InTech, 2012. 978-953-51-0012-6.
12. Lhoťan, Miroslav. *Vytváření hloubkové mapy pro 3D zobrazení*. Praha : s.n., 2016.
13. OpenCV. [Online] [Dátum: 8. 4 2018.] <https://docs.opencv.org/3.4.1/index.html>.
14. Carrvick, Jonathan L., Smith, Mark W. a Quincey, Duncan J. *Structure from Motion in the Geosciences*. s.l. : John Wiley & Sons, 2016.
15. ŠIMÍČEK, Bc. Martin. *IMPLEMENTACE METODY STRUCTURE FROM*. Olomouc : s.n., 2014.
16. Whelan, Thomas, a iní. *Robust Real-Time Visual Odometry for Dense RGB-D Mapping*. 2013.

17. *RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments*. Henry, Peter, a iní. 5, s.l. : The International Journal of Robotics Research, 2012, Zv. 31.
18. *State of the Art on 3D Reconstruction with RGB-D Camera*. Zollhöfer, Michael, a iní. s.l. : Computer Graphics Forum, 2018, Zv. 37, s. 625-652.
19. ZATLOUKAL, TOMÁŠ. *OVLÁDÁNÍ MULTIMEDIÁLNÍHO PŘEHRÁVAČE GESTY*. Brno : s.n., 2016.
20. Reality, Capturing. Product. [Online] [Datum: 10. 1 2018.] <https://www.capturingreality.com/Product>.
21. Wasenmüller, Oliver a Stricker, Didier. *Comparison of kinect v1 and v2 depth images in terms of accuracy and precision*. Cham : Springer, 2016.
22. <http://zugara.com/>. [Online] [Datum: 3. 1 2019.] <http://zugara.com/how-does-the-kinect-2-compare-to-the-kinect-1>.
23. <http://www.meshlab.net/>. [Online] [Datum: 7. 1 2019.] <http://www.meshlab.net/#description>.
24. Overview. VTK. [Online] Kitware. [Datum: 16. 4 2018.] <https://www.vtk.org/overview/>.
25. OpenCV. About. *opencv*. [Online] OpenCV. [Datum: 16. 4 2018.] <https://opencv.org/about.html>.
26. [www.emgu.com](http://www.emgu.com/wiki/index.php/Main_Page). [Online] Emgu, 14. 12 2018. [Datum: 8. 4 2019.] http://www.emgu.com/wiki/index.php/Main_Page.
27. [www.newtonsoft.com](http://www.newtonsoft.com/json/help/html/Introduction.htm). [Online] Newtonsoft. [Datum: 8. 4 2019.] <http://www.newtonsoft.com/json/help/html/Introduction.htm>.
28. <http://ccwu.me/vsfm/>. <http://ccwu.me/>. [Online] [Datum: 5. 1 2019.] <http://ccwu.me/vsfm/>.
29. Rossignac, Jarek R. a Requicha, Aristides A. G. Solid modeling and beyond. *IEEE Computer Graphics and Applications*. September 1992, s. 31 - 44.

30. Constructive solid geometry. [Online] [Dátum: 25. 10 2017.]
https://en.wikipedia.org/wiki/Constructive_solid_geometry#/media/File:Csg_tree.png.
31. DINGES, JONA. Low Poly - Animals. [Online] [Dátum: 20. 10 2017.]
<http://jonadinges.com/low-poly-animals>.
32. Čmarada, Michal. TROJROZMEROVÉ SKENOVACIE SYSTÉMY. [Online]
[Dátum: 29. 10 2017.] <http://www.engineering.sk/clanky2/stroje-a-technologie/912-trojrozmerove-skenovacie-systemy>.
33. www.sparkfun.com. *SparkFun*. [Online] [Dátum: 1. 5 2018.]
https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU9250REV1.0.pdf.
34. www.sparkfun.com. *SparkFun*. [Online] [Dátum: 1. 5 2018.]
https://learn.sparkfun.com/tutorials/9dof-razor-imu-m0-hookup-guide?_ga=2.147474295.1921173044.1525699859-143248281.1525350944.
35. *Whole-body modelling of people from multiview images to populate virtual worlds*. Hilton, Adrian, a iní. 7, 2000, The Visual Computer, Zv. 16, s. 411-436.
36. *3D reality modelling: Photo-realistic 3D models of real world scenes*. Sequeira, Vítor a Goncalves, J. G. M. s.l. : IEEE, 2002, s. 776-783.
37. Micheletti, Natan, Lane, Stuart N. a Chandler, Jim H. *Structure from motion (SFM) photogrammetry*. s.l. : British Society for Geomorphology, 2015. 2047-0371.
38. *3D STRUCTURE FROM MOTION WITH FOURIERDESCRIPTOR TRANSFORMATIO*. Elhady, Gamal F. 5, s.l. : World Scientific, 2013, International Journal of Pattern Recognition and Artificial Intelligence, Zv. 27.
39. <https://docs.microsoft.com/>. [Online] Microsoft, 31. 5 2018. [Dátum: 8. 4 2019.]
<https://docs.microsoft.com/en-us/windows/desktop/directshow/introduction-to-directshow>.

Príloha A – Plán práce 2017/2018







Zimný semester	Popis
1. týždeň	Zoznámenie sa so zadáním
2. týždeň	Rešerš vhodnej literatúry
3. týždeň	Štúdium literatúry
4. týždeň	Nájdienie kamier vhodných pre prácu
5. týždeň	Prieskum elektronických článkov a databáz
6. týždeň	Rozhodovanie o štruktúre práce
7. týždeň	Príprava formálnej úpravy práce
8. týždeň	Hľadanie vhodných techník na mapovanie
9. týždeň	Spísanie techník na mapovanie v analýze práce
10. týždeň	Rešerš špecifických informácií týkajúcich sa stereo vízie
11. týždeň	Hľadanie vhodnej inerciálnej meracej jednotky
12. týždeň	Zoznámenie sa s kamerami
Letný semester	Popis
1. týždeň	Vytvorenie projektu
2. týždeň	Inštalácia potrebných knižníc a programov
3. týždeň	Zapojenie kamier
4. týždeň	Kalibrácia a rektifikácia kamier
5. týždeň	Spísanie doterajších údajov do práce
6. týždeň	Formálna úprava práce
7. týždeň	Vytvorenie hĺbkovej mapy
8. týždeň	Spísanie možností vytvorenia hĺbkovej mapy do práce
9. týždeň	Rozšírenie kapitol v práci o nové informácie
10. týždeň	Vytvorenie mračna bodov
11. týždeň	Testovanie
12. týždeň	Dopisovanie a finalizácia posledných kapitol
13. týždeň	Formálne úpravy, kontrola a odovzdanie práce

Príloha B - Plán práce 2018/2019

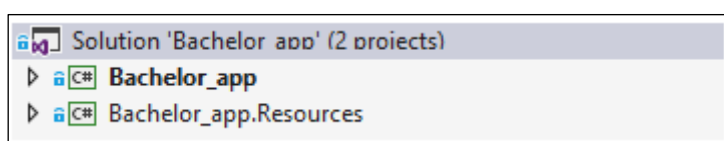
Mesiac	Popis
September	Návrh plánu v plánovacej aplikácii Trello.
Október	Hľadanie ďalších zdrojov.
November	Rozšírenie analýzy.
December	Vytvorenie projektu a implementovanie nástrojov.
Január	Vytvorenie základnej štruktúry programu.
Február	Implementovanie SFM a stereo videnie.
Marec	Úprava a vylepšenie kódu.
Apríl	Doplnenie návrhu, implementácie, testovanie a ďalších kapitol v práci.
Máj	Finalizácia programu a práce

Príloha C – Technická dokumentácia

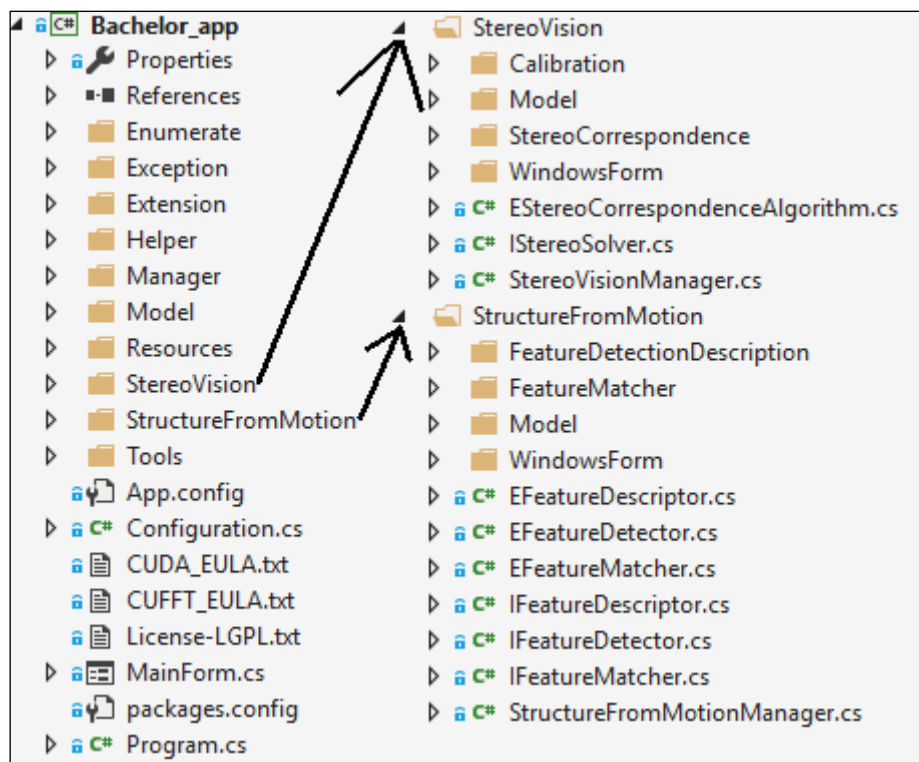
Príloha C obsahuje dokumentáciu k zdrojovému kódu riešenia ako aj k jeho štruktúre. V tejto prílohe sú spomenuté najdôležitejšie časti kódu, a preto treba podotknúť, že táto príloha neobsahuje celý kód ale iba jeho časť.

	Activiz.NET.x64 by Kitware, Inc. 64-bit .NET interface to the Visualization Toolkit (VTK)	v5.8.0
	EMGU.CUFFT by Emgu Corporation This package provides the native CUDA FFT dependencies for EMGU software libraries	v9.1.0
	EMGU.CV-CUDA by Emgu Corporation Emgu CV is a cross platform .Net wrapper to the OpenCV image processing library.	v4.0.1.3373
	EMGU.CV-CUDA.DEPENDENCY by Emgu Corporation This package provides the native CUDA dependencies for EMGU.CV-CUDA	v9.1.0
	Newtonsoft.Json by James Newton-King Json.NET is a popular high-performance JSON framework for .NET	v12.0.2
	ZedGraph by ZedGraph Project ZedGraph is a class library, user control, and web control for .net, written in C#, for drawing 2D Line, Bar, and Pie Charts. It features full, detailed customization capabilities, but most options have defaults for ease of use.	v5.1.7

Na obrázku vyššie sú znázornené použité balíčky v programe aj s číslom použitej verzie. Pre správnu funkčnosť programu sú tieto balíčky dôležité a odporúčame použiť rovnakú verziu ako je na obrázku znázornená.



Riešenie programu je rozdelené do dvoch projektov, kde prvý projekt Bachelor_app predstavuje program s riešením a druhý projekt Bachelor_app.Resources, ktorý slúži pre jazykové mutácie, ktoré by mohli byť v budúcnosti potrebné.



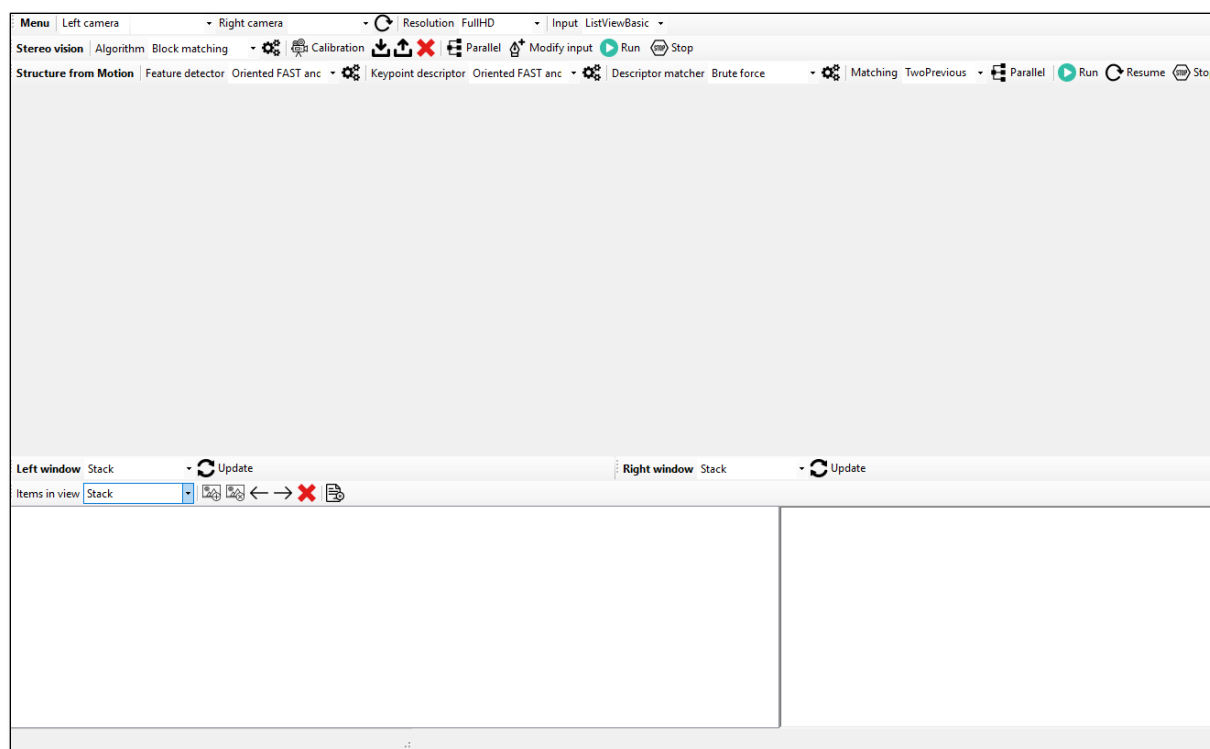
Prvá úroveň programu je znázornená na obrázku vyššie. V tejto úrovni sme si vytvorili priečinky, ktoré súvisia priamo s hlavným oknom programu a jadrom. V jednotlivých priečinkoch sme vytvorili podľa potrieb ďalej. V priečinku StructureFromMotion sme vytvorili všetky potrebné súbory potrebné pre vytvorenie 3D modelu pomocou SFM. Rovnako sme riešili aj StereoCorrespondence, v ktorej sa nachádza aj kalibrácia kamier.

Zdrojový kód obsahuje komentáre, ktoré by mali byť dostatočne popísané pre pochopenie programu. Preto sme sa rozhodli v tejto kapitole opísať len štruktúru projektu a pre detailnejšiu dokumentáciu odporúčame vygenerovať z projektu dokument pomocou na to určených nástrojov.

Príloha D – Inštalačná príručka

Pre fungovanie nášho programu alebo pri manipulácii kódu je potrebné mať program na úložisku, v ktorom je možné vykonávať zápis súborov a nainštalovať si ovládač pre grafickú kartu NVIDIA a CUDA Toolkit, kvôli implementácii riešenia využívajúce akceleráciu grafickej karty. Používaná verzia je CUDA 10, ktorú je možné získať z hlavnej stránky NVIDIA. Programovací jazyk je C# verzie 7.3, preto je dobre zvoliť si vývojové prostredie, ktoré túto verziu podporuje. V našom prípade to je Visual Studio 2017. Pre správnu funkčnosť programu je potrebné nainštalovať doplnky, ktoré boli spomenuté v prílohe obsahujúcej technickú dokumentáciu. Operačný systém určený pre spustenie programu je Windows. Použitá verzia bola Windows 10 x64.

Príloha E - Používateľská príručka



Po spustení programu je potrebné v prvom riadku okna nastaviť kamery, ktoré sa majú v programe použiť na získanie snímok. V prípade, že kamery boli zapojené po spustení programu alebo sa zaznamenali neskôr a nie sú k dispozícii v zozname, je možné tento zoznam obnoviť. Kamery v dnešnej dobe podporujú viacero rozlíšení, preto je dobré pred spustením procesu na spracovanie snímok nastaviť rozlíšenie. V prípade, že k programu poskytneme sadu snímok, je potrebné tento typ vstupu taktiež nastaviť na konci prvého riadku.

Druhý riadok slúži na nastavenie spracovania pomocou stereo videnia. Na začiatku je možnosť si zvoliť algoritmus na spracovanie a pre potreby bližšej špecifikácii hodnôt slúži tlačidlo vedľa výberu na editáciu týchto hodnôt. Ďalej sa tu nachádza kalibrácia, vedľa ktorej sú možnosti na uloženie, načítanie a odstránenie hodnôt z kalibrácie. Celý proces spracovania je možné spustiť paralelne a v prípade, že máme hodnoty z kalibrácie je možné zapnúť úpravu snímok. Na konci máme tlačidlá na spustenie celého procesu spracovania a v prípade získania dostatočného počtu snímok alebo v prípade núdze zastavenie procesu.

Ďalší riadok je určený pre SFM. Rovnako ako pri stereo videní je na začiatku možnosť nastavenia algoritmu pre detektor, deskriptor a matcher. Pre zvolený algoritmus je možné nastaviť hodnoty, ak to daný algoritmus umožňuje. Ďalej je možné nastaviť typ hľadania zhôd

medzi deskriptormi a či má celý proces spracovania prebiehať paralelne. Na konci riadku máme tlačidlá na spustenie, obnovenie a zastavenie procesu spracovania.

Pod týmito riadkami umožňujúce nastavenie programu máme dva elementy slúžiace na zobrazenie informácií z programu. Pod týmito oknami je možnosť nastavenia typu informácií, ktoré sa budú zobrazovať v okne. V prípade zobrazenia mračna bodov slúži na obnovenie modelu tlačidlo vedľa nastavenia typu informácií. Je to z toho dôvodu, že neustále obnovovanie modelu je časovo a pamäťovo náročné.

Na konci máme element na zobrazenie vstupných a výstupných dát z programu, vedľa ktorého je konzola zobrazujúca informácie z prebiehajúcich procesoch. Nad týmito dvoma elementami je riadok slúžiaci na nastavenie typu informácií, ktoré sa zobrazia v elemente pod ním. Vedľa možnosti nastavenia typu informácií sú tlačidlá na manipuláciu so súbormi v programe. Primárne sú to tlačidlá na pridanie a odstránenie vstupných súborov pre procesy spracovania.

Príloha F – Elektronické médium

Súbory, ktoré obsahuje priložené DVD:

- Debug.rar - obsahuje debug verziu programu
- Release.rar - obsahuje release verziu programu
- Zdrojový kód.rar - obsahuje celý kód programu
- Testovacie dáta pre SFM.rar - obsahuje snímky, ktoré boli použité pri testovaní
- BP_ŠimonGašpar.pdf - tento dokument