VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV RADIOELEKTRONIKY
DEPARTMENT OF RADIO ELECTRONICS

ROTAČNÍ 3D SCANNER
ROTATIONAL 3D SCANNER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Ľubomír Švehla
AUTHOR

VEDOUCÍ PRÁCE — Ing. Ondřej Kaller
SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Elektronika a sdělovací technika**
Ústav radioelektroniky

Student: Ľubomír Švehla ***ID:*** 164422
Ročník: 3 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Rotační 3D scanner

POKYNY PRO VYPRACOVÁNÍ:

V úvodní teoretické části projektu se seznamte s principy profilometrie pomocí laserové stopy (in-line laser profilometry). Ve vhodném prostředí (Matlab, v opodstatněném případě jiné) navrhněte systém pro snímání povrchu kompaktního 3D objektu. Výstupem realizovaného systému bude rentovaný 3D model, tedy poloha bodů v 3D prostoru tvořících síť konečných elementů povrchu objektu.

Optimalizujte realizovaný systém z hlediska přesnosti měření pro použitý hardware. Vytvořte uživatelské rozhraní pro řízení a nastavení skenování.

DOPORUČENÁ LITERATURA:

[1] SCHREER. O., KAUFF P., SIKORA T. 3D Videocomunication: Algorithms, concepts and real-time systems in human centred communication, 1/E. Chichester, England: J. Wiley, 2005.

[2] JAVIDI, B., OKAMO, F., SON, J. Three-Dimensional Imaging, Visualiyation and Display, New York: Springer, 2009.

Termín zadání: 8.2.2016 ***Termín odevzdání:*** 26.5.2016

Vedoucí práce: Ing. Ondřej Kaller
Konzultant bakalářské práce:

doc. Ing. Tomáš Kratochvíl, Ph.D., *předseda oborové rady*

Abstrakt

Projekt sa zaoberá návrhom konštrukciou rotačného 3D skeneru a to ako po hardvérovej tak i po softvérovej stránke. Pri hardvérovej časti je dôležitým faktorom čo najnižšia cena a tomu zodpovedá aj jeho jednoduchosť. Softvérová stránka je vyriešená pomocou nástroju FFmpeg a programovacieho prostredia MATLAB. Z nastrihaného synchronizovaného videa je vysegmentovaná ožiarená krivka a následným poskladaním týchto kriviek sa získa 3D model. Pre kvalitnejší výsledok sú využité dve kamery. Program je plne ovládateľný pomocou GUI.

Klíčová slova

3D skener, renderpatch, renderwire, FFmpeg, MATLAB, externá synchronizácia, segmentácia, renderovanie, GUI

Abstract

The point of this project is to design and construct a rotational 3D scanner. It is both about the hardware and software part. Low construction price is important factor in the hardware part and therefore the hardware part is really simple. Software part is solved by the FFmpeg tool and programming environment MATLAB. From the selected number of frames from the synchronized video the irradiated part is segmented. Consequently by compositing these traces a 3D model is gained. To gain a better result, the program uses two cameras. Program can be fully controlled from GUI.

Keywords

3D scanner, renderpatch, renderwire, FFmpeg, MATLAB, external synchronization, rendering, GUI

ŠVEHLA, Ľ. *Rotační 3D scanner*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2016. 18 s., 0 s. příloh. Bakalářská práce. Vedoucí práce: Ing. Ondřej Kaller.

Prohlášení

Prohlašuji, že svoji bakalářskou práci na téma Rotační 3D scanner jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ....

(podpis autora)

Poděkování

Chcel by som sa poďakovať vedúcemu svojej bakalárskej práce Ing. Ondřejovi Kallerovi za odborné vedenie, trpezlivosť a cenné rady pri spracovaní projektu.

Obsah

Zoznam obrázkov **viii**

Zoznam tabuliek **x**

Úvod **1**

1 TEORETICKÝ ÚVOD **2**

1.1 Pohľad na iné 3D skenovacie systémy a využiteľné technológie 2

1.1.1 Systém Autoscan 2

1.1.2 TK Scanner 3

1.2 Funkcie Renderwire a Renderpatch 3

1.3 Nástroj FFmpeg 4

1.3.1 Inštalácia FFmpeg 6

1.4 Segmentácia 7

1.4.1 Segmentácia založená na intenzite 7

1.4.2 Segmentácia založená na oblastiach 8

1.5 3D Renderovanie 9

1.5.1 Renderovacie metódy založené na geometrii 11

1.5.2 Renderovacie metódy založené na obraze 11

2 REALIZÁCIA SKENERU **13**

2.1 Koncepcia rotačného skeneru 13

2.1.1 Synchronizácia rotačnej plošiny 14

2.2 Realizácia skenovania 16

2.2.1 Segmentácia požadovaných dát z obrázku 17

2.2.2 Tvorba 3D modelu a renderovanie obrázku 18

3 APLIKÁCIA DRUHEJ KAMERY, KOREKCIA OKLÚZÍ A SKRESLENIA **20**

3.1 Korekcia skreslenia 20

3.1.1 Hľadanie osi kamery 20

3.1.2 Výpočet uhlov medzi kamerami a laserom 21

3.2 Aplikácia druhej Kamery 24

3.2.1 Stotožnenie obrazu kamier 24
3.2.2 Dodatočná kalibrácia v horizontálnej rovine zobrazenia 25
4 GRAFICKÉ PROSTREDIE 28
4.1 Ovládanie GUI 28
4.1.1 Nastavenie kalibrácí 30
4.2 Princíp fungovania GUI 31
4.2.1 Vzájomná komunikácia prvkov GUI 31
4.2.2 Inicializácia dát 31
4.2.3 Výber Obrázku pre vykresľovanie 32
4.2.4 Nastavenie osi a výškovej kalibrácie 33
4.2.5 Výber vykresľovania osí 33
4.2.6 Spustenie 3D skenovania a renderovanie 33
5 ZÁVER 35
LITERATURA 36
ZOZNAM SKRATIEK 37

Zoznam obrázkov

Obr. 1.1 Skenovanie systémom Autoscan 2

Obr. 1.2 3D mapa metra vytvorená skenovaním robotom TK scanner 3

Obr. 1.3 Príklad vytvorenia matíc .vertices a .faces, potrebných vstupných dát pre funkcie renderwire a renderpatch 4

Obr. 1.4 Proces prekódovania cez ffmpeg [1]. 5

Obr. 1.5 Príklad použitia indexov. Do výstupného vúboru sa nakopíruje tretí dátový prúd z prvého vstupného súboru a siedmy dátový prúd z druhého vstupného súboru [1]. 6

Obr. 1.6 Priradenie cesty „;c:\ffmpeg\bin“ do systému 6

Obr. 1.7 Vykreslený histogram jasovej zložky Y z modelu YCbCr [3]. 8

Obr. 1.8 Vľavo je obrázok ktorý sa segmentuje, v strede obrázok segmentovaný globálne, vpravo je obrázok segmentovaný lokálne [3]. 8

Obr. 1.9 Príklad metódy quadtree. [3] 9

Obr. 1.10 V ľavej časti je zobrazený guľový senzor C, ktorý sníma lúč z bodu M pod uhlami ϕ, θ. Rovina obrazu skutočného senzoru je tangentou ku guli, na ktorej sa bod M premietne ako m. Vpravo je zobrazených 8 plenoptických vzoriek v klasickej mriežke, ktoré snímajú bod M. [5]... 11

Obr. 2.1 Koncepcia rotačného skenera: 1. skenované teleso, 2. stojan s kamerou, 3. rotačná plošina, 4. snímač čiarového kódu, 5. stojan s laserom a šošovkou 13

Obr. 2.2 Snímač čiarového kódu prilepeného na rotačnj plošine. 14

Obr. 2.3 Príklad čiarového kódu a výstupný nezderivovaný signál získaný jeho čítaním [2]. 14

Obr. 2.4 Príklad tvaru audio stopy pre vybrané snímky videa. Z prava: 1. v tomto mieste sa čiarový kód skladá len z krátkych elementov, priebeh sa teda periodicky opakuje, 2. v tomto mieste sa nachádza jeden z širokých elementov 15

Obr. 2.5 Zjednodušený vývojový diagram..... 16

Obr. 2.6 Sprava: 1. pôvodný nasnímaný obrázok v sústave RGB, 2. obrázok prevedený do sústavy YCrCb, 3. obrázok po segmentácií, 4. obrázok po aplikovaní funkcie edge. Posledný obrázok vyzerá v niektorých miestach nespojito z dôvodu, že matlab ho nebol schopný vykresliť v tak veľkom rozlíšení, v skutočnosti je však spojitý tak ako predchádzajúce obrázky. 18

Obr. 2.7 Vyrenderované modely, vľavo pohľad zľava na sieťový model vytvorený cez renderwire, vpravo pohľad spredu na polygónový model vytvorený

cez renderpatch. 19

Obr. 3.1 Kalibračný snímok pre softvérové určovanie osi. Pri určovaní osi sa riadi pomocou ťažidla zaveseného na drôte, ktorý smeruje kolmo na stred rotačnej plošiny. 20

Obr. 3.2 Proces detegovania kľúčových snímkov. 1 – rotačná plošina s kalibračným valcom, 2 – pravá kamera, 3 – laser, 4 – ľavá kamera, α – uhol medzi pravou kamerou a laserom, t23 – čas medzi nasnímaním LED v principal pointe pravej kamery a nasnímaním jej pretnutia s laserom. β - uhol medzi ľavou kamerou a laserom, t34 – čas medzi nasnímaním pretnutia LED s laserom a jej nasnímaním v principal pointe ľavej kamery. 22

Obr. 3.3 Kľúčové polohy LED pri zisťovaní vzájomnej polohy kamier a laseru. 1 – LED je v principal pointe pravej kamery, 2 a 3 – LED sa na pravej, respektíve ľavej kamere pretína s laserom (poloha laseru), 4 – LED je v principal pointe ľavej kamery 23

Obr. 3.4 Vyrenderovaný model z pravej (červená farba) a ľavej (tyrkysová farba) kamery po stotožnení obrazu kamier. 25

Obr. 3.5 Vyrenderované modely ľavej a pravej kamery po dodatočnej kalibrácí v horizontálnej rovine zobrazenia. 26

Obr. 4.1 Okno GUI po spustení programu. 1 – Panel pre výber súborov, 2 – Panel pre výber pohľadu, 3 – Panel pre výber Kamery a talčidlá pre nastavovanie kalibrácí, 4 – panel pre výber renderovania. 28

Obr. 4.2 Priebeh načítavania dát. Pre prvé 3 možnosti boli využité minule použité súbory, pre kalibráciu osi ľavej kamery bola zvolená možnosť vybrať nový súbor. Na zobrazovacej ploche je posledný načítaný snímok z posledného načítaného prvku, výškovej kalibrácie. 29

Obr. 4.3 Printscreen zobrazovacej časti okna v ktorom prebieha nastavovanie výškovej kalibrácie. V programe je zaškrtnutý checkbutton show axis. Zvislá tyrkysová čiara je nastavená os, vodorovné zelené čiary na okrajoch obrázka sú defaultne nastavené hodnoty vyýškovej kalibrácie. Šedý kríž ukazuje aktuálnu polohu kurzora myši. 30

Zoznam tabuliek

Úvod

3D tlač a skenovanie sa stáva stále rozšírenejšou a dostupnejšou technológiou pre verejnosť. 3D skenery sú zariadenia, ktoré slúžia na prenos skutočných tvarov do virtuálnych 3D modelov. Skener pomocou viacerých technológii sníma tvar a súčasne aj povrch skúmaného tvaru. Existuje veľké množstvo 3D skenerov fungujúcich na rozličných princípoch určených pre odlišné účely, keďže svoje uplatnenie si našli 3D skenery vo veľkom množstve odvetví. Napríklad v zdravotníctve sa CT využíva už od 70. rokov 20. storočia. Existujú skenery vhodné na skenovanie menších modelov v kratšej vzdialenosti, ale aj skenery, ktoré skenujú veľké objekty či budovy. Rozlišuje sa taktiež medzi ručnými skenermi a plne automatizovanými systémami.

Tento koncept má ako hlavný cieľ byť nízko nákladový a jednoducho zostrojiteľný. Taktiež by vo svojej finálnej podobe mal byť plne automatizovaný a ovládateľný pomocou grafického prostredia z PC. Táto bakalárska práca je rozdelená na 4 hlavné kapitoly. Prvá sa zaoberá teóriou, zvyšné 3 praktickou realizáciou problematiky.

V teoretickej časti sú popísané varianty, ktorými bolo možné jednotlivé kroky pri spracovaní skenovaných dát robiť a taktiež je tu detailne popísaný princíp externých funkcií, využitých v praktickej časti tejto bakalárskej práce.

Druhá časť sa zaoberá realizáciou rotačného 3D skeneru a to ako po hardvérovej tak i po softvérovej stránke. Zahŕňa konštrukciu, získanie video materiálu a jeho transport, synchronizáciu, získanie potrebných dát z obrazu, ich vyhodnotenie a vytvorenie modelu v PC. Pri praktickej časti je využívaný program MATLAB a nástroj FFmpeg.

Tretia časť sa zaoberá aplikáciou druhej kamery. Jej hlavným účelom je potlačenie oklúzií, ktoré vznikajú pri natáčaní jednou kamerou. Táto kapitola takisto rozoberá okrem redukcie oklúzií aj deformácie spôsobené nesprávnym zarovnaním na os otáčania.

Štvrtá časť práce detailne popisuje grafické prvky GUI a jeho ovládanie. Venuje sa tak popisu ovládania pre koncového užívateľa, ako aj popisu fungovania jednotlivých častí.

1 TEORETICKÝ ÚVOD

1.1 Pohľad na iné 3D skenovacie systémy a využiteľné technológie

3D skenovanie je technológia, ktorá sa začala používať ešte v 20. storočí. Dnes získava uplatnenie stále vo viacerých smeroch, pričom väčšina skenovacích metód je založená na laserovom snímaní.

1.1.1 Systém Autoscan

Skenovací systém bol vyvinutý v roku 1998. Hlavným cieľom jeho vývoja bola flexibilita oproti vtedajším systémom, s ktorými sa veľmi ťažko manipulovalo.

Systém pozostával z laseru, dvojice videokamier, jednotky Elite system, ktorá obraz spracovávala v reálnom čase a z počítača. Elite system sa pri skladaní snímkov riadil pomocou vyhľadania miesta osvieteného laserom. To bolo detegované na základe jeho tvaru. 2D pozícia laseru z oboch kamier je odoslaná do PC a v reálnom čase je vykreslená do 3D priestoru. Vďaka vykresľovaniu v reálnom čase je po celý čas vidieť aktuálny model a tak možno laserom pohybovať podľa potreby a tam kde je to potrebné snímať detailnejšie.

Aparatúra sa dala umiestniť do ľubovoľnej vzdialenosti od objektu, čím bola zaručená dobrá flexibilita. Jedinou podmienkou bolo, aby kamery spolu zvierali uhol aspoň 60°, aby tým bola zabezpečená dostatočne vysoká presnosť vykreslených 3D dát. Pri príklade na Obr. 1.1 bola aparatúra vzdialená 1,2 metra od skenovaného objektu, pričom kamery boli od seba vertikálne vzdialené 1 meter a pomocou približovacích šošoviek boli nastavené tak, aby snímali požadovanú oblasť.

Nevýhodou systému bola na úkor flexibility vyššia časová náročnosť oproti ostatným systémom. Táto podkapitola čerpá zo zdroja [1].

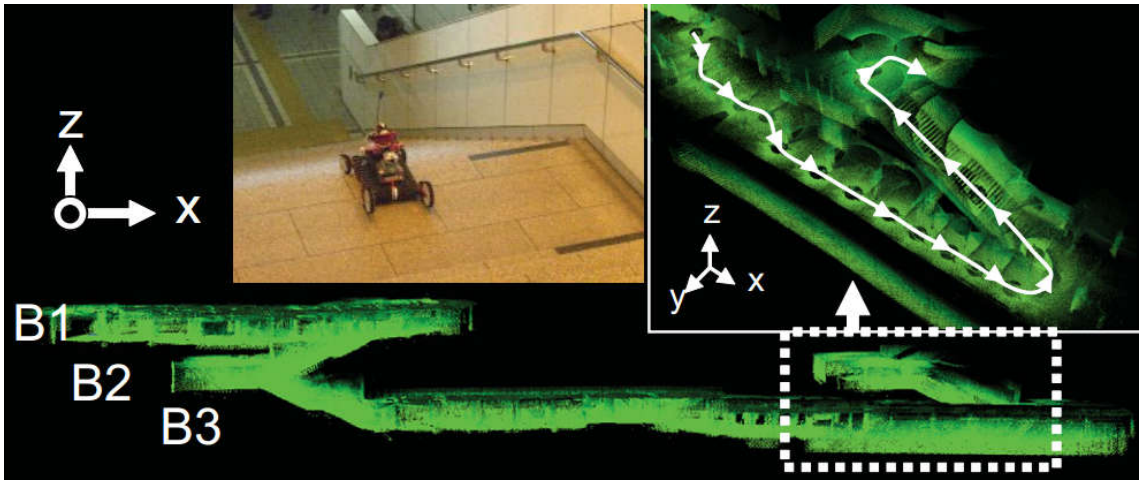
Obr. 1.1 Skenovanie systémom Autoscan

1.1.2 TK Scanner

Jedná sa o 3D skenovací systém na meranie stálosti a hustoty statického telesa v dynamickom prostredí. Projekt bol navrhnutý v roku 2009. Jeho hlavným cieľom je vytvorenie 3D modelu neprístupných lokalít vzniknutých napríklad po zemetrasení či bombovom útoku.

3D skener sa skladá z 2D laserového diaľkomeru a podložky, ktorá je naklápateľná ako vertikálne tak i horizontálne. Naklápanie podložky je ovládané pomocou 2 servomotorov.

Počas prieskumu terénu sa robot zastaví na rôznych miestach a 3D skenovaním zmeria parametre priestoru. Pomocou merania rýchlosti pohybu a pomocou zhôd naskenovaných dát medzi jednotlivými skenovanými pozíciami sa z jednotlivých meraní postupne vytvára 3D mapa. Vďaka krížovému skenovaniu je možno zmerať stálosť a hustotu bodov skenovanej plochy. Zároveň umožňuje odstrániť chybné dáta, ktoré vznikli pri prípadnom pohybe niektorého z objektov. Skener mení hustotu meraných bodov a rozsah skenovanej plochy pomocou zmeny výškového uhlu a uhlovej rýchlosti horizontálneho uhlu podľa potreby. Táto podkapitola čerpá zo zdroja [2].



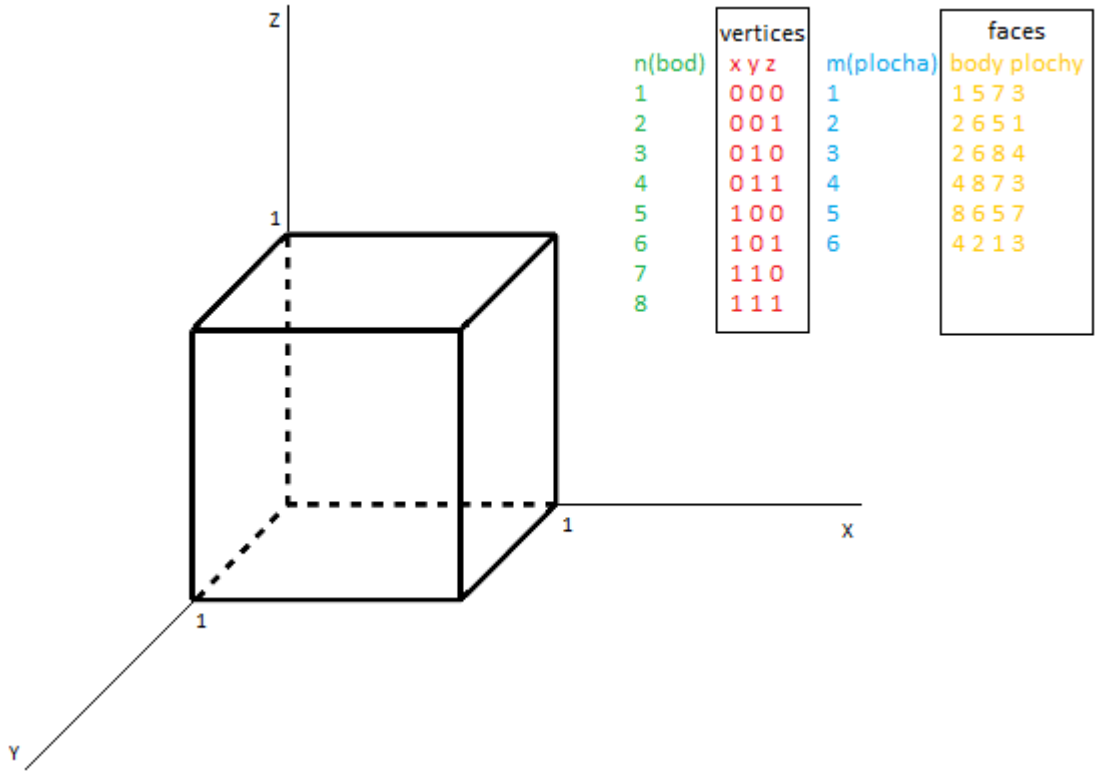
Obr. 1.2 3D mapa metra vytvorená skenovaním robotom TK scanner

1.2 Funkcie Renderwire a Renderpatch

Tieto funkcie slúžia na renderovanie 3D modelu v programe MATLAB. Metóda použitá v týchto funkciách funguje na základe rozloženia objektu na primitívne častice, ako sú trojuholníky či štvorce. Tieto primitívne častice sú usporiadané vo vektore a následným zavolaním jednej z funkcii renderwire, alebo renderpatch sú poskladané dohromady v 3D priestore, čím vznikne požadovaný model. Tieto funkcie nám zároveň vrátia celkový počet renderovaných prvkov, teda počet plôch z ktorých sú zložené. Funkcia renderwire slúži na vykreslenie 3D čiarového modelu z dát v matici. Funkcia renderpatch z matice vykreslí tieňovaný polygónový 3D model.

Vstupné dáta s ktorými tieto funkcie počítajú sú dve matice. Jedna „.vertices“ obsahuje súradnice jednotlivých bodov v kartézskej sústave. Druhá matica „.faces“ obsahuje usporiadanie týchto bodov do n-tíc.

Vzor zostrojenia týchto matíc pre kocku je na Obr. 1.3.

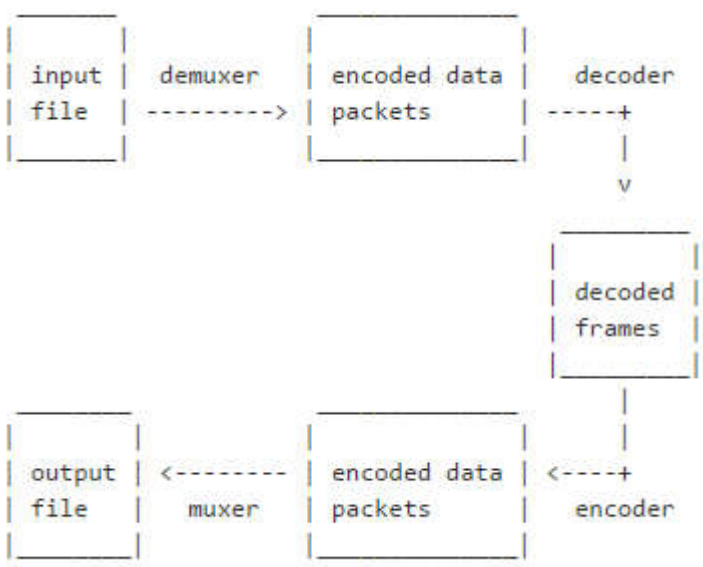


Obr. 1.3 Príklad vytvorenia matíc .vertices a .faces, potrebných vstupných dát pre funkcie renderwire a renderpatch

Obe tieto funkcie majú zároveň viacero vstupných parametrov. Pomocou parametrov „facecolor“ a „edgecolor“ možno definovať farbu plôch a hrán. Farba plôch je pôvodne nastavená na „none“ pre renderwire a farba hrán je pôvodne nastavená na „none“ pre renderpatch. Farby je možné nastavovať z predvolených kombinácii, alebo pomocou RGB matice. Ďalším spoločným vstupným parametrom je viditeľnosť „visibility“. Funkcia Renderpatch okrem toho obsahuje viacero parametrov ktorými sa nastavuje tieňovanie renderovaných plôch.

1.3 Nástroj FFmpeg

FFmpeg je multimediálny framework slúžiaci na kódovanie, dekódovanie, transkódovánie, multiplexovanie, demultiplexovanie, streamovanie, filtrovanie a prehrávanie multimediálneho obsahu. Podporuje obrovské množstvo formátov a funguje na veľkom množstve operačných systémov. V tejto práci je nástroj ffmpeg využitý pri synchronizácií obrazu a zvuku na prekódovanie vsupného video súboru. Proces prekódovania je zobrazený na Obr 1.4.



Obr. 1.4 Proces prekódovania cez ffmpeg [3].

Súčasťou FFmpeg sú 4 nástroje. V prvom rade je to nástroj ffmpeg, ktorý funguje v príkazovom riadku a slúži na spracovávanie video a audio obsahu. Ďalej sú tu ešte nástroje ffserver, ffplay a ffprobe. Nástroj ffserver slúži na streamovanie živých prenosov, s možnosťou timeshift. Nástroj ffplay slúži ako jednoduchý multimediálny prehrávač Posledný z nástrojov, nástroj ffprobe funguje v príkazovom riadku a slúži na analýzu video a audio súborov.

FFmpeg využíva knižnice libavcodec, libavutil, libavformat, libavfilter, libavdevice, libswscale a libswresample. Libavcodec je základnou knižnicou kodekov na kódovanie a dekódovanie video a audio súborov využívanou okrem nástrojov ako je FFmpeg aj v mnohých multimediálnych prehrávačoch. Knižnica Libavutil obsahuje funkcie pre zjednodušenie procesu programovania. Ďalšia knižnica libavformat, slúži ako rámec pre multiplexovanie a demultiplexovanie video a audio obsahu. Knižnica libavfilter, slúži ako univerzálny základ pre filtrovanie audio a video súborov. Libavdevice poskytuje všeobecný rámec pre zaznamenávanie obsahu a renderovanie do výstupných zariadení. Libswscale je knižnica, ktorá vykonáva vysoko optimalizovanú zmenu rozlíšenia obrazu a zmenu farebného modelu videa. Posledná knižnica libswresample, slúži pri prevzorkovaní audio súboru a zmene jeho formátu.

Tento softvér je teda využívaný ako veľmi efektívny konvertor audio a video súborov. Je schopný pracovať so súbormi uloženými na disku aj priamo s video streamom prichádzajúcim z kamery. FFmpeg číta z ľubovoľného počtu vstupných súborov, ktoré sú špecifikované pomocou značky „-i“ a zapisujú sa do ľubovoľného počtu výstupných súborov, ktorých názov sa uvádza bez akejkoľvek značky, pričom všetko čo neobsahuje žiadnu značku sa automaticky považuje za názov výstupného súboru.

Pomocou značiek možno označovať takisto veľké množstvo parametrov, pokiaľ chceme na vstupe, alebo výstupe niečo zmeniť (napríklad –r pre zmenu snímkovej frekvencie, alebo –b pre zmenu prenosovej rýchlosti). Ako základné pravidlo platí že značky sa vzťahujú na najbližší súbor, preto je dôležité dávať pozor na poradie.

Na vstupe i výstupe môže byť ľubovoľné množstvo súborov rôzneho typu (video/audio/titulky/iné dáta). Pomocou značky „-map“ je možné priradiť konkrétne výstupné súbory ku konkrétnym vstupným súborom, v opačnom prípade je to robené automaticky. Pri poukazovaní na konkrétny vstup alebo výstup používame indexy ktoré sú im priradené, pričom prvý v poradí má číslo 0. Ak nejaký súbor pozostáva z viacerých dátových prúdov, tak po indexe súboru nasleduje „:“ a poradové číslo dátového prúdu v rámci súboru. Príklad využitia nástroju ffmpeg v kóde s rôznymi parametrami je na Obr. 1.5. Táto kapitola čerpá zo zdroja [3].

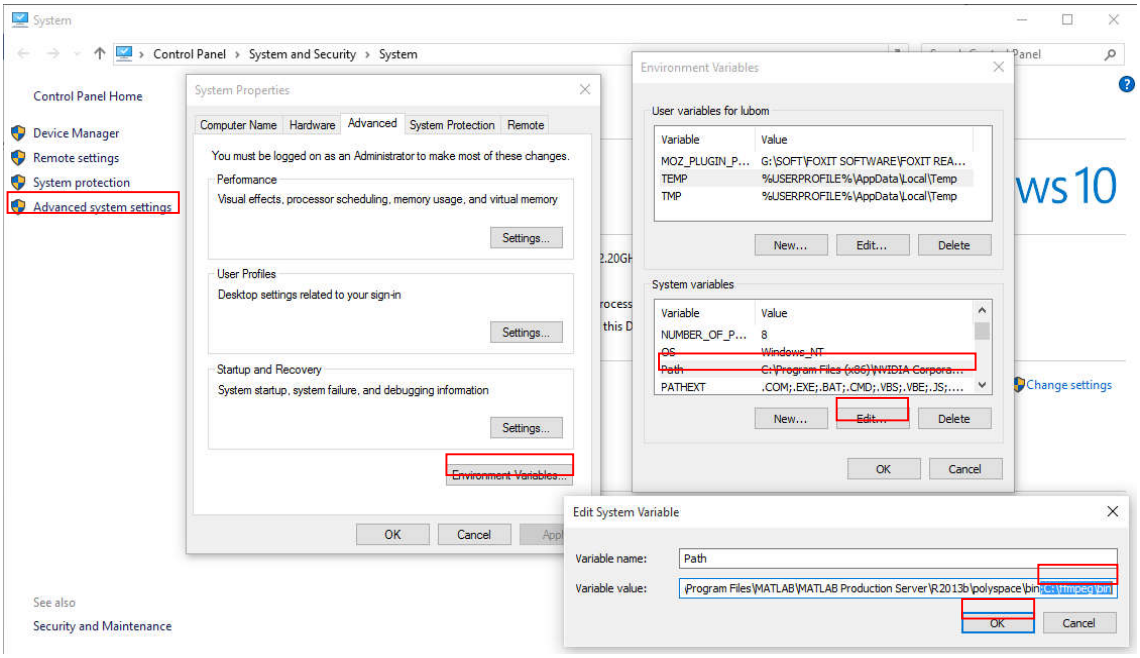
```
ffmpeg -i a.mov -i b.mov -c copy -map 0:2 -map 1:6 out.mov
```

Obr. 1.5 Príklad použitia indexov. Do výstupného vúboru sa nakopíruje tretí dátový prúd z prvého vstupného súboru a siedmy dátový prúd z druhého vstupného súboru [3].

1.3.1 Inštalácia FFmpeg

Pokiaľ má ffmpeg správne fungovať v spojení s programom MATLAB, je vhodné nainštalovať ho do vytvorenej zložky „ffmpeg“ do základného adresára dátovej jednotky na ktorej je nainštalovaný operačný systém (zvyčajne disk C:).

Následne je potrebné udeliť nástroju ffpmeg povolenia cez nastavenia operačného systému. Konkrétne pri systéme Windows je potrebné v rozšírených nastaveniach systému vyhľadať premenné prostredia a do systémovej premennej Path priradiť na koniec reťazca bodkočiarku a cestu do zložky kde je nainštalovaný ffmpeg (napríklad v tomto prípade bude priradený reťazec ;c:\ffmpeg\bin). Celý tento postup je graficky znázornený na Obr 1.6.



Obr. 1.6 Priradenie cesty „;c:\ffmpeg\bin“ do systému

1.4 Segmentácia

Segmentácia patrí medzi najdôležitejšie procesy pri spracovaní obrazového materiálu. Vstupom segmentácie je nejaký snímok a parametre na základe ktorých sa segmentuje. Výstupom sú oblasti obrázka, ktoré splnili dané parametre segmentácie.

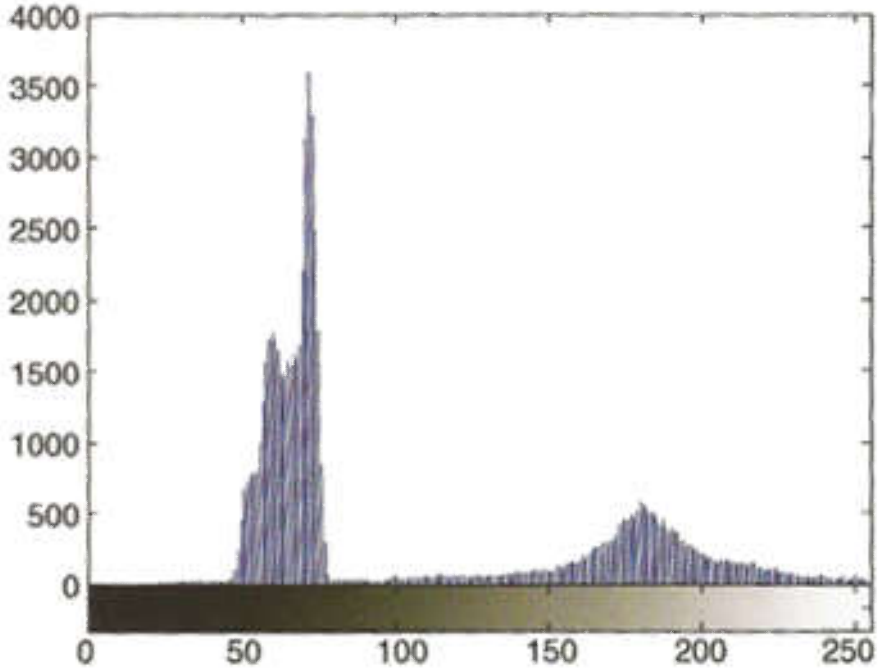
Väčšina segmentačných algoritmov je založená na 2 základných vlastnostiach, ktorými sú diskontinuita a podobnosť. V praxi sa pri segmentácií vyskytuje mnoho problematických faktorov, ako sú nerovnomerné rozloženie svetla, tiene, prekrývanie objektov, zlý kontrast objektu a pozadia a podobne. Preto je potrebné v rozličných prípadoch vyhľadávať rozličné vhodné metódy. Žiadna z nich pritom nemusí byť úplne dokonalá. Lepšie výsledky možno dosiahnuť skombinovaním viacerých metód. Neexistuje však žiadna univerzálna metóda, vhodná sa vyberie na základe druhu segmentovaného obrázku a na základe požadovaného výstupu.

Metódy segmentácie sa dajú rozdeliť do troch základných skupín. Patria sem metódy založené na intenzite, ktoré fungujú na základe rozloženia pixelov a využívajú napríklad histogramy. Druhou skupinou sú metódy založené na oblastiach, ktoré sledujú podobnosť pixelu s pixelmi v jeho okolí a kontrolujú spojitosť daných oblastí. Treťou skupinou sú všetky ostatné metódy, ktoré pracujú napríklad na základe textúr, hrán, pohybu a podobne. Táto kapitola čerpá zo zdroja [4].

1.4.1 Segmentácia založená na intenzite

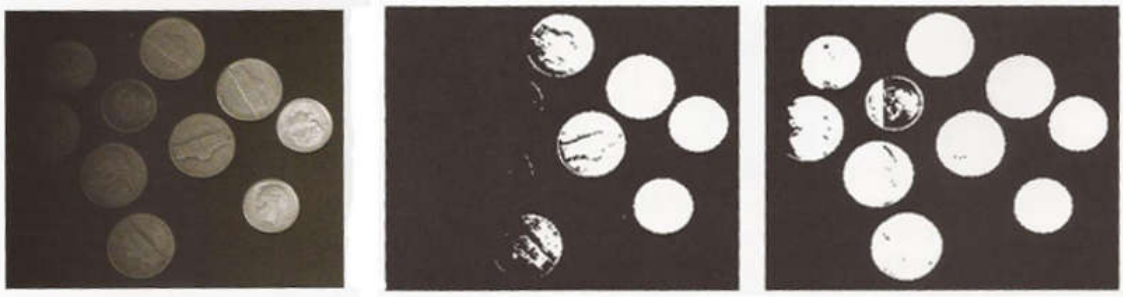
Tento druh segmentácie možno považovať za najjednoduchší a najpoužívanejší. Na základe hodnôt pixelov vyjadrených v histograme sa odčíta prahová hodnota a pomocou nej sa oddelí segmentovaná oblasť od pozadia.

Metóda teda funguje na základe porovnávania hodnoty pixelov s hodnotou referencie a odpovedá hodnotou 1, ak je splnená, alebo 0 ak nie je splnená. Metóda je teda vhodná najmä ak sa kladie dôraz na tvar výslednej oblasti a nie na jej štruktúru a taktiež pokiaľ je táto oblasť svetlejšia ako zvyšok snímku. Ak je teda výrazne svetlejšia oblasť na výrazne tmavšom pozadí, výsledný histogram zložky jasu bude mať dve dobre identifikovateľné lokálne maximá, ako vidno na Obr 1.7.



Obr. 1.7 Vykreslený histogram jasovej zložky Y z modelu YCbCr [4].

Hlavný problém vzniká pokiaľ sa má spracovávať veľké množstvo odlišných snímkov, hladina maxím sa totiž mení a ich manuálne určovanie je zdĺhavé. Pomocou určitých algoritmov je možné automatické vyhľadávanie týchto oblastí, ich použitím sa ale zvýši chybovosť segmentácie. Zabezpečením vhodných podmienok pri snímaní segmentovaných obrázkov však chybovosť možno výrazne potlačiť. Pri určitých problémoch, ako je napríklad nerovnomerné rozloženie svetla môže byť taktiež nápomocné použitie lokálnych prahových hodnôt, čo je zobrazené na Obr 1.8. V tom prípade pixel, ktorý je v tmavšej časti obrázka vyhodnotený ako pozorovaný objekt je vo svetlejšej časti vyhodnotený ako pozadie.



Obr. 1.8 Vľavo je obrázok ktorý sa segmentuje, v strede obrázok segmentovaný globálne, vpravo je obrázok segmentovaný lokálne [4].

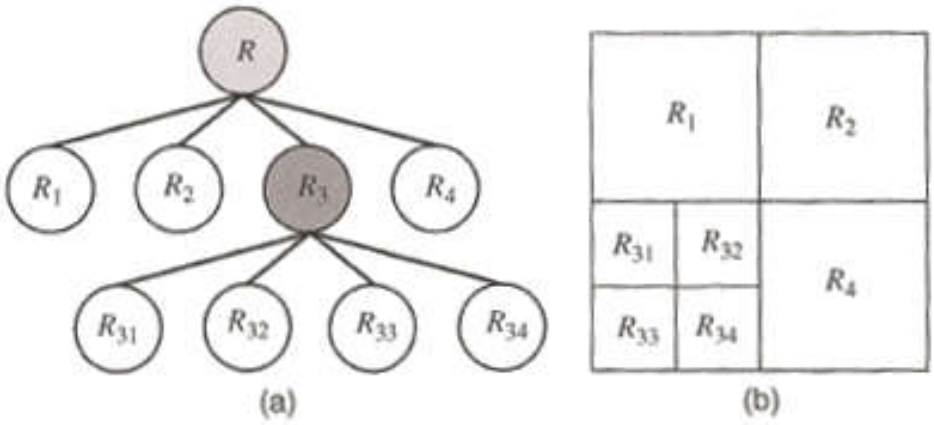
1.4.2 Segmentácia založená na oblastiach

Táto segmentácia uvažuje s tým, že pixel nie je samostatný prvok, ale je prepojený s podobnými pixelmi vo svojom okolí s ktorými vytvára nejakú oblasť. Metódy založené na tejto segmentácii rozdelia na základe tohto predpokladu snímok na n oblastí, ktoré spolu pokrývajú celý snímok a neprekrývajú sa v žiadnom mieste. Pixely

v jednej oblasti pritom splňujú spoločne určitú podmienku, takzvané kritérium homogenity. Takýmto kritériom je napríklad presná hodnota, alebo rozsah intenzity, prípadne podobná textúra. U tohto druhu segmentácie sa využívajú 2 metódy a to metóda rastúcej oblasti a metóda rozdeľovania a spojovania oblastí.

Metódy rastúcej oblasti fungujú na princípe toho, že na začiatku je k dispozícii jeden prednastavený pixel a pomocou podobnosti narastá okolo neho oblasť, až kým okolité pixely prestanú spĺňať kritérium homogenity. Metóda má 3 vstupné faktory. Prvým je výber kritéria homogenity, ktoré je založené na intenzite farieb a vlastnostiach prepojenia pixelov. Ďalej to je výber prednastavených pixelov, ktoré možno vyberať buď manuálne, alebo automaticky na základe predchádzajúcej hrubšej analýzy snímku. Posledným vstupným faktorom je zadefinovanie kritéria, kedy sa má oblasť prestať rozširovať. Táto metóda má množstvo limitácií a preto nie je najvhodnejšia na použitie pri zložitých aplikáciách. Naopak, jej jednoduchosť je jednou z jej hlavných predností u menej zložitých aplikácii. Hlavnými nedostatkami je to, že výsledky sú veľmi citlivé na zadané vstupné parametre a v niektorých prípadoch je možné že zostanú pixely, ktoré nebudú patriť ani do jednej z oblastí. Taktiež sa môže stať že nevhodným umiestnením prednastavených bodov sa jedna oblasť rozdelí na viacero oblastí.

Pri metóde rozdeľovania a spojovania oblastí sa začína delením celého snímku na menšie subsnímky až kým každý z týchto subsnímok nevytvára homogénnu oblasť. Spojovanie potom zabezpečuje zlúčenie susedných regiónov ktoré sú dostatočne podobné a spĺňajú kritérium homogenity. Väčšinou sa využíva quadtree metódy, ktorá delí oblasť na 4 podoblasti až kým sa v danom mieste nedosiahne homogenity. Princíp tejto metódy je na Obr 1.9.



Obr. 1.9 Príklad metódy quadtree [4]. [4]

1.5 3D Renderovanie

Renderovanie je proces generovania nových 2D snímkov z dát 3D modelu, alebo skupiny snímkov pomocou PC softvéru. Renderovanie sa využíva hlavne v architektúre, videohrách, simulátoroch a pri vytváraní špeciálnych efektov v TV technike. Podľa prístupu rozdeľujeme renderovanie na 2 skupiny: renderovanie založené na geometrii a renderovanie založené na obraze. Väčšinou sa v praxi využíva kombinácia oboch prístupov.

Renderovanie založené na geometrii je popísané pomocou primitívnych geometrických častíc. Renderovanie založené na obraze je založené na 7 rozmernej plenoptickej funkcií:

$$Plen_{plná} : R^3 \times (0,2\pi)^2 \times R \times R \rightarrow R, Plen_{plná}(x, y, z, \phi, \theta, \lambda, t) = I \qquad (1.1)$$

, kde I je výsledná intenzita svetla, svetelných lúčov v nejakom bode priestoru (*x,y,z*), pričom zohľadňuje polohu pozorovateľa (ϕ,θ), vlnovú dĺžku ovplyvňujúcu farby (λ) a čas (t) textúru, geometriu, osvetlenie, a tiene ktoré model vrhá. Výsledná funkcia je teda veľmi výpočtovo náročná no pri statickom prostredí možno vylúčiť a svetelným lúčom pohybovať voľne po priestore. Ďalšou možnosťou zjednodušenia je oddelenie časového a priestorových rozmerov (*t, x, y, z*) od rozmerov (ϕ, θ, λ) smeru pozorovania a farby. Ako plenoptickú vzorku (PV) tým definujeme 3D subpriestor, ktorý vytvorí nejaký sférický obraz I_{PV} s parametrami ϕ, θ, λ na pozícii so súradnicami x_t, y_t, z_t a je popísaný funkciou:

$$I_{PV} : R^3 \times (0,2\pi)^2 \times R \times R \rightarrow R, I_{PV}(\phi, \theta, \lambda)\big|_{x,y,z,t} = I \qquad (1.2)$$

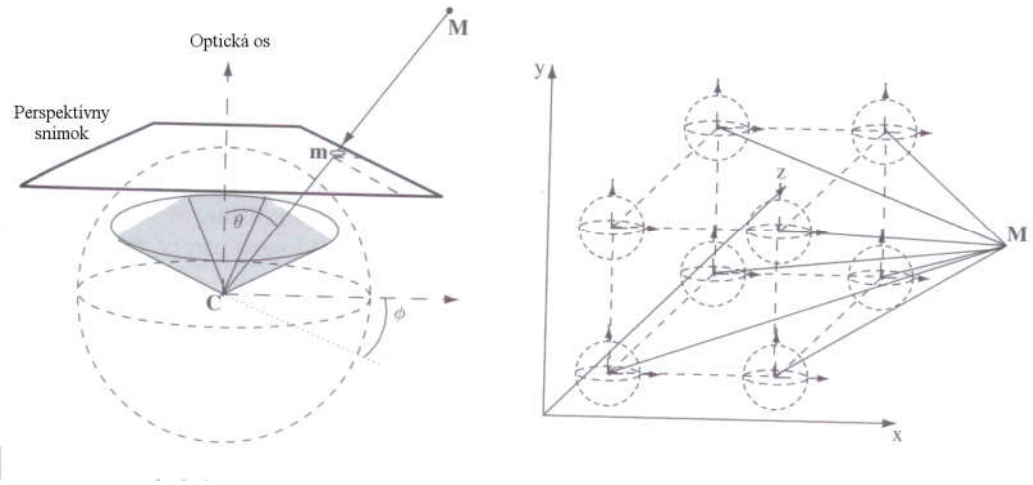
Pokiaľ chceme popísať celý priestor pomocou plenoptickej funkcie tak môžeme vyjadriť ako množinu všetkých plenopických vzoriek v priestore pomocou rovnice:

$$Plen_{priestorová} : R^3 \rightarrow (0,2\pi)^2 \times R, Plen_{priestorová}(x_t, y_t, z_t) = I_{PV}(\phi, \theta, \lambda) \qquad (1.3)$$

Vymodelovanú kameru snímajúcu konvenčnú perspektívu obrazu možno navrhnúť ako tangentuálnu rovinu do gule, ako je zobrazené na Obr. 1.10.

Na základe týchto vlastností možno po renderovaní pozorovať na snímkach javy ako tieňovanie, textúrovanie, hmlu, tiene, odraz svetla, priehľadnosť, priesvitnosť, difrakciu, pohybové rozostrenie a iné optické javy. Táto práca sa však zaoberá len geometrickým vykreslením a optická časť bude vyriešená v neskoršej fáze.

Existuje viacero delení obrazových renderovacích metód, najnovšia z nich ich delí podľa množstva geometrie v nich využitých na renderovanie bez geometrie, renderovanie z implicitnou geometriou a renderovanie s explicitnou geometriou. Okrajovo sú v tejto kapitole spomenuté aj metódy založené na geometrii. V Tejto kapitole sú využité zdroje [5], [6], [7] a [8].



Obr. 1.10 V ľavej časti je zobrazený guľový senzor C, ktorý sníma lúč z bodu M pod uhlami ϕ, θ. Rovina obrazu skutočného senzoru je tangentou ku guli, na ktorej sa bod M premietne ako m. Vpravo je zobrazených 8 plenoptických vzoriek v klasickej mriežke, ktoré snímajú bod M [6].

1.5.1 Renderovacie metódy založené na geometrii

Pri týchto renderovacích metódach je svetelná zložka zaobstaraná tieňovacím modelom. Hlavné nevýhody renderovacích metód založených na geometrii sú nespoľahlivosť a výpočtová náročnosť.

Existujú 2 základné modely. Goraudov model je veľmi jednoduchá technika, ktorá lineárne interpoluje intenzitu farby po polygóne. Phongov model je okrem toho schopný pracovať aj s odrazmi svetla. Nedokáže síce zobrazovať úplne všetky optické javy, je však veľmi efektívny a používaný. Existuje taktiež druhá trieda týchto modelov, takzvané globálne svetelné modely. Na rozdiel od spomínaných lokálnych metód sú schopné uvažovať odrazy medzi povrchmi objektov. Tieto metódy sú však značne výpočtovo náročnejšie a tie zložitejšie nie sú schopné pracovať v reálnom čase.

1.5.2 Renderovacie metódy založené na obraze

Pri renderovaní je možné vyhnúť sa modelovaniu objektu, či prostredia ktoré sa má zobraziť. Dá sa to docieliť fotografovaním tohto prostredia z viacerých uhlov. Pomocou interpolácie je potom možné dopočítať snímky z uhlov z ktorých prostredie vyfotografované nebolo. Metódy renderovania, ktoré tento jav využívajú sa nazývajú metódami založenými na obraze. V angličtine sa pre ne zaužívala skratka DIBR (depth image-based rendering).

Namiesto toho aby boli použité milióny polygónov tak môže byť prostredie reprezentované pomocou skupiny snímkov a veľmi zjednodušeným geometrickým modelom. Toto renderovanie možno rozdeliť na 3 skupiny.

Renderovanie bez geometrie spravidla potrebuje veľké množstvo snímkov. Na úkor toho však nevyužíva vôbec geometriu. Nevýhodou je však že nedokáže dobre pracovať s prostredím, ktoré sa v čase mení.

Druhá metóda, renderovanie s implicitnou geometriou sa spolieha na pozičnú

zhodu bodov v snímkach na základe čoho je vytvorená geometria a generujú sa nové snímky. Pre získanie správneho výsledku je treba vedieť relatívnu polohu snímkov s ktorých geometriu vytvárame.

Tretie je renderovanie s explicitnou geometriou, ktoré má k dispozícií vopred geometriu prostredia, či textúrové mapy.

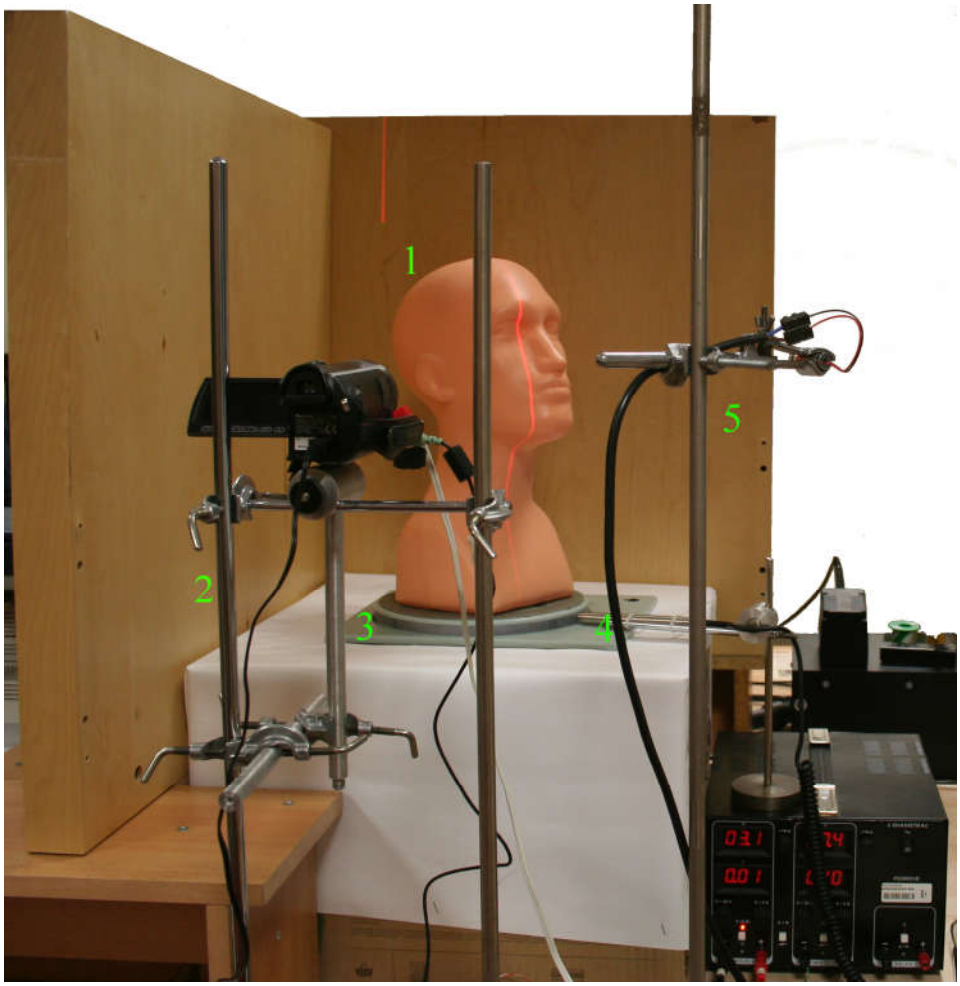
2 REALIZÁCIA SKENERU

2.1 Koncepcia rotačného skeneru

Základná koncepcia rotačného skeneru pozostáva z rotačnej plošiny, na ktorej je umiestnené skenované teleso, z laseru, optickej šošovky a z kamery.

Laser je fixovaný na stojane v určitej vzdialenosti od snímaného telesa a je orientovaný na jeho stred. Optická šošovka tento vzniknutý kruhový zväzok lúčov laseru rozptýli na svetelný pruh. Lúč je orientovaný vertikálne a svieti po celej výške telesa.

Rovnako na stred rotačnej plošiny je orientovaná i kamera. Tá zviera s lúčom uhol α, ktorý bol experimentálne určený tak, aby sme získali čo najkratšie úseky diskontinuity. Kamera sníma vo FullHD. Videozáznam je ukladaný na SD kartu a následne nahraný do PC. Celá koncepcia skeneru je zobrazená na Obr. 2.1.

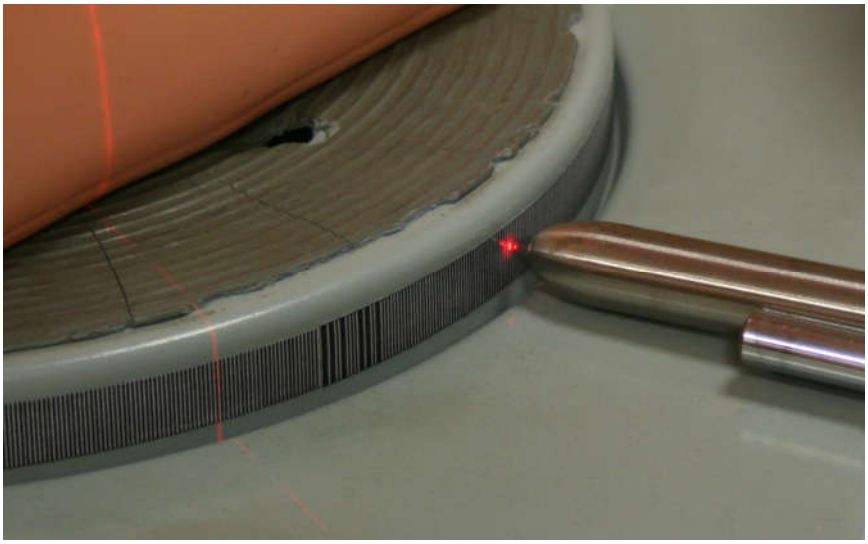


Obr. 2.1 Koncepcia rotačného skenera: 1. skenované teleso, 2. stojan s kamerou, 3. rotačná plošina, 4. snímač čiarového kódu, 5. stojan s laserom a šošovkou

Rýchlosť rotačnej plošiny je riadená pomocou potenciometra ktorým je možné regulovať napájacie napätie. Dobu rotácie je vhodné nastaviť na približne 20 sekúnd. Je možné tak dosiahnuť dostatočný počet snímkov a zároveň nasnímané video nie je príliš veľké. Rotačná plošina je nesynchrónna a preto je potrebné dodatočnú synchronizáciu zabezpečiť externe.

2.1.1 Synchronizácia rotačnej plošiny

Synchronizácia rotačnej plošiny je vyriešená pomocou 4 rovnako dlhých čiarových kódov, ktoré sú nalepené po celom obvode rotačnej plošiny a sú snímané laserovým čítačom. Táto aparatúra je zobrazená na Obr. 2.2.



Obr. 2.2 Snímač čiarového kódu prilepeného na rotačnej plošine.

Čiarové kódy obsahujú úzke a široké tmavé elementy, ktorých šírky sú v pomere 1:3. Každý zo štyroch kódov sa začína odlišnou sekvenciou, pozostávajúcou zo štartovej sekvencie, sekvencie pre číslo 1 až 4 a stop sekvencie. Tmavé elementy predstavujú logickú 0 a biele predstavujú logickú 1.Vplyvom jednosmerného oddelenia tu teda vzniká derivovaný obdĺžnikový signál. Dĺžka stavov v ňom je daná rýchlosťou otáčania. Príklad prečítaného signálu z čiarového kódu je na Obr. 2.3.

Obr. 2.3 Príklad čiarového kódu a výstupný nezderivovaný signál získaný jeho čítaním [9].

Tieto čiarové kódy sú čítané pomocou laserového optického čítaču a nasnímaný signál je odosielaný do kamery ako audio stopa. Po vyexportovaní dát z SD karty kamery do PC video rozdelíme na obrazovú časť a zvukovú časť pomocou FFmpeg softvéru. Následne je ku každému snímku videa priradený rovnaký počet prvkov zo zvukovej matice. V prípade, že nebude priradiť každému snímku rovnaký počet prvkov z audio stopy, tak audio stopa posledného snímku bude doplnená nulami. Na Obr. 2.4 sú 2 takéto stopy zobrazené. Kód vykonávajúci vzorkovanie videa, oddelenie video a audio stopy má nasledovnú podobu:

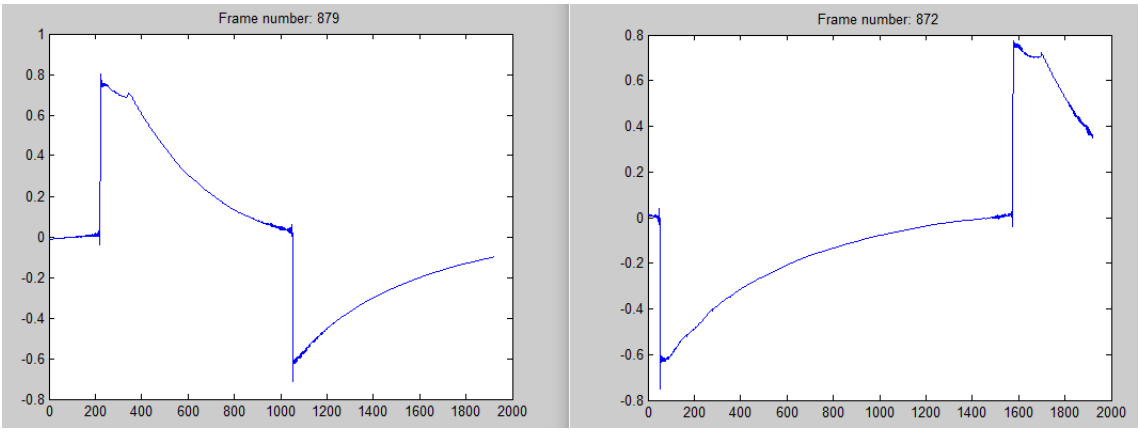
```
clearvars

%vzorkovanie videa
system('ffmpeg -i 00003.MTS -vcodec libx264 -crf 20 -acodec ac3 -vf
"yadif" output_deinterlaced2.avi');
 obj=VideoReader('output_deinterlaced2.avi');
 nFrames=obj.NumberOfFrames;

%oddelenie video a audio stopy
 system('ffmpeg -i 00003.MTS audio2.wav');

 nAS = audioinfo('audio2.wav');
 nAS=nAS.TotalSamples;        %počet audio vzorkov
 nSpF=ceil(nAS/nFrames);      %počet audio vzorkov na jeden frame videa
 SpF= zeros(nFrames,nSpF,2);  %audio stopy jedenotlivých frameov videa
 audio = dsp.AudioFileReader('audio2.wav','SamplesPerFrame',nSpF)
 k=1;

 while ~isDone(audio)
       A = step(audio);
       SpF(k,:,:) = A;
       k=k+1;
 end
 release(audio); % release the input file
```



Obr. 2.4 Príklad tvaru audio stopy pre vybrané snímky videa. Z prava: 1. v tomto mieste sa čiarový kód skladá len z krátkych elementov, priebeh sa teda periodicky opakuje, 2. v tomto mieste sa nachádza jeden z širokých elementov

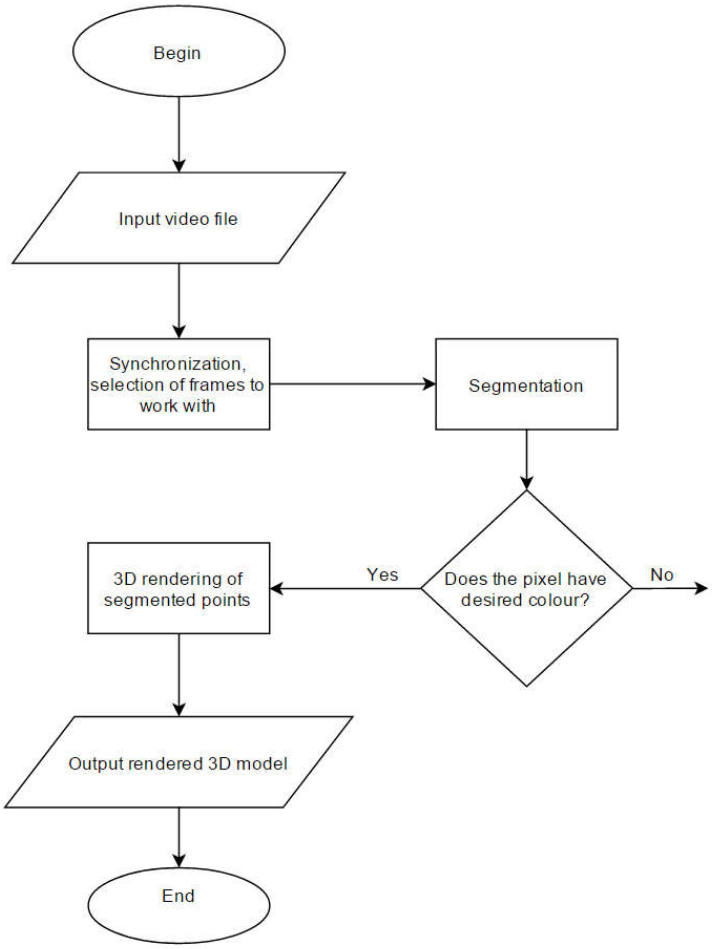
V audio stope je najskôr potrebné vyhľadať kedy sa prvý krát zopakuje prvá sekvencia. Tým je možné z celého nasnímaného obsahu vyfiltrovať len prvú rotáciu. Ďalej sa teda bude pracovať s menším obsahom dát a bude sa v ňom jednoduchšie

orientovať.

Vďaka odlišným sekvenciám v každej štvrtine je možné nielen určiť presný čas jednej rotácie, ale takisto túto rotáciu rozdeliť na 4 úseky, ktoré by mali byť pri synchrónnom pohybe rovnako dlhé. V prípade nesynchrónneho pohybu sa dĺžky týchto sekvencií vyrovnajú pomocou rovnomerného vystrihnutia niektorých snímkov z daných štvrtín videa, tak aby boli všetky rovnako dlhé.

2.2 Realizácia skenovania

Proces skenovania využíva ako vstupné dáta obrazovú časť z videa, ktorá prebehla vystrihnutím jedného cyklu a časovou synchronizáciou pomocou FFmpeg. Pre zrýchlenie procesu sa na základe požadovanej presnosti sa nastaví počet snímkov s ktorými sa bude pracovať. Skenovanie pozostáva z niekoľkých krokov. Prvým je synchronizácia, ktorej bola venovaná predchádzajúca podkapitola a získanie dát s ktorými sa bude pracovať. Ďalším krokom je Segmentácia požadovaných dát z obrázku a v poslednom rade to je renderovanie obrázku. Zjednodušený vývojový diagram je na Obr. 2.5.



Obr. 2.5 Zjednodušený vývojový diagram

2.2.1 Segmentácia požadovaných dát z obrázku

Z testovaných metód segmentácie sa nakoniec ako najvýhodnejšia ukázala segmentácia na základe farieb z chromatických zložiek farebného modelu YCbCr. Pôvodné snímky sú vo farebnom formáte RGB. Najskôr sa teda obrázok prevedie z RGB sústavy do sústavy YCbCr. Tento prevod možno zabezpečiť násobením špeciálnou maticou, ktorá je zobrazená v nasledujúcej rovnici [1]:

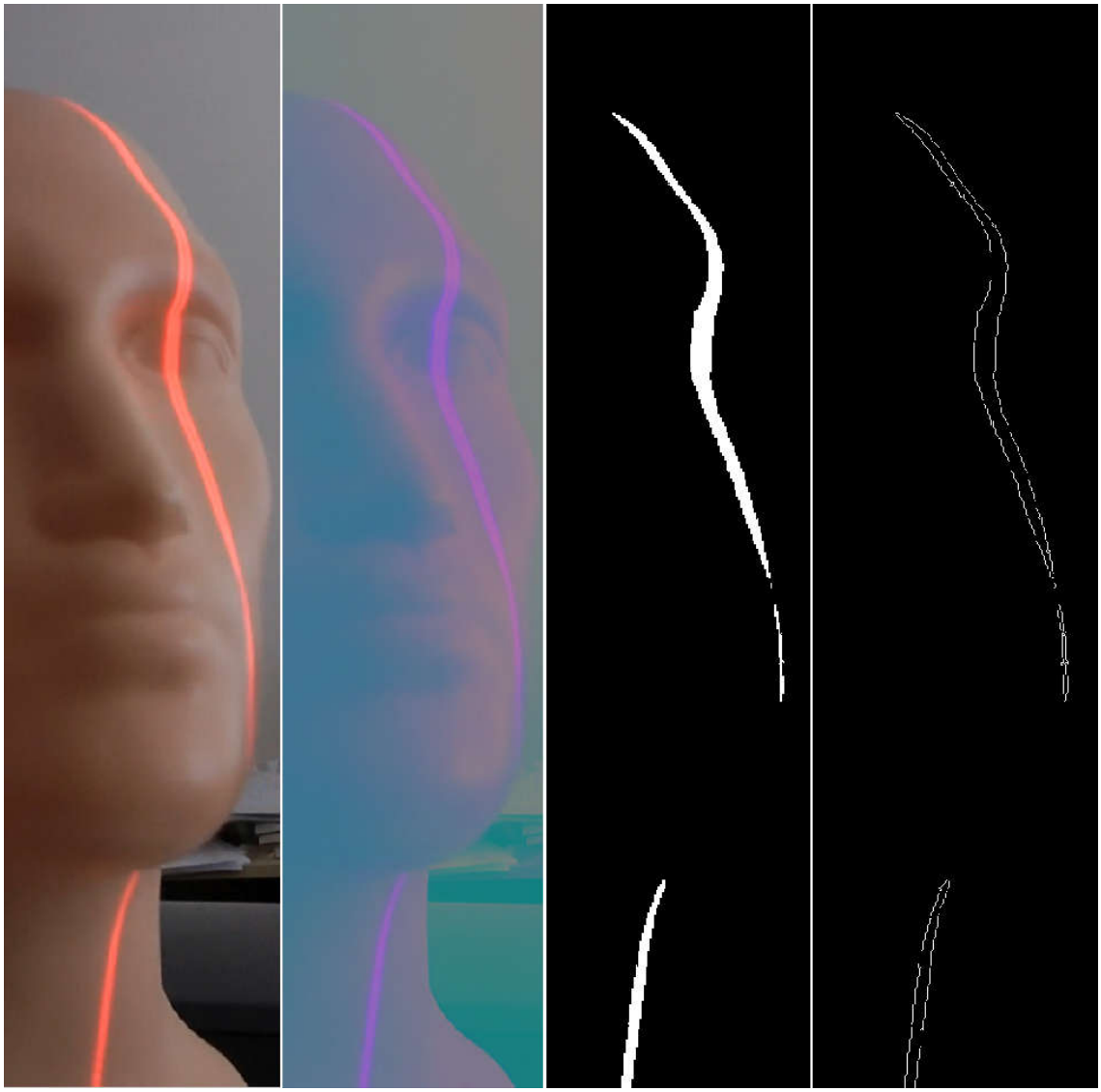
$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0{,}299 & 0{,}587 & 0{,}114 \\ -0{,}169 & -0{,}331 & 0{,}500 \\ 0{,}500 & -0{,}419 & -0{,}081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.1}$$

V programe MATLAB na toto násobenie existuje funkcia „rgb2ycbcr“. Následne je potrebné chromatické zložky Cb a Cr vykresliť do histogramu a vyhľadať v ňom oblasť farby laseru. Z histogramu sa vymedzia sa hodnoty Cr a Cb medzi ktorými sú jednotlivé body osvietené laserom. Na základe tohto je vytvorená funkcia ktorá filtruje, nami požadované farebné rozpätie zo snímku. Vytvorí sa nový snímok, s potrebnými rozmermi, pričom každý z pixelov má pôvodne RGB hodnotu [0,0,0]. Vždy keď niektorý z pixelov pôvodného snímku splní podmienky funkcie tak sa hodnota zodpovedajúceho pixelu na novom snímku prepíše na bielu [255,255,255]. Vytvorí sa tak nový dvojfarebný snímok, v ktorom je oblasť osvietená laserom biela a zvyšok snímku čierny.

Ďalším krokom je vykreslenie hrán tejto vyfiltrovanej oblasti pomocou funkcie edge. Táto funkcia však pracuje len s 0 a 1. Pomocou funkcie rgb2gray, ktorá podľa jasu priraďuje jednotlivým pixelom hodnoty 0 až 1 sú hodnoty pixelov prispôsobené na požadovaný tvar. Všetky fázy, ktorými obrázok prechádza sú zobrazené na Obr. 2.5.

Následne je potrebné vybrať z každého riadku obrázku len jeden bod. Vyberá sa posledný segmentovaný bod v riadku, ktorý predstavuje vonkajší obrys laseru na skenovanom telese. Výsledkom je matica, ktorej jedným z údajov je výška, ďalším je číslo rezu a posledným je vzdialenosť od osi vo vodorovnej rovine. Rezy sú rozložené rovnomerne, takže je možné určiť aký uhol zviera daný rez s prvým rezom.

Pri ďalšom spracovaní dát sa postupne zo všetkých bodov matice vyhľadá táto jediná nenulová hodnota z každého riadku. Následne pomocou sínusu a kosínusu uhlu a vzdialenosti od osi môžeme určiť súradnice X a Y kartézskej sústavy. Súradnica Z je výšková súradnica, ktorá bola známa od začiatku. Vytvoríme novú maticu, ktorá obsahuje tieto súradnice. Takáto matica sa vo vhodnom tvare už dá pomocou základných funkcií programu MATLAB vykresliť. Pre docielenie čo najlepších výsledkov je však vhodné renderovať ju pomocou obsiahlejšej funkcie.



Obr. 2.6 Sprava: 1. pôvodný nasnímaný obrázok v sústave RGB, 2. obrázok prevedený do sústavy YCrCb, 3. obrázok po segmentácií, 4. obrázok po aplikovaní funkcie edge. Posledný obrázok vyzerá v niektorých miestach nespojito z dôvodu, že matlab ho nebol schopný vykresliť v tak veľkom rozlíšení, v skutočnosti je však spojitý tak ako predchádzajúce obrázky.

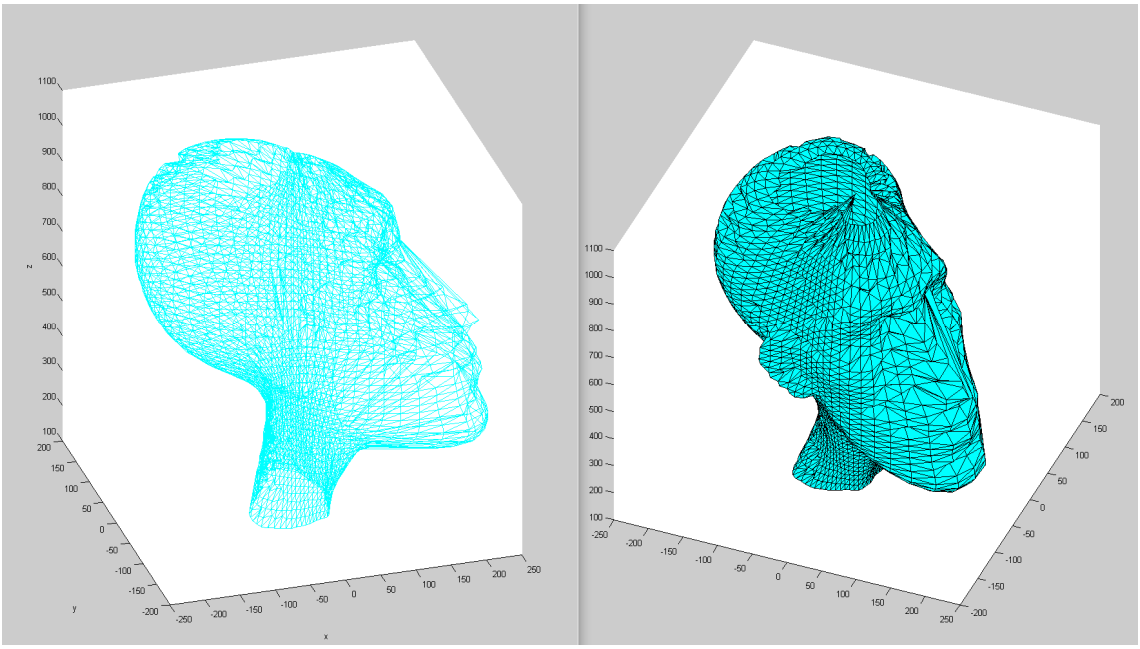
2.2.2 Tvorba 3D modelu a renderovanie obrázku

Renderovanie 3D objektu prebieha pomocou funkcii renderwire pre drôtový model a renderpatch pre vyplnený model. Vstupnými hodnotami oboch týchto funkcii sú 2 2-rozmerné matice. Prvá z matíc „.vertices“ má rozmer M*3, kde M je počet bodov ktoré sa budú vykresľovať a v druhej súradnici v tejto matici sú obsiahnuté súradnice X Y a Z bodov v kartézskej sústave. Druhá matica „.faces“ má rozmer M*n, kde M predstavuje počet plôšok z ktorých bude vytvorený renderovaný model a n predstavuje z koľkých bodov sú tieto plôšky zložené.

Využívané sú trojice bodov. Tieto trojice bodov sú generované pomocou funkcie, ktorá najskôr nájde prvý a posledný riadok každého stĺpca a zaznamená

súradnice týchto bodov a číslo riadku v ktorom boli nájdené. Body nad hornou a pod dolnou hranicou majú iný algoritmus ktorým sú spájané do trojíc ako body obsiahnuté medzi nimi.

Algoritmus spájajúci body v strede spočíva v tom, že v danom stĺpci matice sa vytvorí dvojica bodov pod sebou. V prípade že nie sú 2 body priamo pod sebou, tak program hľadá najbližší ďalší bod v stĺpci. Tretí bod sa vyberá zo susedného stĺpca na jednej i na druhej strane. Pre stĺpce vpravo sa primárne priraďuje bod na úrovni prvého bodu z dvojice a body nad ním. Pre stĺpce vľavo sa primárne priraďuje bod na úrovni spodného bodu z dvojice a body pod ním. Pri generácii trojíc sa dbá na to aby sa plôšky po renderovaní neprekrývali.



Obr. 2.7 Vyrenderované modely, vľavo pohľad zľava na sieťový model vytvorený cez renderwire, vpravo pohľad spredu na polygónový model vytvorený cez renderpatch.

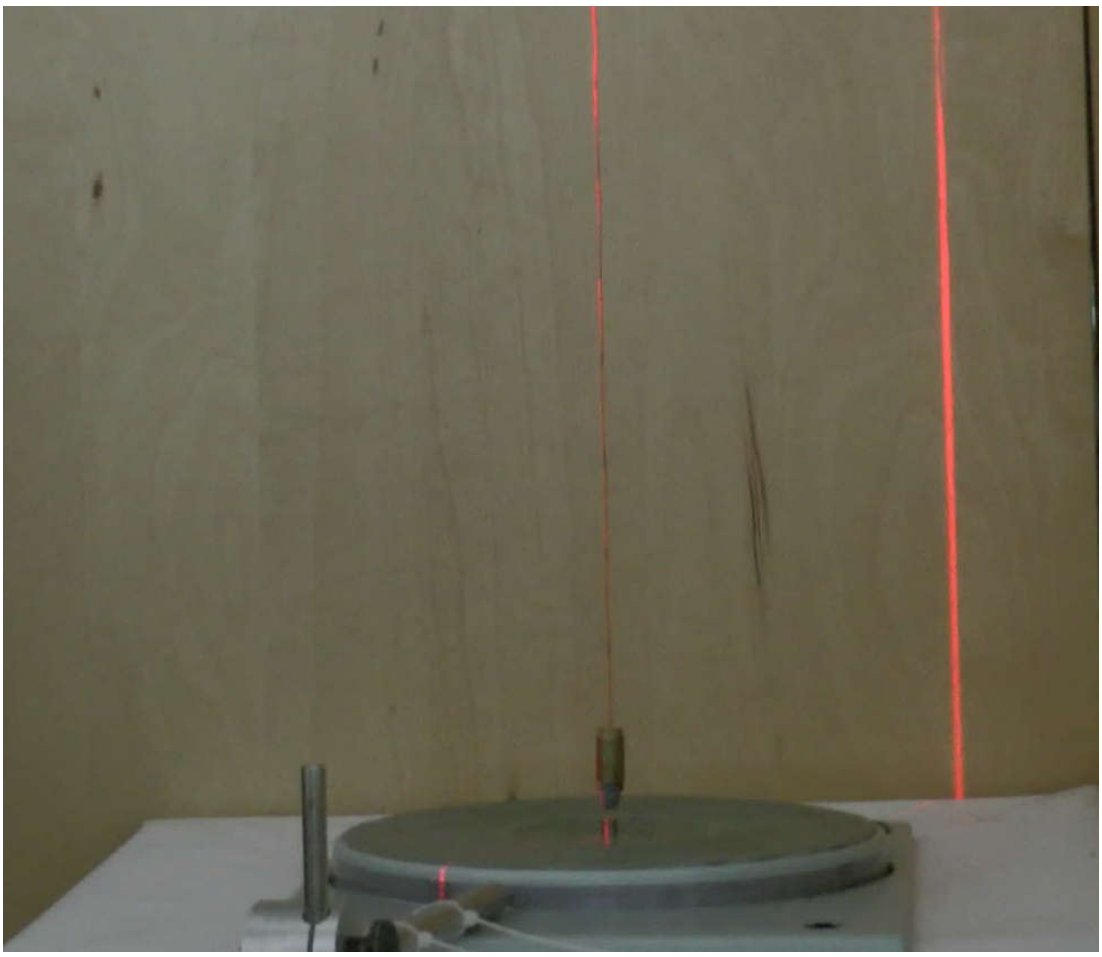
3 APLIKÁCIA DRUHEJ KAMERY, KOREKCIA OKLÚZÍ A SKRESLENIA

3.1 Korekcia skreslenia

Dvoma hlavnými príčinami skreslenia je posunutie osi otáčania mimo principal point kamery a samotné vlastnosti kamery. Z tohto dôvodu je pre korekciu týchto nedostatkov potrebné získať vonkajšie a vnútorné parametre kamery.

3.1.1 Hľadanie osi kamery

Veľmi výhodné je pokúsiť sa nastaviť kameru tak, aby os otáčania prechádzala jej principal pointom. Principal point je bod, ktorý sa premieta bez skreslenia. Pre zjednodušenie možno predpokladať, že principal point je umiestnený v strede plochy, ktorú kamera sníma.



Obr. 3.1 Kalibračný snímok pre softvérové určovanie osi. Pri určovaní osi sa riadi pomocou ťažidla zaveseného na drôte, ktorý smeruje kolmo na stred rotačnej plošiny.

Namierenie kamery na stred osi otáčania je riešené pomocou umiestnenia na spodnej strane zaťaženého drôtu nad os otáčania. Týmto sa získa dlhá zvislá čiara, ktorá je osvietená laserovou stopou. Na kamere je možné aktivovať pomocnú mriežku. Kamera sa potom nastaví tak, aby celý drôt umiestnený v osi otáčania bol zakrytý strednou vertikálnou čiarou mriežky kamery.

Os otáčania vo videu možno korigovať aj dodatočne, softvérovo. Nastavovanie osi otáčania je súčasťou grafického prostredia (GUI) obslužného programu 3D skeneru. V kalibračnom náhľade GUI je umiestnený pre každú kameru snímok s drôtom v osi otáčania a súčasne snímok modelu z každej štvrtiny otáčania. Po prepnutí zobrazenia na snímok s drôtom možno nastaviť príslušným tlačidlom nastavovanie osi. Následne je treba kliknúť na zobrazenom snímku na príslušné miesto, kde by mala byť os otáčania. Zaškrtnutím check buttonu je potom možné zapnúť zobrazovanie tejto osi na všetkých snímkach a skontrolovať si správnosť jej umiestnenia. Kalibračná snímka pre nastavovanie osi je zobrazená na Obr. 3.1.

3.1.2 Výpočet uhlov medzi kamerami a laserom

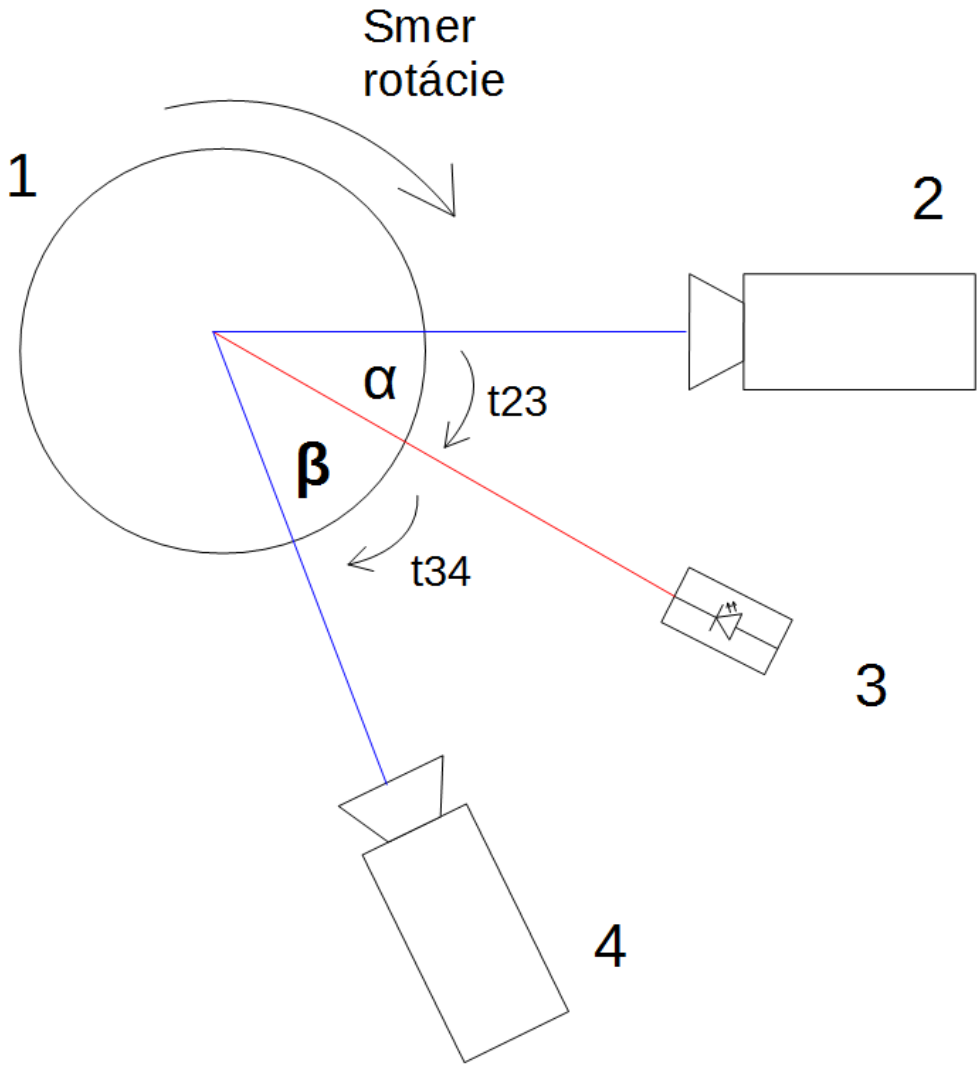
Kľúčovým vonkajším parametrom kamery je uhol pod ktorým daná kamera laserovú stopu sníma. Výpočtu tohto uhlu musí nutne predchádzať časová synchronizácia videa a hľadanie osi kamery.

Výpočet možno dosiahnuť meraním doby za ktorú sa značka na kalibračnom telese prejde z principal pointu jednej kamery cez laserovú stopu do principal pointu druhej kamery.

Pre tento druh kalibrácie sa využíva kalibračný valec. Tento valec je položený na stred rotačnej plošiny. Valec je oblepený bielym papierom a v jednom bode je vyrezaná diera v ktorej je umiestnená zelená LED dióda. Nad a pod touto diódou sú ďalej umiestnené značky, ktoré sa využívajú pre ďalšiu kalibráciu. Umiesteniu valca prípadne ešte predchádza umiestnenie skenovaného telesa na plošinu a nastavenie aparatúry podľa požiadaviek. Po tom ako umiestnime valec na plošinu sa už s aparatúrou nehýbe. Natočeniu videa so skenovaným objektom teda predchádza ešte natočenie videa s kalibračným valcom.

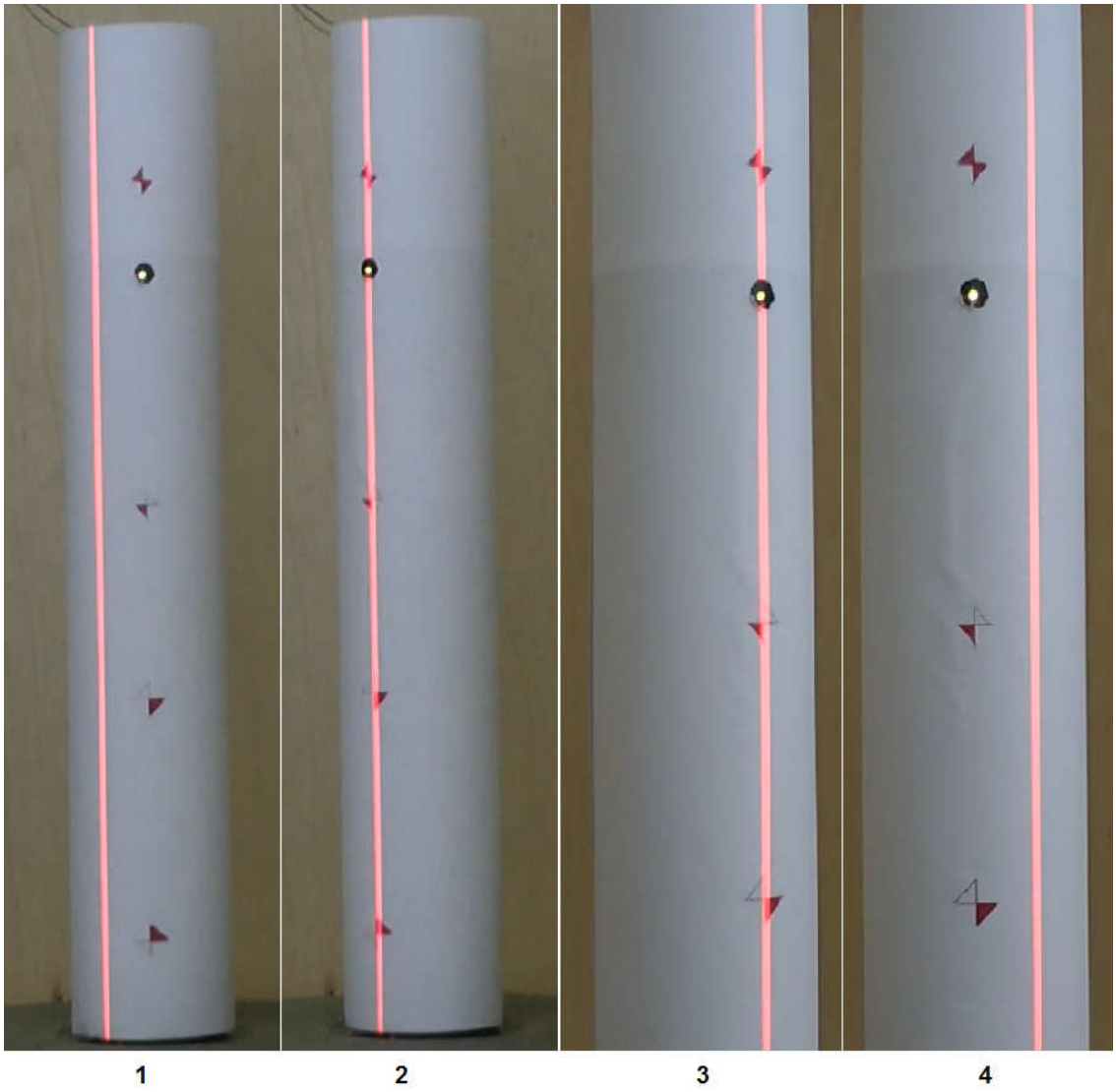
Kamery sú najskôr zosynchronizované pomocou zvukovej stopy. Poloha diódy je zo snímku zisťovaná pomocou farebnej segmentácie. Sledovaný je prechod stredu vysegmentovanej diódy cez určité medzné body.

Ďalším krokom je stanovenie medzných bodov kamier a laseru. Pre kamery sa sleduje horizontálny stred obrazovky, teda oblasť v ktorej by sa mal nachádzať principal point kamery. Pre laser sa segmentuje zo snímkov okrem zelenej diódy aj červená laserová stopa a sleduje sa prienik týchto dvoch prvkov. Ako prvý sa teda deteguje prechod ohniskom pravej kamery, nasleduje prechod laserom a posledný detegovaný medzný bod je prechod ohniskom ľavej kamery. Princíp detegovania bodov je znázornený na Obr. 3.2.



Obr. 3.2 Proces detegovania kľúčových snímkov. 1 – rotačná plošina s kalibračným valcom, 2 – pravá kamera, 3 – laser, 4 – ľavá kamera, α – uhol medzi pravou kamerou a laserom, t23 – čas medzi nasnímaním LED v principal pointe pravej kamery a nasnímaním jej pretnutia s laserom. β - uhol medzi ľavou kamerou a laserom, t34 – čas medzi nasnímaním pretnutia LED s laserom a jej nasnímaním v principal pointe ľavej kamery.

Následne sú zaznamenané čísla snímkov pri ktorých k daným prienikom došlo. Snímky v ktorých dochádza k prieniku s laserovou stopou sú navyše uložené ako ďalší výstupný parameter funkcie, pretože sú potrebné pre ďalšiu kalibráciu a sú zobrazené v GUI. Snímky, ktoré sú softvérom detegované sú zobrazené na Obr. 3.3.



Obr. 3.3 Kľúčové polohy LED pri zisťovaní vzájomnej polohy kamier a laseru. 1 – LED je v principal pointe pravej kamery, 2 a 3 – LED sa na pravej, respektíve ľavej kamere pretína s laserom (poloha laseru), 4 – LED je v principal pointe ľavej kamery

Keď sú známe poradové čísla snímkov prieniku diódy a jednotlivých medzných bodov je možné určiť uhol, ktorý tieto body spolu zvierajú. Výstupom funkcie, slúžiacej na zvukovú synchronizáciu je aj počet snímkov na jednu rotáciu telesa. Keď je známy tento údaj a takisto počet snímkov za ktorý prejde dióda z jedného medzného bodu do druhého, tak uhol medzi týmito medznými bodmi je možné dopočítať trojčlenkou.

Pre dosiahnutie čo najefektívnejšieho času segmentácie nie sú sledované všetky snímky, ale po nájdení bodu sa konverguje k správnemu snímku pripočítaním rozdielu medzi medzným bodom a diódou v pixeloch a vynásobením tejto hodnoty rozlíšením kamery, ktoré sa určilo na začiatku.

3.2 Aplikácia druhej Kamery

3.2.1 Stotožnenie obrazu kamier

Ďalšia kalibrácia je potrebná pre redukciu oklúzií. Toho možno docieliť zavedením druhej kamery. Druhá kamera je umiestnená z opačnej strany lasera a vo väčšej diaľke. Pomocou zvukovej synchronizácie je obraz z oboch kamier dokonale synchronizovaný a tak v horizontálnej rovine nie je problém zaistiť snímanie tých istých bodov oboma kamerami súčasne. Obe kamery sú centrované na os otáčania, výškovo si však jednotlivé body nemusia zodpovedať. Rozdiel môže byť spôsobený tým, že sú kamery vo vertikálnej rovine vzájomne posunuté a takisto aj tým, že kamery sú rôzne priblížené. Pokiaľ je teda treba zobraziť takto zábery z dvoch kamier a porovnať výsledok renderovania tak nastane jav, že vykreslené modely sa neprekrývajú.

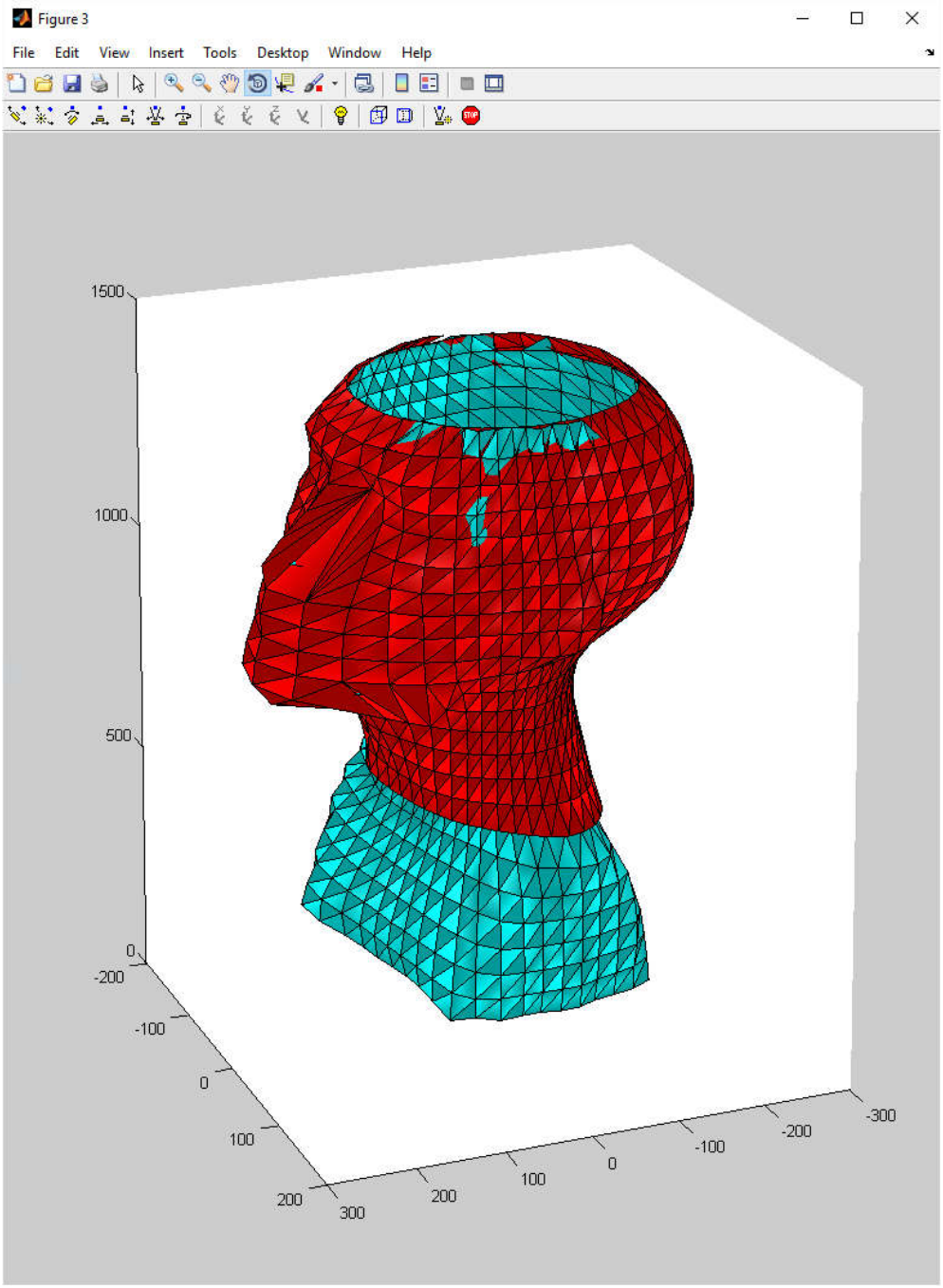
Z tohto dôvodu sa zavádza vertikálna kalibrácia. Táto vertikálna kalibrácia funguje pomocou štyroch pridaných značiek na kalibračnom valci, z ktorých jedna je umiestnená nad detegovanou diódou a tri pod ňou. Pomocou týchto značiek sa dá odčítať dynamický rozsah jednej a druhej kamery a takisto vertikálny posuv medzi oboma kamerami.

Značky sa sledujú v čase, keď dióda prechádza laserovou stopou, teda v momente, ktorý je jednoznačne detegovateľný súčasne oboma kamerami. Snímka, na ktorej je detegovaný priesečník diódy s laserovou stopou sa uloží do pamäti. Následne je možné v jednom z kalibračných funkcií GUI manuálne nastaviť pomocou funkcie ginput hornú a dolnú medzu pre jednu kameru a pre druhú kameru. Detegované body sú uložené do premennej *CalibHeight*, ktorá je typu cell. Ďalšie spracovanie je zahrnuté do renderovacej funkcie.

Na začiatku sú z premennej *CalibHeight* porovnané dynamické rozsahy dvojíc bodov zodpovedajúcich jednej a druhej kamere. Kamera, ktorá má menší dynamický rozsah sa vyberie ako prvá. Kalibračný snímok tejto kamery sa preto pomocou funkcie zväčší tak, aby si dynamické rozsahy oboch kamier zodpovedali.

Následne sa sleduje vertikálny posuv kamier. Sledovaným bodom je kalibračná dioda, ktorej poloha sa dá pomocou farebnej segmentácie ľahko zistiť. Polohu pre obe kamery porovnáme. V prípade, že kamera, ktorej dynamický rozsah bol zvyšovaný je vertikálne nižšie umiestnená, tak sú prvé riadky z obrazu vymazané, tak aby si vertikálne obe kamery zodpovedali. V prípade, že kamera, ktorej dynamický rozsah bol zvyšovaný je vertikálne vyššie umiestnená, treba pre vertikálne vyrovnanie kamier na začiatok matice obrazu tejto kamery vložíť nulové riadky.

Následne prebieha samotné renderovanie obrazu z tejto kamery s tým, že každý spracovávaný snímok je upravený zistenými parametrami tak, aby zodpovedal snímkam z druhej kamery. Potom prebehne renderovanie obrazu z druhej kamery bez zmeny akýchkoľvek parametrov. Výsledkom je že obrazy kamier si vo vertikálnom smere zodpovedajú, v horizontálnej rovine je však obraz jednej kamery stále viac roztiahnutý ako obraz druhej kamery. To je dané sčasti vlastnosťami kamery a sčasti rozdielnym uhlom natočenia voči laserovej stope. Preto je potrebná ďalšia kalibrácia. Výsledok po tejto kalibrácií je zobrazený na Obr. 3.4.



Obr. 3.4 Vyrenderovaný model z pravej (červená farba) a ľavej (tyrkysová farba) kamery po stotožnení obrazu kamier.

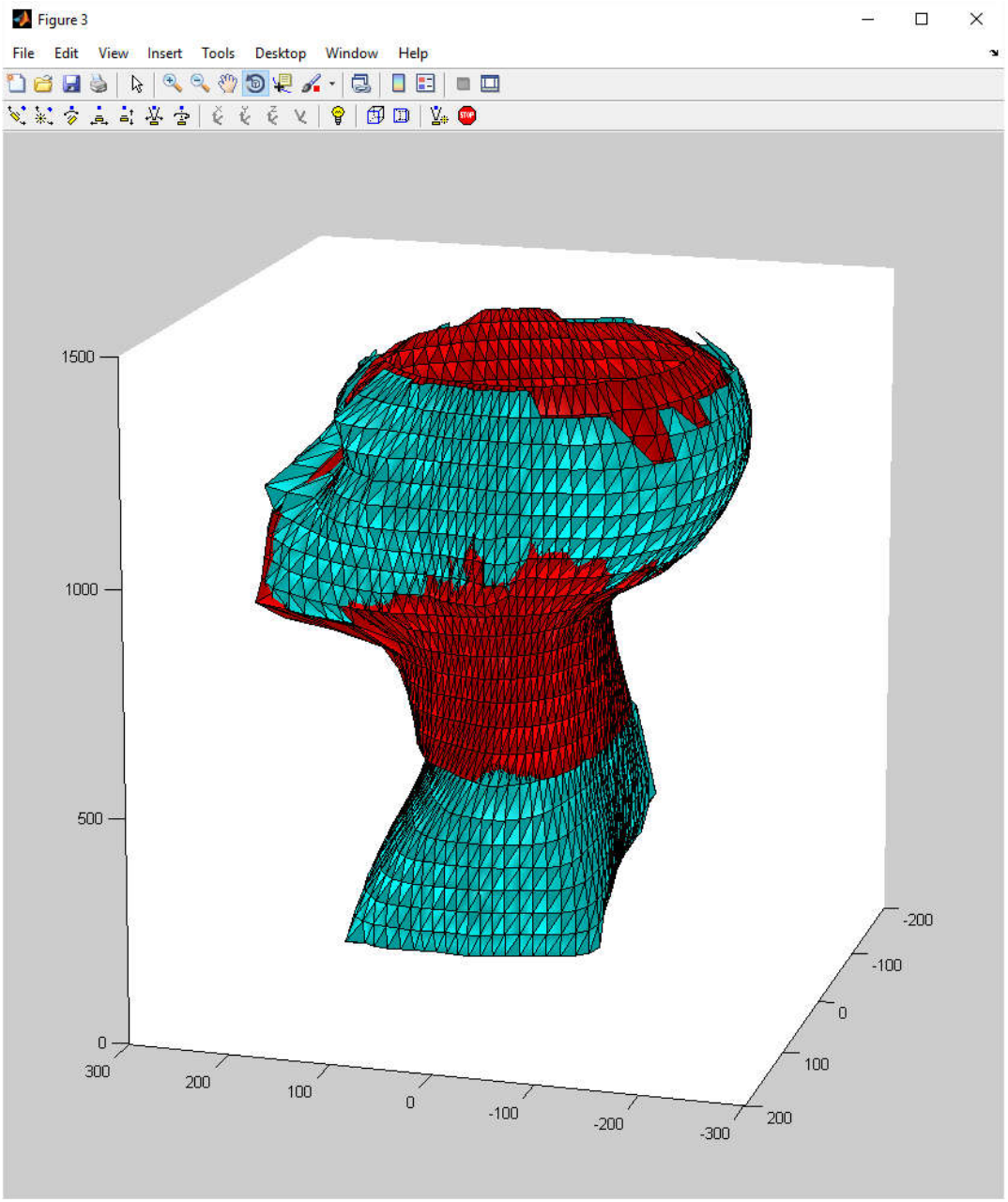
3.2.2 Dodatočná kalibrácia v horizontálnej rovine zobrazenia

Pre realizáciu tejto kalibrácie by bolo potrebné poznať vlastnosti kamier, čo je síce možné docieliť, ale v tejto bakalárskej práci to zahrnuté nie je. V tejto práci je preto aspoň experimentálne ukázané ako sa obrazy oboch kamier prekrývajú.

Tento proces nadväzuje na stotožnenie obrazu kamier popísané v podkapitole 3.1.1. Renderovanie obrazu z prvej kamery, teda kamery s menším dynamickým rozsahom

prebieha bezo zmeny. U druhej kamery nastáva zmena po získaní bodov pre renderovanie z vysegmentovaných snímok. Matica obsahujúca tieto informácie má 3 rozmery. V prvých dvoch je popísané rozmiestnenie bodov a tretí rozmer hovorí či sa jedná o výšku bodu, jeho vzdialenosť od osi otáčania, alebo o jeho natočenie voči počiatku. V popísaných úpravách matice sa upravuje iba vzdialenosť osi od otáčania.

Po tom čo sú známe zodpovedajúce si body pre renderovanie, nasleduje porovnanie plochy, ktorú vzhľadom na os otáčania pokrývajú. Tieto plochy predstavuje suma vzdialeností jednotlivých bodov kamery od osi otáčania v danom reze.



Obr. 3.5 Vyrenderované modely ľavej a pravej kamery po dodatočnej kalibrácií v horizontálnej rovine zobrazenia.

Vzhľadom na to, že každá kamera sníma inú časť tak sa porovnáva plocha iba

v časti, ktorú snímajú obe kamery. Toho je docielené tým, že sa využíva rozmeru matice, ktorá zodpovedá menšiemu modelu. Pre redukciu chýb spôsobených prípadnými oklúziami sa berú do úvahy len body, ktoré nemajú nulovú hodnotu. Súčet vzdialeností všetkých bodov je podelený počtom nenulových bodov.

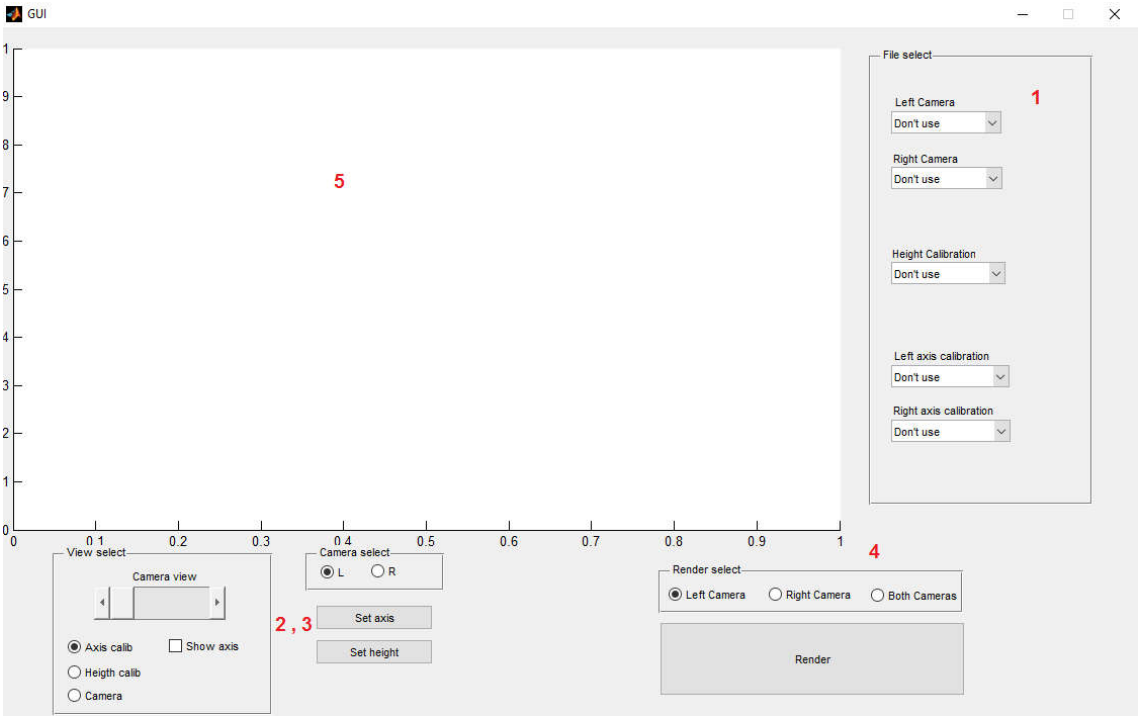
Následne sú všetky prvky v danom reze vynásobené podielom plochy prvej kamery ku ploche druhej kamery a ďalej je model z druhej kamery vyrenderovaný. Body si nezodpovedajú presne. Vo vzdialenejších bodoch od osi je stále model druhej kamery širší, v bodoch bližšie k ose otáčania je naopak užší. Celkový dojem je však o poznanie lepší ako bez tejto kalibrácie. Výsledok renderovania je vyobrazený na Obr. 3.5.

U kalibračného valca je známa presná vzdialenosť medzi jednotlivými značkami a tak by bolo možné v budúcnosti určiť aj presné fyzické rozmery skenovaného objektu. K tomuto by bolo však okrem tohto presného rozmeru bolo opäť potrebné poznať aj vlastnosti kamery.

4 GRAFICKÉ PROSTREDIE

4.1 Ovládanie GUI

Grafické prostredie (Graphical user interface - GUI) slúži pre zjednodušenie ovládania pre verejnosť a pre plnú automatizáciu celého procesu skenovania. GUI obslužného programu ku skeneru pozostáva z jedného okna. Najväčšiu časť tohto okna zaberá zobrazovacia plocha určená pre zobrazovanie snímkov z nasnímaných video súborov. Zvyšná časť je rozdelená na niekoľko ovládacích panelov. Rozloženie GUI je zobrazené na obr. 4.1.



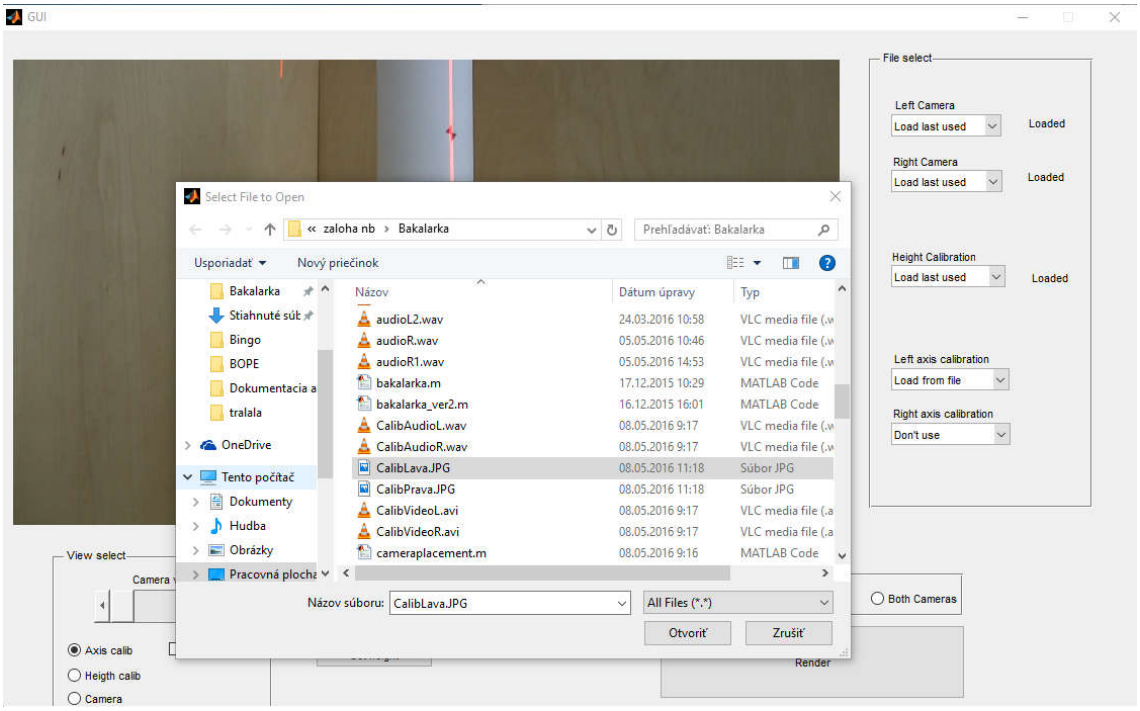
Obr. 4.1 Okno GUI po spustení programu. 1 – Panel pre výber súborov, 2 – Panel pre výber pohľadu, 3 – Panel pre výber Kamery a talčidlá pre nastavovanie kalibrácí, 4 – panel pre výber renderovania.

Prvý panel, ktorý v programe užívateľ využíva je panel *File select*. Panel je umiestnený v pravej časti okna. V tomto paneli užívateľ zvolí, aké dáta chce načítať, podľa toho, ako plánuje skener využiť. Jedná sa o videlo z ľavej kamery, video z pravej kamery, videá s kalibračným valcom slúžiace pre výškovú kalibráciu a synchronizáciu kamier, kalibračný snímok pre stanovenie osi ľavej kamery a kalibračný snímok pre stanovenie osi pravej kamery.

Pre renderovanie z jednej kamery stačí video so skenovaným objektom z danej kamery. Pre použitie renderovania z oboch kamier treba použiť skenované videá z oboch kamier tak so skenovaným objektom, ako aj s kalibračným valcom. Ak je to žiadúce, tak je možné pre zlepšenie výsledku takisto využiť snímky slúžiace na

kalibráciu osi.

Pre každý prvok je implementované popup menu, nad ktorým je popis, pre ktorý prvok slúži. V popup menu je pôvodne nastavená možnosť „*Don't use*“ a teda nie je vybraný žiaden video súbor. Ďalšou možnosťou je „*Load from file*“, po zvolení ktorej sa otvorí okno, v ktorom je možné vybrať požadovaný súbor z kamery, ktorý chce užívateľ použiť. Zvolený súbor sa následne načíta a program pracuje s audio a video stopou, ktorú z neho vytvorí. Treťou možnosťou je „*Load last used*“, ktorá slúži na použitie audio a video súboru využitého pri poslednom použití programu. Vedľa popup menu sa po začatí načítavania zobrazí text „*Loading*“ a po dokončení načítavania sa prepíše na „*Loaded*“. V prvom okne sa vždy po načítaní niektorého z prvkov zobrazí na zobrazovacej ploche snímok z tohto prvku. Priebeh načítavania dát je zobrazený na Obr. 4.2.



Obr. 4.2 Priebeh načítavania dát. Pre prvé 3 možnosti boli využité minule použité súbory, pre kalibráciu osi ľavej kamery bola zvolená možnosť vybrať nový súbor. Na zobrazovacej ploche je posledný načítaný snímok z posledného načítaného prvku, výškovej kalibrácie.

Druhý a tretí panel sú v spodnej časti sú umiestnené v spodnej ľavej časti okna. Tieto panely sa nazývajú *Camera select a View select.* Panely slúžia na zmenu snímku zobrazovaného na zobrazovacej ploche a na nastavovanie kalibrácií.

Panel *Camera select* obsahuje dva radiobuttony *L* a *R*. Tento panel slúži na výber kamery, ktorá sa má zobrazovať. Zaškrtnutím *L* sa vyberie ľavá kamera, zaškrtnutím *R* pravá kamera.

Panel *View select*, obsahuje slider *Camera view*, radiobuttony *Axis calib, Height calib* a *Camera* a checkbutton *Show axis*. Výberom medzi radiobuttonmi sa volí medzi zobrazením pre nastavenie osi, zobrazením pre nastavenie výškovej kalibrácie pre stotožnenie obrazu kamier a zobrazením záberov skenovaného telesa z kamery. Pre

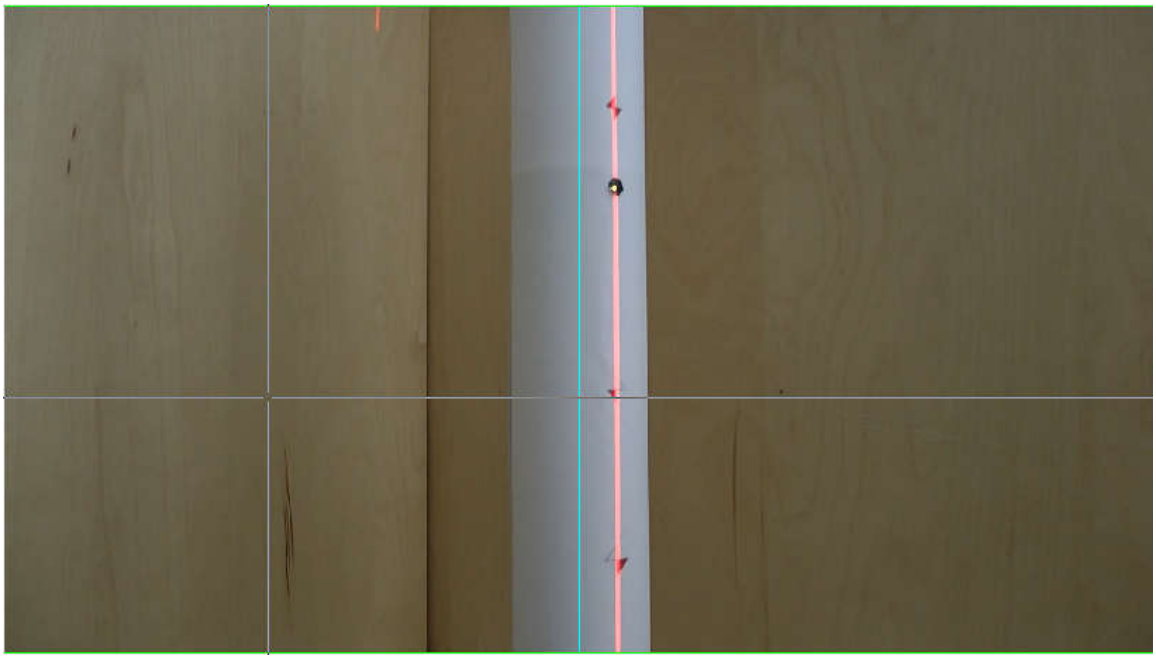
radiobutton *Camera,* teda zobrazenie skenovaného telesa z kamery, je možné využiť taktiež slider *Camera view*, ktorý má 4 polohy a slúži na natočenie skenovaného telesa zo 4 rôznych strán. Checkbutton *Show axis* slúži na zobrazenie nastavených kalibračných hodnôt, tak aby ich bolo možné vidieť na všetkých snímkach.

Posledným, štvrtým z panelov je panel *Render select*. Tento panel je umiestnený v pravej spodnej časti okna. Panel obsahuje 3 radiobuttony na výber požadovaného vstupu renderovania, teda ľavej kamery, pravej kamery, alebo oboch kamier. Po vybraní požadovaného vstupu užívateľ klikne na tlačidlo *Render* umiestnené pod týmto panelom, čím spustí renderovanie modelu.

4.1.1 Nastavenie kalibrácí

Súčasťou GUI je aj stanovenie osi otáčania. Na to slúži pushbutton *Set Axis*. Po stlačení tohto tlačila sa klikne na miesto na obrázku v ktorom sa požaduje nastaviť os otáčania. Pre presné nastavenie by sa teda mal zobraziť snímok kalibrácie osi zaškrtnutím radiobuttonu *Axis calib*, následne je potrebné kliknúť na *Set axis* a potom na obrázok v mieste kde laserová stopa osvecuje zvislý drôt. Pomocou checkbuttonu *Show axis* je možné túto stanovenú os zobraziť na kalibračných snímkach i na snímkach z kamery. Rovnakým spôsobom je možné nastaviť túto kalibráciu pre obe kamery, medzi kamerami sa prepína pomocou tlačidiel v paneli *Camera select.*

Obdobným spôsobom sa nastavuje aj výšková kalibrácia slúžiaca na stotožnenie obrazu kamier. Nastavovanie výškovej kalibrácie je zobrazené na obr. 4.3.



Obr. 4.3 Printscreen zobrazovacej časti okna v ktorom prebieha nastavovanie výškovej kalibrácie. V programe je zaškrtnutý checkbutton show axis. Zvislá tyrkysová čiara je nastavená os, vodorovné zelené čiary na okrajoch obrázka sú pôvodne nastavené hodnoty výškovej kalibrácie. Šedý kríž ukazuje aktuálnu polohu kurzora myši.

Pre nastavenie tejto kalibrácie sa odporúča využiť snímok s kalibračným valcom,

ktorého zobrazenie sa prepína radiobuttonom *Height calib* v paneli *View select.* Odporúča sa začať nastavovaním kamery s väčším dynamickým rozsahom, pretože na nej nemusia byť zobrazené všetky značky. Samotné body sa nastavujú po kliknutí na pushbutton *Set height*. Kliknutím na toto tlačidlo sa nápis na tomto tlačidle prepíše na *„Top point“* a zmení farbu na zelenú. Je teda potrebné zadať výškovú súradnicu horného bodu kliknutím na obrázok. Po prvom kliknutí sa nápis tlačidla opäť zmení, tentokrát na *„Bottom point*“ a je treba zadať rovnakým spôsobom výškovú súradnicu spodného bodu. Rovnako sa nastavia súradnice pre druhú kameru. Tak ako u kalibrácie osi je aj u tejto kalibrácie možné zobraziť nastavené hodnoty pomocou checkbuttonu *Show axis*.

4.2 Princíp fungovania GUI

Po spustení programu GUI nasleduje načítanie grafických prvkov, teda všetkých tlačidiel a bielej plochy určenej pre zobrazovanie.

4.2.1 Vzájomná komunikácia prvkov GUI

Každý grafický prvok funguje ako samostatná funkcia a tak je potrebné zabezpečiť, aby tieto funkcie mohli spolu komunikovať. Každý grafický prvok je v GUI charakterizovaný pomocou štruktúry *handles*. Druhou úrovňou štruktúry je názov konkrétneho grafického prvku a treťou úrovňou je niektorý z parametrov štruktúry, ktorý obsahuje nejaké dáta. Tieto parametre môžu byť napríklad súradnice polohy, rozmery, zobrazovaný textový reťazec, hodnota a mnohé iné. Jedným z parametrov je parameter *UserData*, ktorý má tvar štruktúry a je ponechaný na zapisovanie ľubovoľných dát pre užívateľa. Parameter *UserData* objektu *ShowVideo* bol teda vybraný pre účel uchovávania dát zo všetkých funkcií tak aby mohli vzájomne komunikovať.

Na začiatku každej funkcie je teda použitý príkaz *get*, ktorý zoberie dáta z danej štruktúry a uloží ich do premennej *mojedata*. S touto premennou sa v každej funkcií pracuje a ukladajú sa sem ďalšie dáta. Na konci funkcie je pomocou príkazu *set* aktualizovaná premenná *mojedata* uložená do danej štruktúry . Časť kódu zobrazujúca prenášanie dát medzi funkciami je zobrazená na obr.

4.2.2 Inicializácia dát

Dáta sú inicializované pomocou vedľajšieho GUI *GUIinit*, ktoré je schopné komunikovať s hlavným GUI. Dáta je možné inicializovať jednotlivo, pokiaľ teda užívateľ chce preskočiť všetky kalibrácie, je možné načítať iba skenovaný obraz z kamier.

Súčasťou inicializácie je aj možnosť vybrať súbory, ktoré sa majú do požadovaných premenných uložiť. Ak je v paneli slúžiacom na nahrávanie dát vybraná možnosť „*Load last used“*, tak sa vykoná zvuková synchronizácia s videom a audiom použitým pri poslednom skenovaní. Pokiaľ je vybraná možnosť „*Load form file“*, tak sa ešte predtým pomocou príkazu *system* cez *ffmpeg* vytvorí nový audio a video súbor pre spracovanie. Pre výber súboru a uloženie jeho mena ako reťazca do premennej sa využíva funkcia *filename*. Pre použitie názvu súboru v príkaze *system* je do príkazu

integrovaný príkaz *sprintf* a na požadované miesto je doplnený reťazec z názvom programu. Nasleduje prepísanie pôvodného video a audio súboru, ktoré vďaka pridaniu direktívy „*-y*“ do reťazca použitého pre *ffmpeg* nie je potrebné potvrdzovať z konzoly, ale je potvrdené automaticky. Časť kódu, ktorá má túto funkciu vyzerá nasledovne:

```
switch str
    case 'Load from file'
        filename = uigetfile;
        system(sprintf('ffmpeg -y -i %s -ss 00:00:00.0 -t 00:00:30.0
        vcodec libx264 -crf 10 -acodec ac3 -vf "yadif"
        ScanVideoR.avi',filename));
        system(sprintf('ffmpeg -y -i %s -ss 00:00:00.0 -t 00:00:30.0
        ScanAudioR.wav',filename));
end
```

Po vykonaní časovej synchronizácie sa vybrané parametre uložia do štruktúry a na zobrazovacej ploche sa vykreslí aktuálne načítaný snímok a to nezávisle od toho čo je radiobuttonmi a sliderom na prepínanie snímkov zvolené.

Systém funguje rovnako pre načítávanie všetých dát, iba pri načítavaní výškovej kalibrácie sa načítava obraz z oboch kamier a následne sa počíta výšková kalibrácia. Z tohto dôvodu trvá načítanie mierne dlhšie ako v ostatných prípadoch.

U všetkých premenných je prvým prvkom v poslednom stupni štruktúry údaj pre ľavú kameru a druhým prvkom údaj pre pravú kameru. Obrázky z kamery sa ukladajú do štruktúrovanej premennej *handles.ShowVideo.img*, ktorá obsahuje 8 prvkov, pričom prvé 4 zodpovedajú ľavej kamere a zvyšné 4 pravej. Pre kalibračné zábery v štruktúrovanej premennej *handles.ShowVideo.imgcal* sú dáta ukladané tak, že prvé 2 prvky sú osová kalibrácia z ľavej a pravej kamery a zvyšne 2 sú výšková kalibrácia z ľavej a pravej kamery.

4.2.3 Výber Obrázku pre vykresľovanie

Pre vždy aktuálne vykreslený obrázok, podľa navolených hodnôt na radiobuttonoch a slideri je potrebné v každej callback funkcii zistiť, ktorý z obrázkov je aktuálne zvolený. Tento výber je zabezpečený pomocou série switchov.

Pomocou prvého switchu program zistí či sa jedná o ľavú, alebo pravú kameru, čím priradí aj číselnú hodnotu a vie, ktoré premenné má používať. Nasleduje switch, ktorý sleduje, ktorý z trojice radiobuttonov v poli *View select* je zaškrtnutý. Slider už následne nevyužíva switch ale jeho hodnota je získavaná podobným postupom.

V prípade switchov aj v prípade zistenia hodnoty slideru sa sleduje pomocou funkcie *get* niektorý z parametrov štruktúry daného grafického prvku. U radiobuttonov zoskupených do polí tlačidiel sú použité dve funkcie *get* vnorené do seba.

Vnútorná funkcia pomocou parametru *SelectedObject* zistí, ktorý radiobutton je zaškrtnutý. Následne vonkajšia funkcia get prečíta pomocou parametru *Tag* názov tohto tlačidla v závislosti od ktorého potom switch začne vykonávať funkciu.

U slideru sa využíva iba jedna funkcia *get*, ktorá sleduje parameter *Value*, zaokrúhlený na celé číslo. Nadobúda hodnoty 0 až 3, teda 4 hodnoty pre každú kameru.

Po tom čo je známy vybraný prvok tak je príslušný obrázok vykreslený pomocou

funkcie *imshow*.

4.2.4 Nastavenie osi a výškovej kalibrácie

Pre tento účel slúžia pushbuttony umiestené v strede pod zobrazovacou plochou. Po kliknutí na tlačidlo je potrebné získať dáta kliknutím na zobrazovaný obrázok.

Pri nastavení osi otáčania aj výškovej kalibrácie je najskôr spôsobom popísaným v podkapitole 4.2.3 zistené či sa jedná o pravú alebo ľavú kameru. Potom je pomocou funkcie *ginput* zistená požadovaná súradnica z bodu ktorý bol kliknutím zvolený.

Pri nastavovaní osi otáčania je pomocou funkcie *ginput* zistený jeden bod. Ten je uložený do štruktúrovanej premennej *handles.ShowVideo* na miesto prislúchajúce danej kamere a následne je vykreslená funkciou *line* zvislá čiara s touto získanou súradnicou v ose x.

Pri nastavovaní výškovej kalibrácie sú pomocou funkcie *ginput* získavané 2 body. Pri zadávaní prvého bodu sa prepíše nápis na tlačidle v GUI pomocou príkazu *set* na *„Top point“*. Ako prvý sa teda zadáva horný bod a po jeho zadaní sa obdobným spôsobom prepíše text na tlačidle na *„Bottom point“* a zadáva sa druhý bod. Rovnakým spôsobom ako u kalibrácie osi sú súradnice uložené do štruktúrovanej premennej *handles.ShowVideo* a pomocou funkcie *line* sú vykreslené tentokrát vodorovné čiary s prislúchajúcou súradnicou v ose y.

4.2.5 Výber vykresľovania osí

Pomocou checkbuttonu *Show axis* je možné zobraziť kalibračné priamky na ktoromkoľvek obrázku.

Za týmto účelom bol do každej rovnice, ktorá vykresľuje nejaký obrázok pridaný príkaz *if*, ktorý sleduje pomocou funkcie *get* hodnotu parametru *Value*, ktorá vraví či je stlačený alebo nie. Pokiaľ je hodnota *Value* v stave 1, tak po vykreslení samotného snímku sú spôsobom popísaným v podkapitole 4.2.4 do neho vykreslené kalibračné čiary.

4.2.6 Spustenie 3D skenovania a renderovanie

Posledným bodom v celom procese je spustenie hlavného programu skenovania pomocou pushbuttonu *Render*.

Užívateľ má na výber, či chce spraviť sken z ľavej kamery, pravej kamery, alebo oboch kamier súčasne. Sken z jednej kamery má výhodu, že je časovo a dátovo menej náročný. Takémuto skenu stačí nahrať jediné video a síce video s nasnímaným telesom z danej kamery. Druhou možnosťou je využitie oboch kamier, vďaka ktorému sa redukujú oklúzie a tak je video kvalitnejšie. Takýto sken potrebuje nahrať videá z oboch kamier a video s kalibračným valcom.

Keďže skenovanie je proces, ktorý zaberá viac času, tak je po kliknutí na tlačidlo pomocou funkcie set zmenený text pushbuttonu na *„In process“,* aby užívateľ vedel že program pracuje. Následne je pomocou funkcie *Renderovacia_funkcia*, ktorý je dôkladne popísaný v kapitole 2 spustený program na spracovanie dát a vytvorenie 3D modelu. Pokiľ je zvolená možnosť použitia len jednej z kamier tak sa používa

zjednodušená verzia funkcie *Renderovacia_funkcia.*

Funkcia *Renderovacia_funkcia* obsahuje mnoho vstupných premenných medzi ktorými sú zahrnuté údaje získané synchronizáciu, kalibráciami a samotné nasnímané dáta.

5 ZÁVER

Úlohou tejto práce bolo zhotoviť prototyp rotačného 3D skeneru. Boli tu predstavené spôsoby, ktorými možno realizovať jednotlivé kroky procesu skenovania objektu. Z nich boli vybrané tie najlepšie aplikovateľné a bol vytvorený funkčný prototyp skeneru, ktorý sa dá po nasnímaní video záznamu a jeho prenose do PC spustiť cez program MATLAB. Systém je plne ovládateľný pomocou GUI a to od načítania dát, cez nastavovanie kalibrácií, až po vyrenderovanie 3D modelu.

Po natáčaní jednou kamerou boli vo výslednom modeli pri akomkoľvek natočení kamery a laseru oklúzie spôsobené tým, že kamera nebola schopná laserovú stopu zachytiť úplne vždy. Z toho dôvodu bola zavedená druhá kamera a kamery boli spolu zosynchronizované a stotožnené, aby bol dosiahnutý čo najpresnejší výsledok.

Z dôvodu neznámych parametrov kamery nebolo možné zabezpečiť prekrývanie kamier úplne dokonale. Parametre kamery však možné získať je. Získanie týchto parametrov a zabezpečenie dokonalého prekrývania objektu z jednej a druhej kamery je teda základom prípadného budúceho zdokonalenia tohto projektu. Pokiaľ by boli známe parametre kamery, tak by bolo možné za pomoci kalibračného valca určiť aj presné fyzické rozmery skenovaných objektov.

LITERATURA

[1] N A BORGHESE, G FERRIGNO, G BARONI, S FERRARI a R SAVARÈ. Autoscan: A Flexible and Portable 3D Scanner. *IEEE Computer Graphics and Applications*. 1998, 38-41.

[2] OHNO, K, T KAWAHARA a S TADOKORO. Development of 3D Laser Scanner for Measuring Uniform and Dense 3D Shapes of Static Objects in Dynamic Environment. *Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics*. 2009, 2161-2167.

[3] *FFmpeg Documentation* [online]. Dostupné z: https://www.ffmpeg.org/ffmpeg.html

[4] MARQUES, O. *Practical image and video processing using MATLAB*. Wiley-IEEE Press; 1st edition (September 21, 2011), 696 s. ISBN 978-0470048153

[5] VERMA, V., WALIA, E. Rendering - Techniques and Challenges, *International Journal of Engineering and Technology* Vol.2(2), 2010, 29-33, Dostupné z: http://www.enggjournals.com/ijet/docs/IJET10-02-02-01.pdf

[6] SCHREER, O,, KAUFF, P., SIKORA, T. *3D Videocommunication: Algorithms, concepts and real-time systems in human centered communication.* John Wiley & sons Ltd, 2005, 340 s.. ISBN 978-0-470-02271-9

[7] DOBBINS, P. 3D Rendering in Computer Graphics. White Word Publications, 1st edition, 2012, ISBN 978-81-323-4264-9. Dostupné z: http://www.pdfiles.com/pdf/files/English/Designing_&_Graphics/3D_Rendering_In_Computer_Graphics.pdf

[8] KAUSHIK, M., NAGWANSHI, K. K., SHARMA, L. K., A Overview of Point-based Rendering Techniques, *International Journal of Computer Trends and Technology*- volume3, Issue1, 2012. Dostupné z: http://www.academia.edu/1275281/A_Overview_of_Point-based_Rendering_Techniques

[9] Čiarový kód [online]. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/f/f7/Barcode_reader.png

[10] JAVIDI, B., OKAMO, F., SON, J. *Three-Dimensional Imaging, Visualiyation and Display,* New York: Springer, 2009.

ZOZNAM SKRATIEK

DIBR	Depth image based rendering, metódy renderovania založené na obraze
RGB	Red-Green-Blue colour model, farebný model Červená-Zelená-Modrá
YCbCr	Luminance-Chromatic red-Chromatic bluecolour model, farebný model Jas-Chromatická červená-Chromatická modrá
LED	Light emmiting diode, dióda emitujúca svetlo
GUI	Graphical user interface, grafické užívateľské rozhranie