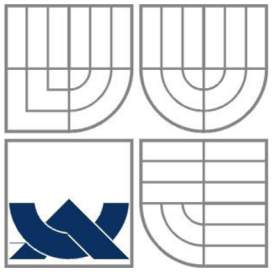
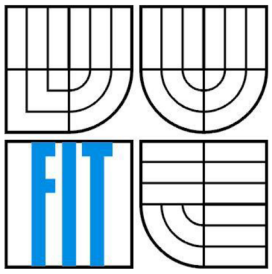


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

3D REKONSTRUKCE POHYBU LIDSKÉHO TĚLA (MOCAP)

3D Reconstruction of Human Body Motion (MOCAP)

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

FRANTIŠEK ČERNÁK

VEDOUCÍ PRÁCE
SUPERVISOR

ING. MAREK ŠOLONY

BRNO 2014

Abstrakt

Tato práce řeší rozpoznávaní značek v obrazu, zachytávání pohybu a jeho následné převedení do 3D souřadnic a 3D rekonstrukci. Zvolený problém byl vyřešen sestavením stereo kamerového systému, detekci značek na pohybujícím se objektu a získáním polohy objektu v prostoru pomocí triangulace. V práci se podařilo dosáhnout úspěšnosti detekce značek na objektu 95% na blízkou vzdálenost a rekonstrukce pohybu do určité míry odpovídá skutečnosti. Hlavním zjištěním této práce je, že pomocí dvou obyčejných webových kamer je možné zrekonstruovat pohyby těla.

Abstract

This thesis deals with object recognition in image, motion capturing and his transformation into 3D coordination system and 3D reconstruction. The problem was solved by building stereo camera system, detection of markers on a moving object and getting his position in space using triangulation. Hit rate of 95% was achieved in marker detection at close range and accuracy of 3D reconstruction answers to reality at certain extend. The main finding of this work is that it is possible to reconstruct body motions with a pair of simple web cameras.

Klíčová slova

kamerový model, stereo kamera, detekce kruhů, epipolární geometrie, triangulace, 3D rekonstrukce, zachytávaní pohybu

Keywords

Camera model, circle detection, epipolar geometry, triangulation, 3D reconstruction, motion capture

Citace

Černák František: 3D rekonštrukcia pohybu ľudského tela (MOCAP), bakalářská práce, Brno, FIT VUT v Brne, 2014

3D REKONŠTRUKCIA POHYBU ĽUDSKÉHO TELA (MOCAP)

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Mareka Šolonyho.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
František Černák
20.1.2014

Poďakovanie

Chcel by som poďakovať svojmu vedúcemu bakalárskej práce Ing. Marekovi Šolonymu za odborné vedenie pri vypracovávaní témy.

Obsah

Obsah 1
1 Úvod 2
1.1 Motion capture 2
2 Kamerový model 4
3 Epipolárna geometria 7
3.1 Fundamentálna matica 8
4 Houghove transformácie 10
4.1 Detekcia priamok 10
4.2 Detekcia kruhov 11
5 Predspracovanie obrazu 12
5.1 Vyhladenie obrazu Gaussovým filtrom 12
5.2 HSV farebný model 13
5.3 Morfologické operácie erózia a dilatácia 14
6 Sledovanie objektu v obraze 15
6.1 Optický tok 15
6.2 Kalmanov filter 16
7 Implementácia 19
7.1 Knižnica OpenCV 19
7.2 Stereo kalibrácia 19
7.3 Detekcia značiek 21
7.4 Sledovanie značiek pomocou Kalmanovho filtra 24
7.5 Triangulácia bodov 26
8 Testovanie 28
8.1 Pracovné nástroje 28
8.2 Úspešnosť detekcie značiek pri rôznej vzdialenosti 28
8.3 Úspešnosť vyhľadania korešpondenčných bodov 30
8.4 Presnosť triangulácie 31
8.5 Úspešnosť rekonštrukcie na základe bodov daných triangulóciou 33
9 Záver 35
Literatúra 36

1 Úvod

V modernom svete sa môžeme čoraz častejšie stretnúť s rýchlo sa rozvíjajúcimi počítačovými technológiami. Využívame ich v našom každodennom živote, možno aj bez toho aby sme si to často uvedomovali. To platí aj o počítačovej 3D grafike, rozpoznávaní objektov v obraze a sledovaní objektov.

Táto technológia sa využíva v mnohých užitočných oblastiach, ako aj v zábavnom priemysle. Príkladom môžu byť zachytávania ľudského pohybu napr. pri interaktívnych herných konzolách, pri natáčaní špeciálnych efektov vo filmoch, pri vedeckých výskumoch ohľadom ľudského tela, jeho fungovania a skúmaní jeho hraníc a možností.

Ale so zachytávaním pohybu je často úzko spätá aj jeho následná 3D rekonštrukcia. Aby sme to mohli dosiahnuť, potrebujeme kamerový stereo systém, základné informácie o našich kamerách (vzájomná poloha kamier voči sebe, vnútorné parametre kamier), algoritmy, ktoré nám umožnia rozoznať potrebné objekty v obrazoch, zistiť ich polohu v reálnom svete a následne objekty vykresliť pomocou zvolenej aplikácie.

Táto práca sa zaoberá predstavením knižnice OpenCV, algoritmami a funkciami používanými pre kalibráciu kamier, detekciu a sledovanie objektov a ľudského pohybu ako aj jeho následnou rekonštrukciou.

1.1 Motion capture

Pod pojmom motion capture rozumieme proces vzorkovania informácií o pohybe a lokácii subjektu v čase. Subjekt môže byť prístroj, ale aj osoba či zviera. V takomto prípade sa niekedy o ňom hovorí ako o hercovi. Pre motion capture sa najmä v zábavnom priemysle často používa skratka 'mocap'.

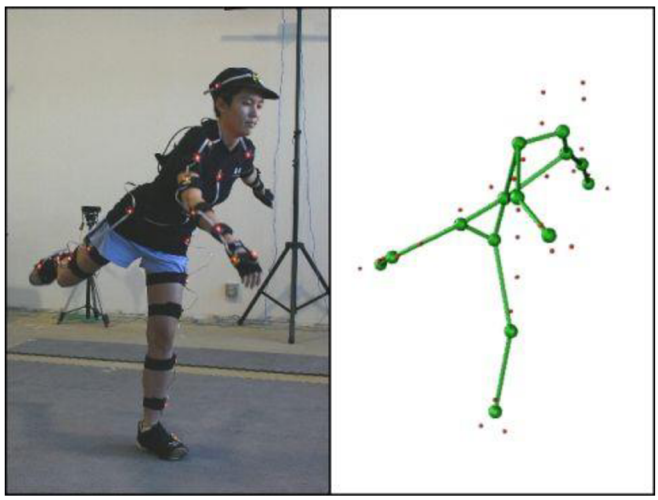
Technickým cieľom mocapu je získavanie pohybových dát z určitých bodov alebo značiek na subjekte. Z týchto dát sa ďalej získavajú parametre pohybu, ako napríklad rýchlosť, uhol alebo vzdialenosť. V takom prípade môže byť aplikácia analýza pohybu, športová analýza, biomechanika, biodynamika apod. V ďalšom prípade sa získané dáta môžu použiť na kontrolu, prípadne ovládanie iných zariadení. V praxi sa s takýmto využitím môžeme stretnúť v oblastiach ako sú teleoperácia(remote surgery), telerobotika, interaktívne hry, virtuálny tréning, virtuálna rehabilitácia či pohybom riadená hudba.

Poznáme viacero druhov zaznamenávania pohybu v závislosti na použitých značkách. Subjekt je buď snímaný bez značiek, alebo má značky pri kĺboch, aby bolo možné pohyb identifikovať pozíciami značiek alebo uhlami medzi nimi.

Prvé formy motion capture boli avšak vykonávané bez značiek. Zachytené obrazy potom analyzovali ľudia, neskôr sa pre analýzu vyvinuli rôzne softvéry. Táto technológia sa dnes stále používa, aj keď presnosť softwarového rozpoznávania bodov záujmu je pre mnoho aplikácií nedostatočná.

Jednými z najpoužívanejších druhov značiek v súčasnosti sú optické. Poznáme optické značky aktívne a pasívne. Aktívne značky vydávajú vlastné svetlo, ktoré je detekované kamerami. Príkladom aktívnych značiek sú napríklad LED značky. Pasívne značky nevydávajú vlastné svetlo ale odrážajú svetlo generované pri kamerových šošovkách. Preto sú pre lepšiu detekovateľnosť často vyrábané z odrazového materiálu. Prahová úroveň kamier môže byť nastavená tak, aby zachytávala iba svetlé značky a ignorovala kožu a látku či okolie. Technika pasívnych značiek bola použitá aj pri tejto práci.

Ďalším možným spôsobom označenia bodov záujmu môžu byť tzv. senzory. Tie môžu fungovať na princípe elektromagnetickom, gyro senzorov, akcelerometrov a na princípe optických vlákien.

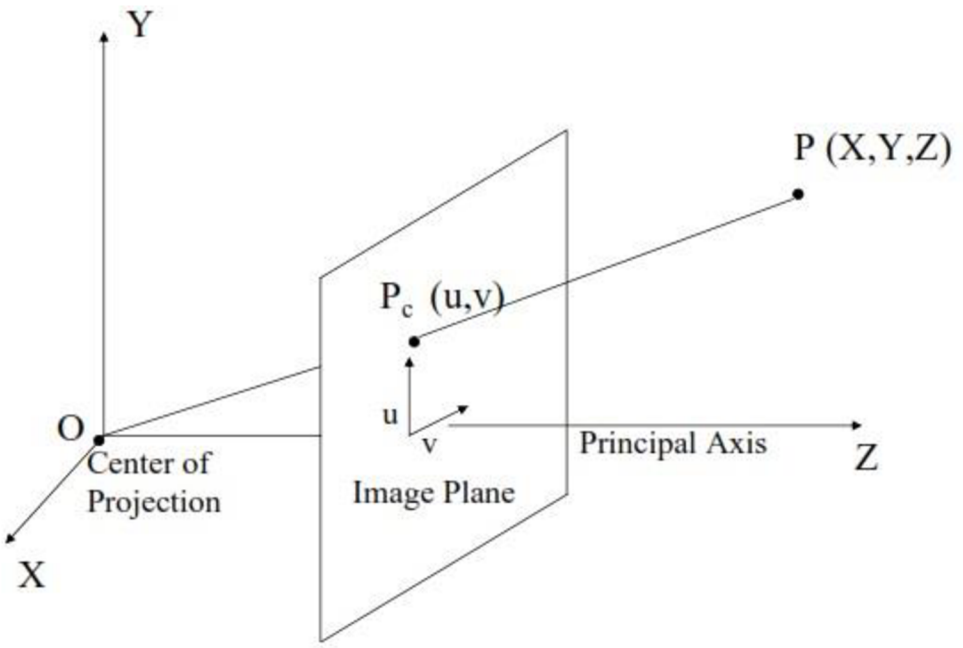


Obrázok 1-1 Príklad použitia aktívnych značiek na zaznamenanie pohybu [14].

2 Kamerový model

Kamerový model je funkcia, ktorá mapuje náš trojrozmerný svet do dvojrozmernej roviny, nazývanej obrazová rovina. Vo všeobecnosti, táto funkcia modeluje skutočnú fyzickú kameru. Poznáme mnoho druhov kamerových modelov s rôznou zložitosťou. Jedným z kritérií pre ich kategorizáciu je schopnosť zachytávať perspektívu. Perspektíva je optický jav, ktorý spôsobuje, že sa nám bližšie objekty javia väčšie a vzdialenejšie objekty menšie. Tento efekt môžeme pozorovať ako u ľudského zraku, tak aj pri väčšine kamier v reálnom svete [2].

Najjednoduchším kamerovým modelom ktorý zachytáva perspektívu je model dierkovej komory.



Obrázok 2-1 Schéma dierkovej komory [12].

Na Obrázok 2-1 je znázornená kamera so stredom premietania *O* a hlavnou osou paralelnou s osou *Z*. Obrazová rovina je od stredu premietania vzdialená o ohniskovú vzdialenosť *f*. 3D bod P z reálneho sveta sa zobrazí na obrazovú rovinu do bodu $P_c = (u,v)$. Pozíciu bodu P_c môžeme potom nájsť s využitím podobnosti trojuholníkov rovnicou

$$\frac{f}{Z} = \frac{u}{X} = \frac{v}{Y} \tag{1}$$

z čoho dostaneme následnou úpravou rovnice

$$u = \frac{fX}{Z} \tag{2}$$

$$v = \frac{fY}{Z} \tag{3}$$

V prípade, že sa začiatok súradnicového systému na obrazovej rovine nenachádza v bode, kde sa rovina pretína s osou *Z*, musíme bod P_c presunúť podľa skutočnej pozície začiatku. Ak parametre t_u a t_v predstavujú možné posunutie začiatku súradnicového systému na obrazovej rovine, potom sa bod *P* so súradnicami *(X,Y,Z)* premietne do 2D roviny podľa nasledujúcich vzťahov:

$$u = \frac{fX}{Z} + t_u \tag{4}$$

$$v = \frac{fY}{Z} + t_v \tag{5}$$

Pri práci s projekciou je vhodné používať homogénne súradnice. Ak na reprezentáciu bodu v euklidovských súradniciach použijeme vektor s n prvkami, pri homogénnych súradniciach použijeme vektor o $n + 1$ prvkoch. Potom môžeme vyššie uvedené vzťahy zapísať pomocou matice ako:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & t_u \\ 0 & f & t_v \\ 0 & 0 & 1 \end{bmatrix} * \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{6}$$

V rovnici (6) sú súradnice bodu P_c vyjadrené v palcoch, pretože sa jedná o obraz kamery. Aby sme mohli nájsť pozíciu bodu P_c v pixeloch, potrebujeme poznať rozlíšenie kamery v pixeloch/palec. Ak máme rozlíšenie m_u a m_v pixelov/palec, potom pozíciu hľadaného bodu nájdeme rovnicou

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} m_u f & 0 & m_u t_u \\ 0 & m_v f & m_v t_v \\ 0 & 0 & 1 \end{bmatrix} * \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \Rightarrow \begin{bmatrix} u \\ v \\ w \end{bmatrix} = K * \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{7}$$

Vidíme, že matica K závisí len od vnútorných parametrov kamery ako sú ohnisková vzdialenosť či smerovanie hlavnej osi. Preto sa táto matica zvyčajne nazýva matica vnútorných parametrov kamery.

V prípade, že kamera nemá svoj stred premietania v bode (0,0,0) a jej obrazová rovina nie je nutne kolmá na os *Z*, potrebujeme jej posunutie a rotáciu, aby sme mohli mať systém zhodný s Obrázok 2-1. Nech je posunutie stredu premietania do začiatku súradnicového systému *XYZ* dané bodom $T(\boldsymbol{T_x}, T_y, T_z)$ a rotácia aby sa hlavná os kamery zhodovala s osou *Z*, nech je vyjadrená rotačnou maticou *R* rozmerov 3x3. Potom matica vytvorená aplikovaním posunutia a následne rotácie je daná 3x4 maticou

$$E = (R \mid RT) \tag{8}$$

Táto matica sa nazýva matica vonkajších parametrov kamery. Takže kompletná transformácia kamery môže byť reprezentovaná vzťahom

$$K(R \mid RT) = (KR \mid KRT) = KR(I \mid T) \tag{9}$$

Z toho vyplýva, že bod P_c, projekcia bodu P, je daný vzťahom

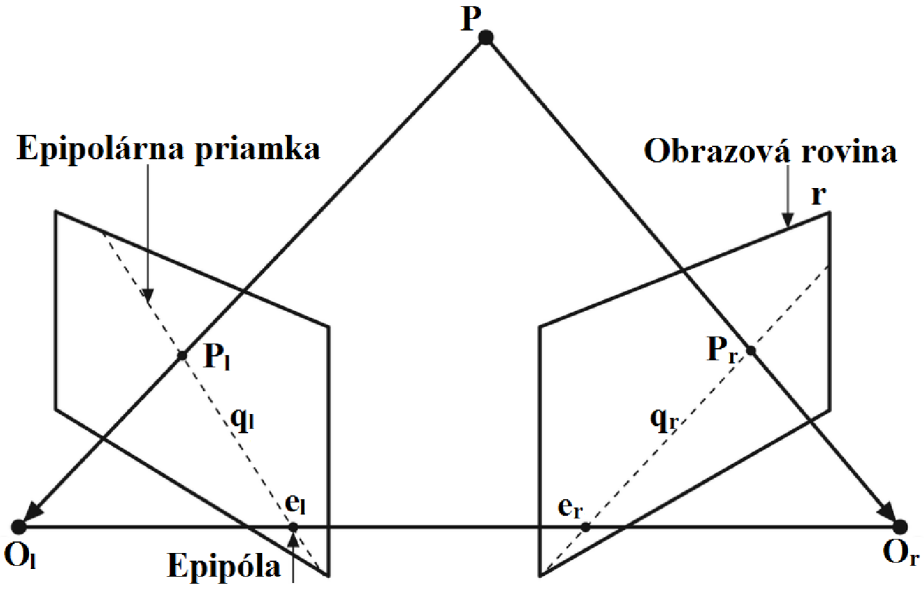
$$P_c = KR(I \mid T)P = CP \tag{10}$$

C je 3x4 matica ktorú zvyčajne nazývame kompletná kalibračná matica kamery [2].

3 Epipolárna geometria

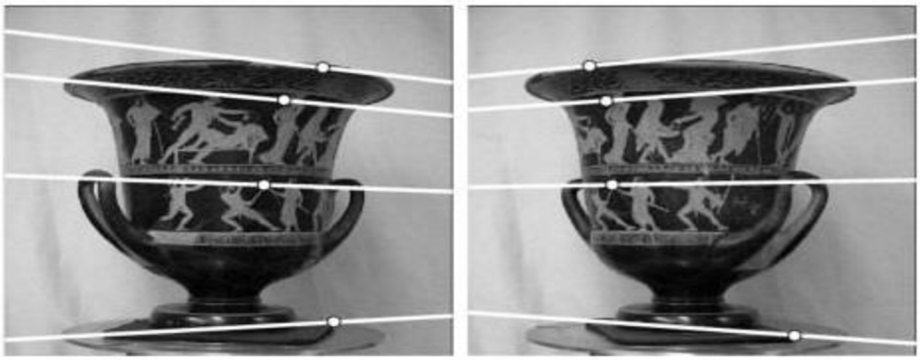
Epipolárna geometria je prirodzenou geometriou medzi dvomi a viacerými pohľadmi na scénu. Pohľadom na scénu rozumieme projekciu objektov scény do roviny. Epipolárna geometria je nezávislá na štruktúre scény. Odvodzuje sa iba od vnútorných parametrov kamery a vzájomnej polohy kamier [3].

Epipolárna geometria sa využíva najmä na identifikáciu vzájomne si odpovedajúcich bodov zachytených na snímkach dvoma alebo viacerými kamerami a rekonštrukciu ich 3D súradníc.



Obrázok 3-1 Schéma 2 kamier snímajúcich scénu.

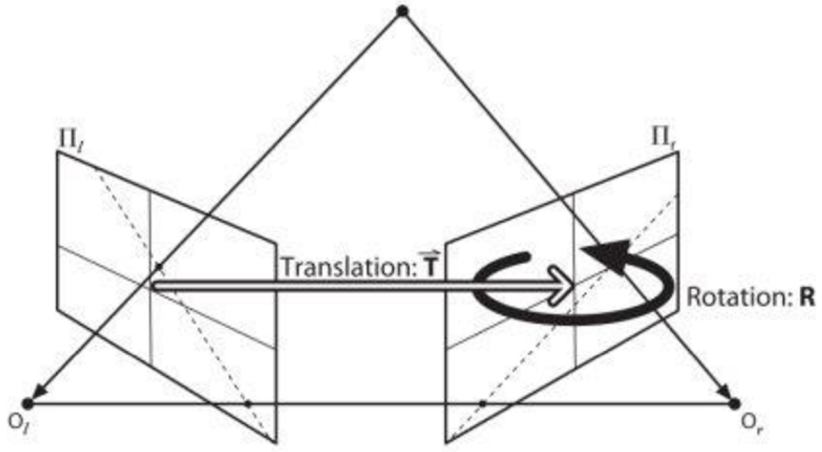
Na Obrázok 3-1 vidíme stredy premietania prvej a druhej kamery označené O_l a O_r. Body e_l a e_r nazývame epipóly. Označujú miesta, kde na snímke vidíme druhú kameru. Vznikajú ako prienik spojnice stredov premietania O_l a O_r s obrazovými rovinami. Ak zobrazujeme bod P na prvej rovine do bodu P_l, potom tento bod leží kdekoľvek na priamke q_r pri pohľade z druhej kamery. Rovnako bod p_r leží kdekoľvek na priamke q_l pri pohľade z prvej kamery. Tieto priamky nazývame epipolárne priamky. Vznikajú ako priesečnica epipolárnej roviny a obrazovej roviny. Epipolárna rovina je rovina, ktorá obsahuje základnú líniu a ľubovoľný bod P v priestore.



Obrázok 3-2 Príklad vykreslených epipolárnych čiar [15].

3.1 Fundamentálna matica

Algebrickou reprezentáciou epipolárnej geometrie je fundamentálna matica [4]. Fundamentálna matica obsahuje informácie o geometrickej polohe kamier v kamerovom systéme, ale aj vnútorné parametre samotných kamier. Avšak jej odvodeniu predchádza získanie esenciálnej matice, ktorá obsahuje len samotné informácie o vzájomnom geometrickom umiestnení kamier.



Obrázok 3-3 Geometria stereo snímania je zaznamenaná esenciálnou maticou E [13].

Pri zobrazovaní bodu P ľavou kamerou je jeho pozícia P_l v súradnicovom systéme danej kamery stred premietania pravej kamery je posunutý o vektor T. Jej rotácia je vyjadrená cez R. Zobrazenie bodu P na obrazovú plochu pravej kamery do bodu P_r môžeme potom vyjadriť ako:

$$P_r = \mathrm{R}(P_l - \mathrm{T}) \tag{11}$$

Všetky tieto komponenty sú obsiahnuté v epipolárnej rovine. Tú môžeme vyjadriť vzťahom, ktorý hovorí, že každý bod x v rovine sme schopní dostať s pomocou normálového vektora n, ktorý prechádza bodom y podľa vzťahu:

$$(x - y).n = 0 \tag{12}$$

Keďže normálový vektor môžeme dostať vzťahom P_l x T, ten môžeme dosadiť namiesto n a rovnica pre všetky možné P_l bude:

$$(P_l - T)^{\mathrm{T}}(T\ x\ P_l) = 0 \tag{13}$$

z čoho po využití vlastnosti $R^{\mathrm{T}} = R^{-1}$a dosadení s rovnicou (11) dostaneme tvar:

$$(R^{\mathrm{T}}P_r)^{\mathrm{T}}(TxP_l) = 0 \tag{14}$$

Keďže vektorový súčin je možné zapísať ako maticový súčin, môžeme so zadefinovať takú maticu S, že:

$$TxP_l = SP_l \Rightarrow S = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \quad (15)$$

Po dosadení tejto substitúcie do rovnice (14) dostaneme výsledok:

$$(P_r)^{\mathrm{T}}\, RSP_l = 0 \quad (16)$$

Súčin R a S je to, čo nazývame esenciálna matica E. Esenciálna matica je taká matica, ktorá obsahuje informácie o posunutí a rotácii, ktoré vyjadrujú lokáciu druhej kamery v závislosti k prvej kamere v globálnych súradniciach [4].
Pre výsledný tvar s esenciálnou maticou môžeme zapísať vzťah ako:

$$(P_r)^{\mathrm{T}}\, EP_l = 0 \quad (17)$$

Keďže esenciálna matica E neobsahuje žiadne vnútorné parametre kamery, ale len vyjadruje vzájomnú pozíciu jednej kamery voči druhej, mapuje body na seba vo fyzických, alebo kamerových súradniciach, nie v pixelových. Na to potrebujeme maticu, ktorá obsahuje aj vnútorné parametre kamery, teda fundamentálnu maticu.

Cieľom epipolárnej geometrie je nájsť závislosti medzi obrazmi. Potrebujeme nájsť vzťah, ktorý bodu m z prvého obrazu priradí bod m' z druhej snímky. Pre ľubovoľnú dvojicu korešpondujúcich bodov existuje fundamentálna matica F rozmerov 3x3, pre ktorú platí $m'^{\mathrm{T}}.F.m = 0$. Jej hodnoty závisia na vzájomnej pozícii kamier a ich kalibrácii. To znamená, že obsahuje aj vnútorné parametre kamery [3].

Aby sme túto maticu získali dosadíme do rovnice esenciálnej matice (17) vzťah s maticou vnútorných parametrov kamery K, ktorý sme získali v rovnici (7). Pri fundamentálnej matici pracujeme už v pixelových súradniciach, nie vo fyzických súradniciach ako pri esenciálnej matici, preto už na označenie korešpondenčných bodov nepoužijeme označenie P_l a P_r alebo q_l a q_r.

$$q_r^{\mathrm{T}}\, (K_r^{-1})^{\mathrm{T}}\, EK_l^{-1} q_l = 0 \quad (18)$$

Fundamentálnu maticu F potom môžeme zadefinovať ako:

$$F = (K_r^{-1})^{\mathrm{T}}\, EK_l^{-1} \quad (19)$$

Následne môžeme vyjadriť vzťah medzi korešpondenčnými bodmi už v pixelových súradniciach ako:

$$q_r^{\mathrm{T}} F q_l = 0 \quad (20)$$

Fundamentálna matica je matica 2. stupňa. Obsahuje 7 parametrov, 2 pre každú epipólu a 4 pre homografiu, ktorá spojuje 2 obrazové roviny [1].

4 Houghove transformácie

Houghova transformácia je algoritmus, ktorý ponúka významné riešenie problémov pri detekcii a definícii geometrických objektov v digitálnom obraze, ako sú napríklad priamky, kružnice či elipsy. Metóda spočíva v nájdení parametrického vyjadrenia hľadaných objektov v obraze a transformácii karteziánskych súradníc do polárnych [5].

4.1 Detekcia priamok

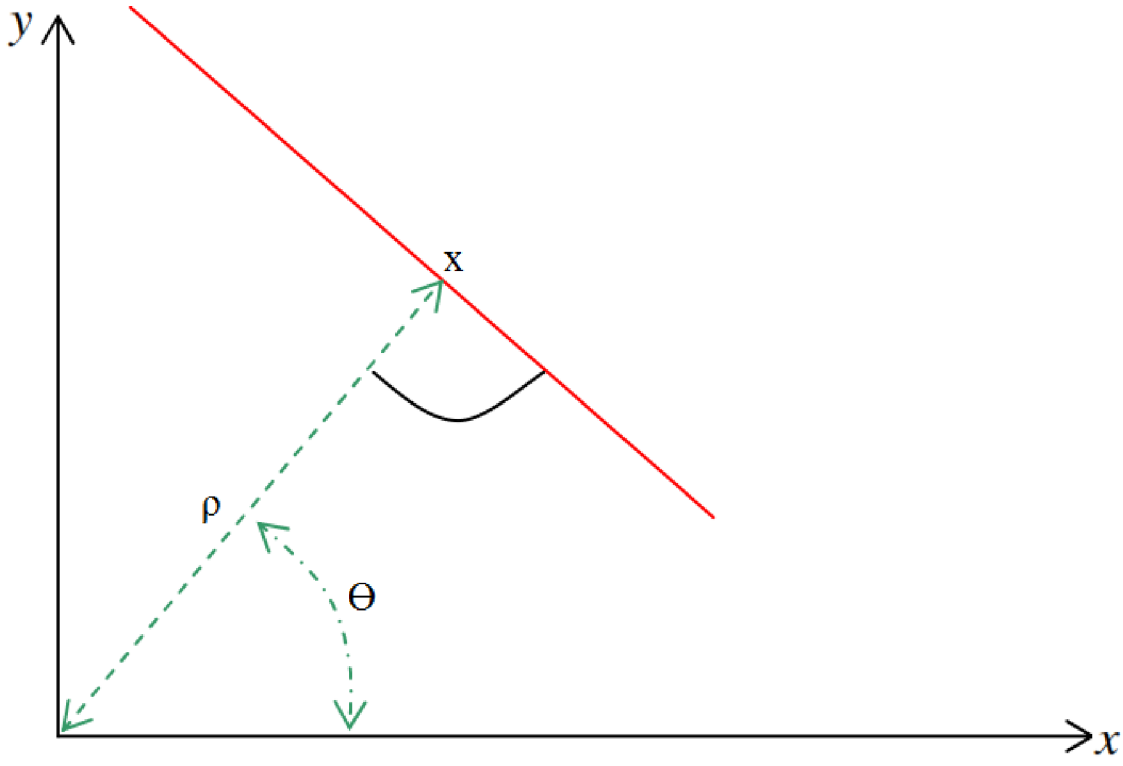
Základná Houghova transformácia využíva parametrický popis priamky. Ten má tvar rovnice:

$$y = mx + b \tag{21}$$

Ak pre ľubovoľný bod (x, y) je možné zmenou parametra m a dopočítaním b vykresliť priamku, potom je tento bod transformovaný do množiny bodov v rovine (m, b), korešpondujúcich so všetkými priamkami prechádzajúcimi bodom (x, y). Keďže parameter m môže nadobúdať hodnoty (-∞, ∞), je pre detekciu priamok výhodnejšie použiť parametrické vyjadrenie v polárnych súradniciach (Obrázok 4-1):

$$\rho = x.cos\theta + y.sin\theta \tag{22}$$

Každá priamka je potom reprezentovaná ako bod x v polárnych súradniciach, pričom spomínaná priamka je tá priamka, ktorá prechádza týmto bodom, a je kolmá na vektor, vedúci z počiatku súradnicového systému do bodu x [1].



Obrázok 4-1 Parametrické vyjadrenie priamky v polárnych súradniciach [6].

Postupnou zmenou parametru Θ od 0 po $\boldsymbol{\pi}$ a dopočítaním parametru ρ získame sínusoidu, ktorá je jedinečná pre daný bod. Bod, kde sa sínusoidy pretínajú v parametrickom priestore určuje priamku v priestore obrazu.

Priestor, kde sú jednotlivé krivky vykresľované sa nazýva akumulátor. Pri implementácii je akumulátor najprv vynulovaný, potom je každý bod (x, y) transformovaný na krivku a body tejto krivky sú v akumulátore inkrementované. Vyhľadaním maxima v akumulátore je potom možné získať parametre priamky nájdenej v obraze. Pri hľadaní priamok má akumulátor iba 2 rozmery (Θ, $\boldsymbol{\pi}$), teda je na jeho implementáciu možné použiť 2D pole [6].

4.2 Detekcia kruhov

Houghova transformácia pre detekciu kruhov pracuje na podobnom princípe ako metóda na detekciu priamok, ale je tu aj zopár odlišností.

Kým pri detekcii priamok bola parametrická rovina priamky popísaná dvoma parametrami (Θ, $\boldsymbol{\pi}$), pri detekcii kruhov potrebujeme parametre tri, ako uvidíme pri parametrickom vyjadrení kružnice (23)

$$r^2 = (x - x_0) + (y - y_0) \tag{23}$$

kde x, y sú súradnice bodu v priestore, x_0, y_0 sú súradnice stredu kružnice a r je polomer kružnice. Takže parametrický priestor sa bude pri detekcii kružníc skladať z troch parametrov (x_0, y_0, r). Z tohto dôvodu nám už nestačí akumulátor z 2D poľa, ale potrebovali by sme pole o 3-rozmerných prvkoch. Keďže by to zvýšilo nároky na pamäť a spomalilo by to rýchlosť algoritmu, je v knižnici OpenCV detekcia kruhov implementovaná metódou Hough gradient method.

Algoritmus je taký, že najprv sa na obrázok aplikuje detektor hrán (v OpenCV cvCanny()). Potom, pre každý nenulový bod v obrázku sa zoberie lokálny gradient. Pomocou tohto gradientu je každý bod na priamke danej uhlom gradientu, od špecifikovaného minima po maximum, inkrementovaný v akumulátore. Pritom je pozícia každého nenulového hranového bodu uložená. Z akumulátora sú potom vybraní takí kandidáti na stredy kruhov, ktorí sú nad daným prahom, a sú väčšie ako ich priami susedia. Títo kandidáti sú potom zoradení v zostupujúcom poradí, aby stredy s najväčším počtom podporujúcich pixelov boli zobrazené prvé.

Potom, pre každý stred, všetky nenulové pixely sú zoradené podľa vzdialenosti od daného stredu. Potom sa postupuje od najmenšieho polomeru kruhu po najväčší, a vybraný je ten, ktorému odpovedá najviac nenulových pixelov. Ďalšou podmienkou pre detekciu tohto kruhu je, aby jeho stred bol v dostatočnej vzdialenosti od predošlého vybraného stredu [1]. Avšak nedostatkom tejto metódy je, že nedokáže odhaliť sústredné kruhy, teda kruhy s rovnakým stredom a odlišným polomerom (Obrázok 4-2).

Obrázok 4-2 Detekcia kruhov pomocou Houghovej transformácie, nedokáže odhaliť sústredné kruhy [13].

5 Predspracovanie obrazu

Cieľom predspracovania obrazu je väčšinou potlačiť šum a skreslenie, ktoré vzniká pri digitalizácii a prenose obrazu. Iným príkladom predspracovania môže byť zvýraznenie určitých rysov obrazu, podstatných pre ďalšie spracovanie, napríklad hľadanie hrán.

Problémom je, že šum je tvorený hlavne vysokými frekvenciami a hrany tieto vysoké frekvencie obsahujú tiež. Metódy pre odstránenie šumu orezávajú vysoké frekvencie v obraze, čo zároveň poškodzuje hrany. Preto je potrebné nájsť vhodné metódy pre odstránenie šumu, ktoré v čo najväčšej miere zachovávajú hrany.

5.1 Vyhladenie obrazu Gaussovým filtrom

OpenCV ponúka viacero metód pre vyhladzovanie obrazu, ako sú napríklad bilaterálny filter, či mediánový filter. V tejto práci bol využitý Gaussov filter.

Pre popis šumu sa používa teória pravdepodobnosti. Pre biely šum je pravdepodobnosť výskytu rovnaká pre všetky frekvencie. Pre pravdepodobnosť Gaussovho šumu platí vzťah:

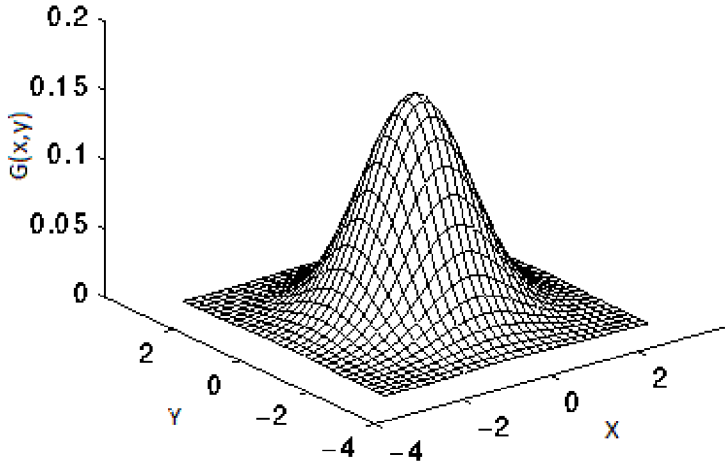
$$G(x,y) = \frac{1}{2\pi\,\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{24}$$

kde μ je stredná hodnota a σ je stredná kvadratická odchýlka Gaussovej distribúcie [7]. Rozloženie tejto funkcie je na Obrázok 5-2.

Gaussov filter je považovaný za najpoužívanejší a najužitočnejší, ale je pomalší. Filtrovanie prebieha konvolúciou každého bodu vo vstupnom poli s Gaussovým konvolučným jadrom, a následným sčítaním do výstupného poľa. Príklad konvolučného jadra pre $\sigma = 0$ je na Obrázok 5-1.

0,000007225	0,000394505	0,004806062	0,000394505	0,000007225
0,000394505	0,035512267	0,096532352	0,035512267	0,000394505
0,004806062	0,096532352	0,159154943	0,096532352	0,004806062
0,000394505	0,035512267	0,096532352	0,035512267	0,000394505
0,000007225	0,000394505	0,004806062	0,000394505	0,000007225

Obrázok 5-1 Príklad konvolučného jadra rozmerov 5x5 pre σ = 0 [6].

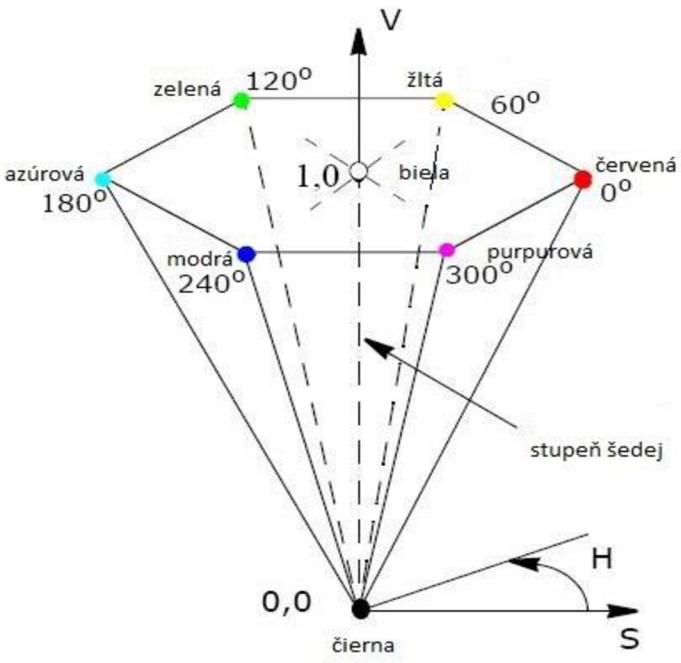


Obrázok 5-2 Gaussove rozloženie pre (0,0) a σ = 0 [7].

5.2 HSV farebný model

Model nazývaný HSV odpovedá popisu farieb, na ktorý je človek zvyknutý, jedná sa teda o intuitívny popis farieb.

Model HSV má tri základné zložky, farebný tón (hue), sýtosť (saturation), a jas (value). Farebný tón určuje prevládajúcu spektrálnu farbu, sýtosť reprezentuje prímes iných spektrálnych farieb a jas prímes bielej farby. Niekedy sa HSV model zvykne označovať aj ako HSB.



Obrázok 5-3 HSV model farieb.

Pre popis farieb v HSV modeli sa používa šesťboký ihlan ako je zobrazené na Obrázok 5-3. Ten je umiestnený do súradnicového systému tak, že vrchol ihlanu sa nachádza v počiatku systému a os ihlanu je zhodná so zvislou osou, ktorá zároveň znázorňuje zmeny úrovne jasu. Jas aj sýtosť sa menia

v intervale <0,1>, na obvode podstavy sa nachádzajú čisté farby. Farebný tón je určený ako veľkosť uhla, ktorý sa meria od osi S proti smeru hodinových ručičiek. Nadobúda hodnoty od 0° po 360° [11].

5.3 Morfologické operácie erózia a dilatácia

Morfologické operácie v obrazoch sú vzťahom dvoch množín. Jednou je obraz, reprezentovaný bodovou množinou X a druhou malá bodová množina B, nazývaná štrukturálny prvok. Tá systematicky prechádza celý obraz a jej vzťah k obrazu v každej pozícii sa ukladá vo výstupnom obraze [8].

Základnými operáciami matematickej morfológie sú erózia a dilatácia. Využívajú sa k rôznym úkonom, ako napríklad odstránenie šumu, izolácia jednotlivých objektov alebo spájanie rozličných objektov v obraze [1]. Dilatácia rozťahuje objekt o najbližšie pixely z jeho okolia. Erózia stenčuje objekt. Erózia a dilatácia nie sú inverzné operácie. Ich kombináciou vznikajú nové operácie, otvorenie a uzavretie [8].

Dilatácia je konvolúcia daného obrázku s nejakým jadrom. Jadro môže mať rozličný tvar a veľkosť a má zadefinovaný jeden pevný bod. Väčšinou je jadro v tvare štvorca s pevným bodom uprostred. Jadro je postupne posúvané po obrázku. Potom sa vypočíta maximálna pixelová hodnota v časti obrázku prekrytej jadrom, a táto hodnota je umiestnená do pixela obrázku, ktorý sa nachádza pod pevným bodom jadra. Tento postup spôsobí rozšírenie svetlých oblastí obrázku, ako je ukázané na Obrázok 5-4.

Obrázok 5-4 Obrázok 5 4 Dilatácia spôsobí rozšírenie svetlých oblastí objektu okolo čiernych oblastí písmena [13].

Pri erózii sa postupuje rovnako ako pri dilatácii, s tým rozdielom, že miesto pixelového maxima hľadáme minimum a touto hodnotou nahradíme hodnotu pixela pod pevným bodom jadra, čo spôsobí rozšírenie tmavých oblastí obrázku a stenčenie svetlých oblastí, ako je ukázané na Obrázok 5-5.

Obrázok 5-5 Erózia spôsobí rozšírenie tmavých oblastí objektu [13].

Vo všeobecnosti pre bežne používané jadrá platí, že dilatácia spôsobí vyhladenie dutín v obraze a erózia spôsobí vyhladenie výčnelkov.

6 Sledovanie objektu v obraze

Keď sa v počítačovej grafike zaoberáme video zdrojmi, a nie len obrázkami, často je v našom záujme možnosť sledovať určitý objekt v čase. Po úspešnom zdetekovaní určitého objektu, je preto ďalšou dôležitou časťou pochopenie pohybu jeho pohybu. Tento problém sa skladá z dvoch častí: identifikácia a modelácia.

Identifikácia zahrňuje vyhľadanie objektu, ktorý chceme sledovať v po sebe nasledujúcich rámcoch. Na identifikáciu sa dajú v OpenCV použiť rôzne techniky, ako napríklad farebné histogramy.

Sledovanie neidentifikovaných objektov je súvisiaci problém. Sledovanie neidentifikovaných objektov je dôležité vtedy, keď chceme určiť, ktoré objekty sú dôležité na základe ich pohybu, alebo keď pohyb je presne to, čo robí objekt zaujímavým pre sledovanie. Techniky pre sledovanie neidentifikovaných objektov zvyčajne zahrňujú sledovanie vizuálne významných kľúčových bodov. OpenCV ponúka dve metódy pre ich výber. Prvá sa volá Lucas-Kanadeova metóda a druhá Horn-Schunckova. Tieto metódy reprezentujú čo voláme riedky a hustý optický tok [1].

6.1 Optický tok

Optický tok je metóda často využívaná v situácii, keď chceme určiť pohyb medzi dvomi po sebe nasledujúcimi rámcami bez predošlej vedomosti o obsahu týchto rámcov. Zvyčajne pohyb samotný indikuje, že sa deje niečo zaujímavé. K tomu je možné priradiť každému pixelu v rámci určité posunutie, ktoré reprezentuje vzdialenosť, o ktorú sa pixel pohol oproti predošlému rámcu. Táto technika sa zvyčajne nazýva hustý optický tok. Horn-Schunckova metóda sa pokúša vypočítať takéto vektory pohybu pixelov. Nevýhodou hustého optického toku je, že v prípade, že mnoho pixelov na obrázku je rovnakej farby, budú sa pri prechode z jedného rámca do druhého javiť rovnako, a ťažko zachytíme nejaký pohyb. Preto potrebuje táto technika nejakú metódu interpolácie medzi bodmi, aby boli ľahšie detekovateľné, čo sa prejavuje na výpočetnej náročnosti algoritmu. Preto sa táto technika bežne moc nepoužíva, ale pre praktické využitie v aplikáciách sa volí riedky optický tok.

Algoritmus riedkeho optického toku sa spolieha na špecifikáciu podmnožiny bodov, ktoré budú ďalej sledované. Ak tieto body majú vhodné vlastnosti, napríklad že sa jedná o rohové pixely na subpixelovej úrovni, je sledovanie týchto bodov relatívne spoľahlivé. Knižnica OpenCV ponúka niekoľko možností, pre identifikáciu takýchto vhodných bodov v obraze, napríklad cvGoodFeaturesToTrack() [1].

6.1.1 Lucas-Kanadeova metóda

Lucas-Kanadeov (LK) algoritmus bol pôvodne navrhnutý na výpočet hustého optického toku. Ale pretože sa táto metóda ľahko aplikuje na podmnožinu pixelov v obrázku, stal sa dôležitou metódou riedkeho optického toku. Metóda sa spolieha iba na lokálne informácie získané z malého okna, ktoré obklopuje každý zo sledovaných bodov, čím sa líši od Horn-Schunckovej metódy, ktorá pracuje globálne.

Nevýhodou používania malého okna v LK metóde je, že veľké pohyby môžu posunúť bod záujmu mimo spomínané okno a pre algoritmus sa stáva tento bod ťažko sledovateľný. Tento problém viedol k vyvinutiu pyramídového LK algoritmu, ktorý sleduje objekty od najvyššej úrovne pyramídy

obrázku, ktorá obsahuje najmenej detailov, po nižšie úrovne s vyššími detailmi. Sledovanie pomocou pyramíd umožňuje zachytiť veľké pohyby pomocou lokálneho okna [1].
Výpočet optického toku na základe LK metódy je založený na predpoklade stability jasnosti obrazu, ktorý hovorí, že pre pohyb (u, v) bodu v obraze I sa jasnosť bodu nemení, čo vyjadruje vzťah

$$I(x, y, t) = I(x + u, y + v, t + 1) \tag{25}$$

Použitie Taylorovho rozvoja nás dovedie k rovnici

$$I_x u + I_y v + I_t = 0 \tag{26}$$

Túto rovnicu je možné vyriešiť metódou najmenších štvorcov cez obraz danú výrazom [9]

$$(u\ v)^T = \left[\sum (I_x\ I_y)^T (I_x\ I_y)\right]^{-1} \sum I_t (I_x\ I_y)^T \tag{27}$$

6.2 Kalmanov filter

Kalmanov filter je nástrojom, ktorý vie odhadnúť premenné v širokom spektre procesov. Matematicky povedané, Kalmanov filter odhaduje stavy lineárneho systému. Je možné povedať, že zo všetkých možných filtrov je práve Kalmanov ten, ktorý minimalizuje varianciu chyby odhadu.

6.2.1 Lineárny systém

Ak teda chceme, aby Kalmanov filter odstránil šum zo signálu, proces v ktorom ho chceme použiť musí byť opísateľný lineárnym systémom. Veľa fyzikálnych procesov, napríklad auto pohybujúce sa po ceste, satelit krúžiaci na obežnej dráhe Zeme, hriadeľ poháňaný od elektromotora alebo sínusový signál nosnej rádiovej vlny môžu byť aproximované ako lineárne systémy. Lineárny systém je teda, jednoducho povedané, proces, ktorý vieme opísať nasledovnými dvoma rovnicami:

- stavová rovnica:

$$x_{k+1} = Ax_k + Bu_k + w_k \tag{28}$$

- rovnica výstupu:

$$y_k = Cx_k + z_k \tag{29}$$

V uvedených rovniciach *A*, *B* a *C* sú matice, *k* je časový index, *x* je tzv. stav systému, *u* je známy vstup do systému a *w* a *z* sú šum. Premenná *w* reprezentuje tzv. procesný šum a *z* tzv. merací šum. Každá z týchto veličín je obvykle vektor a preto obsahuje viac ako jednu zložku. Vektor *x* obsahuje všetky dostupné informácie o súčasnom stave systému, ale *x* sa nemôže merať priamo. Namiesto toho sa meria *y*, ktoré je funkciou *x* , ale je ovplyvnená šumom *z* . Môže sa preto použiť *y* ako pomoc na

získanie odhadu *x*, ale vziať informáciu z *y* ako reprezentujúcu hodnotu sa nedá, pretože obsahuje aj šum.

6.2.2 Teória Kalmanovho filtra a algoritmus

Uvažujme lineárny model systému ako bolo uvedené vyššie. Chceme využiť dostupné merania *y* na odhad stavu systému *x* . Vieme, ako sa systém správa vzhľadom na stavovú rovnicu a máme merania polohy. Chceme, aby prostriedok, ktorý zabezpečuje presný odhad správneho stavu, tento stav určoval presne aj keď nemáme k dispozícii možnosť priamo ho merať. K tomu sú potrebné dve významné veci.

Najprv potrebujeme priemernú hodnotu odhadu správneho stavu. To znamená, že nechceme, aby náš odhad bol posunutý nahor alebo nadol. Matematicky povedané: očakávaná hodnota odhadu by mala byť rovná očakávanej hodnote stavu. Druhá potreba je, aby sa odhad stavu od skutočnosti, teda správnej hodnoty, líšil čo najmenej.

To znamená, že chceme nielen to, aby bol priemer odhadu rovný priemeru správnej hodnoty, ale aj prostriedok odhadu, ktorého výsledkom je čo najmenšia variácia odhadu tohto stavu, teda chceme odhadovať správne a s čo najmenšou možnou chybou.

Kalmanov filter spĺňa obe tieto kritériá. Ale riešenie Kalmanovým filtrom nie je možné aplikovať kým nie sú splnené určité predpoklady o šume, ktorý systém ovplyvňuje. Je treba pripomenúť si, že v rovniciach modelu systému (rovnice 28, 29) vystupujú dve premenné - *w* je šum v procese a *z* je šum v meraní. Môžeme konštatovať, že priemerná hodnota *w* je rovná nule a priemerná hodnota *z* je tiež rovná nule, pretože ide o náhodné javy. Tiež potom môžeme ďalej skonštatovať, že neexistuje korelácia medzi *w* a *z* - teda v akomkoľvek čase *k* , w_k a z_k sú nezávislé náhodné premenné. Potom kovariančné matice šumu S_w a S_z vieme zadefinovať ako:

- Kovariancia šumu v procese:

$$S_w = (w_k w_k^T) \tag{30}$$

- Kovariancia šumu v meraní:

$$S_z = (z_k z_k^T) \tag{31}$$

kde w^T a z^T označujú transpozíciu náhodných vektorov.

Existuje veľa alternatívnych, ale ekvivalentných spôsobov ako vyjadriť rovnice Kalmanovho filtra. Jednou z formulácií je táto:

$$K_k = AP_k C^T (CP_k C^T + S_z)^{-1} \tag{32}$$

$$x_{k+1} = (Ax_k + Bu_k) + K_k(y_{k+1} - Cx_k) \tag{33}$$

$$P_{k+1} = AP_k A^T + S_w + AP_k C^T S_z^{-1} CP_k A^T \tag{34}$$

Kalmanov filter pozostáva z troch rovníc, z ktorých každá zahŕňa maticové operácie. Matica *K* je tzv. Kalmanov zisk (zosilnenie) a matica P je nazývaná kovariancia chyby odhadu.

Prvý výraz použitý pre odvodenie odhadu stavu v čase k+1 je vlastne len A-krát odhad stavu v čase k plus B-krát známy vstup v čase k . Toto by bol odhad stavu ak by sme nemali merania. Inými slovami, odhad stavu by sa v čase šíril rovnako ako stavový vektor v modeli systému. Druhý výraz v rovnici je tzv. korekcia a reprezentuje veľkosť korekcie, akou treba korigovať šírený odhad stavu na základe meraní.

Rozborom rovnice pre K zistíme, že ak je šum z merania veľký, S_z bude veľké, takže K bude malé a nebudeme dávať veľkú váhu meraniu y , keď budeme počítať ďalšie. Na druhej strane však ak je šum v meraní malý, S_z bude malé a K veľké, čo znamená, že meraniu priradíme veľkú váhu pri výpočte ďalšieho [10].

7 Implementácia

Implementácia programu pre zachytávanie pohybu kamerovým systémom sa skladá z viacerých hlavných častí. Najprv je potrebné zostaviť si kamerový systém. K tomu som využil 2 webové kamery.

Samotný program sa skladá z častí, ktoré vykonávajú stereo kalibráciu, potrebnú pre získanie vnútorných a vonkajších parametrov týchto kamier, ďalej aplikácia pomocou filtrovania na základe farieb zachytáva 2D súradnice značiek pre detekciu pohybu. Následne je potrebné tieto značky sledovať v čase, k čomu sa dajú využiť rôzne techniky sledovania. Získané 2D súradnice je potrebné previesť do 3D súradnicového systému, pomocou triangulácie. Takto získané súradnice je už potom možné využiť pre 3D rekonštrukciu a animáciu.

Všetky tieto časti aplikácie som sa rozhodol implementovať v jazyku C++ pomocou knižnice OpenCV. Pre prácu s touto knižnicou som si zvolil vývojové prostredie Visual Studio 2013.

7.1 Knižnica OpenCV

OpenCV je open-source knižnica distribuovaná pod licenciou BSD, ktorá obsahuje stovky algoritmov pre manipuláciu s obrazom. Je špecializovaná predovšetkým na počítačové videnie a spracovanie obrazu v reálnom čase. Funguje pod operačnými systémami Linux, Windows, Mac OS X, FreeBSD, OpenBSD, Android, Maemo, iOS a BlackBerry.

OpenCV bola vyvinutá pre výpočtovú efektivitu, hlavne pri real-time aplikáciách. Je preto napísaná v optimalizovanom jazyku C a C++ a je schopná využívať možnosti viacjadrových procesorov. Má modulárnu štruktúru, takže obsahuje niekoľko zdieľaných a statických knižníc.
K dispozícii sú tiež plne funkčné rozhrania v jazykoch Python, Java a Matlab/Octave.

Knižnica OpenCV bola vytvorená s cieľom umožniť programátorom vyvinúť sofistikované aplikácie v relatívne krátkom čase. Množstvo jej algoritmov a funkcií sa využíva napríklad pri kontrole produktov vo výrobe, pri bezpečnostných a obranných systémoch (monitorovanie neidentifikovaných lietajúcich objektov), biomedicínska analýza, tvorba satelitných a webových máp, rozpoznávanie objektov ale aj vo vojenskom priemysle, napr. sa využíva pri bezpilotných lietadlách.

Medzi oblasti, na ktoré sa aplikácie v OpenCV zameriavajú patria napr. rozpoznávanie gest, rozpoznávanie tváre, pohyblivé roboty, sledovanie pohybu, rozšírená realita a pod. Pre podporu vymenovaných oblastí OpenCV podporuje knižnicu pre strojové učenie, ktorá obsahuje metódy ako sú boosting, gradient boosting, umelá neurónová sieť, decision tree learning, random forest a pod.

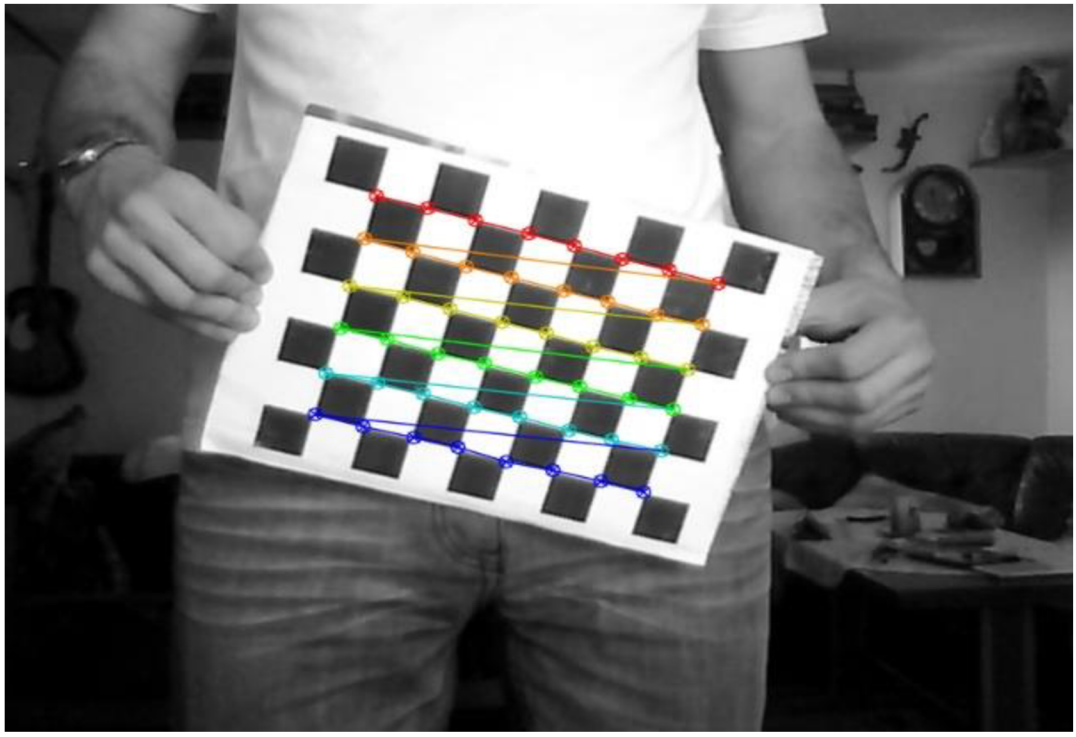
7.2 Stereo kalibrácia

Pre získanie informácii o vzájomnej polohe kamier a ich vnútorných parametrov sa využíva kalibrácia kamier. Stereo kalibrácia je proces vypočítavania geometrického vzťahu medzi dvoma kamerami v priestore [1].

Túto kalibráciu je možné v OpenCV implementovať buď pomocou funkcie cvStereoCalibrate(), ktorá pracuje naraz s oboma kamerami, alebo pomocou cvCalibrateCamera(), ktorá počíta parametre pre každú kameru zvlášť. Tieto parametre je potom potrebné ďalej spracovať, aby vyjadrovali vzťah medzi oboma kamerami. Vo svojej práci som si zvolil použiť funkciu cvStereoCalibrate(). Pri jej implementácii som využil aj vzorový kód pre stereo kalibráciu, ktorý sa nachádza v priečinku medzi vzorovými kódmi knižnice OpenCV.

Pre úspešnú kalibráciu je potrebné programu dodať niekoľko fotiek z oboch kamier, na ktorých je zachytená šachovnica. Pre čo najväčšiu presnosť je vhodné, aby boli tieto fotky zachytené v rovnakom momente z oboch kamier, alebo aspoň aby sa fotený objekt podľa možností nepohyboval, kým bude zachytený oboma kamerami.

Počet potrebných párov fotiek nie je presne určený, ale platí pravidlo, že čím väčší počet párov fotiek, a čím rôznorodejšie pozície foteného objektu, v tomto prípade šachovnice, tým presnejšie dáta je funkcia pre stereo kalibráciu schopná vrátiť.

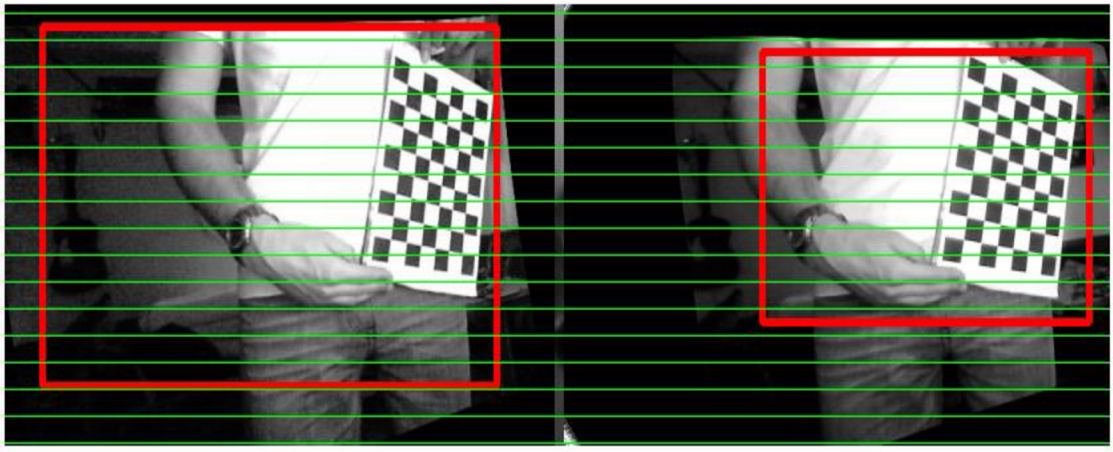


Obrázok 7-1 Detekcia bodov na šachovnici.

V programe sa fotky so šachovnicami načítavajú pomocou XML súboru, ktorý obsahuje zoznam fotiek a cestu k nim. Pre presnú kalibráciu je potom potrebné nastaviť reálnu veľkosť jedného štvorčeka na šachovnici. Následne prebieha detekcia bodov na šachovnici (Obrázok 7-1) a program kontroluje vhodnosť párov, prípadne odstráni nevhodné páry. Aby mohla kalibrácia vôbec prebehnúť, musí program nájsť aspoň 2 páry fotiek. Potom tieto odpovedajúce body z párov fotiek dodáme funkcii cvStereoCalibrate(), ktorá nám vráti potrebné informácie, ako sú matice vnútorných parametrov oboch kamier, koeficienty skreslenia, fundamentálna matica, esenciálna matica, geometriu kamier cez vektor posunutia T a rotačnú maticu R. Funkcia má aj viacero voliteľných parametrov. Pre úspešnú kalibráciu bolo potrebné odstrániť parameter, ktorý znamená rovnakú fokálnu vzdialenosť kamier, pretože použité kamery boli odlišné modely.

Program potom vypíše priemernú chybu, ktorú vracia funkcia cvStereoCalibrate() ako návratovú hodnotu, a podľa jej hodnoty je potom buď potrebné nafotiť kvalitnejšie fotky alebo je výsledky možné použiť ďalej v programe.

Tiež je možné v programe zobraziť si úspešnosť kalibrácie použitím funkcií Remap() a StereoRectify() a následným vykreslením výsledku. Ak prebehla kalibrácia a rektifikácia správne, mali by zelené horizontálne čiary prechádzať odpovedajúcimi bodmi na obrázkoch z oboch kamier (Obrázok 7-2). Táto časť programu sa nachádza v zdrojovom kóde s názvom Calibration.cpp.



Obrázok 7-2 Výsledné zobrazenie kalibrácie a rektifikácie, horizontálne čiary prechádzajú korešpondujúcimi bodmi na obrázkoch.

7.3 Detekcia značiek

Po kalibrácii program prechádza do časti, kde prebieha čítanie vstupného video po jednotlivých obrazových rámcoch a ich následné spracovanie. Hlavným cieľom tejto časti programu je získanie súradníc značiek na obrázkoch.

Na začiatku sa dá nastaviť, či chceme načítať rámce z videí uložených na disku alebo či chceme rámce načítavať pomocou kamier v reálnom čase. V tomto prípade je ale nutné nafotiť aj nové kalibračné fotky, pretože je pravdepodobné, že kamery sa nenachádzajú v rovnakej pozícii ako pri fotkách, ktoré boli nafotené pri zaznamenaní spomínaných videí.

7.3.1 Úprava snímok pred detekciou

Po získaní rámcov sú odoslané do funkcie prepare_frames() na ďalšie spracovanie. Tu sú najprv rámce vyhladené pomocou filtra. Knižnica OpenCV ponúka viacero možností na odstránenie šumu z obrázkov, ako je napríklad funkcia GaussianBlur (). Potom sú prevedené z BRG modelu do HSV farebného modelu (Obrázok 5-3), ktorý pri detekcii farieb podáva lepšie výsledky, pretože pri snímaní kamerami môže farbu dosť skresľovať osvetlenie a odraz svetla od sledovaného objektu. Pri použití modelu HSV sa s týmto problémom dá lepšie vysporiadať ako pri BGR modeli, pretože hlavným nositeľom informácie je tam tón farby.

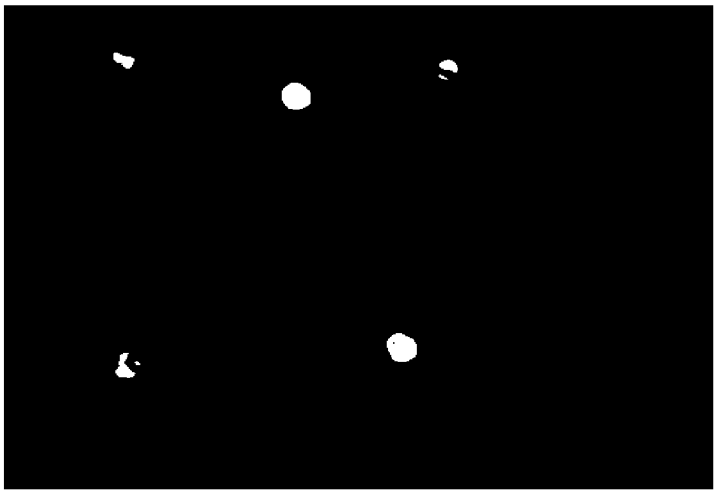
Obrázok 7-3 Pred a po použití GaussianBlur().

7.3.2 Filtrácia farieb

Po prevedení obrazu do HSV modelu v programe nasleduje samotná detekcia značiek, a to za pomoci filtrácie farieb. K tomu som použil funkciu inRange(). Táto funkcia umožňuje kontrolu, či prvky vstupného poľa, v tomto prípade matice, ktorá obsahuje zachytený rámec (Obrázok 7-4), spadajú do hraničných hodnôt uvedených medzi parametrami funkcie. Tieto parametre sú v podobe skalárnych vektorov nastavené pomocou trackbarov. Hodnoty na trackbaroch sú pri spustení aplikácie prednastavené na čo najpresnejšiu detekciu farby značiek, ale je s nimi možné počas behu programu manipulovať pre prípad, že by sa zmenili podmienky svetla a pozadia a detekcia značiek by nebola dostatočne úspešná. V prípade, že daný prvok sa nachádza medzi hranicami, je mu pridelená biela farba. V opačnom prípade bude vyfarbený na čierno. Takže výsledkom tejto operácie je matica rovnakej veľkosti, prevedená do čierno-bieleho formátu a pretypovaná z pôvodného typu CV_64F do CV_8U (Obrázok 7-5). V ideálnom prípade by mali byť nabielo vyfarbené len pozície značiek, načierno všetko okolie.

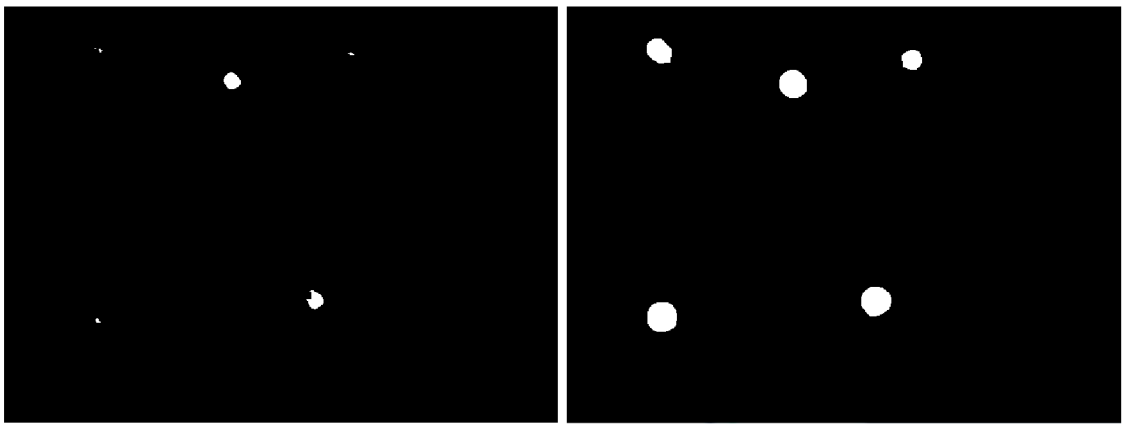
Vo svojej práci som sa rozhodol za farbu značiek zvoliť zelenú. Pri filtrovaní farieb bližších k červenej dochádzalo k nežiadanej detekcii ľudskej kože a pri modrej sa často medzi hranice filtrovania farieb zmestila aj časť pozadia obrazu. Za útvar značiek som si zvolil nazeleno zafarbené pingpongové loptičky, pretože majú tvar gule, teda sú dobre viditeľné aj z profilu.

Obrázok 7-4 Snímka zachytávajúca objekt so značkami.



Obrázok 7-5 Snímka vyfiltrovaná pomocou funkcie inRange().

Po získaní čiernobielej matice som použil funkcie erode() a dilate(), aby sa biele oblasti na obrázku rozšírili a mali celistvejší tvar (Obrázok 7-6). Po ich aplikácii majú biele oblasti približný tvar kruhu, a dajú sa potom ďalej vyhľadať v obrázku pomocou funkcie na vyhľadanie kruhových tvarov.



Obrázok 7-6 Výsledok operácii erode() (vľavo) a následne dilate() (vpravo).

7.3.3 Vyhľadávanie kruhov

Po úspešnom vyfiltrovaní farebných značiek nasleduje získanie súradníc ich stredov v obrázku (Obrázok 7-7). K tomuto účelu som v programe použil funkciu HoughCircles(), ktorá využíva k vyhľadávaniu nedokonalých kruhov v obraze extrakciu príznakov pomocou Houghovej transformácie.

Táto funkcia prijíma ako vstupný parameter čiernobiely alebo šedotónový obraz. Ja vkladám ako vstup výslednú maticu po aplikácii funkcií erode() a dilate(). Funkcia má aj viacero užitočných parametrov, ako napríklad metóda detekcie, kde som použil CV_HOUGH_GRADIENT, ktorá využíva k detekcii gradient, teda smer rastu hrán. Ďalšími parametrami funkcie sú minimálna vzdialenosť medzi kruhmi, horný prah pre detektor hrán.

Parameter, ktorý veľmi vplýva na úspešnosť detekcie kruhov a teda na počet kruhov, ktoré funkcia v obraze nájde, je prah pre detekciu stredov kruhov. Tento parameter je možné počas behu programu nastavovať pomocou trackbaru, podobne ako hranice pri detekcii farieb. Čím nižšia je hodnota tohto parametra, tým menšie nároky na kruhový tvar objektov funkcia má. Poslednými parametrami funkcie sú minimálne a maximálne rozmery kruhov, ktoré chceme detekovať. Tieto parametre je možné tiež nastavovať pomocou trackbarov, pre prípad, že pohyb zachytávame real-time.

Výsledkom funkcie HoughCircles() je vektor, ktorý obsahuje 3 kanálové vektory. Tie obsahuje súradnice stredov kruhov x, y a polomery pre každý nájdený kruh. Tieto súradnice predstavujú stredy značiek, teda ich pozíciu v pixelových súradniciach.

Všetky tieto úkony, ktoré obsahuje detekcia značiek sa vykonávajú pre snímky z oboch kamier zvlášť, takže na konci tohto sledu príkazov sú výstupom 2 vektory, ktoré obsahujú pozície značiek. Táto časť programu sa nachádza v zdrojovom súbore s názvom Main.cpp.

Obrázok 7-7 Pozície značiek nájdené pomocou metódy HughCircles().

7.4 Sledovanie značiek pomocou Kalmanovho filtra

Po úspešnej lokalizácii značiek nasleduje sledovanie týchto objektov v čase. K tomu som v programe využil Kalmanove filtre. Pre každú značku sa vytvorí jeden filter a uloží sa do vektora, ktorý ich obsahuje všetky. Potom prebieha načítavanie ďalších obrázkov z videa v cykle a pomocou Kalmanových filtrov je možné určiť, ktoré značky sa kam pohli, aby sa nepomiešali medzi sebou. Sledovanie značiek prebieha vždy len na obrázkoch z jednej, po celý beh programu tej istej kamery, kvôli nižšej výpočetnej náročnosti. Nie je potrebné sledovať značky aj na obrázkoch druhej kamery, pretože správne poradie značiek sa určí pomocou sledovaných značiek z prvej kamery a použitia funkcie match_markers() v ďalšej časti programu.

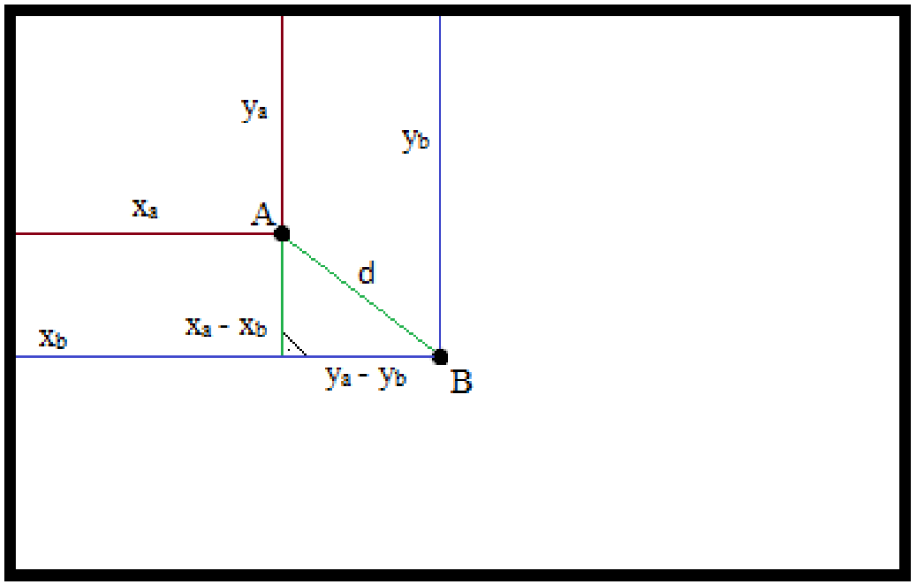
Avšak, vždy pri načítaní nových rámcov videa je potrebné zaistiť správne priradzovanie Kalmanových filtrov k odpovedajúci bodom, ktoré dané filtre sledujú. Najprv filtre vykonajú predikciu pohybu naposledy sledovaných značiek,. Potom sa týmto predikovaným súradniciam program pokúsi priradiť odpovedajúce súradnice značiek. To sa vykoná pomocou funkcie match_marker_points(). V tejto funkcii sa k priradzovaniu využíva euklidovská vzdialenosť.

Keďže súradnice predikovaných pozícii aj súradnice značiek ležia v jednom súradnicovom systéme a predikované súradnice sa snažia predpovedať pozíciu určitej značky, mali by byť ich súradnice približne rovnaké a vzdialenosť medzi týmito bodmi by mala byť menšia ako vzdialenosť medzi bodmi, ktoré si neodpovedajú.

Pretože sú súradnicové osy na seba kolmé, dá sa táto vzdialenosť vypočítať ako dĺžka prepony podľa vzorca

$$d = \sqrt{x^2 + y^2} \tag{35}$$

Kde *x* je rozdiel vzdialeností bodov na horizontálnej osi a *y* je rozdiel vzdialeností na vertikálnej osi, ako je zobrazené na Obrázok 7-8.



Obrázok 7-8 Výpočet vzdialenosti medzi súradnicami značky a jej predikovanou pozíciou pomocou euklidovskej vzdialenosti.

V prípade, že program bol spustený v móde zachytávania videa v reálnom čase a zmení sa počet zdetekovaných značiek, je potrebné nájsť opäť na základe euklidovskej vzdialenosti správne filtre k bodom čo ostali na obrázku. Filtre s predikovanými pozíciami najvzdialenejšími od pozícii značiek ostanú nepriradené a budú odstránené z vektora, kým sa počet filtrov a zistených značiek nebude rovnať. V prípade, že sa počas behu programu počet značiek zvýši, je jednoducho vytvorený odpovedajúci počet filtrov.

V prípade, že rámce sú načítavané z video súboru, je v programe vykonávaná kontrola, či je na obrázkoch správny počet detekovaných značiek. Ak nie, rámec sa ďalej nespracúva ale pokračuje načítanie ďalších rámcov z videa. Toto je zavedené z toho dôvodu, že nesprávny počet značiek môže spôsobiť to, že sú si priradené nesprávne pozície značiek, čo výrazne zhorší presnosť získavania 3D súradníc a rekonštrukcie scény v ďalšej časti programu.

Funkcie slúžiace na vytvorenie Kalmanových filtrov a ich obsluhu sa nachádzajú v zdrojom súbore s názvom KalmanFilter.cpp.

7.4.1 Vyhľadanie korešpondujúcich značiek

Po získaní súradníc stredov značiek a ich identifikácii v čase pomocou Kalmanových filtrov je v mojom programe ďalším krokom nájsť korešpondujúce stredy značiek na odpovedajúcich snímkach z oboch kamier. K tomu som vytvoril funkciu match_markers(). Funkcia prijíma ako vstupné parametre dva vektory s bodmi stredov značiek a polomermi ich kruhov. Podľa prvého vektora zoradí

prvky druhého vektora tak, aby si ich poradie odpovedalo a súradnice značiek z prvého obrázku odpovedali značkám z druhého. Funkcia vráti výsledný zoradený vektor.

K zoradeniu vektora a vyhľadaniu odpovedajúcich súradníc som využil fundamentálnu maticu a prevedenie súradníc pozícii značiek do homogénneho tvaru. Keďže fundamentálna matica obsahuje vnútorné aj vonkajšie parametre kamerového systému, je možné zistiť pozíciu bodu P_r na obrázku z druhej kamery, ktorý odpovedá bodu P_l z obrázku prvej kamery vzťahom

$$P_l^T * F * P_r = 0 \tag{36}$$

Keďže pri kalibrácii sa vyskytuje určitá chyba, správne riešenie sa v realite pravdepodobne bude len približovať nule, nie rovnať. Vo svojom riešení preto skúšam priradiť si navzájom všetky body, a tie body, ktoré spolu dávajú číslo najbližšie nule priradím. V riešení bolo potrebné ošetriť situáciu, aby sa nepriradzovalo jednému bodu viacero korešpondenčných bodov, a to postupným kopírovaním priradených bodov do výsledného vektora a odstraňovaním týchto bodov z pôvodného vektora.

Toto riešenie ale nefunguje samo o sebe v dostatočnej miere, pretože ak leží viacero bodov blízko jednej epipolárnej čiary, je možné, že si tieto body budú priradené opačne. Preto musí po priradzovaní pomocou fundamentálnej matice nasledovať kontrola, či nejaké body neležia na približne rovnakej výškovej úrovni. Ak áno, porovnávajú sa súradnice týchto bodov na osi x. Ak je hodnota súradnice x bodu P_l menšia ako u bodu Q_l, potom v prípade, že body P_l a P_r sú korešpondenčnými bodmi, mal by rovnaký vzťah platiť aj u bodov P_r a Q_r. V prípade, že tento vzťah platí opačne, sú priradené body vymenené. Zavedenie tejto kontroly výrazne zvýšilo úspešnosť priradzovania korešpondenčných bodov, avšak úspešnosť stále nie je stopercentná. Úspešné priradzovanie korešpondenčných bodov je možné vidieť na Obrázok 7-9.

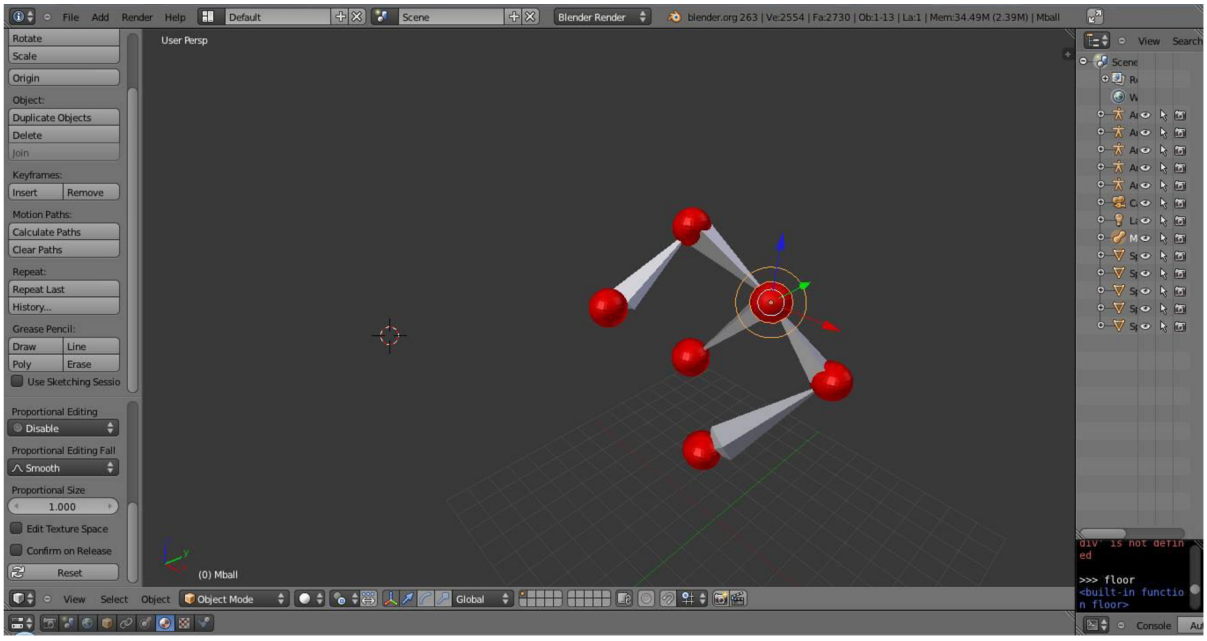
Obrázok 7-9 Funkcia match_marker_points() dokáže nájsť odpovedajúce body z dvoch obrázkov.

7.5 Triangulácia bodov

Na konci každej slučky cyklu pre načítanie nových rámcov videa prebieha samotná triangulácia, teda získavanie 3D súradníc zo vstupných 2D súradníc. K tomu som vo svojej aplikácii využil funkciu cvTriangulatePoints(), ktorú ponúka knižnica OpenCV.

Vstupné parametre tejto funkcie tvoria projekčné matice oboch kamier, ktoré som získal z kalibračnej časti programu, dva vektory obsahujúce vstupné súradnice, v tomto prípade sú to súradnice značiek zachytených oboma kamerami zoradenými tak, aby si korešpondenčné body odpovedali a matica, do ktorej funkcia uloží výsledné 3D súradnice v homogénnom súradnicovom systéme.

Homogénne súradnice sú také, pri ktorých používame pri N rozmerných súradniciach vektor o N + 1 prvkoch. Tento prvok navyše predstavuje váhu. Aby boli po triangulácii výsledné súradnice v Euklidovskom súradnicovom systéme, je potrebné súradnice jednotlivých bodov vydeliť touto váhou. Výsledné súradnice predstavujú 3D súradnice značiek. Tieto súradnice sú potom programom zapisované do výstupného súboru, odkiaľ je možné ich načítať a dodať aplikácii slúžiacej na 3D rekonštrukciu a zrekonštruovať pohyb snímaného objektu. Ja som si na rekonštrukciu zvolil program Blender. Vytvorené animácie sú dostupné v príslušnom adresári na dodanom CD ako príloha.
Príklad zrekonštruovaných bodov je možné vidieť na Obrázok 7-10.



Obrázok 7-10 Zrekonštruované 3D súradnice pomocou programu Blender.

8 Testovanie

V tejto kapitole popíšem experimentálne výsledky testovania systému, ktorého implementáciu a detaily som popísal v kapitole 7. Pri testovaní a následnom vyhodnocovaní som sa zameral hlavne na úspešnosť rozpoznávania značiek na detekciu pohybu a ich správne sledovanie v čase. Ďalšou významnou časťou bolo správna identifikácia značiek tak, aby korešpondenčné značky z obrázkov prvej a druhej kamery boli priradené správne. K tomu som skúsil využiť rôzne algoritmy a prístupy pre čo najvyššiu úspešnosť. Poslednou testovanou časťou bola úspešnosť a hlavne presnosť triangulácie, teda získavania hĺbkovej súradnice. Úspešnosť tejto operácie je veľmi závislá na správnom priradení korešpondenčných bodov, preto som ladeniu a testovaniu tej časti venoval zvýšenú pozornosť a čas.
Ďalším faktorom, ktorý do značnej miery vplýval na dosiahnuté výsledky bolo úroveň a spôsob osvetlenia rekonštruovanej scény, preto sa na túto problematiku zameral pri niektorých testoch tiež.

8.1 Pracovné nástroje

Na vytvorenie stereo kamerového systému som využil dve obyčajné webové kamery. Tieto kamery neboli príliš veľkej kvality, čo sa podpísalo aj pod niektoré výsledky pri tomto testovaní.

Na zaznamenanie pohybov som ako detekčné značky použil sadu loptičiek na stolný tenis. Tieto loptičky som nafarbil na zeleno, pretože som túto farbu zvolil ako tú, ktorú sa mi darilo detekovať s najväčšou úspešnosťou. Tieto loptičky som potom upevnil na tmavé oblečenie, pretože implementovaný systém je v dosť veľkej citlivý na odrazené svetlo, a svetlejšie oblečenie tieto podmienky zhoršovalo.

8.2 Úspešnosť detekcie značiek pri rôznej vzdialenosti

Ako prvú časť programu som testoval úspešnosť detekcie značiek a pokúsil sa zistiť najefektívnejšiu vzdialenosť od kamier a aj maximálnu akceptovateľnú vzdialenosť.

Chýbajúce značky	0	1	2	3	4	5	Celkovo chybných rámcov	Úspešných rámcov
Kamera 1	254	5	0	0	0	0	5	98,06%
Kamera 2	252	7	0	0	0	0	7	97,30%
Celkovo							12	95,37%

Tabuľka 8-1 Úspešnosti detekcie značiek zo vzdialenosti 0,45 metra.

Tabuľka 8-1 ukazuje úspešnosť detekcie 5 značiek zo vzdialenosti 0,45 metra. Z celkového počtu 259 rámcov tak bol správny počet značiek nájdený v 247 rámcoch, keďže je potrebné, aby boli všetky značky nájdené na oboch kamerách. Preto sa od počtu správnych rámcov odpočítavajú všetky rámce

s chybne zaznamenanými značkami z oboch kamier. Zo vzdialenosti 0,45 metra je teda detekcia značiek úspešná v 95,37% rámcov.

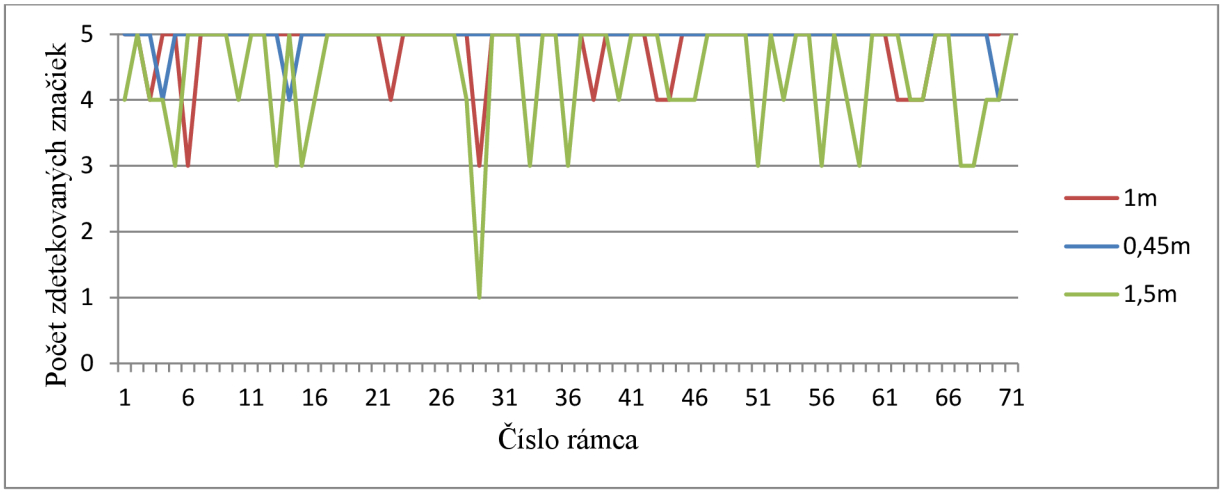
Chýbajúce značky	0	1	2	3	4	5	Celkovo chybných rámcov	Úspešných rámcov
Kamera 1	144	13	2	0	0	0	15	90,57%
Kamera 2	147	12	0	0	0	0	12	92,45%
Celkovo							27	83,01%

Tabuľka 8-2 Úspešnosti detekcie značiek zo vzdialenosti 1 metra.

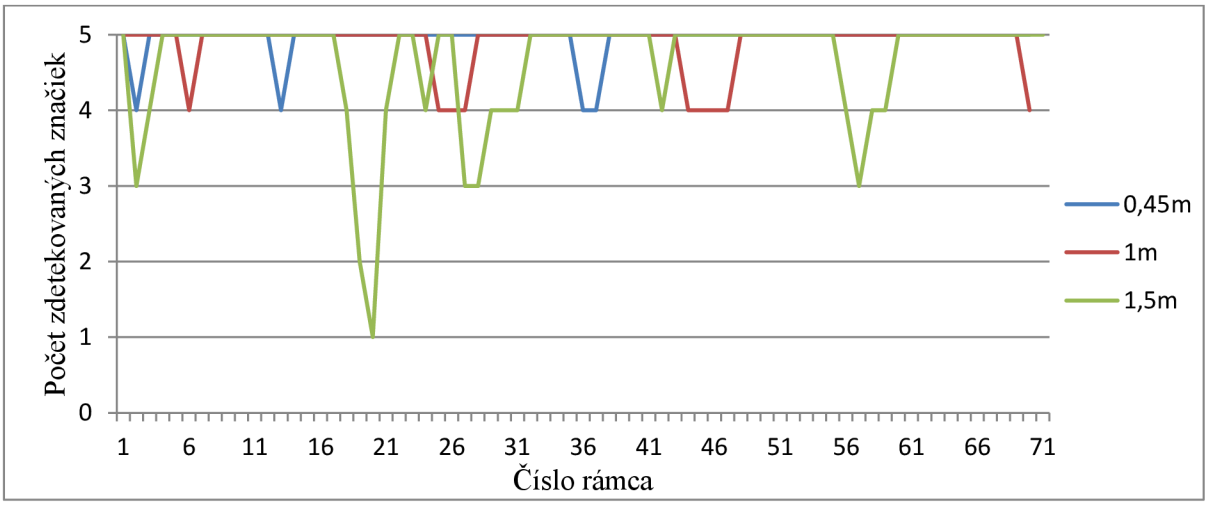
Zo vzdialenosti 1 metra už úspešnosť o niečo nižšia, ako ukazuje Tabuľka 8-2. Z celkového počtu 159 rámcov boli značky správne zaznamenané na 144, čo predstavuje 83,80%. Úspešnosti jednotlivých kamier ešte nie sú také nízke, stále presahujú 90%, ale po sčítaní rámcov s neúspešne zdetekovanými značkami je to už značná časť celkového počtu rámcov, a na plynulý beh rekonštrukcie už má táto chybovosť nezanedbateľný vplyv. Tento pokles úspešnosti je prevažne spôsobený nižšou kvalitou kamier. Kamery neboli spoľahlivo schopné rozoznať objekty veľkosti loptičiek na stolný tenis pri náraste vzdialenosti. To sa ešte viac prejavilo pri testovaní detekcie značiek zo vzdialenosti 1,5 metra, ako ukazuje Tabuľka 8-3, kde podiel pre rekonštrukciu použiteľných rámcov, teda so správnym počtom zdetekovaných značiek, je len 64,78%.

Chýbajúce značky	0	1	2	3	4	5	Celkovo chybných rámcov	Úspešných rámcov
Kamera 1	180	30	15	3	2	0	50	78,26%
Kamera 2	199	23	6	1	1	0	31	86,52%
Celkovo							81	64,78%

Tabuľka 8-3 Úspešnosti detekcie značiek zo vzdialenosti 1,5 metra.



Graf 8-1 Úspešnosť detekcie značiek pri rôznych vzdialenostiach počas prvých 70 rámcov u prvej kamery.



Graf 8-2 Úspešnosť detekcie značiek pri rôznych vzdialenostiach počas prvých 70 rámcov u druhej kamery.

Na grafoch Graf 8-1 a Graf 8-2 je možné vidieť priebeh detekovania značiek pre každú z kamier zvlášť. Z dôvodu prehľadnosti grafov je na nich zaznamenaný priebeh len počas prvých 70 rámcov.

8.3 Úspešnosť vyhľadania korešpondenčných bodov

Ďalšou testovanou zložkou bolo priradzovanie odpovedajúcich bodov z rámcov jednej kamery na body rámcov druhej kamery. Správne priradenie odpovedajúcich bodov je veľmi dôležité pre správne výstupy triangulácie, a pri nesprávnom poradí korešpondenčných bodov sa výsledok rekonštrukcie vzďaľuje realite.

Na otestovanie úspešnosti tejto časti programu som použil značky uchytené na kruhový podnos, ktorým som pred kamerami pomaly otáčal o 720°. Rámce som zaznamenával približne každú polovicu sekundy. Týmto spôsobom sa značky vzájomne dostali do rôznych polôh voči kamerám, aj epipolárnym rovinám. Z výsledkov tohto testu som okrem spomínaného priradzovania bodov zaznamenal aj úspešnosť detekcie značiek a úspešnosť sledovania značiek pomocou Kalmanových filtrov. Tento test som opakoval 2-krát a výsledky zaznamenal do následných tabuliek.

Detekcia značiek				
Chybné rámce		Kamera 1	Kamera 2	Úspešnosť
	1. pokus	7 z 85	6 z 85	84,70%
	2. pokus	1 zo 110	1 zo 110	98,18%

Tabuľka 8-4 Úspešnosť detekcie značiek pri kružnicovom teste.

Priradzovanie korešpondenčných značiek			
	Počet rámcov	Chybné priradenia	Úspešnosť
1. pokus	85	0	100%
2. pokus	110	0	100%

Tabuľka 8-5 Úspešnosť priradzovania odpovedajúcich značiek.

Priradzovanie značiek prebehlo veľmi úspešne v oboch testoch, ako ukazuje Tabuľka 8-5. Jedno zlé priradenie sa síce vyskytlo, ale to bolo spôsobené jednou zlou detekciou značiek zaznamenanou medzi chybami v Tabuľka 8-4, preto ju už v tomto prípade nerátam medzi chyby.

Veľmi úspešne dopadlo aj sledovanie objektov Kalmanovými filtrami. Tento test som medzi priradzovanie značiek zaradil preto, lebo dôležitou časťou správneho sledovania v mojom programe je práve priradzovanie filtrov k značkám podľa súradníc. Úspešnosť sledovania môžeme vidieť v Tabuľka 8-6.

Sledovanie objektov Kalmanovými filtrami			
	Počet rámcov	Chybné priradenia	Úspešnosť
1. pokus	85	0	100%
2. pokus	110	2	98,18%

Tabuľka 8-6 Úspešnosť sledovania Kalmanovými filtrami.

8.4 Presnosť triangulácie

Ďalšou časťou testov je získanie presnosti procesu triangulácie. Pomocou triangulácie sa program pokúša o 3D rekonštrukciu scény.

Prvý test na trianguláciu sa zameriava na presnosť získanej hĺbkovej súradnice z. Pomocou metra som značku umiestnil presne 1m od kamier a zaznamenával som počas niekoľkých rámcov hodnotu súradnice z, ktorú mi vracala funkcia pre trianguláciu (Obrázok 8-1). Keďže som pri kalibrácii kamier dodal programu veľkosť jedného na šachovnici v centimetroch (Obrázok 7-1), mala by triangulácia vracať vzdialenosť rovnako v centimetroch. Teda správne riešenie by sa malo čo najviac približovať číslu 100. Pokus som opakoval 2.krát. Zaznamenané výsledky je možné vidieť v Tabuľka 8-7.

Číslo rámca	1. meranie	2. meranie
1	98,47	96,4
2	99,34	96,77
3	105,85	96,5
4	105,85	100,07
5	104,98	93,87
6	88,84	93,87
7	98,8	93,65
8	102,8	96,86
9	102,8	93,62
10	99,02	96,23
11	94,12	99,65
12	100,65	102.29
Priemer	100,12	96,13

Tabuľka 8-7 Priemerná vzdialenosť daná trianguláciu pri meraní metrovej vzdialenosti.

Hoci sú výsledky pomerne presné, systém bol veľmi citlivý na posunutie na zvislej osi y. Pri zvýšenom posunutí značky smerom nadol, vzdialenosť daná trianguláciou začala od správneho riešenia divergovať.

Obrázok 8-1 Testovanie triangulácie metrovou vzdialenosťou od kamery.

Ďalším testom bolo porovnanie reálnej veľkosti a veľkosti v súradniciach po triangulácii, podobne ako tomu bolo s testom vzdialenosti od kamier spomenutým vyššie (Tabuľka 8-7). V tomto prípade som mal značky vzdialené od seba 25 centimetrov a vo vodorovnej polohe. Vzdialenosť daná tento

krát rozdielom súradníc na osi x by sa mala čo najviac približovať číslu 25. Výsledky testu je možné vidieť v Tabuľka 8-1.

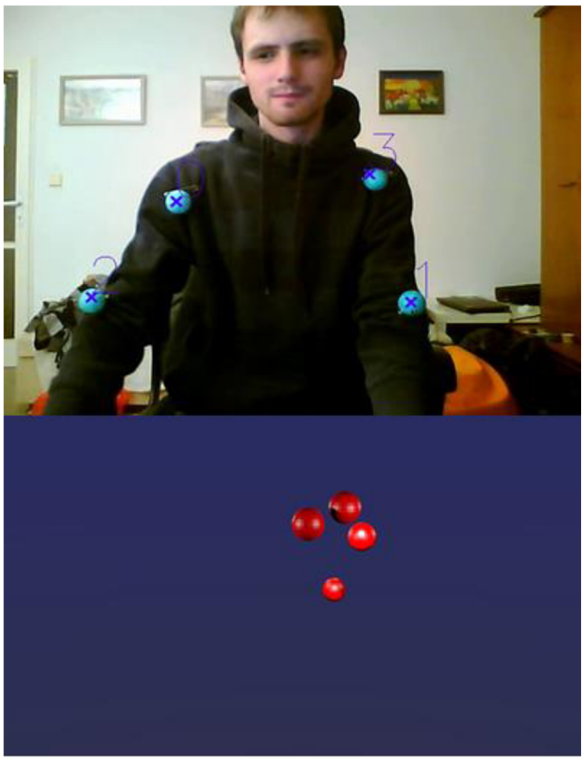
Číslo rámca	Súradnica na osi x prvá značka	Súradnica na osi x druhá značka	Vzdialenosť medzi značkami
1	-18,87	3,52	22,39
2	-19,08	2,91	22,34
3	-20,64	3,26	23,9
4	-21,31	2,25	23,56
5	-21,04	2,14	23,18
6	-22,92	2,05	24,97
7	-23,29	1,83	25,12
8	-22,95	2,77	25,72
9	-22,83	2,21	25,04
10	-23,64	2,37	26,01
11	-22,62	2,37	24,99
12	-21,58	2,55	24,13
Priemer	-21,73083333	2,519166667	24,27916667

Tabuľka 8-8 Testovanie triangulácie meraním vzdialenosti medzi značkami.

8.5 Úspešnosť rekonštrukcie na základe bodov daných triangulániou

Rekonštrukcia pohybu bola vykonaná vykresľovaním 3D bodov získaných triangulániou pomocou programu Blender. Presnosť nebola ideálna, súradnice dané triangulániou neboli vždy presné. To môže byť spojené s odlišnou rýchlosťou načítavania rámcov u jednotlivých kamier, ako aj kalibračnou chybou.

Na dodaných videách je na rekonštruovaných bodoch možné vidieť aj chaotické chvenie, ktoré kazí presnosť rekonštrukcie. To je spôsobené tým, že značky sú v rozličných rámcoch detekované nie vždy plnohodnotne, a tak sa stred týchto značiek posúva o niekoľko pixelov rôznymi smermi. Príklady výsledkov rekonštrukcie je možné vidieť na Obrázok 8-2 a Obrázok 8-3.



Obrázok 8-2 Prvý príklad rekonštrukcie nasnímaného pohybu.

Obrázok 8-3 Druhý príklad rekonštrukcie nasnímaného pohybu .

9 Záver

V tejto práci som sa zaoberal zostavením stereo kamerového systému a jeho kalibráciou, zaznamenávaním pohybu pomocou tohto systému a jeho následnou 3D rekonštrukciou. Po preštudovaní doterajších prác s podobnou tematikou som sa rozhodol implementovať systém zachytávania pohybu pomocou pasívnych značiek, ktoré rozoznávam na snímkach na základe ich farieb. Implementovaný systém som otestoval.

V testoch som sa zameral na viaceré oblasti projektu. Pri overovaní spoľahlivosti detekcie značiek som dosiahol úspešnosť 95,37% správne vyhodnotených snímok na krátku vzdialenosť trištvrte metra. Pri vzdialenosti jedného metra bola dosiahnutá nižšia úspešnosť 83,01%. Len 64,78%. bolo dosiahnutých pri vzdialenosti 1 a pol metra od kamerového systému. Vysoké úspešnosti boli dosiahnuté pri sledovaní objektov v čase, 99% a aj pri identifikácii korešpondenčných bodov na snímkach z kamier, ktorá bola 100% pri špecifických testoch , a o niečo nižšia pri vyhodnotení vzorkového videa. Rovnako špecifické testy na presnosť triangulácie dopadli uspokojivo, keď program určil metrovú vzdialenosť od kamier v priemere na 96,13cm a veľkosť meraného objektu o veľkosti 25cm v priemere na 24,27 cm. Ale rovnako aj tu, pri výslednej rekonštrukcii vzorkového videa boli výsledky horšie. Dojem kazí aj chaotické chvenie bodov pri rekonštrukcii, to je ale spôsobené obmedzenou presnosťou detekcie stredu bodu na základe filtrácie farby.

Myslím, si, že nižšia úspešnosť pri získavaní 3D súradníc je z veľkej časti spôsobená nižšou kvalitou vybavenia, hlavne webových kamier. Rovnako aj nízka úspešnosť detekcie značiek pri zväčšenej vzdialenosti by mohla byť zlepšená kvalitnejšími kamerami ako aj lepšie zvolenými značkami.

Na základe tejto práce a jej výsledkoch som zistil, že je do určitej miery možné vytvoriť systém na rekonštrukciu pohybu na princípe dvoch obyčajných webových kamier a farebných značkách na sledovanom objekte. Pre určité obmedzenia a nepresnosti v meraní tento systém nie je vhodný na technické využitie, ale jeho koncept by sa dal využiť pri zostrojovaní podobného systému s lepším vybavením.

Čo sa týka ďalšieho vývoja projektu, rád by som vylepšil celkovú úspešnosť rekonštrukcie scény a presnosť systému využitím lepších značiek, napríklad pomocou LED diód, ktoré by umožňovali detekciu na efektívnu vzdialenosť väčšiu ako poskytuje detekcia na základe farby použitá v tomto projekte . Taktiež by sa mohla zvýšiť presnosť triangulácie využitím väčšieho počtu kamier vyššej kvality a ich pevným zostrojením do systému napríklad pomocou statívov.

Literatúra

[1] Bradski, G., Kaehler, A.: *Learning OpenCV: Computer vision with the OpenCV library*. CA: O'Reilly, Sebastopol, 2008. ISBN 978-0-596-51613-0

[2] Poling, B.: *A tutorial on camera models* [online]. Dostupné na URL: http://math.umn.edu/~poli0048/CameraModels.pdf

[3] Blanárik, I.: *Epipolárna geometria* [online]. Slovenská technická univerzita v Bratislave, Fakulta informatiky a informčných technológií, 2005. Dostupné na URL: http://labss2.fiit.stuba.sk/TeamProject/2005/team05/doc/clanok_IB.pdf

[4] Hartley, R., Zisserman, A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, 2003. ISBN 0521540518. Dostupné na URL: http://dl.acm.org/citation.cfm?id=861369

[5] Keywook, L.: *Application of the Hough transform* [online]. Lowell: University of Massachusetts, 2006. Dostupné na URL: http://wenku.baidu.com/view/bbfafd07e87101f69e3195ba.html

[6] Kazík, M.: *Houghova transformace pro detekci kružnic.* Brno, 2009. Bakalářská práce. Vysoké učení technické v Brně, Ústav telekomunikací. Dostupné na URL: https://dspace.vutbr.cz/handle/11012/11704

[7] Fisher, R., Perkins, S., Walker, A., Wolfart, E.: *The Hypermedia Image Processing Reference* [online]. Wiley, 1997. ISBN 978-0471962434. Dostupné na URL: http://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm

[8] Ftáčnik, M.: *Matematická morfológia* [online]. Dostupné na URL: http://www.google.cz/url?sa=t&rct=j&q=erozia%20%20ad%20dilatacia%20%3Ask&source=web&cd=2&ved=0CDIQFjAB&url=http%3A%2F%2Fsccg.sk%2F~ftacnik%2FIP-8.pdf&ei=VnhzU96qLe_X7AaByIHIBQ&usg=AFQjCNH0RHF9vKstdb3pMtZGP9zpVKWzFw&sig2=8KRmC_D_o-aernvyBjNbrw

[9] *Lucas Kanade optic flow* [online]. Dostupné na URL: http://www.curvace.org/index.php?option=com_content&view=article&id=104%3Alucas-kanade-optic-flow&catid=57&Itemid=94

[10] *Kalmanov filter* [online]. Katedra leteckej technickej prípravy, 2009. Dostupné na URL: http://www.senzorika.leteckafakulta.sk/?q=node/269

[11] Krištof, M.: *Barevný model* [online]. 2007. Dostupné na URL: http://www.dmp.spsei.cz/digi/model.php

[12] *Camera calibration* [online]. Dostupné na URL: http://www.ics.uci.edu/~majumder/docs/cameracalib.pdf

[13] *OpenCV 3.0.0-dev documentation* [online]. 2011. Dostupné na URL: http://docs.opencv.org/trunk/index.html

[14] Kirk, A. G., O'Brien, J. F., Forsyth, D. A.: *Skeletal Parameter Estimation from Optical Motion Capture Data* [online]. San Deigo, 2005. Dostupné na URL: https://buffy.eecs.berkeley.edu/PHP/resabs/resabs.php?f_year=2006&f_submit=one&f_absid=101318

[15] *Two-view geometry* [online]. 2010. Dostupné na URL: http://www.docstoc.com/docs/48455210/Two-view-geometry

Príloha – obsah cd

Source/ - adresár so zdrojovými súbormi
DataVideos/ - adresár so zdrojovými video nahrávkami
Calibration/ - adresár s kalibračnými fotkami
OutVideos/ - adresár s videami obsahujúcimi rekonštrukciu pohybu
Poster/ - adresár s prezentačným plagátom