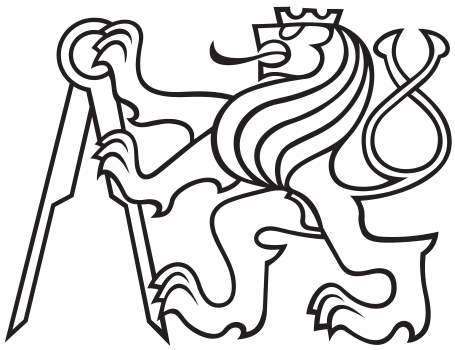


ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra radioelektroniky

Analýza videozáznamu pohybu pacientů během epileptického záchvatu

Movement Analysis of Video Footage of Patients During Epileptic Seizure

Bakalářská práce

Studijní program: Komunikace, multimédia a elektronika

Studijní obor: Multimediální technika

Vedoucí práce: Ing. Petr Ježdík, PhD.

Jiří Šebek

Praha 2017

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Šebek** Jméno: **Jiří** Osobní číslo: **434637**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra radioelektroniky**

Studijní program: **Komunikace, multimédia a elektronika**

Studijní obor: **Multimediální technika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Analýza videozáznamu pohybu pacientů během epileptického záchvatu

Název bakalářské práce anglicky:

Movement Analysis of Video Footage of Patients During Epileptic Seizure

Pokyny pro vypracování:

Prostudujte v praxi používané metody detekce objektů ve videozáznamu. Vyberte nejvhodnější metodu a v programu MATLAB ji upravte, aby byla použitelná pro analýzu pohybu pacienta prožívajícího epileptický záchvat s cílem kvantifikace síly záchvatu pro objektivní parametrizaci semiologie záchvatu.

Seznam doporučené literatury:

Siddiqui, Matheen, and Gérard Medioni. "Robust real-time upper body limb detection and tracking." Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks. ACM, 2006.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Petr Ježdík Ph.D., katedra měření, katedra teorie obvodů, LVR

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **17.02.2017** Termín odevzdání bakalářské práce: **26.05.2017**

Platnost zadání bakalářské práce: **31.08.2018**

Podpis vedoucí(ho) práce	Podpis vedoucí(ho) ústavu/katedry	Podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

Datum převzetí zadání Podpis studenta

Abstrakt

Tato práce se zabývá možnostmi automatizované analýzy pohybu pacienta, který prodělává epileptický záchvat, aby bylo možné objektivně určit vývoj nemoci. V teoretické části rozebereme existující způsoby trasování částí lidského těla a objektů v obraze a jejich použitelnost pro případ videozáznamu snížené kvality.

Praktická část se zabývá návrhem algoritmu pro zpracování polohových dat sledovaného objektu a řešením v případech, kdy sledování selže. Návrh bude dále implementován v prostředí Matlab a ověřován nejprve na ideálních testovacích videozáznamech a později na skutečných záznamech dlouhodobě hospitalizovaných pacientů s epilepsií.

Klíčová slova

Trasování objektů, počítačové vidění, Matlab, RANSAC, KLT, epilepsie

Abstract

This thesis deals with various methods of automatized analysis of a patient having an epileptic seizure for objectively define state of the illness. In theoretical part we discuss existing methods of tracking human body parts in a video and whether they are usable for our case.

Practical parts introduces an algorithm propose for processing positioning data of a tracked object and solution for tracking failures. Algorithm will then be implemented in Matlab software and tested by variety of videos, starting with single simple movements and ending with real recordings of long-term hospitalized patients.

Keywords

Object Tracking, Computer Vision, Matlab, RANSAC, KLT, Epilepsy

Prohlášení

Prohlašuji, že jsem bakalářskou práci na téma Analýza videozáznamu pohybu pacientů během epileptického záchvatu zpracoval samostatně za použití uvedené literatury a pramenů.

Dále prohlašuji, že nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne …… ……

Podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Petru Ježdíkovi, PhD. za účinnou pomoc metodickou, odbornou i pedagogickou a Fakultní nemocnici Motol za poskytnutí materiálů pro testování vytvořených algoritmů. Děkuji rovněž všem figurantům za pomoc při tvorbě testovacích videí.

Obsah

Seznam obrázků

Seznam zkratek

ROI – Region of Interest

GUI – Graphic User Interface

Bbox – Bounding Box

EEG – Elektroencefalogram

AVI – Audio Video Interleave

KLT – Kanade-Lucas-Tomasi

MSS – Minimal Sample Set

RANSAC – Random Sample and Consensus

DSLR – Digital Single-Lens Reflex

1 ÚVOD

Epileptické záchvaty již od starověku postihují přibližně 1% populace. V dnešní době se pro určení diagnózy a léčby používá sémiologie záchvatů. Pacient je dlouhodobě hospitalizován a je pod neustálým dohledem videokamer. Doktoři poté sledují záznamy a na základě toho, jak se pacient chová, stanovují diagnózu, léčbu, v případě již probíhající léčby její úspěšnost. Vyhodnocování stavu, respektive vývoje stavu pacienta je především subjektivní záležitostí. Doktor se nemůže opřít o přesná polohová data jednotlivých končetin nebo částí těla, jako například frekvence kmitání rukou, rozkmit nebo prohnutí. Metoda je také kvůli množství získaného videozáznamu časově velmi náročná.

Abychom získali časový vývoj pohybu končetin, respektive částí těla, aniž bychom museli každý jednotlivý snímek videa přesně mapovat, vytvoříme program, který dokáže automaticky sbírat polohu končetin v průběhu celého videa. Jádrem programu bude algoritmus, který dokáže detekovat klíčové rysy obrazu a poté sledovat změny jejich umístění. Součástí programu bude také možnost opětovné inicializace při selhání a závěrečné výpočty polohových údajů.

K implementaci vhodného algoritmu využijeme vývojové prostředí Mathworks Matlab R2014b rozšířený o Image Processing toolbox a Computer Vision toolbox. První zmíněný je specializovaný na práci s obrazy, umožňuje úpravy jako ve fotografických editorech, druhý zmíněný toolbox obsahuje předpřipravené základy algoritmů pro detekci snadno odlišitelných rysů obrazu a jejich následné sledování. Tyto algoritmy upravíme, aby byly použitelné pro videozáznamy, které máme k dispozici.

Další kapitola této práce bude obsahovat popis a výsledky testů námi sestaveného programu. Testování začneme na videozáznamech figurantů zachycených v ideálních podmínkách, tedy ve vysokém rozlišení, při dostatečném osvětlení a kamerou kolmo k figurantovi. Testovací videa budou mít vzestupnou komplexitu; nejprve se bude jednat o samostatné pohyby, poté jejich kombinace, dále snížení kvality videí a nakonec budeme testovat reálné záběry pacientů z Fakultní nemocnice Motol.

Závěrem této práce bude interpretace výsledků a s ní spojená úvaha, nakolik je algoritmus použitelný pro analýzu pohybu pacienta na lůžku, co všechno může metoda poskytnout a jakým směrem by bylo vhodné či nevhodné v jejím rozvoji pokračovat.

2 SNÍMÁNÍ POHYBU

2.1 Definice a využití

Snímáním pohybu rozumíme proces nebo techniku zaznamenávání pohybu digitálně. Tyto techniky se hojně využívají především při vytváření animovaných postav ve filmech nebo videohrách. V poslední době nachází využití i ve zdravotnictví, především pro analýzu pohybu nebo hodnocení vývoje rehabilitace pacientů s poškozením nervového nebo motorického ústrojí.

Mezi nejpoužívanější systém snímání pohybu patří optický. Objekt, nejčastěji herec, je označen retroflexivními značkami, které odráží nebo samy vyzařují světlo do kamery, čímž je snímána poloha objektu vůči kameře. Pro tyto účely se používá několik červených nebo infračervených kamer, objekt/osoba má jednobarevný (nejčastěji černý) oblek a je snímán(a) před zeleným pozadím.

2.2 Detekce rohů

Snímání pohybu optickými systémy je běžně rozšířené. Nevýhodou těchto systémů je nutnost, aby v každém okamžiku byla každý značka viděna alespoň dvěma kamerami. Dále je potřeba speciální oblek, který z finančních a etických důvodů vylučuje použití v nemocnici. Potřebujeme systém, který dokáže sledovat člověka v běžném oděvu jedinou kamerou.

Využijeme detekce rysů (v anglicky psané literatuře *feature*) obrazu. V teorii strojového učení a rozpoznávání obrazců je rys individuální měřitelný prvek či vlastnost pozorovaného jevu [2]. Pro zajištění efektivity algoritmu musí být rys snadno rozlišitelný a nezávislý. Mezi typické rysy řadíme okraj, roh, „blob“, linii a bod.

Úkolem algoritmu bude sledovat části lidského těla ve videozáznamu s nízkým rozlišením. V těchto podmínkách zabírají části těla maximálně desítky pixelů a okraje prstů, předěly mezi prsty, obočí, nosní dírky nebo oči mají proto relativně ostré okraje. Pro naše účely tedy nejlépe poslouží detekce rohů. Roh je definovaný jako průsečík dvou okrajů, může být také definován jako bod, pro který existují dva dominantní a odlišné směry okraje v sousedství bodu. Nezáleží, jestli jsou okraje světlé (mají hodnotu téměř 1 ve všech barevných kanálech) a jejich okolí tmavé (hodnoty okolo 0) nebo naopak, klíčový pro spolehlivou detekci je kontrast. Další výhodou rohů je, že zatímco například svislá konstantní linie bude při pohybu po ose Y po celé délce stejná a algoritmus tedy může chybně označit její jinou část jako detekovanou transformaci předchozí, roh je při jakémkoliv posunu a rotaci jasně určitelný.

2.4 Algoritmy pro detekci rohů

2.4.1 Moravcův operátor

Tento algoritmus patří ke starším a nejjednodušším detekcím rohů. Určuje významnost bodu pouze na základě co největší odlišnosti malé části obrazu od nejbližšího okolí. Princip Moravcova operátoru vychází z rovnice (1), kde funkce *g*(i,j) představuje vstupní snímek ve stupních šedi a *f*(i,j) výsledný obraz, který reprezentuje průměrný rozdíl jasu bodu v porovnání se všemi jeho osmi sousedy.

$$f(i,j) = \frac{1}{8}\sum_{k=i-1}^{k=i+1}\sum_{e=j-1}^{e=j+1}|g(k,e) - g(i,j)| \qquad (2.1)$$

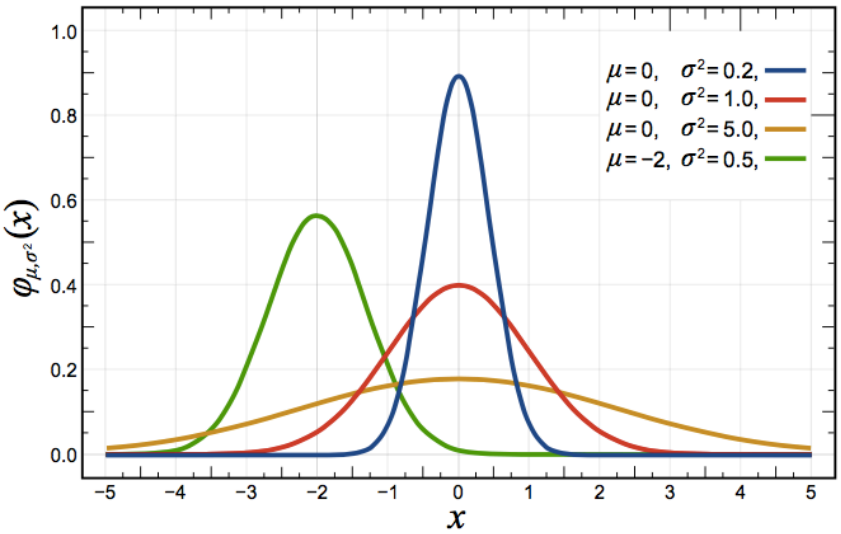
Tento výpočet se provede pro všechny body obrazu a tím vytvoří takzvanou mapu rohovitosti. Při výpočtech je třeba ošetřit případy, ve kterých bude *k* nebo *e* mimo oblast snímku, speciální vyjímkou, jinak by došlo ke značnému zkreslení okrajů výsledného snímku [3].

2.4.2 Harrisův operátor

Jedná se o zdokonalenou verzi Moravcova operátoru, v současné době se jedná o jednu z nejpoužívanějších metod. Její výhodou je, že je rotačně invariantní. Moravcův operátor používá čtvercové okénko pro vyhledávání rysů, Harrisův operátor volí pro vyhledávání jako okénko Gaussovu funkci. Je dána vztahem (2) a popsána na obrázku 2.3.

$$f(x) = ae^{\frac{(x-\mu)^2}{2\sigma^2}} \qquad (2.2)$$

Čísla *a* a σ (2) musí být kladná, *e* je Eulerovo číslo. V bodě x = μ má funkce vrchol, σ určuje výšku a zároveň strmost funkce. Integrál z této funkce podle *x* je roven jedné.



Obrázek 2.1 – graf Gaussovy funkce pro vybrané σ a μ

Rohový detektor posouvá okénko W vždy o polovinu jeho velikosti v okolí každého pixelu kromě hraničních pixelů. Nevýhodou Harrisova detektoru je závislost na změně měřítka snímku. Například při videozáznamu postupného zoomování na objekt se jeho hrany zjemňují; místo ostrých rohů máme plynulé křivky. Při stejné velikosti okénka W se v takto zblízka snímaném objektu vytvoří více detekcí tam, kde při nízkém rozlišení jediná [4].

2.5 KLT algoritmus

Pomocí Harrisova detektoru získáme ve snímku množství detekovaných rohů, které zaznamenáme jako body o souřadnicích *x* a *y*. Tyto rysy mění v každém snímku svoji polohu, někdy i natočení nebo velikost (při přibližování). Pokud bychom potřebovali v každém snímku lokalizovat daný objekty obsahující tyto charakteristické rysy, vytvořili bychom for smyčku a pro každý snímek provedli Harrisovu detekci. Pro naše účely ale potřebujeme znát vývoj objektu a k tomu je zapotřebí používat jednou detekované body a ty sledovat. K tomu nám poslouží Kanade-Lucas-Tomasi (dále jen KLT) algoritmus.

Registrace obrazů dříve spočívala v prohledání celého obrazu za účelem nalezení hodnoty vektoru neslučitelnosti **h**, reprezentující rozdíl mezi 2 snímky. Lucas a Kanade přišli s myšlenkou lokálního vyhledávání pomocí gradientů vážených aproximací druhého derivátu obrazu. Pokud uvažujeme, že *h* je posuv mezi dvěma obrazy $F(x)$ a $G(x) = (x + h)$, tak aproximace bude

$$F'(x) \approx \frac{F(x+h) - F(x)}{h} = \frac{G(x) - F(x)}{h} \qquad (2.3)$$

a tedy

$$h \approx \frac{G(x) - F(x)}{F'(x)} \qquad (2.4)$$

Tato aproximace ke gradientu obrazu je přesná pouze pro malé posuvy o obraze. Aproximované *h* závisí na *x*. Pro kombinace více odhadů *h* při různých *x* dochází k průměrování

$$h \approx \frac{\sum_x \frac{G(x) - F(x)}{F'(x)}}{\sum_x 1} \quad . \qquad (2.5)$$

Průměrování může být dále zpřesněno přidělením váhy každému příspěvku, což je nepřímo úměrné odhadovanému $|F''(x)|$, kde

$$F''(x) \approx \frac{G'(x) - F'(x)}{h} \quad . \qquad (2.6)$$

Pro zjednodušení je definována váhovací funkce:

$$w(x) = \frac{1}{|G'(x) - F'(x)|} \quad . \qquad (2.7)$$

Vážený průměr potom dostáváme

$$h = \frac{\sum_x \frac{w(x)[G(x) - F(x)]}{F'(x)}}{\sum_x w(x)} \quad . \tag{2.8}$$

Po získání odhadu lze posunout $F(x)$ podle odhadované h. Procedura je provedena opakovaně. Sekvence odhadů bude ideálně konvergovat k nejpřesnějšímu h.

Tato metoda byla později vylepšena sledováním vhodných rysů, konkrétně rohů a samostatných bodů. Tyto rysy budou vybrány ke sledování, když splní podmínku, že vlastní hodnoty matice gradientu musí být větší než zadaný práh. Pro náš algoritmus použijeme v Matlabu existující algoritmus, který pracuje na bázi výše popsaného KLT algoritmu a jehož výstupem jsou pro každý snímek dvoudimenzionální souřadnice každého detekovaného bodu [6] [7].

2.6 Pohyb ve 3D prostoru

Pomocí Harrisova detektoru a KLT algoritmu získáme pro každý sledovaný objekt skupinu rysů se známou polohou ve 2D snímku. Pro ucelenou analýzu nicméně ještě budeme potřebovat třetí rozměr. Získáme ho výpočtem ze zvětšení bboxu. Pokud chceme přepočítat tento pohyb na jednotky vzdálenosti, musíme znát vzdálenost objektivu od snímané osoby, reálnou výšku osoby a kolik pixelů zabírá v obraze. Nemusí se jednat konkrétně o výšku, ale obecně o jeden z rozměrů, u něhož známe tyto odpovídající hodnoty.

Pro přepočet využijeme princip podobných trojúhelníků, jak ilustruje Obrázek 2.2. Pokud použijeme model jednobodové perspektivy, díky podobnosti vychází

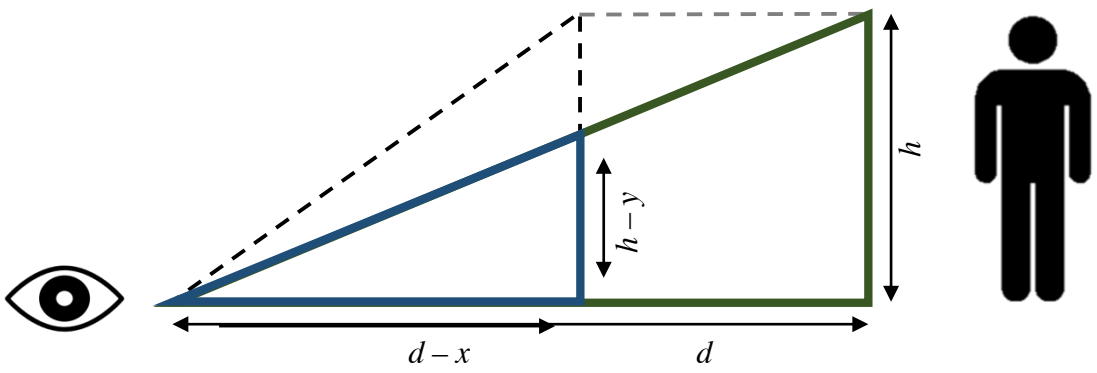
$$\frac{d}{d-x} = \frac{h}{h-y} \quad . \tag{2.3}$$

V této rovnice se vyskytují 2 neznámé x a y a 2 parametry d a h. Využijeme tedy známého zvětšení objektu, které získáme z výpočtu vzdáleností jednotlivých rysů od středu bboxu. Tento koeficient zvětšení nazveme A a platí, že

$$A = \frac{h}{h-y} \quad . \tag{2.4}$$

Výsledná rovnice pro výpočet tedy bude mít tvar

$$d - x = \frac{d}{A} \quad . \tag{2.5}$$



Obrázek 2.2 – Ilustrace podobnosti trojúhelníků při perspektivním vnímání

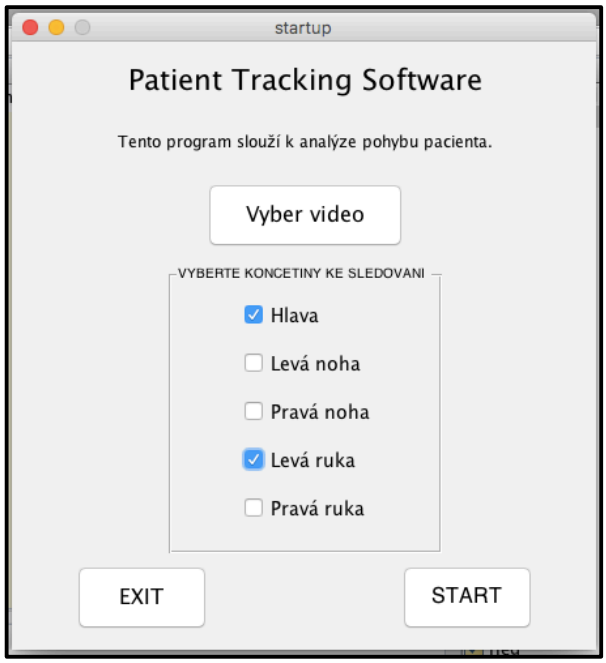
3 IMPLEMENTACE

Výše uvedené algoritmy a metody nyní implementujeme ve vývojovém prostředí Matlab. Pro detekci hran a jejich sledování napříč videozáznamem vytvořila společnost Mathworks specializovaný balíček nástrojů s názvem "Computer Vision Toolbox", jehož systémové objekty budou tvořit jádro našeho programu. Program musí být snadno ovladatelný a musí umožňovat reinicializaci sledování, protože vlastnosti videozáznamů jsou vzdálené od ideálních. Funkčnost a použitelnost programu budeme testovat nejprve sérií idealizovaných videí s přibývající „obtížností" a poté reálnými záznamy.

Náš program umožňuje sledování 1 – 5 objektů (definujeme pomocí GUI, viz oddíl 3.1). Pro zjednodušení a značné zkrácení kódu definujeme třídu `tracker`, jejíž instance budou vytvořeny pro každý sledovaný objekt a budou sloužit jako úložiště většiny proměnných. Program je rozdělený do samostatně spustitelných oddílů pro výběr videa, jeho úpravy, vytvoření sledovaných bodů, samotná analýza a zpracování analyzovaných dat.

3.1 GUI

Pro dosažení maximálního možného uživatelského komfortu a jednoduchosti ovládání je prvotní nastavení dělané pomocí grafického uživatelského rozhraní (dále jen GUI). GUI můžete vidět na obrázku 3.1. Uživatel nejprve zvolí, jaký videozáznam chce analyzovat a poté navolí, jaké končetiny bude sledovat. Každé zaškrtnuté políčko vytvoří objekt `tracker`, který obsahuje identifikaci objektu a prázdné matice pro zapisování polohových údajů, počtu chyb a pomocných výpočtů (kompletní popis v oddílu 3.3.1). GUI rovněž disponuje možností přerušení operace.



Obrázek 3.1 – Grafické uživatelské rozhraní

3.2 Úpravy videa

Výsledný program bude mít za úkol zpracovávat retrospektivně záznamy hospitalizovaných pacientů. Ti leží nebo sedí na bledě zeleném lůžku ve standardním oblečení, v některých případech s připojenými snímači EEG na hlavě, které jsou omotané obvazy. Videa trpí několika významnými neduhy: mají nízké rozlišení, konkrétně 352x288 obrazových bodů; sledovaný není vždy po celou dobu v záběru v důsledku rozsáhlejších pohybů, špatně nastavené kamery nebo interference jinou osobou; světelné podmínky nejsou konstantní nebo v důsledku nedostatku světla dochází k výraznému šumu; kamera v důsledku velikosti snímače neposkytuje příliš ostrý obraz. Tyto neduhy výrazně snižují šance na úspěšné sledování pacienta. Interference jiných osob či část objektu mimo záběr nezměníme, ale kvalitu obrazu můžeme pro naše účely výrazně vylepšit.

3.2.1 Funkce ImageEnhance a ImageEnhanceBig

Matlab samotný neumí pracovat přímo upravovat videozáznam způsobem, jaký známe z editačních programů jako je Adobe Premiere nebo Apple Final Cut Pro. Je tedy potřeba nejprve videozáznam rozdělit na jednotlivé snímky, každý samostatně upravit a nakonec je opět spojit. Pro tyto účely napíšeme funkce `ImageEnhance` a její úpravu `ImageEnhanceBig` s možností změny rozlišení videa.

Funkce `ImageEnhance` nejprve vytvoří složky „original” a „enhanced”, do první zmíněné budou zapisovány původní snímky, do druhé jejich upravená verze. Následuje vytvoření objektů `videoReader` a `videoWriter`. Z prvního zmíněného získáme

původní framerate videozáznamu, který bude použit ve videu novém, druhý zmíněný provede na konci funkce zápis videa ve formátu AVI.

Mezi těmito kroky jsou snímky probíhají úpravy, zaostření a úprava rozlišení. V důsledku malé rychlosti závěrky kamery je pohybující se objekt rozmazaný, tudíž vzhledem k nízkému rozlišení snímku nastávají případy, kdy je například prst zachycen „mezi dvěma pixely" a do videozáznamu přepočítán jako kombinace barvy pozadí a ruky. To vede ke značnému snížení kontrastu mezi rukou a pozadím, v důsledku čehož zaniká sledovaný roh. Proto je třeba provést zvětšení rozlišení interpolací. Zvolíme interpolaci bikubickou, která výstupní hodnotu pixelu počítá váženým průměrem z okolí 4x4 pixelů. Interpolace vykazuje velmi dobré výsledky při dvojnásobném zvětšení a poskytuje nám mnohem lépe zpracovatelný snímek.

Druhým neduhem je nedostatečnou ostrost, tzn. přechody mezi jednotlivými rozdílnými barvami jsou postupné. Funkcí `imsharpen` zkrátíme tyto přechody zvýšením kontrastu mezi rozdílnými barvami, čímž vytvoříme snadněji detekovatelné hrany a rohy.

Pro testovací videa není třeba zvyšovat rozlišení, proto vytváříme dvě funkce, z nichž jedna umožňuje i zvětšení a druhá zaostřuje. Posledním krokem je samotný zápis upraveného videa videa. Protože potřebujeme znát framerate původního videa, abychom mohli sledovat časový vývoj pohybu nezkreslený, musíme využít starší metody zahrnující právě `videoReader`, který je značně pomalá. Stále však v porovnání s obyčejným prohlížením jde o značnou úsporu času, neboť úprava videa po nastavení poměrného zvětšení – pokud je prováděno – pracuje zcela automaticky.

3.3 Inicializace trasování

3.3.1 Výběr oblasti

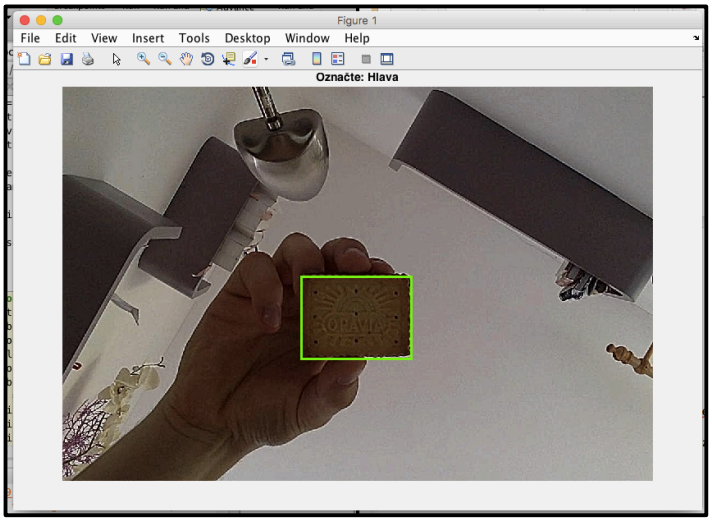
V upraveném videu bude algoritmus hledat snadno odlišitelné rysy, v našem případě rohy. Protože ale nebudeme pracovat s ideálními videozáznamy, musíme předpokládat, že kromě žádaných objektů, jako jsou dlaně a hlava, budou v pohybu i objekty nežádoucí. Navíc je pravděpodobné, že video nezačíná přesně v momentě začátku záchvatu nebo že od začátku záchvatu má pacient několik snímků zakryté končetiny, které později odhalí. Pro tyto účely program obsahuje výběr počátečního snímku a oblastí zájmu (ROI – region of interest). Po spuštění se zobrazí první snímek videozáznamu vybraného pomocí GUI a uživatel může přetočit video o libovolný počet snímků, než se dostane k vhodnému. Poté je mu nabídnut výběr ROI, kde tahem ukazatele vybere obdélník ve snímku tak, aby uvnitř něho byl objekt, který chceme sledovat. Jak vypadá vybírání můžete vidět na obrázku 3.2.

3.3.2 Vyhledání rohových bodů

Ve vybrané oblasti je poté inicializován systémový objekt `vision.pointTracker`, který pomocí Harrisova detektoru (popsán v oddíle 2.3.2) vyhledá v ROI rohy. Tyto rohy jsou zaznamenány do dalšího systémového objektu, `cornerPoints`. Tento objekt po

zavolání metody `Location` vrátí matici m-krát-2 bodů, kde m je počet všech detekovaných bodů v ROI. Tato matice obsahuje souřadnice všech detekovaných rysů v ROI, kterou jsme vymezili obdélníkovým výběrem.

Obdélníkovým výběrem také vytvoříme bounding box (dále jen bbox). Jedná se o obdélník přesně odpovídající našemu výběru, který je definovaný čtyřmi rohovými souřadnicemi. V průběhu sledování se velikost, pozice i natočení bboxu mění (dále popsáno v oddíle 3.4.x) a slouží nám k určení polohy středu sledovaného objektu a jako jeden z indikátorů selhání sledování. Bbox je uložený jako pole o osmi bodech; liché hodnoty znázorňují souřadnice 4 rohů ve směru osy X, sudé souřadnice ve směru osy Y. Z těchto bodů je pro každý snímek funkcí `center` vypočtený střed ROI, viz Obrázek 3.3. Bbox je rovněž zapisován v objektu třídy `tracker`.



Obrázek 3.2 – ukázka výběru sledovaného objektu

```
sx = (bbox(1)+bbox(3)+bbox(5)+bbox(7))/4;
sy = (bbox(2)+bbox(4)+bbox(6)+bbox(8))/4;
souradnice = [sx;sy];
```

Obrázek 3.3 – část kódu funkce `center`

3.3.3 Identifikace bodů

Harrisův detektor a KLT algoritmus implementovaný v systémových objektech Computer Vision toolboxu detekují v ROI rohy a zaznamenají jejich polohu. Tyto algoritmy ale nedokáží přidělit bodům vlastní identitu, která je klíčová pro určení polohy vy směru osy Z – „do/od kamery“ na základě zvětšení objektu a pro určení jeho rotace. Je třeba upravit program, aby body nebyly jen změněny s každým provedením sledovací smyčky, ale by také zapsány do nezávislé proměnné.

Problém zapisování bodů tak, aby se daly identifikovat, je totiž ve způsobu ověřování

jejich validity. Algoritmus vytvořený vývojáři Matlabu v rámci Computer Vision toolboxu vždy neplatný bod z `cornerPoints` objektu odstraní, ale jeho pozici nezaplní nulovou, neplatnou či jinak jasně rozlišitelnou hodnotou, ale nahradí ho v pořadí následující hodnota. Celá matice se tak v případě ztráty *n* bodů zmenší o *n* řádků. V tu chvíli řádek obsahující *x*, respektive *y* souřadnice jednoho rysu dostává hodnotu zcela jiného rysu a vývoj polohy rysu původního již nelze sledovat.

Abychom identitu zachovali, vytvoříme si 2 matice o velikosti *n*-krát-2, kde *n* je počet rysů zaznamenaných při prvotní aplikaci Harrisova detektoru. Matice bude v prvním sloupci obsahovat hodnoty od 1 do *n*, v druhém sloupci *x* souřadnice všech bodů nalezených v prvním snímku (první matice) a v druhé matici obdobně *y* souřadnice. Detekované body každého *k*-tého snímku se tedy zapíšou do *k*+1 sloupce.

3.3.4 Error checking a reinicializace

Jak bylo dříve několikrát uvedeno, program je cílený na videa nízké kvality a rozlišení a my proto očekáváme, že samotný proces sledování bude kolabovat. Aby byl program použitelný, je třeba vytvořit záchytný a reinicializační algoritmus. S pomocí detekovaných bodů stanovíme počáteční stav sledovaného objektu a určíme maximální únosnou změnu, při které bude objekt stále dostatečně přesně zachycen a popsán. V případě nedodržení této hraniční změny bude třeba provést počáteční inicializaci a detekci znovu.

První podmínkou bude počet stále validních bodů. Když počet bodů klesne na příliš nízké hodnoty oproti původnímu stavu, je příliš velká šance že body nebudou rovnoměrně pokrývat sledovanou oblast a dojde k příliš velkému zkreslení při zaznamenávání polohy. Vzhledem k tomu, že podle transformace všech bodů v ROI se transformuje i bbox, nastala by chyba při určení středu. Tím se dostáváme k další podmínce a tou je velikost bboxu. Vycházíme z předpokladu, že libovolný pohyb pacienta nesníží jeho vzdálenost od kamery na méně než polovinu a tedy každý objekt i jeho ROI bude v případě bezchybného sledování vždy méně než dvojnásobně zvětšený oproti počáteční inicializaci. Jako druhou podmínku tedy stanovíme, že pokud se úhlopříčky bboxu zvětší na více než dvojnásobek, považujeme to za selhání sledovacího systému a vyžadujeme reinicializaci.

Pokud chyba nastane, systém zvýší počet zaznamenaných chyb pro daný objekt o jednu a označí snímek, ve kterém se provede reinicializace. Ta je naprogramovaná stejně, jako počáteční inicializace, s malými úpravami. Po detekci bodů ve znovu vybrané ROI, zaznamenání jejich souřadnic do příslušného objektu se resetují původní pomocné proměnné sledování, které slouží jako kontrola, zda máme dostatek informací pro pokračování sledování. Poté je logická proměnná `wrong.objName`, kde objName je unikátní název pro každou končetinu, přepnuta na hodnotu false, počet chyb pro daný objekt zvýšen o 1, zaznamenán snímek, na kterém byla prováděna reinicializace a sledování této končetiny (objektu) běží na chybový režim, ve kterém doběhne do konce videozáznamu, nehledě na další případné reinicializace.

Chybový režim se od standardního liší kontrolou návaznosti nových polohových hodnot detekovaných rysů na rysy snímek staré. Reinicializace totiž vytváří sadu zcela nových rysů s polohovými hodnotami a tak by navázání nebylo možné. S využitím informace, ve kterém snímku došlo k chybě můžeme videozáznam pomyslně rozdělit na bloky nepřerušeného sledování, z nichž každý obsahuje nepoškozené informace o poloze objektu.

Reinicializační systém je klíčový pro použití na neideální videozáznamy. Vhodně zvolenými podmínkami a správně provedeným opětovným výběrem ROI zajistíme kvalitní analýzu i v případech, je nutné inicializaci opakovat víckrát.

3.4 Průběh sledování

Vybrali jsme ROI, v ní jsme detekovali body a přiřadili jim identitu a připravili jsme algoritmus, který v případě selhání umožní pokračovat v analyzování. Můžeme zahájit samotné sledování detekovaných bodů, které v průběhu budou měnit svoji polohu. Využijeme k tomu KLT algoritmus, jehož základ připravený v rámci Computer Vision toolboxu upravíme, aby sbíral veškeré polohové údaje v průběhu videozáznamu.

V Matlabu se pro běh systémových objektů `vision` používá `step` metoda. Nejprve posuneme touto metodou `videoReader` o jeden snímek. Poté provedeme test chybovosti, zda máme dostatečný počet nezkreslených informací ke sledování; pokud ne, v tuto chvíli nastává reinicializace popsaná v oddílu 3.3.4. Následuje aktualizace systémového objektu `tracker`, kde `step` metoda rovněž způsobí posun o jeden snímek videozáznamu a tím se srovná s `videoReader`em. Podle nových údajů ze ze sledovaného objektu jsou aktualizovány polohy validních bodů a ostatní jsou vyřazeny. Do tohoto procesu musíme vstupujeme a zjišťujeme, které body byly vyřazeny a do matic na jejich místo dosadíme hodnotu 0, čímž zachováme identitu všech bodů, více viz 3.3.3. Po této aktualizaci je třeba na základě nových souřadnic bodů transformovat bbox.

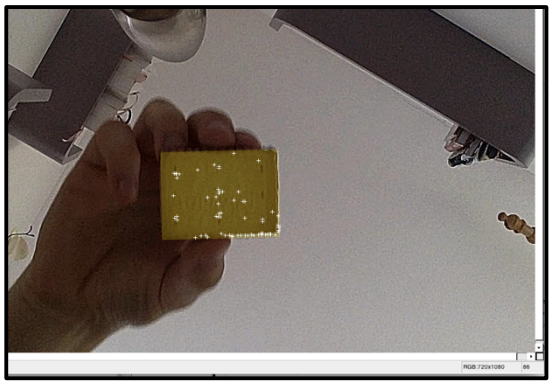
3.4.1 RANSAC a funkce estimateGeometricTransform

Algoritmus jménem RANSAC (= RANdom Sample And Consensus) funguje ve dvou krocích, které se opakují. Prvním z nich je odhad. První minimální sety vzorků (MSS) jsou náhodně vybrány ze vstupních dat a parametry modelu jsou vypočítány pouze z MSS. Počet prvků MSS je nejmenší možný, ze kterého je možné determinovat parametry modelu, na rozdíl od jiných metod, například nejmenších čtverců, kdy se používají veškerá data a případně jejich významnost. Druhým krokem je test, při kterém RANSAC prověřuje, které prvky z množiny vstupních dat jsou konzistentní s parametry modelu, který vznikl odhady v kroku 1. Soubor takových prvků se nazývá konsenzuální soubor (v angličtině consensus set) [8].

Implementaci algoritmu RANSAC je zajištěna pomocí přednastavené funkce `estimateGeometricTransform`. V programu tato funkce zjišťuje, tak se mění poloha všech bodů detekovaných Harrisovým detektorem a následně transformuje velikost, pozici a natočení bboxu a z něho následně získáme novou polohu středu sledovaného objektu.

3.5 Výstup trasování

Celý proces sledování je v reálném čase vizualizován; pro každý snímek můžeme vidět, kde se na něm vyskytují bboxy a v nich detekované body a jak se mění jejich poloha a tvar. Ukázky sledování můžete vidět na obrázcích 3.4 a 3.5. Proces sledování proběhne pro všechny zbývající snímky daného videa, jejich počet je dán výběrem počátečního snímku. Po skončení procesu máme pro každý sledovaný objekt 2 matice se zaznamenanými souřadnicemi všech validních bodů v průběhu videa a souřadnice středu sledovaného objektu. Z nich nyní můžeme snadno získat časový průběh pozice ve 2D prostoru.



Obrázek 3.4 – testovací video „sušenka“

Obrázek 3.5 – testovací video „úklony“

3.5.1 Z souřadnice

Výpočet souřadnic třetího rozměru vychází ze vzorce 2.5. Aby ho bylo možné použít, je nejprve třeba zjistit koeficient zvětšení A. K tomu využijeme matice obsahující body s identifikacemi, výpočet napíšeme do funkce `Scale` a její variace `ScaleWrong` pro případ, že ztratíme objekt a jsme nuceni reinicializovat. Výslednou hodnotu zobrazíme spolu s dalšími souřadnicemi v trojrozměrném grafu. Za předpokladu, že známe vzdálenost objektu od kamery, výšku objektu v reálném světě a výšku objektu v pixelech, budeme mít osy popsané v centimetrech a zaznamenaný pohyb bude přesně zmapován.

Funkce `Scale` vymaže všechny řádky souřadnicových matic, které alespoň jednou obsahují hodnotu 0; zůstanou nám jen údaje o bodech, jejichž souřadnice známe ve všech snímcích. Poté vytvoříme pomocí dvojitého `for` cyklu matici obsahující vzdálenost každého bodu od středu bboxu v každém snímku. K výpočtu této vzdálenosti používáme klasický vzorec pro výpočet vzdálenosti 2 bodů vycházející z Pythagorovy věty, tedy

$$d = \sqrt{(x_1 - x_c)^2 + (y_1 - y_c)^2} \qquad (3.1)$$

kde x_i a y_i jsou hodnoty z matic souřadnic a x_c, y_c jsou souřadnice středu bboxu. Následně vytvoříme druhou matici, která tyto vzdálenosti vydělí hodnotou z prvního snímku. Tato matice bude v prvním sloupci obsahovat identifikační čísla, v druhém hodnoty 1 a poté hodnoty relativního zvětšení. Posledním krokem je zprůměrování těchto hodnot, abychom

dostali matici o jediném řádku a informaci o celkovém relativním zvětšení pro každý snímek.

Před vykreslením grafu se nás systém zeptá na vzdálenost objektu od kamery, výšku objektu v reálném světě a v pixelech. Pokud je zadáme, vykreslí se graf s hodnotami popsanými v centimetrech, pokud údaje neznáme, zadáme hodnoty 1 a dostaneme stejný graf, pouze bez měřítka.

3.5.2 Natočení objektu

Rotace objektu je počítána z rozdílu směrových vektorů od středu do vybraného bodu. Tento výpočet je poté pomocí dvojitého `for` cyklu aplikován na všechny body v průběhu videa. Výpočet probíhá podle upraveného vzorce pro skalární součin vektorů, konkrétně

$$\alpha = \cos^{-1} \frac{\overrightarrow{u_i}.\overrightarrow{u_1}}{|\overrightarrow{u_i}| + |\overrightarrow{u_1}|} \qquad (3.2)$$

kde $\mathbf{u}_i$ je vektor od středu bbox do vybraného bodu a $\mathbf{u}_1$ je vektor od středu bboxu v prvním snímku. Výsledkem je matice obsahující rozdíly v natočení vektorů každého bodu v každém snímku. Jako v předchozím případě jsou tyto zprůměrovány na konečný rozdíl ve stupních, kladný nehledě na směr.

4 TESTOVÁNÍ

Vytvořili jsme algoritmus, který detekuje v námi vybraných oblastech klíčové rysy a podle nich sleduje vybrané objekty, jejichž počet je na počátku určen. Algoritmus má reinicializační systém v případě selhání a jeho výstupem jsou polohová data ve třech rozměrech a údaje o rotaci vybraných objektů.

Nyní je třeba ověřit, zda je námi navržený algoritmus použitelný v praxi. K tomu nám poslouží nejprve idealizovaná videa s jednoduchými pohyby, poté složitější videa s nahranými figuranty provádějícími komplexní pohyby a nakonec reálné videozáznamy pacientů poskytnuté Fakultní nemocnicí Motol.

Pro každé video volíme nejvyšší možný počet objektů ke sledování, pokud předem nevíme, že je v celém videozáznamu například přikrytá peřinou a sledování tedy bude možné jen u hlavy. ROI volíme vždy tak, aby obsahoval celou hlavu, celou dlaň nebo celou část nohy od kotníku na zem, pokud je to možné. Pravou stranu volíme z pohledu diváka, tzn. ruka, která je z pohledu figuranta otočeného čelem na objektiv levá, bude pro nás pravá.

Kritéria hodnocení rozdělíme do tří skupin:

- schopnost algoritmu sledovat vybraný objekt, který bude určen celkovým počtem selhání v poměru k množství snímků a typem pohybu, který zapříčinil selhání, pokud nastane

- přesnost analyzovaných dat, respektive míra, v jaké odpovídají skutečným pohybům, zvlášť všechny 3 rozměry a rotaci
- jiné chyby nebo nedostatky, které neočekáváme

Výsledky analýzy porovnáme se skutečnými hodnotami, které získáme změřením pixelového rozdílu s pomocí fotografického editoru v aplikaci Náhled v operačním systému macOS. Pokud známe v daném videozáznamu reálné vzdálenosti a rozměry, přepočítám na tyto hodnoty pixelové vzdálenosti změřené v Náhledu.

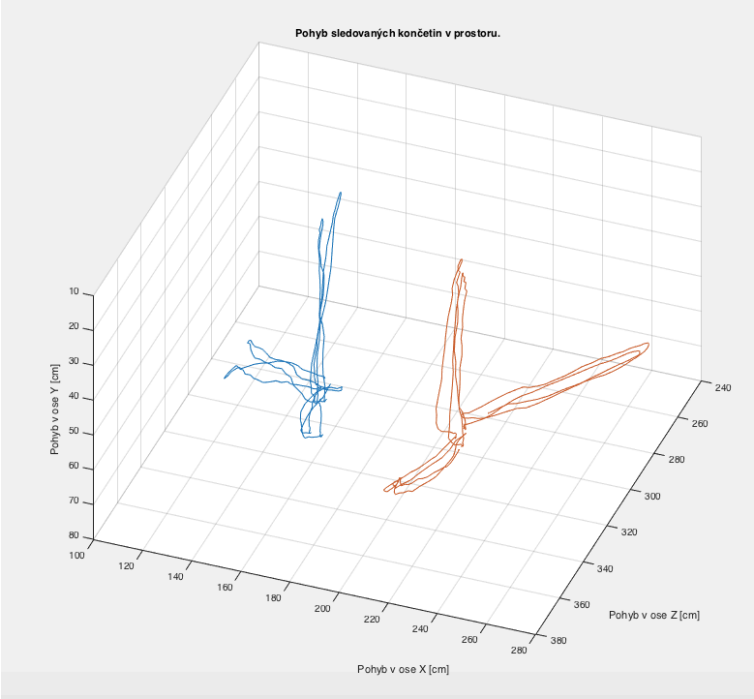
4.1 Fáze 1 – ideální videozáznamy

Všechny videozáznamy testované ve Fázi 1 byly natočeny DSLR (= Digital Single-Lens Reflex) fotoaparátem v exteriérech s běžným denním světlem. Figurant stál ve vzdálenosti do 20 cm od betonové jednolité zdi, nevrhal tedy žádný stín a jeho tělo svíralo s objektivem pravý úhel se zanedbatelnou odchylkou. Rozlišení videa je 1280x720 pixelů, počet snímků za sekundu 24. Vzdálenost od objektu byla 300 cm, figurant měl na výšku 180 cm a na záznamu tedy zabíral na výšku 600 pixelů.

4.1.1 Základní pohyby rukou v osách X, Y a Z

Prvním testovaným subjektem je video, podle kterého jsme prováděli pomocnou kalibraci algoritmu. Figurant drží ruce s dlaněmi otevřenými viz Obrázek 4.1. Provádí pohyby do stran, nahoru a vpřed. Nohami ani hlavou nehýbe. Následující tabulka demonstruje rozdíl mezi minimem a maximem polohy v dané ose, začátek všech pohybů předpokládáme v [0, 0, 0] s 0° rotací.

Název videa	**Osa X [cm]**		**Osa Y [cm]**		**Osa Z [cm]**		**Rotace [°]**	
rucexyz	**analýza**	**reálný**	**analýza**	**reálný**	**analýza**	**reálný**	**analýza**	**reálný**
Hlava	0	0	0	0	1	0	4,5°	0°
Levá noha	0	0	0	0	2,5	0	2°	0°
Pravá noha	0	0	0	0	2,5	0	2°	0°
Levá ruka	48,8	45,4	49,5	49,8	38	40	27,7°	26,6°
Pravá ruka	44,5	42	55	52,3	60	45	27°	20,1°

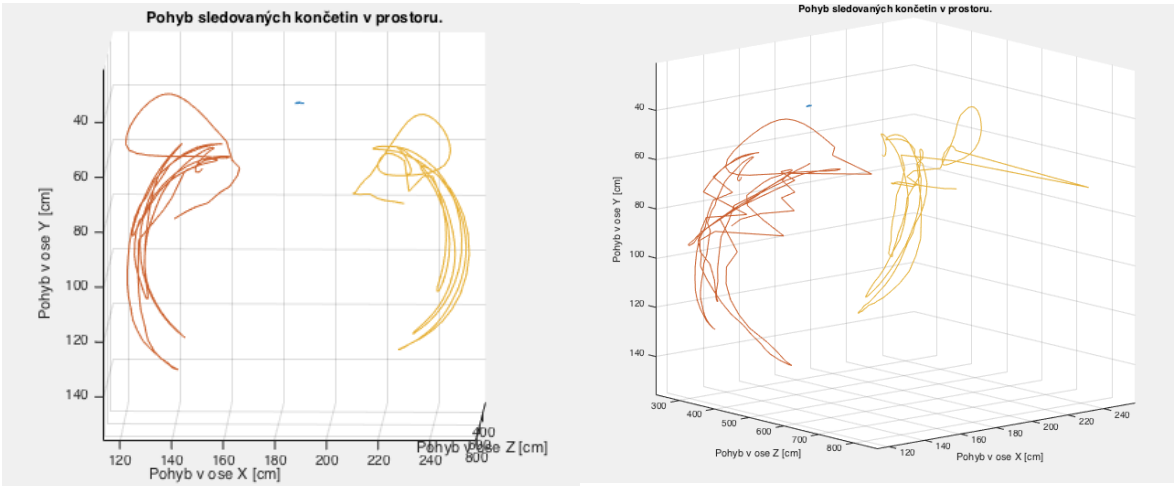


Obrázek 4.1 – graf pohybu sledovaných končetin ve videu „rucexyz“

Podle očekávání ani jeden z 5 objektů není po celou dobu třeba reinicializovat. Měření dopadlo velmi dobře, v rámci osy *x* i osy *y* je u obou rukou nepřesnost v řádu několika centimetrů. V rámci osy *z* je u levé ruky nepřesnost také malá, u pravé už větší. Je to způsobeno natočením ruky, algoritmus natočenou ruku vyhodnocuje jako menší a tedy vzdálenější. U rotace shledáváme obdobnou situaci, pravá ruka vlivem naklopení od těla vykazuje nižší natočení než je ve skutečnosti, oproti tomu levá ruka je popsána velmi přesně.

4.1.2 Pohyby rukou v půlkruhách

Druhé video obsahuje figuranta dělajícího „kruhy“ s otevřenými dlaněmi. Výsledky analýzy nejlépe ilustrují následující grafy v Obrázku 5.2.

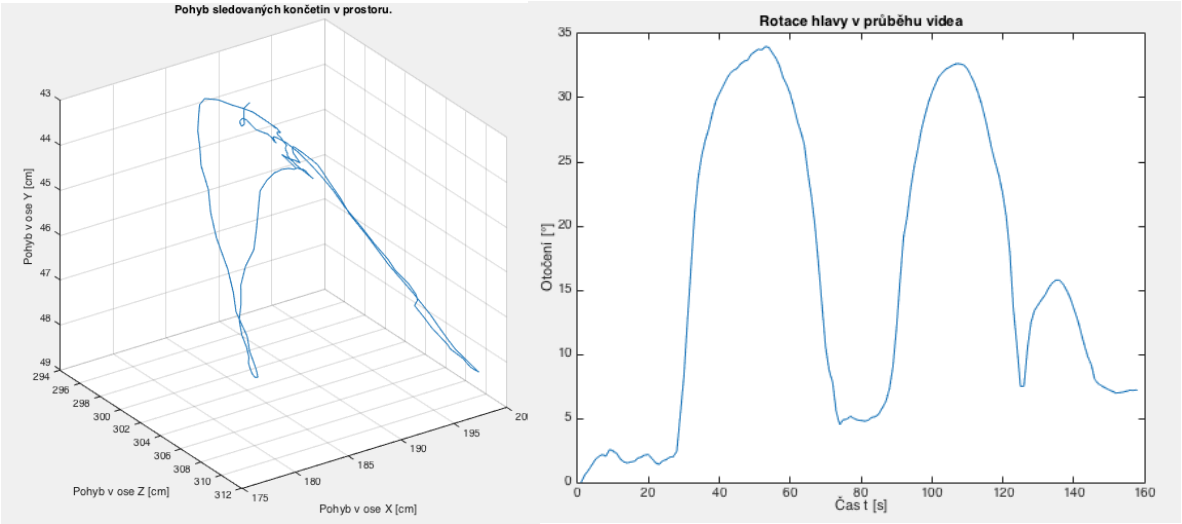


Obrázek 4.2 – Grafy analýzy videa „ruce_kruhy“; vlevo omezení na 2D souřadnice, vpravo včetně osy Z

Zatímco v rámci *x* a *y* souřadnic odvádí algoritmus znovu dobrou práci, přepočet na je zatížen chybami ve sledování. Během analýzy bylo třeba znovu inicializovat levou ruku a dvakrát ruku pravou, protože transformace vzdálenosti bodů vytvořila chybné vzdálenosti, viz výrazný výstupek na světle oranžové křivce, který by ve skutečnosti odpovídal natažení ruky o 5 metrů vpřed. Z této analýzy jsou použitelná pouze data v rámci os *x* a *y*.

4.1.3 Úklony hlavy

Dalším testovacím videem je „hlava_uklony.mov“. Figurant v něm uklání hlavu na strany za stálého pohledu do kamery a neměnného natočení hlavy. Výsledky ukazuje Obrázek 5.3.



Obrázek 5.3 – Grafy analýzy videa „hlava_uklony”; vlevo trojrozměrný pohyb, vpravo úhlová výchylka

Během analýzy nebylo zaznamenáno selhání. I v tomto případě je algoritmus velmi přesný pro dvojrozměrný obraz a vykazuje nejistoty při výpočtu *z*-souřadnice. Nicméně tentokrát jde pouze o jednotky cm, což je vzhledem k vzdálenosti objektivu od figuranta pouze několikaprocentní a tedy zanedbatelná nejistota.

U výpočtu výchylky rotace dochází k chybě v důsledku nerovnoměrného naklopení hlavy (krk, jehož část je sledována, se naklápí méně než hlava). Skutečné hodnoty naklonění hlavy se pohybují mezi 45° a 50°, z analýzy vyplývá údaj okolo 30° až 35°. Tato nejistota může být v některých případech příliš velká, můžeme však použít korigování, pokud budeme vědět, že největší odchylka odpovídá určité hodnotě ve stupních, a na základě tohoto předpokladu pak můžeme hodnotit vývoj náklonů hlavy z hlediska intenzity.

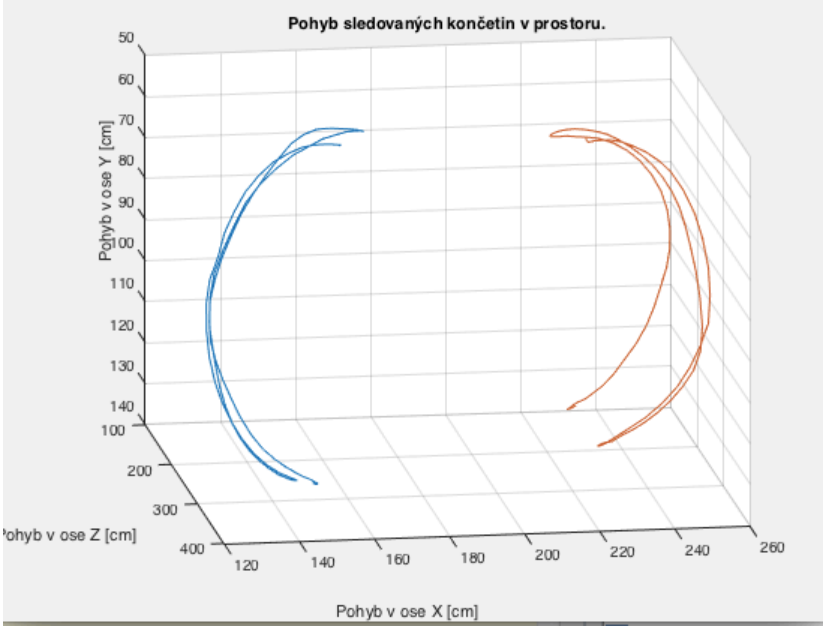
4.2 Fáze 2 – videozáznamy s deformací nebo překrytím

Testovací videa ve Fázi 2 mají všechny kvalitativní parametry stejné jako ve Fázi 1.

4.2.1 Zavírání dlaní

První z řady videozáznamů obsahující obtížnější situaci pro program. Během zavření ruky nejen že se změní velikost, ale značná část dlaně je zakryta a hrozí příliš velká ztráta

detekovaných bodů.

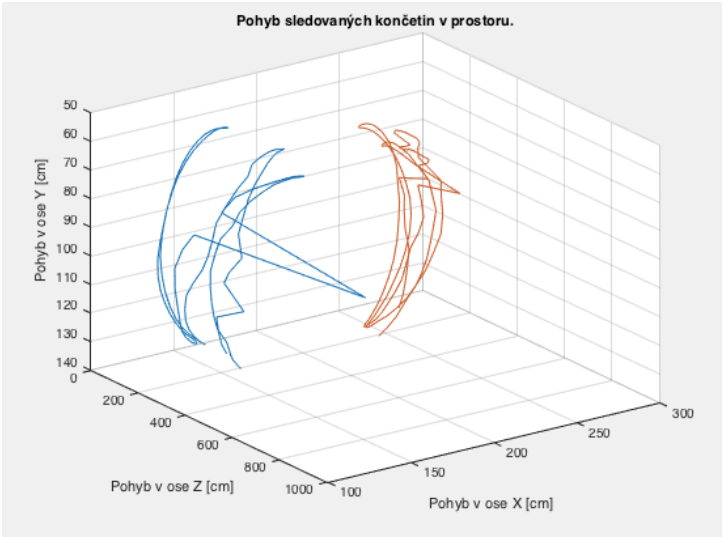


Obrázek 4.4 – výsledný 3D graf videa obsahující pohyb rukou po obloucích se zavření rukou

Test dopadl velmi dobře, algoritmus i přes změnu velikosti sledovaného objektu – ruky – a její textury v důsledku zavření neselhal. Pouze se mírně měnila velikost bboxu, což vede ke zkreslení v ose *z*.

4.2.2 Zavírání a otevírání dlaní

V návaznosti na předchozí video přichází na řadu střídavé otevírání a zavírání dlaní během půlkruhového pohybu.



Obrázek 4.5 – Graf analýzy pohybu opakovaného zavírání a otevírání dlaní v pohybu

Na rozdíl od předchozího videa vidíme, že zde došlo po jednom selhání u obou rukou, pravděpodobně kombinací změněné struktury ruky po zavření a otevření a rozmazání obrazu,

v jehož důsledku se snížil kontrast a detekované hrany byly vyhlazeny. Díky reinicializaci je ve výsledné analýze kvalitně zachycený obloukový pohyb v dvourozměrném prostředí, zatímco vypočítaný pohyb v ose *z* vykazuje obrovské chyby, což je dáno změnou velikosti bboxu při zavření ruky a ztrátě množství sledovaných bodů.

4.2.3 Komplexní pohyby ve vysokém rozlišení

Výstupem tohoto testu není grafické zobrazení; u náhodných pohybů je téměř nemožné určit, zda byly po celou dobu sledovány správně. V průběhu sledování končetin v tomto videu došlo k mnoha selháním a tedy nutné reinicializaci.

Nejprve zhodnotíme pohyb nohou. Pravá noha byla reinicializována čtyřikrát, přičemž vykonala třikrát pohyb nahoru a zpět dolu. Levá noha byla reinicializována osmkrát během čtyř pohybů, přičemž první byl prosté zvednutí a ostatní pohyby navíc zahrnovaly i třesavý pohyb chodidla. Důvodem k selhání zde byl výběr bot značky Crocs©, které mají jedinou barvu a jejich hladký povrch narušují otvory, které jsou bohužel v důsledku své velikosti a vzdálenosti od objektivu i při pomalém pohybu značně rozmazané.

Pohyb rukou je oproti nohám výrazně rychlejší a o tom svědčí i množství reinicializací – 13 pro levou ruku a 8 pro pravou, kde byl díky tmavým hodinkám častěji zachován dostatečný počet bodů. Ačkoliv je množství reinicializací značné, vzhledem k množství pohybu se nejedná o špatný výsledek, neboť bbox byl na základě zbývajících bodu vždy až do momentu selhání transformován korektně a zachovával svůj střed na stejném místě dlaně, jako jsme zvolili.

4.3 Fáze 3 – skutečné záznamy

Nyní se dostáváme k testování algoritmu pomocí skutečných záznamů pacientů. Každé video otestujeme s úpravami podle 3.2.1. Všechny záznamy zpracované ve Fázi 3 jsou natočené blíže nespecifikovanou kamerou, mají rozlišení 352x288 pixelů při 25 snímcích za sekundu a údaje o vzdálenosti a velikosti objektu nám nejsou známy.

4.3.1 Video 1

Na prvním testovacím videozáznamu je kojenec trpící spazmy, jak můžete vidět na Obrázku 5.6. Z etických důvodů v této práci pro ilustraci rozoostřujeme na reálných videozáznamech obličeje, analýza však probíhá na nerozostřených záznamech.

Z Obrázku 5.6 si lze všimnout, že video trpí několika neduhy. Zaprvé pacientova pravá noha je velkou část videa mimo záběr a není proto dobře možné ji sledovat. Zadruhé si všimneme další osoby, která střídavě chytá kojence za pravou ruku a zpravidla ji zcela zakryje. Posledním neduhem je jednobarevný komplet, který má kojenec na sobě a díky kterému se v oblasti nohou, respektive chodidel a kotníků prakticky nevyskytují jasně distingované rohy či body.

Ke sledování aktivujeme čtyři objekty s tím, že vynecháme pravou nohu. Vždy se snažíme

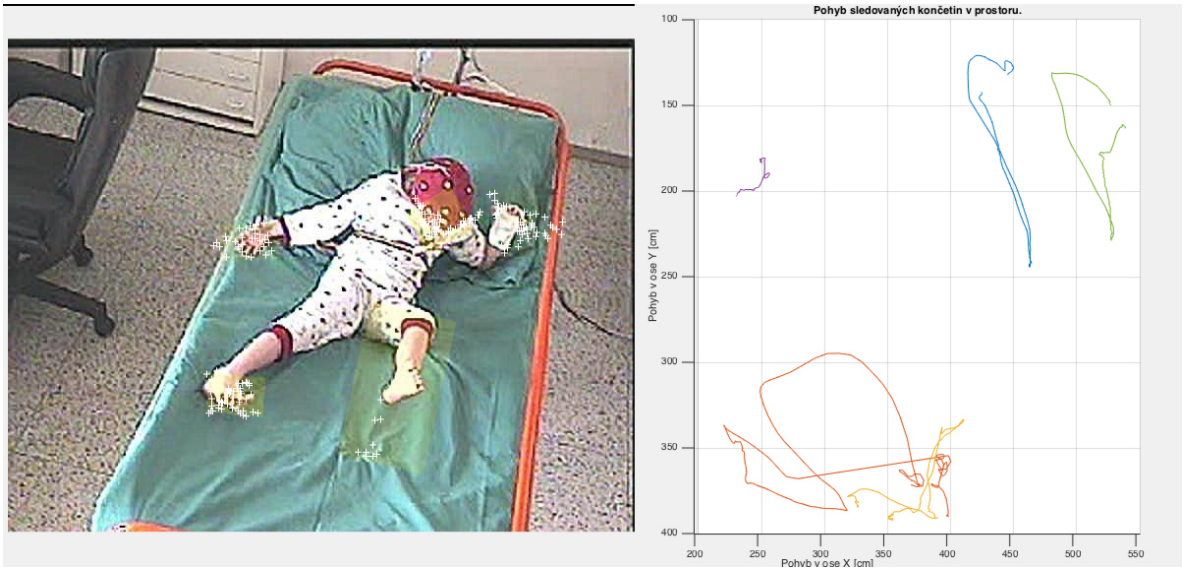
označit kupříkladu celou dlaň, celý obličej nebo celé chodidlo.

Obrázek 4.6 – Průběh analýzy videozáznamu Video 1

Při sledování tohoto videa se v rámci 1750 snímků (~ 10 minut) vyskytlo 16 chyb ve sledování pravé nohy, 5 chyb ve sledování levé ruky a 4 chyby při sledování ruky pravé. Množství chyb při sledování nohy je způsobeno výše zmíněným jednobarevným kompletem, který nemá příliš jasně odlišitelných bodů. Naproti tomu sledování levé ruky má velmi dobrý výsledek, přestože jde o objekt poměrně malý a jehož pozadí má téměř stejnou barvu. K selhání u něho došlo během rychlejšího pohybu při zvednutí paže. Veškerá selhání sledování pravé ruky jsou v důsledku zakrytí rukou druhé osoby.

4.3.2 Video 2

Druhé video zachycuje pacientku ve školním věku, která prodělá krátký záchvat bez záškubů. Pro sledování vybíráme všech 5 objektů. Na grafu vpravo v Obrázku 4.7 můžeme



vidět dvourozměrný pohyb všech končetin, vlevo vidíme inicializované body a bboxy jimi vypočítané.

Obrázek 4.7 – Ukázka průběhu analýzy Videa 2 a graf pohybu 5 sledovaných objektů

V tomto videu došlo k jevu, který si na výše uvedeném obrázku můžeme prohlédnout u pacientovy levé nohy (z našeho pohledu). V oblasti označené okolo chodidla se detekovaly body i mimo chodidlo a poté nebyly vyřazeny. V důsledku toho se bbox zvětšil špatně a ačkoliv jeho střed kupodivu stále koresponduje se středem chodidla, nebude kvůli této chybě možné vypočítat jeho z souřadnice. Další problém nastal při sledování hlavy; původní výběr byl proveden na obličeji, nicméně pacientka v průběhu záchvatu značně předklonila hlavu a obličej si zcela zakryla, což vedlo k nutnosti reinicializace. Přes tyto neduhy jsou výsledky sledování jednotlivých končetin ve 2 rozměrech dobré.

4.3.3 Video 3

Poslední video zachycuje ležícího pacienta školního věku, který dostane záchvat s gradujícím třesem nejprve hlavy a hrudi a poté rukou, zatímco nohy zůstávají víceméně v klidu.

Obrázek 4.8 – Ukázka analýzy Videa 3

V tomto videu se setkáváme s podobnými problémy, jako v předchozích dvou. Při jednom z výběrů pravé ruky zhodnotil algoritmus jako zvolený objekt texturu tepláků, na kterých měl pacient v té chvíli položenou ruku.

Selhání hlavy ani levé ruky (té, kterou si podepírá hlavu) neproběhlo, oproti tomu máme 11 selhání pravé ruky, ve které byl nejsilnější třes a po 5 selháních pro každou nohu, které byly s největší pravděpodobností způsobené tím, jak pacient obě nohy překládal přes sebe.

4.4 Zhodnocení výsledků

Na začátku jsme jako testovací video použili postupné střídání jednoduchých pomalých pohybů v HD rozlišení. Všech 5 sledovaných objektů v tomto videu bylo analyzováno velmi dobře, odchylky byly v řádu procent. Jedinou výjimkou bylo pootočení dlaně figuranta směrem od sebe, které algoritmus vyhodnotil jako zmenšení a tedy oddálení objektu. V prvním testu nebylo třeba reinicializovat žádný objekt. Podobných výsledků bylo dosaženo při analýze videa jednoduchých úklon hlavy do stran a obloukových pohybů dlaní. V obou případech se ukázalo, že systém výpočtu osy *z* selhává kvůli drobnému pootočení objektu na stranu nebo nejistotami, které vzniknou v důsledku nedokonalého sledování rohových rysů.

Další testovací videa obsahovaly rychlejší pohyby rukou, zavírání a znovuotevírání dlaní a nakonec komplexní náhodné pohyby těla. Během analýzy zavírání a otevírání paží byl algoritmus velmi dobře schopen objekt „udržet“, selhání přišlo v momentě, kdy ruka po zavření a následném otevření – tedy při značně sníženém počtu bodů oproti počátečnímu stavu – vykonala rychlejší pohyb. Výpočet osy *z* byl i v těchto případech špatný, u zavírání a otevírání dlaní to způsobil fakt, že detekované body z konečků prstů změnily svoji polohu do středu dlaně a průměrná vzdálenost bodů, ze které se počítá pohyb v ose *z*, se zmenšil na polovinu; toto zmenšení podle výpočtů v dané situaci (objekt vzdálený 300 cm) odpovídá oddálení ruky o zhruba 300 cm.

Při analýze komplexních pohybů vykazoval algoritmus relativně špatné výsledky. Kombinace rychlých pohybů, způsobujících rozmazání a obuvi, která není příliš strukturovaná vyústila ve velké množství selhání všech čtyř končetin. Ve všech případech byl naopak algoritmus schopný sledovat hlavu, pravděpodobně díky značnému kontrastu očí, obočí, pusy a nosních dírek ke světlému zbytku tváře.

Při analýze reálných videozáznamů jsme se ve všech případech potýkali s problémy netechnické rázu, jako například zakrytí končetiny jiným objektem. V některých případech proto nebylo možné sledovat všechny končetiny pacienta. V některých případech se ve vybrané oblasti inicializovaly body mimo objekt, který jsme chtěli sledovat, a tyto body byly shledány vybraným objektem. V důsledku toho „zůstal“ shluk bodů například na vzorovaných kalhotách, zatímco pacient svou ruku posunul na zcela jiné místo. V ostatních případech sledování se potvrdily principy z testovacích videí, především že nejobtížnější je sledovat objekt ve vysoké rychlosti, protože podléhá rozmazání.

5 ZÁVĚR

Cílem této práce bylo seznámit se s dostupnými metodami sledování objektů ve videozáznamu a vytvořit ve vývojovém prostředí Matlab program, který dokáže pomocí

trasovacího algoritmu parametrizovat pohyb pacienta v průběhu epileptického záchvatu.

Námi vytvořený algoritmus se skládá z několika částí. V první části pomocí GUI vytvoříme objekty, do kterých později zapisujeme většinu dat a vybereme video, které chceme analyzovat. Poté video upravíme a ve vybraných oblastech spustíme Harrisův detektor, který vyhledá a zaznamená významné odlišitelné rysy, konkrétně rohy. Následuje samotný proces sledování, kde snímek po snímku pomocí KLT sledovacího algoritmu a estimačního algoritmu RANSAC sledujeme detekovaný body a jejich polohu v čase zapisujeme. Tato část má automaticky volaný reinicializační program, s jehož pomocí v případě selhání sledování vybereme ROI pro sledování znovu. Po skončení sledování jsou data zpracována a přepočítána na prostorové údaje a vykreslena do grafů.

Funkčnost algoritmu silně závisí na kvalitě videa, druhu pohybu objektu a případných netechnických překážkách, jako například překrytí jiným objektem. Algoritmus zvládá sledovat objekt, který se libovolně pohybuje vertikálně, horizontálně, rotuje nebo se přibližuje či oddaluje, problém nastává v případě naklopení do strany, kdy je tento pohyb chybně vyložen jako změna velikosti objektu. Pro konstantně kvalitní výsledek platí, že čím horší je technická stránka videa, tím musí být pomalejší pohyby. Při dostatečném rozlišení, kvalitě a množství snímků za sekundu, popř. rychlosti závěrky v poměru k rychlosti pohybu tak, aby rozmazání pohybujícího se objektu bylo minimální, sleduje algoritmus objekty s odchylkou jednotek procent. V případě, že objekt nekoná žádný pohyb překlápěním, lze totéž říci i o výpočtu pohybu směrem do/od kamery.

Závěrem lze potvrdit, že vytvořený algoritmus lze použít pro naše účely. Jeho využití je limitované kvalitou videozáznamů. Jednou z možností rozšíření použitelnosti je udělat více reinicializačních metod, více či méně automatické, aby postupně pokryly všechny možné příčiny selhání systému. Dalšími možnostmi je kombinace tohoto algoritmu s jiným nebo úprava obrazu nízké kvality způsobem, aby bylo co nejvíce redukováno rozmazání okrajů objektu vlivem jejich pohybu.

Zdroje

[1] Chang, B. S., Lowenstein, D. H. (2003). Epilepsy. N. Engl. J. Med. 349 (13): 1257–66.

[2] Bishop, Ch. (2006). Pattern recognition and machine learning. *Berlin: Springer*. ISBN 0-387-31073-8.

[3] Picek, L. (2014). Automatická detekce dopravních objektů na pozemních komunikacích pro pasportizaci a navigaci. Západočeská univerzita v Plzni. Bakalářská práce.

[4] Hasmata, M. (2007). Detekce a korespondence významných bodů v obraze. Vysoké učení technické v Brně. Bakalářská práce

[5] The MathWorks, Inc (2014). *MATLAB r2014b Documentation*. K dispozici online: https://www.mathworks.com/help/matlab/

[6] Lucas, B. D., Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. 674-679.

[7] SVOBODA, Tomáš (2008). *Kanade–Lucas–Tomasi Tracking (KLT tracker)*. Czech Technical University in Prague, Center for Machine Perception http://cmp.felk.cvut.cz

[8] Zuliani, M (2014). RANSAC for Dummies. Dostupné online: http://vision.ece.ucsb.edu/~zuliani