

bakalářská práce

# Hidden

*Hidden*



Červen 2018

Hidden

České vysoké učení technické v Praze  
Fakulta elektrotechnická, Hidden



## **Poděkování**

Text of acknowledgement...

## **Prohlášení**

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

## **Abstrakt**

Text abstraktu česky...

## **Klíčová slova**

První klíčové slovo; druhé; třetí; ...

## **Abstrakt**

Text of abstract in English...

## **Keywords**

First keyword; second; third; ...

# Obsah

<b>1 Úvod</b>	<b>1</b>
1.1 Důvody . . . . .	1
<b>2 Stávající metody a aplikace</b>	<b>3</b>
2.1 Stávající metody . . . . .	3
2.1.1 Předzpracování obrazu . . . . .	3
Binární obraz . . . . .	3
Rozostření . . . . .	3
2.1.2 Rozpoznávání postav . . . . .	4
2.1.3 Rozpoznávání částí těla . . . . .	7
2.1.4 Rozpoznávání gest . . . . .	8
Rozpoznávání prstů . . . . .	8
2.2 Aktuální aplikace . . . . .	10
2.2.1 Konkrétní aplikace . . . . .	10
Rehabilitace . . . . .	10
UI během operace . . . . .	10
Bixi . . . . .	10
<b>3 Implementace rozpoznávání gest</b>	<b>11</b>
3.1 Software . . . . .	11
3.1.1 openFrameworks . . . . .	11
3.2 Hardware . . . . .	11
3.2.1 Kinect . . . . .	11
3.3 Zpracovatelné vstupy . . . . .	11
3.3.1 RGB video . . . . .	12
3.3.2 Hloubkové video . . . . .	12
Úvodní nastavení . . . . .	12
Nalezení ruky . . . . .	12
Identifikace gesta . . . . .	13
Vykreslení ruky . . . . .	14
3.4 Předzpracování . . . . .	14
3.4.1 Filtrace . . . . .	14
3.4.2 Prahování . . . . .	14
3.5 Detekce ruky . . . . .	15
3.6 Detekce dlaně . . . . .	16
3.7 Detekce prstů . . . . .	17
3.7.1 Detekce směru prstů . . . . .	18
3.7.2 Detekce konců prstů . . . . .	18
3.8 Statistika . . . . .	19
<b>4 Závěr</b>	<b>20</b>
4.1 Možnosti využití gest při ovládání mobilního robotu . . . . .	20
<b>Literatura</b>	<b>21</b>

## Zkratky

Preliminary text...

OF	openFrameworks ... nástroj pro kreativní programování
FBO	"frame buffer object"... objekt pro vykreslování v OF
HMM	"Hidden Markov Model"... Skrytý Markovův model
P2DHMM	"Pseudo-2D Hidden MARKov Model
VR	virtuální realita
DOF	"degree of freedom"... stupeň volnosti
SVM	"support vector machines"... SVM klasifikátor s podpůrnými vektory
ToF	"Time-of-Flight"... doba letu aneb hloubkové kamery založené na vý- počtu vzdálenosti z doby letu laseru
ROI	"Region of interest"... oblast zájmu
RBF	"Radial Basis Functions"
...	...





# 1 Úvod

V dnešní době robotizace, kdy lidé přicházejí na výhody využívání inteligentních (VLOŽIT REFERENCI podle jaké definice)

zařízení, je snaha rozšířit jejich využití co nejvíce.

POZN:vložit delší povídání o tom co se již děje atd?

Řízení zařízen za pomoci kamery se řadí mezi nejjednodušší způsoby, jelikož kamery jsou relativně levné a dostupné. Není k tomu potřeba nic jiného, než co většina populace (alespoň cílové skupiny - lidí používající počítače) už má. Jedná se o jednodušší variantu i s ohledem na implementační čas.

## 1.1 Důvody

Mnoho podobných aplikací již existuje, nebo se zaměřuje na jednotlivé podčásti, často ale vyžadují složitou instalaci a konfiguraci programů. Velké množství pak ani nepodporuje zpracování v reálném čase a jsou výpočetně složité a časově náročné. Cílem bakalářské práce je vytvořit program relativně jednoduše nastavitelná s tak rychlou odezvou, jak to jen bude robustnost dovolovat.

Následně řešené přístupy jsou voleny tak, aby měly co nejmenší odezvu. S lepší robustností již lze navázat na konci.



## 2 Stávající metody a aplikace

### 2.1 Stávající metody

#### 2.1.1 Předzpracování obrazu

##### Binární obraz

Většina metod se zakládá na zpracování binárního obrazu. Probíhá to zvolením prahové hodnoty, se kterou se všechny hodnoty porovnají a ve výsledku vznikne binární obraz. Objekt, který je středem zájmu, se skládá z pixelů s hodnotou 1 zatímco ostatní mají hodnotu 0.

##### Rozostření

Rozostření slouží k eliminaci šumu v obraze. Nejosvědčenější metody jsou nízkofrekvenční filtry jako je Gauss a medián. Tato kapitola čerpá z Pikoly [15] kapitoly 5 a 6.

Filtr medián se zaměřuje na vyhlazování lokálních extrémů z obrazu. Nezachovává jemné čáry a ostré rohy, takže je vhodný pouze v aplikacích, kde na tom nezávisí aplikace. Výraznou výhodou představuje v odstranění extrémů, které v obraze mohou vzniknout. Narozdíl od průměrování obraz nezkreslí chybné pixely s extrémními hodnotami.

Velikost okolí zahrnutého v analýze se může zvětšit při větší hustotě šumu. V aplikaci na hloubkový obraz, ve kterém se detekují prsty je ale důležité zachovat určité rozlišení. Na obrázku 1 lze pozorovat rozdíly mezi filtrem s průměrováním (b) a mediánem (c).



**Obrázek 1** a) původní obraz b) průměrování c) medián  
Převzato z [15]

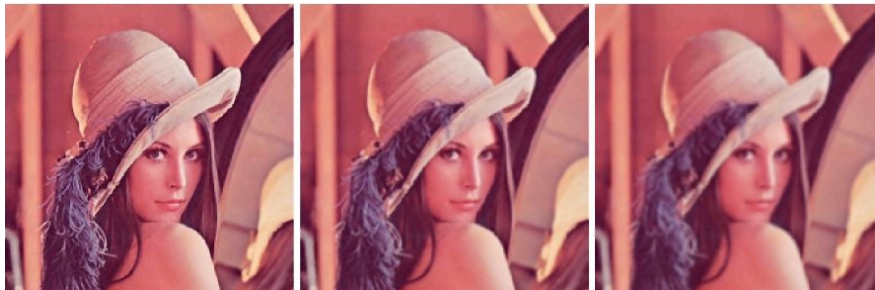
Gaussovský filtr je průměrování s Gaussovským rozložením. Využívá se k vyhlazování obrazu a odstranění detailů a šumu. Obrázek 2 zachycuje rozdíl mezi opakovanou aplikací Gaussovského filtru.

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

kde  $x$  a  $y$  jsou souřadnice a  $\sigma$  je směrodatná odchylka.

Aby se zabránilo rozšíření do nekonečna se v konvoluční masce přidělí větší váha bodu ve středu. Aby se aplikací konvoluční masky neměnila světlost, součet všech složek konvoluční matice dává hodnotu 1. Příklad konvoluční masky:

$$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad (2)$$



**Obrázek 2** a) původní obraz b) jednou aplikovaný Gaussův filtr c) třikrát aplikovaný Gaussovský filtr Převzato z [15]

Mezi vysokofrekvenční filtry, které se používají na zpracování digitálního obrazu patří i Laplaceův filtr. Ve výpočtu se užívá gradient, což je vektorová veličina určující směr a strmost největšího růstu funkce. Laplaceův filtr využívá pouze velikost a pro její odhad se používá všesměrný operátor vycházející s parciálních derivací.

$$\nabla^2 g(x, y) = \frac{\delta^2 g(x, y)}{\delta x^2} + \frac{\delta^2 g(x, y)}{\delta y^2} \quad (3)$$

Laplacián je aproximován diskretní konvolucí. Příklad konvolučního jádra:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad (4)$$

### 2.1.2 Rozpoznávání postav

Možnosti:

podle tvaru

- na základě modelu
- na základě prvků
- strojově naučené programy
- neuronové sítě
- transformační matice vzdálenostních funkcí (chamfer)

podle barvy

- strojově naučené programy

Jednou z metod je dle Ch. Nakajima a spol. [2]<sup>6</sup> ukládání několika obrázků, z nichž se detekuje pohybující se objekt a vše, co se nehýbe se označí za pozadí a ignoruje. Následně se naleznou kraje možné postavy, aby se vylepšil výsledek a vymazal šum. Pokud se jedinec pohybuje v druhém kroce, oříznutí probíhá z posledního získaného obrazu. Pokud nebyl detekován žádný jedinec, tak se do paměti uloží pozadí, které se z následujících obrazů rovnou vymaže pro větší přesnost. K vlastní identifikaci se použila SVM (support vector machines) a k-NN klasifikátor.

Hussein a spol. [3]<sup>9</sup> používají k rozpoznání postavy prohledávání obrazu a porovnávání siluet s databází, ve které jsou vzory lidských postav. Shoda je detekována pomocí zkosením objektů. Vzory jsou ve formě binárního obrazu. Vzdálenost  $D$  mezi vzorem  $V$  a objektem  $O$  z obrazu  $O$  se počítá vzorcem

$$D(O, V) = \frac{1}{|V|} \sum_i C_i V_i, \quad (5)$$

kde  $|V|$  je počet pixelů v siluetě ve vzoru,  $T_i$  je hodnota pixelu  $i$  ve vzoru a  $C_i$  je rozdíl zkosení pixelu  $i$  v obraze. Čím menší hodnota mezi vzorem a obrazem, tím lepší shoda je detekována.

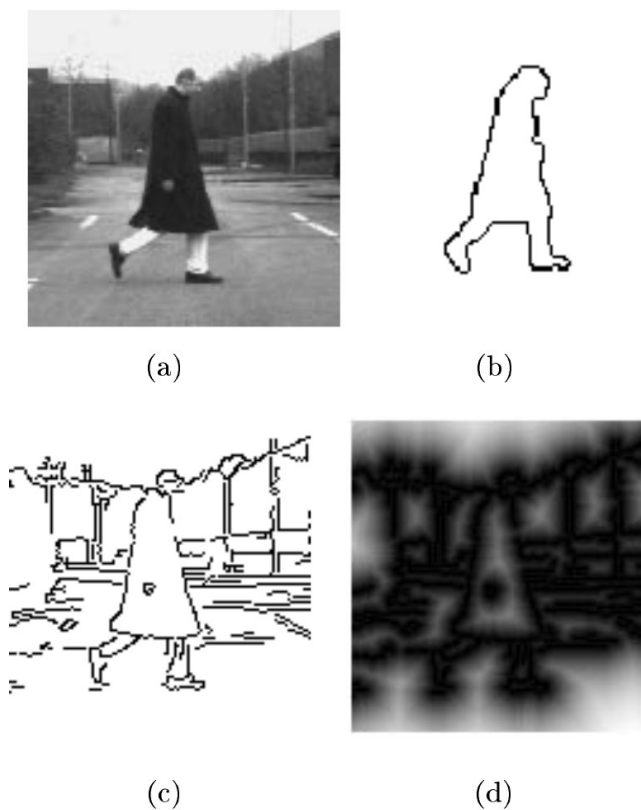
Nalezení podobnosti zkosením předpokládá již nalezené hrany objektů v obraze. Vzor se překryje co nejvíce na objekt a transformací bodů jednoho objektu pomocí parametrických transformačních rovnic.

Jiný přístup poskytuje nalezení objektu v zorném poli pomocí hloubkové kamery, následným rozhodnutím, zda objekt má lidské nohy, nebo aspoň dva objekty odpovídající tvaru, a pokud ano, tak se přejde k dalšímu kroku, ve kterém se detekuje barva kůže z RGB vstupu. Nalezený výsledek slouží jako oblast k detekci obličeje, čímž je definitivně nalezen člověk [4]<sup>10</sup>.

Gavrila [5]<sup>7</sup> se snaží snížit výpočetní náročnost tak, že nahrazuje procházení obrazu postupně schopností zaměřit se na jeden úryvek přímo a ve druhém kroce teprve hledá podobnosti tvarů. Dosahuje tak zpracování v reálném čase. Shodu podle tvaru detekuje pomocí vzdálenosti křivek, která se počítá podobně jako u Hussein a spol. [3]<sup>9</sup>

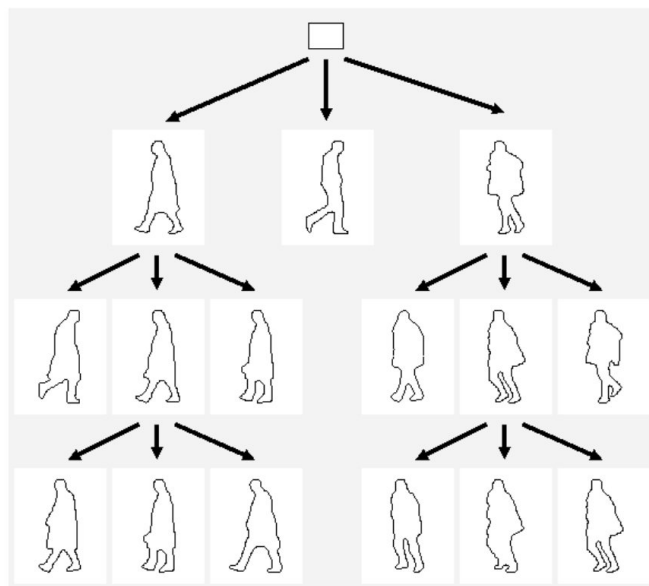
$$D(O, V) = \frac{1}{|V|} \sum_{t \in T} d_I(t), \quad (6)$$

kde  $|V|$  je počet prvků ve vzoru a  $d_I(t)$  představuje vzdálenost daného prvku ve vzoru a odpovídajícímu v obraze. Pokud výsledná hodnota je menší než předem určená prahová hodnota, považuje se objekt v obraze za postavu. Detekce chodce podle vzoru lze pozorovat na obrázku 3.



**Obrázek 3** a) původní obraz b) vzor c) nalezené hrany d) vyobrazené vzdálenosti Převzato [5]

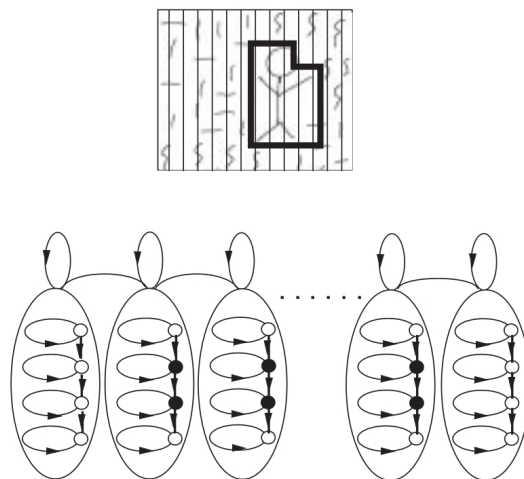
V rámci optimalizace se tato metoda (jak překládat chemfer matching system??) rozšiřuje na hierarchizovanou, která sdružuje podobné vzory do shluků a tudíž se prohledávání databáze zrychlí (viz obrázek 4).



**Obrázek 4** Využití hierarchie v databázi vzorů. Převzato z [5]

Ověření výsledku probíhá RBF (radial basis functions) klasifikátorem. Nejdříve se z původního obrazu vybere čtyřúhelník obsahující možnou postavu a na základě euklidovské vzdálenosti určuje, jestli jednotlivé pixely jsou součástí postavy či nikoliv.

Riggol a spol. [6]<sup>11</sup> kombinují rozpoznávací metody založené na prvcích a na modelech za cílem dosažení spolehlivého a efektivního programu. Využívají P2DHMM (Pseudo-2D Hidden Markov Model). Jedná se o předem naučený algoritmus, který dokáže na základě pravděpodobností určit, jestli se jedná o postavu nebo ne. Příklad detekované postavy a postupné rozhodování je vidět na obrázku 5.



**Obrázek 5** Stochastický model 2D objektu. Převzato z [6]<sup>11</sup>

### 2.1.3 Rozpoznávání částí těla

Ruka má 23 stupňů vlastností a v kombinaci s různým osvětlením a změnami v pozadí se jedná o značně komplikovaný problém na řešení pomocí podobnosti.

Jestliže aplikace nevyžaduje dynamické využití, lze použít rozdíl mezi objektem a pozadím [7]<sup>14</sup>. Vyžaduje to co nejvíce neměnné pozadí a dostatečný rozdíl mezi objektem a zbytkem. Rozdíl lze hledat jednoduše procházením pole nebo využitím "chytřejšího" algoritmu. Spočívá v identifikaci čístejšího obrazu, která se hýbe oproti posledním obrazům. Příklad detekce ruky odstraněním pozadí je ukázán na obrázku 6.



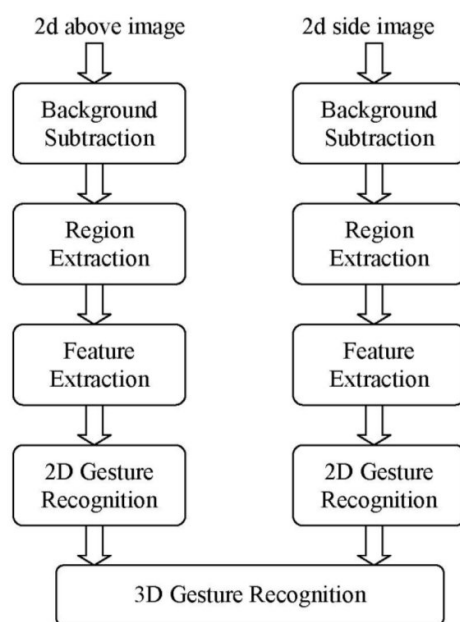
**Obrázek 6** a) vstup b) referenční obraz c) výstup Převzato z [7]<sup>14</sup>

### 2.1.4 Rozpoznávání gest

Mezi rozpoznávání na základě vzhledu se řadí mnoho metod zahrnujících strojově naučené algoritmy jako neuronové sítě, HMM (Skrytý Markovův model) a jiné [8]. Gesto lze identifikovat i pomocí určení pozice a směru ruky a jednotlivých prstů.

Kim a spol. [9] používají bílé značky na špičkách prstů, které sledují černým světlem a detekují jednoznačně špičky prstů, což umožňuje velkou škálu možných gest. Na druhou stranu vzniká i omezení barvy pozadí, které lze eliminovat jen v určitém prostředí, tudíž se jedná o vhodnou variantu k využití při hrách ve VR (virtuální reality).

Shaker a Zliekha [10] získávají směr prstu na ruce, kterou snímají dvěma kamerami. Jedna kamera natáčí shora, zatímco druhá z boku. Oba vstupy se zpracovávají zvlášť do rozpoznávání gesta. Nejprve přepočítají obraz na stupně šedi a poté dle určené prahové hodnoty odfiltruje pozadí a vytvoří binární obraz. V tomto kroce je třeba, aby bylo pozadí jednoduché a jasně rozlišitelné od ruky. Kvůli odstranění šumu se obrázek rozmaže a následně zaostří. V tomto kroce vzniká podmínka, že největší objekt v obraze je právě ruka. Poté se pomocí Laplaciána získají hrany daného objektu a z něj se vytvoří kontury. Gesto se identifikuje pomocí nalezení největšího vrcholu v obraze a nalezení čáry, kterou opisuje natažený prst a následnou kombinací obou směrů do 3D. Postup je znázorněn na obrázku 7.



**Obrázek 7** Popis algoritmu. Převzato z [10]

### Rozpoznávání prstů

Hackenberg a spol. [10] implementovali identifikaci prstů a dlaně na základě několika kroků. Nejprve projde hloubkový obraz z kamery a hledá tvary podobné špičkám prstů nebo rourovitého tvaru podobného prstům. Poté se zaměří na detailnější vlastnosti dlaně. Z nalezených vhodných polí, která mohou představovat dlaně se vyřadí ta, která nejsou napojena nijak na prsty, jsou menší než předem určená hodnota (odvozená od

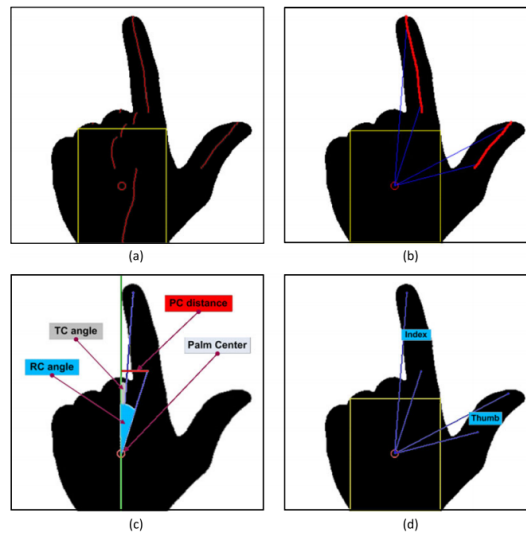


průměrně velikosti hlavy) a vzdáleností špiček prstů od dlaně.

Bez potřeby porovnávání tvarů a zdrojů se dají rozpoznat jednotlivé špičky prstů pomocí nalezení lokálního maxima v obrysu ruky. Ve všech adekvátních směrech se naleznou kandidáti a následně vyloučí mylné identifikace. Jedná se ale o postup náchylný k šumu v obraze, podobná metoda odolnější navrhuje porovnávání vzdálenosti obrysu k pozici ruky [8].

Jednotlivé prsty lze rozpoznat například pomocí vyhledávání na základě tvaru [11]. Jde o zjednodušení představy objektu na geometrický objekt a vyhledávání vhodných kandidátů v obraze.

Mezi další programy rozpoznávající špičky prstů z již relativně čistého čtyřúhelníku, který obsahuje zájmový objekt, patří identifikace na základě dvou specifických vlastností. Centrum špičky je obklopen kolem pixelů patřící prstu a ten je obklopen dlouhou řadou pixelů nepatřících prstu [7]. Ruce lze identifikovat i až po prstech a to tak, že se vyloučí objekty podobného tvaru (propisky, fixy), které nesplňují vlastnosti rukou, například nepatří k žádné dlani.



**Obrázek 8** Vyobrazení definovaných pojmů, které lze využít k výpočtům pozic, na základě proporčních vlastností prstů. Převzato z [12]

$$RC_{angle} = 90 - \tan^{-1} \frac{y_r - y_{pc}}{x_r - x_{pc}}$$

$$TC_{angle} = 90 - \tan^{-1} \frac{y_{ft} - y_{pc}}{x_{ft} - x_{pc}}$$

## 2.2 Aktuální aplikace

Všechny aplikace, které mohou využít výhod [14]:

- Ovládání na velmi malém prostoru
- Ovládání z větší vzdálenosti
- Snížený počet součástí
- Zjednodušené ovládání elektroniky
- Potenciál zabezpečení proti zneužití (při implementaci ochranných prvků)
- Minimalistické (méně tlačítek a obdobných prvků)

Potenciál pro využití v:

- Překlad znakové řeči
- Ovládání chytrých zařízení (chytrá domácnost - světla, hudba, televize, telefon)
- Prezentace před publikem (ovládání počítače)

Již využíváno v:

- Rozpoznání chodce v dohledu vozidla
- Sociální experimenty (vyslání robota stopovat)
- Ovládání a hraní na konzolích

### 2.2.1 Konkrétní aplikace

#### Rehabilitace

Mezi využití v lékařské oblasti patří převážně rehabilitační cvičení. V nemocnici ve městě Reading využívají kameru Kinect pacienti po mrtvici na zlepšení pohyblivosti a koordinace [13].

#### UI během operace

Další možnosti se nabízí i při procházení záznamů pacienta při operaci, aniž by se kdokoliv dotýkal nesterilních věcí. Další možnosti se nabízí i na operačním sále, když chce doktor přístup k záznamům pacienta a nemusí se tak dotýkat žádných nesterilních věcí [24].

//cite22 preliminary investigation of movement

#### Bixi

Bixi je bezdotykový ovladač k mobilnímu zařízení. Hlavním zaměřením je usnadnění ovládání vedlejších aplikací během řízení auta tak, aby se řidič mohl soustředit na provoz. Po spárování se zařízením pomocí bluetooth lze gesty volit adresu pro navigaci, ztlumit či zastavit hudbu, ztlumit jas obrazovky, přijímat hovory a jiné [bixi].

## 3 Implementace rozpoznávání gest

Následné algoritmy byly testovány na Kinectu v2 pomocí projectGeneratoru od openFrameworks. Program má i vývojové prostředí, ze kterého to lze spouštět. V příloženém README.txt bude uveden postup pro generátor pomocí příkazové řádky.

### 3.1 Software

#### 3.1.1 openFrameworks

OpenfFrameworks <sup>[1]</sup><sub>[14]</sub> je open source C++ nástroj pro kreativní programování. Využívá se doplněk ofxKinectV2 <sup>[2]</sup><sub>[15]</sub>. Oproti zabudovanému ofxKinect je optimalizovaný pro aktuální openFrameworks (verze 0.9.0), je stabilnější, rychlejší a podporuje pro případné potřeby i více kamer.

Kód v openFrameworks se dělí do třech hlavních částí. Jedná se o setup(), update() a draw(). Sekce setup slouží pro počáteční nastavení programu, proměnných apod., update obsahuje výpočetní a aktualizací část a draw má na starosti vykreslování.

Program se snaží vykonávat všechny části tak často, jak to lze. V update i draw se může využít funkce ofGetElapsedTimef(), která vrací vteřiny v jednotkách float od spuštění programu nebo ofGetElapsedTimeMillis(), která vrací milisekundy od resetování čítače. Pomocí využití modulo funkce (i jiných) můžeme ovlivnit jak často se bude vykonávat každá aktualizací část nebo její podčást, jelikož lze předpokládat, že člověk nemění gesta rychleji, než je počítač zpracovává. Omezení snímků určených ke zpracování programem lze povést i pomocí funkce isFrameNew(), která vrací boolean hodnotu určující, jestli se snímek změnil či nikoliv.

### 3.2 Hardware

#### 3.2.1 Kinect

K implementaci této práce je využita kamera Kinect v2. Zdroje, ze kterých lze data využít k potřebám aplikace jsou RGB kamera s rozlišením 1920 x 1080 pixelů a kamera na snímání hloubky s rozlišením 512 x 424 pixelů.

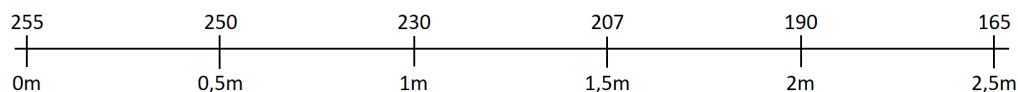
Jedná se o relativně levnou a dostupnou kameru, která se v době zpracování této práce (začátek roku 2018) dá pořídit do tří tisíc korun.

### 3.3 Zpracovatelné vstupy

Obě kamery mohou poskytovat užitečné informace pro účely projektu. RGB kamera nabízí pole pixelů, kde každý má složky RGB (0 až 255). Lze využít v kombinaci s robustním programem, který by na základě barvy, tvarů a dalších vylučovacích prvků správně detekoval nejdříve ruku a poté i gesta. Jedná se o znatelně náročnější způsob.

Hloubková kamera poskytuje pole s hodnotami 0 až 255 dle vzorce pro výpočet vzdálenosti na základě doby letu infračerveného světla. Stupnice hodnot je klesající,

to jest nejbližší objekty mají hodnotu 255. Ve vzdálenosti 3 metry hodnoty klesnou na 150.



**Obrázek 9** Stupnice hodnot, kterých nabývá hloubková kamera v závislosti na vzdálenosti

Minimální vzdálenost pro správné měření vzdálenosti je 0.5 m a maximální 4.5 m. Při větší vzdálenosti bude Kinect stále ještě detekovat věci v zorném poli, ale bude poskytovat nepřesná data a hloubková kamera pozbyde užitku. Velký vliv má i rozlišení, které je pro velké vzdálenosti nedostačující. V dálce 5 m od kamery je rozlišení 7 cm.

Využitelné parametry:

- Vzdálenost od kamery
- Vzdálenost mezi pixely
- Úhly
- Barvy pixelů

#### 3.3.1 RGB video

Zpracování obrazu z RGB videa nebude v této práci implementováno, ale patří do jedné z variant, jak detekovat ruku a následně i gesta.

komentář: nejsem si jistá, jestli to sem patří, ale do osnovy/obsahu práce to vytváří kompletnější dojem

#### 3.3.2 Hloubkové video

Doporučený postup by měl sestávat ze čtyř částí:

1. Úvodní nastavení
2. Nalezení ruky
3. Identifikace gesta
4. Vykreslení ruky (volitelné)

##### Úvodní nastavení

Lze zde dát uživateli na vědomí podmínky interakce s kamerou. Například pozice ruky musí být v prostřední třetině. Alternativou může být počáteční umístění ruky do prostředí kamery, aby mohla dále vycházet z umístění za využívání historie nebo aby se zkrátila doba hledání ruky. V této aplikaci je nejen počáteční podmínkou, že ruka musí být nejbližší objekt před kamerou.

##### Nalezení ruky

Jedná se o prostor pro určení podmínek a způsobu vyhledávání ruky. Tato část rozhoduje o robustnosti kódu a využitelnosti aplikace. Je vhodné nejdříve odstranit nulové a jinak nevyhovující pixely, vyplnit díry v obraze a počítat s dalšími chybami hardwaru a vlivy okolí. Pokud má být aplikace využívána v jakkoliv náročnějším okolí, musí obsahovat i vyloučení veškerých nevyhovujících objektů, jako jsou věci podobného tvaru či případně cizí ruce.

Předpoklad pro následující text je zpracování videa ohledně nulových pixelů. Měly by být ignorovány v rámci chybného vstupu. Zároveň platí, že čím více podmínek se uživateli na začátku předloží, tím jednodušší je nalezení ruky.

U správně ukazovaného gesta se dá předpokládat stejná vzdálenost jednotlivých bodů od kamery. Mírné odchylky se dají buď zahrnout offsetem, který rozdíl v určitém rozmezí bude považovat za ekvivalentní, nebo vyloučit úplně, jelikož se dá předpokládat, že to nebude mít vliv na zpracování, pokud se jedná o krajní body. Aby byl program uživatelsky přívětivý, v programu je zvolena odchylka na zachycení těchto rozdílů.

V tomto programu se vychází z předem definované vzdálenosti, ve které je nalezen největší možný čtverec, který by představoval dlaň. Pro účely nalezení by muselo existovat pole pixelů, které by reprezentovalo dlaň, či případně podobný objekt, který by se následně vyloučil podle dalších kritérií. Nadále by byla dlaň definovaná středem nalezeného čtverce a šířkou.

Následuje vylučování mylných objektů podle toho, jak robustní program má být a v jak obtížném prostředí bude detekce probíhat. Tato část je předmětem budoucích rozšíření, jelikož se jedná o komplexní problematiku.

Nutnou podmínkou ruky je, že obsahuje prsty, které mají proporcionálně ke dlani určitou velikost a vzdálenost od středu dlaně. Jelikož známe střed nalezené dlaně, lze identifikovat i prsty, respektive konečky prstů. Ze začátku musí být známo jaké gesto bude ruka zobrazovat při inicializaci, aby se dala nalézt. Pro ilustraci bude rozebráno gesto se všemi prsty nataženými.

Nejjednodušší metodou je nalezení lokálních maxim po šíři dlaně. Když se dlaň rozdělí na čtyři části a v každé se nalezne maximum po ose y (ose x pro horizontální polohu ruky), pro každý prst se změří vzdálenost špičky prstu od středu dlaně. Pokud se bude výrazně odchylovat od proporcí běžné ruky, bude nalezený objekt vyloučen.

Rafinovanější programy využívají tvorbu skeletonu pro sledování a identifikaci jak postavy, tak i ruky. Jedná se o postup nalezení kostí a jejich kloubů, které jsou reprezentovány čarami, což pak reprezentuje ruku (případně tělo). Jedná se o možný, ale mírně náročnější a zdlouhavější postup.

### Identifikace gesta

Výběr jednotlivých gest je vhodné prodiskutovat s jejich potenciálními uživateli a rozhodnout se mezi dynamickými a statickými gesty. Případně by obě možnosti šly kombinovat, ale byl by to zbytečně komplikovaný postup. Pro gesta dynamická je potřeba udržovat si v paměti minulé stavy a stanovit dobu, po kterou by bylo přijatelné pro uživatele provádět gesto a zároveň to nemohlo být zaměnitelné nebo mylně identifikované. Nejjednodušší je identifikovat gesta statická, kde s využitím souřadnic lze vypočítat počet prstů, a tak v základní verzi bude prostor pro minimálně pět variant, pokud se pro zjednodušení vyloučí zavřená pěst. V momentě, kdy bude program podporovat identifikaci jednotlivých prstů od sebe, je k dispozici variant hned více, než by kdy bylo třeba nebo by bylo zapamatovatelné pro uživatele.

Počet prstů lze vypočítat po správné identifikaci jednotlivých prstů, a jedná se o nejjednodušší způsob klasifikace gesta. Pro přesnost se lze opřít i o úhly a vzdálenosti mezi středem dlaně a špičkami prstů podle vzorců uvedených v kapitole 2.1.4 u obrázku 8.

Sledování gesta může také probíhat pomocí ukládání historie. Za předpokladu, že se člověk hýbe pomaleji, než se střídají snímky k analýze, lze procházet menší část pole, například jen okolí místa, kde se posledně nacházela ruka s prsty a kontrolovat změny, zda počet ukázaných prstů se zmenšil nebo zvětšil. K tomuto účelu by stále stačilo gesto

definované počtem prstů bez závislosti na tom, o které konkrétně se jedná.

Otázkou je i jaký objekt má představovat gesto. Jedná-li se o číslo, je to nejjednodušší. Může to být ale i objekt, který obsahuje pět prstů, z nichž každý může být zvednutý nebo nikoliv. Pak je podstatně přehlednější implementace více než pěti gest. Identifikace je ale náročnější, jelikož každý nalezený prst se musí nejdříve identifikovat, poté aktualizovat objekt představující ruku před kamerou a porovnat s implementovanými gesty.

#### Vykreslení ruky

Objekty se přes video vykreslují pomocí FBO ("frame buffer object"). Jedná se o buffery s objekty, které je třeba vykreslit. Reprezentují plátno, na které se pomocí pomocí příkazů 'begin()' a 'end()' vykreslují 3D objekty a jednou za snímek či méně často (podle požadavků) se vykreslí pomocí příkazu 'draw()'. Pro lepší přehled se můžou jednotlivé prsty (konečky prstů) zvýraznit koulí, zatímco celá ruka krychlí. Vykreslují se pouze základní objekty, které upřesňují nalezenou pozici rukou a prstů, aby nebylo potřeba udržovat v paměti přesné okraje ruky, které ani nejsou potřeba.

## 3.4 Předzpracování

Kvůli přehlednější práci s matematickými parametry obrazu se nejdříve načte video, které je uloženo v 1D poli do 2D pole pro další zpracování.

### 3.4.1 Filtrace

Jelikož se do obrazu dostane značné množství chybných hodnot, je potřeba je nejdříve co nejvíce eliminovat. Za tímto účelem je použit medián z okolí. Kvůli překreslování aktuálních hodnot jsou pro další výpočty použita data z pole zrcadlicího originální data. Pro každý pixel je načteno okolí o požadované velikosti, které se seřadí podle velikosti a jako výsledná hodnota se vezme hodnota z prostřední pozice.

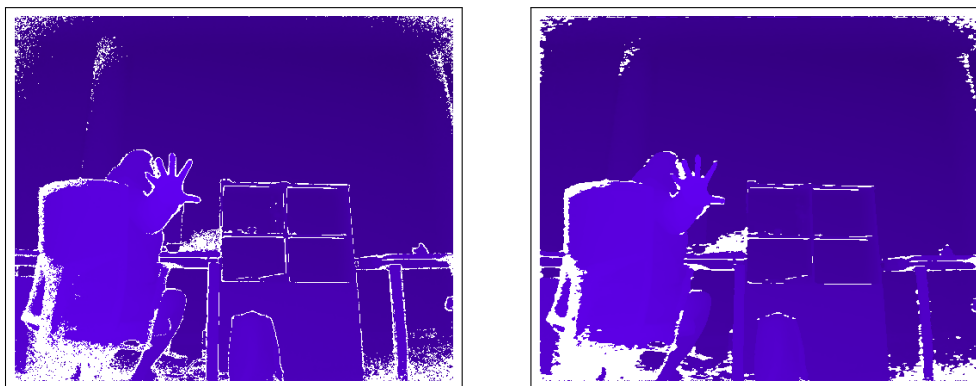
Velikost okolí by se měla vybrat empiricky. Čím větší okolí, tím více je výsledek rozmazaný, ale zato obsahuje menší počet odchýlených hodnot, které by mohly narušit průběh zpracování. Pokud se vezme okolí malé, zůstane jich více, ale lépe se zachovají tvary. Na následujících obrázcích lze pozorovat rozdíl mezi okolím 9 (obrázek 9) a okolím 25 (obrázek 10).

### 3.4.2 Prahování

Pro manipulaci s načtenými tvary a výpočty mezi jednotlivými body je snazší pracovat s binárním obrazem. Nejdříve se obraz projde a najde nejbližší objekt (největší hodnota). Díky předchozí filtraci nezkreslí tuto hodnotu žádný špatně naměřený pixel. Následně se již porovná každý jednotlivý pixel s hodnotou a vytvoří se pole s hodnotami 1 a 0.

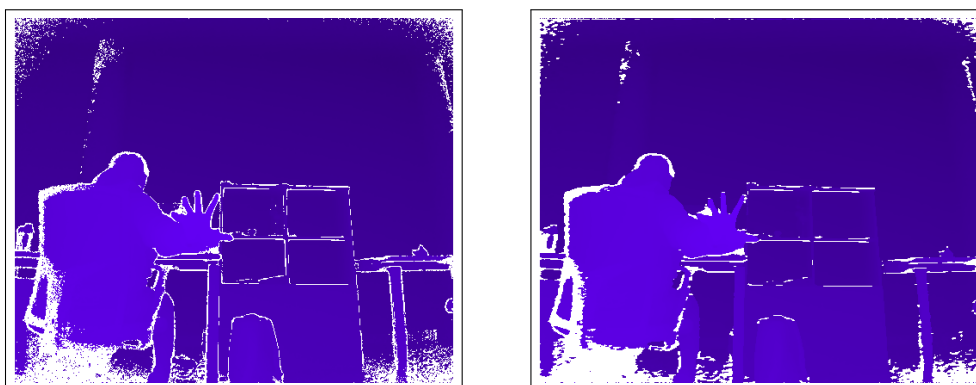
Jelikož lze předpokládat, že člověk nebude mít vždy ruku striktně kolmo k pohledu kamery, je záhodno odečíst od této hodnoty odchylku. Výsledná hodnota tedy záleží na tom, zda původní pixel byl blíže, než nejbližší objekt s ohledem na odchylku.

Na obrázcích 11, 12 a 13 lze pozorovat změny s ohledem na různé tolerance naklonění ruky. S odchylkou 10 vidíme, že ještě kus dlaně chybí, při odchylce 15 je s rukou načtený i znatelný kus předloktí a k tomu i nejednotný kus jiného objektu. Při toleranci pouze



**Obrázek 10** Medián vzatý z okolí mohutnosti 9  
a) originální obraz b) po úpravě

fig:neigh9



**Obrázek 11** Medián vzatý z okolí mohutnosti 25  
a) originální obraz b) po úpravě

fig:neigh25

12 je stále načteno moc pixelů a tak se jeví 10 jako nejvhodnější tolerance.



**Obrázek 12** Prahování s odchylkou 10  
a) originální obraz b) po filtraci c) binární obraz

fig:tresh10

### 3.5 Detekce ruky

Definujeme-li ruku jako nejbližší objekt, pak již při prahování je nalezena. Pro lepší názornost se ukládá umístění objektu pomocí pole souřadnic. Na videu je vizualizován pomocí červeného čtverce, který má střed v zprůměrovaných souřadnicích ( $X_{avg}$ ,  $Y_{avg}$ ) a velikost odvozenou od velikosti pole. Na následujících obrázcích je vidět rozdíl mezi



**Obrázek 13** Prahování s odchylkou 12

a) originální obraz b) po filtraci c) binární obraz

fig:tresh11

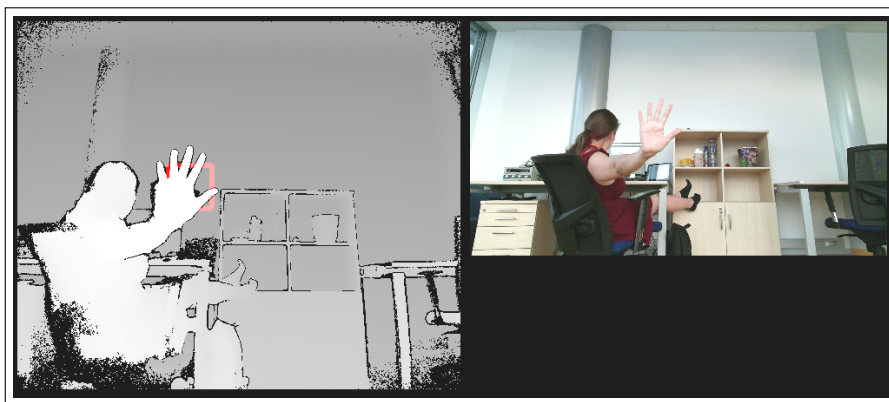


**Obrázek 14** Prahování s odchylkou 15

a) originální obraz b) po filtraci c) binární obraz

fig:tresh12

vyhledáváním v originálních datech a nebere v potaz chybné hodnoty a vyhledáváním v binárním obraze s vyfiltrovaným šumem.



**Obrázek 15** Obrázek ukazuje největší nalezený nejbližší objekt z originálních dat.

quare\_depth

Již z obrázků lze poznat, že vyhledávání v binárních datech je přesnější.

### 3.6 Detekce dlaně

V binárním obraze se vyhledá největší čtvercová matice, kterou lze považovat za dlaň. Vytvoří se pomocné matice, do které se zapíše krajové hodnoty binárního obrazu a poté se binární pole prochází z levého horního rohu.

Pokud je na dané pozici binárního obrazu nula, přepíše se i do matice. Pokud je tam ale jedna, pak se nalezne minimum třech sousedů z levého horního rohu. K minumu se přičte daná jedna, čímž se zvětší velikost čtvercové matice. Ve výsledku tak největší





**Obrázek 16** Největší nejbližší objekt nalezený v binárních vyfiltrovaných datech.

fig:square\_bin

číslo, které je nalezené v pomocné matici, představuje pravý dolní roh největší matice a zároveň její velikost [23].

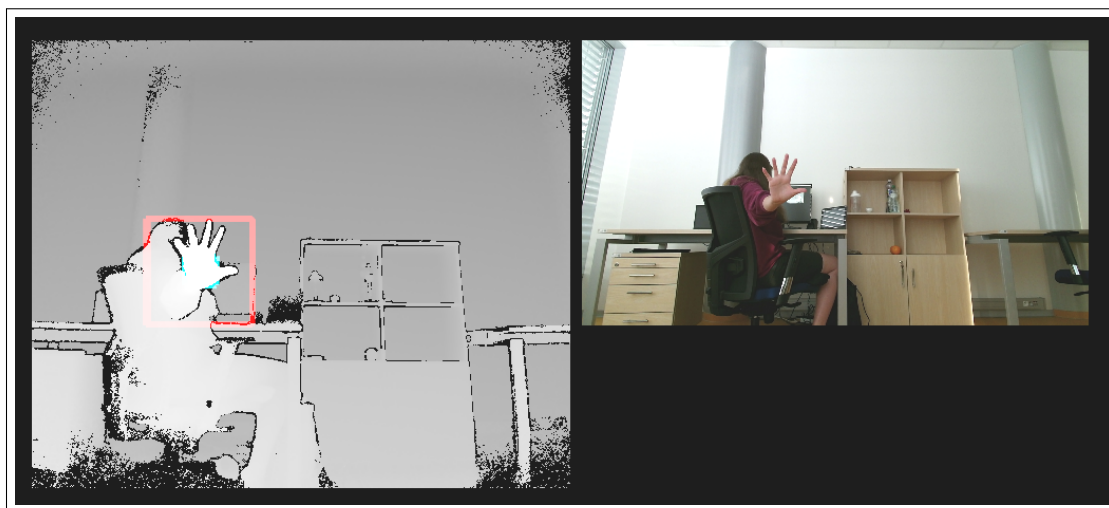
0	1	0	1
0	1	1	1
1	1	1	1
0	1	1	1

0	1	0	1
0	1	1	1
1	1	2	2
0	1	2	3

fig: palm

**Obrázek 17** Nalezení největší jednotkové matice v binárním obraze.

Vykresluje se pomocí zeleného čtverce, jenž má velikost nalezené jednotkové matice a jeho střed má souřadnice  $X_c$ ,  $Y_c$ , které budou následně využívány jako střed dlaně.



**Obrázek 18** Červený čtverec znázorňuje polohu a velikost nalezeného nejbližšího objektu. Zelený čtverec obkresluje největší nalezený čtverec v objektu.

fig:square

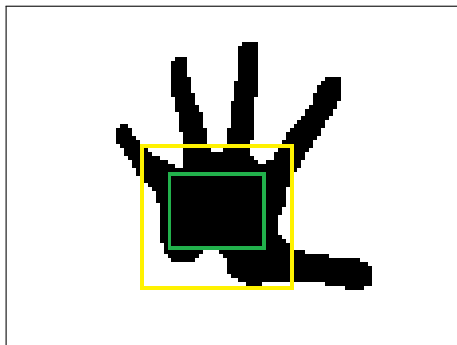
### 3.7 Detekce prstů

V aplikaci je implementováno několik způsobů nalezení prstů. Ne všechny jsou efektivní a od některých se upustilo (viz kapitola statistika), ale pro různost postupu jsou zde uvedeny.

### 3.7.1 Detekce směru prstů

Za předpokladu, že jsou prsty nezaměnitelně hubenější, než předloktí, lze okolo dlaně ve vzdálenosti *offset\_palm* od okraje vést čáru ve všech čtyřech směrem. Následně se iteruje podél jednotlivých čar a detekují změny v binárním obraze a zaznamenává se počet předpokládaných prstů. Jelikož je potřeba znát i pozici palce, ukládají se dva směry prstů.

Empiricky nalezená nejlepší hodnota pro *offset\_palm* je  $\frac{5}{7}$  šířky dlaně.



Obrázek 19 todo Popisek.

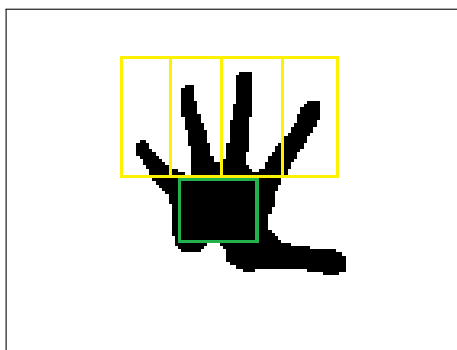
hereFingers

### 3.7.2 Detekce konců prstů

Všechny postupy se zakládají na prohledávání v okolí dlaně. Maximální délka prstů se odvozuje od nalezené velikosti dlaně. Prsty nesmí být ani příliš dlouhé ani příliš krátké.

Nejjednodušší postup spočívá v rozdělení strany, na které se nachází prsty, na čtyři části, kde každá náleží jednotlivým prstům. Rozdělení je rovnoměrné a nepodporuje tak extrémní odchylky. Celkový obdélník musí být širší než dlaň a nebere se v potaz, který prst je v daném obdélníku nalezen. To znamená, že pokud se ukazují dva prsty, přičemž se některý/oba nachází ve vedlejším obdélníku, zaznamená se správně pouze počet prstů.

Jednotlivé části se následně procházejí a hledá se pixel, který ještě patří prstu a je nejvýše/nejniže dle orientace.



Obrázek 20 todo Popisek.

ndFingerTips

Další způsob zvyšuje použitelnost předchozí metody. Stále se prochází pole, ale celkový obdélník se předá v kuse a hledá se největší vzdálenost od středu dlaně. Stále platí

omezení délky, tudíž je přesnější největší možná vzdálenost, která stále patří prstu.

Funkce `findFingerTips3` si ukládá souřadnice, na kterých byly detekovány prsty z předchozí kapitoly. Následně si dopočítá střed prstu a po přímce prohledává binární obraz tak daleko, dokud nenarazí na konec prstu (nulu v binárním obraze).

## 3.8 Statistika

Vyhodnocení funkčnosti a použitelnosti navržených metod.

## **4 Závěr**

### **4.1 Možnosti využití gest při ovládání mobilního robotu**

# Literatura

- [15] [1] Jan Pikora. “Implementace grafických filtrů pro zpracování rastrového obrazu”. bakalářská práce. Masarykova univerzita, 2008.
- [6] [2] Chikahito Nakajima a spol. “Full-body person recognition system”. In: *Pattern Recognition 36* (2003).
- [9] [3] Yang Ran a Larry Davis Mohamed Hussein Wael Abd-Elmageed. “Real-Time Human Detection in Uncontrolled Camera Motion Environments”.
- [10] [4] Michael Van den Bergh. “Real-time 3D Hand Gesture Interaction with a Robot for Understanding Directions from Humans”. In: *IEEE International Symposium on Robot and Human Interactive Communication* (2011).
- [7] [5] D Gavrilu. “Pedestrian detection from a moving vehicle”. In: *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part II* (2000).
- [11] [6] B. Winterstein a S. Müller G. Rigoll. “Robust Person Tracking in Real Scenarios with Non-Stationary Background Using a Statistical Computer Vision Approach”.
- [14] [7] Christian von Hardenberg a François Bérard. “Bare-Hand Human-Computer Interaction”. In: *ACM Workshop on Perceptive User Interfaces*. 2001.
- [3] [8] A. Erol et al. “Computer Vision and Image Understanding”. 2007.
- [5] [9] Dieter W Fellner Hyosun Kim. “Interaction with Hand Gesture for a Back-Projection Wall”. 2003.
- [12] [10] M. Abou Zliekha Noor Shaker. “Real-time Finger Tracking for Interaction”. In: *International Symposium on image and Signal Processing and Analysis*. 2007.
- [4] [11] François Bérard Julien Letessier. “Visual Tracking of Bare Fingers for Interactive Surfaces”.
- [13] [12] Rod McCall a Wolfgang Broll Georg Hackenberg. “Lightweight Palm and Finger Tracking for Real-Time 3D Gesture Control”.
- [21] [13] *Microsoft Kinect used in rehabilitation of stroke patients*. 4. lis. 2011. URL: <http://metro.co.uk/2011/11/04/microsoft-kinect-used-in-rehabilitation-of-stroke-patients-208625/>.
- [1] [14] *openFrameworks*. URL: <http://openframeworks.cc> (cit. 13.10.2017).
- [2] [15] *ofxKinectV2*. URL: <http://github.com/ofTheo/ofxKinectV2> (cit. 13.09.2017).