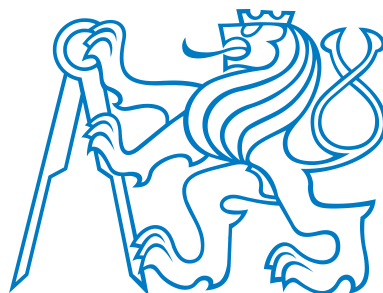


České vysoké učení technické v Praze  
Fakulta elektrotechnická



Diplomová práce

## **Toolbox pro tvorbu interaktivních multimediálních instalací**

*Filip Pokorný*

Vedoucí práce: Ing. Roman Berka, Ph.D.

Studijní program: Elektrotechnika a informatika

Obor: Informatika a výpočetní technika

květen 2011



## Poděkování

Rád bych poděkoval Ing. Arch. Markétě Veselé za možnost realizovat tento projekt v prostorách kanceláří TV Nova, Janu Herodesovi za toleranci a úlevy v zaměstnání a Ing. Romanu Berkovi, Ph.D. za podporu ve vypracování tohoto projektu.



## Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil pouze podklady uvedené v příloženém seznamu.

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 13.5. 2011

.....



## **Abstrakt**

Tato práce dokumentuje vytvoření a implementaci systému detekujícího pohyby uživatele, dále pak diskretizaci těchto pohybů a prototyp prezentující zde využití techniky. Hlavní důraz bude kladen na modularitu instalace. Vstup bude zajištěn kamerovým systémem, výstup projekto-rem se zpětnou projekcí. Systém simuluje velkoplošnou dotykovou obrazovku. Dokument shrnuje řešení problémů rozvržení kamer, sběru obrazu, jeho vyhodnocení, eliminaci artefaktů, diskretizaci pohybu a transformaci do izometrického světa.

## **Abstract**

This disertation describes creation and implementation of system, which detects movements of user, discretization of its motion and prototype, which presents techniques used in project. Main emphasis will be on modularity of instalation. Input will be provided by camera system, output by overhead projector. System simulates large scale touchscreen. This document sumarizes solution for camera layout problem, image capture, its evaluation, elimination of artefacts, discretization of movements and transformaion to isometric world.





# Obsah

## Seznam obrázků

xiii

<b>1</b>	<b>Úvod</b>	<b>1</b>
1.1	Smysl projektu . . . . .	1
1.2	Terminologie . . . . .	1
1.3	State of The Art . . . . .	2
1.3.1	Detekce ruky . . . . .	2
1.3.2	Detekce člověka . . . . .	2
1.3.3	Rychlá rekonstrukce pohybu . . . . .	2
1.3.4	Stabilní a flexibilní rekonstrukce pohybu . . . . .	2
1.3.5	Obecný nástroj . . . . .	2
1.4	Vytyčení cílů projektu . . . . .	3
1.5	Zaměření inspirace v existujících projektech . . . . .	3
<b>2</b>	<b>Specifikace problému snímání polohy</b>	<b>5</b>
2.1	Příbuzné technologie . . . . .	5
2.1.1	Sony Move . . . . .	5
2.1.2	Nintendo Wii Motion Controller . . . . .	5
2.1.3	Microsoft Kinect . . . . .	6
2.1.4	Shrnutí existujících technologií . . . . .	6
2.2	Potenciální Hardware . . . . .	6
2.2.1	Výstupní rozhraní . . . . .	6
2.2.2	Vstupní rozhraní . . . . .	7
2.2.2.1	Kamery . . . . .	7
2.2.2.2	Ultrazvukové snímače . . . . .	7
2.2.2.3	Laserové vysílače a optické brány . . . . .	7
2.2.2.4	Indukční a kapacitní snímače . . . . .	7
2.2.3	Výběr vhodné technologie . . . . .	7
2.2.3.1	Kamery . . . . .	8
2.3	Obecné schéma problematiky . . . . .	8
<b>3</b>	<b>Analýza scény</b>	<b>9</b>
3.1	Vliv prostředí . . . . .	9
3.1.1	Materiál průmětny . . . . .	9
3.1.1.1	Matný LCD panel, či projekční matnice . . . . .	9
3.1.1.2	Lesklý povrch či sklo . . . . .	9
3.1.2	Světelné podmínky a pracovní spektrum světla . . . . .	9
3.1.3	Shrnutí - Optimální prostředí . . . . .	10
3.1.3.1	Modularita . . . . .	10
3.1.3.2	Citlivost detekce . . . . .	10
3.1.3.3	Rychlost . . . . .	10
<b>4</b>	<b>Algoritmy pro zpracování obrazu</b>	<b>13</b>
4.1	Metody potlačení šumu a zvýraznění hran . . . . .	13
4.1.1	Metoda lokálních extrém[7] . . . . .	13
4.1.2	Metoda dvojitě konvoluce[13] . . . . .	13
4.2	Transformace a mapování obrazu[14][12] . . . . .	13
4.3	Canny algoritmus pro detekci hran[3][17] . . . . .	14

<b>5</b>	<b>Zpracování obrazu</b>	<b>15</b>
5.1	Metody detekce snímání polohy . . . . .	15
5.1.1	Metoda rozdílového snímku . . . . .	15
5.1.1.1	Pseudokód zpracování obrazu . . . . .	15
5.1.1.2	Výhody metody . . . . .	15
5.1.1.3	Nevýhody metody . . . . .	16
5.1.2	Metoda detekce hran původního snímku . . . . .	16
5.1.2.1	Pseudokód zpracování obrazu . . . . .	16
5.1.2.2	Výhody metody . . . . .	16
5.1.2.3	Nevýhody metody . . . . .	16
5.1.3	Detekce několika disjunktních objektů . . . . .	16
5.2	Kalibrace a korelace . . . . .	17
5.2.1	Inicializační parametry . . . . .	17
5.2.1.1	Automatická kalibrace . . . . .	18
5.2.1.2	Poloautomatická kalibrace . . . . .	18
5.2.1.3	Manuální kalibrace . . . . .	20
5.2.1.4	Zajímavý poznatek ohledně funkce cvWarpPerspective . . . . .	20
5.2.2	Dynamické parametry . . . . .	21
5.2.2.1	Optimalizace detekce kontur . . . . .	21
5.2.2.2	Dynamické změny prostředí . . . . .	23
<b>6</b>	<b>Implementace programu</b>	<b>25</b>
6.1	Pomocná knihovna pro segmentaci obrazu . . . . .	25
6.1.1	Pomocné funkce knihovny OpenCV . . . . .	25
6.1.2	Předpokládaná rychlost odezvy . . . . .	27
6.2	Objektové schéma programu . . . . .	30
6.3	Problémy zpracování . . . . .	30
<b>7</b>	<b>Testování</b>	<b>31</b>
7.1	Metody Testů . . . . .	31
7.2	Výsledky Testů . . . . .	32
7.2.1	Latence . . . . .	32
7.2.2	Měření odchylky pozice . . . . .	32
7.2.3	Měření odchylky pozice bez doteku . . . . .	32
7.2.4	Měření odchylky pozice dvou bodů . . . . .	35
7.2.5	Měření stability . . . . .	35
<b>8</b>	<b>Diskuze</b>	<b>37</b>
8.1	Výsledky testů . . . . .	37
8.1.1	Latence . . . . .	37
8.1.2	Přesnost doteku . . . . .	37
8.1.3	Odchylka pozice nad obrazovkou bez doteku . . . . .	38
8.1.4	Stabilita . . . . .	38
8.2	Naplnění vytyčených cílů . . . . .	39
8.2.1	Bezchybná detekce polohy ruky . . . . .	39
8.2.2	Rychlost Odezvy . . . . .	39
8.2.3	Dobrá škálovatelnost technologie . . . . .	39
8.2.4	Spolehlivá detekce bez doteku . . . . .	40
8.2.5	Automatická přizpůsobivost technologie dynamickým změnám prostředí . . . . .	40
<b>9</b>	<b>Závěr</b>	<b>41</b>

9.1	Zhodnocení projektu . . . . .	41
9.2	Budoucí rozšíření tohoto projektu . . . . .	41
<b>A</b>	<b>Slovník pojmů</b>	<b>43</b>
<b>B</b>	<b>Uživatelská příručka</b>	<b>45</b>
B.1	Kalibrace . . . . .	45
B.2	Korelace . . . . .	45
B.3	Shrnutí . . . . .	46
<b>C</b>	<b>Aplikační příručka</b>	<b>47</b>
<b>D</b>	<b>Přiložené DVD</b>	<b>49</b>
<b>E</b>	<b>Literatura</b>	<b>51</b>



## Seznam obrázků

2.1	objektové schéma problematiky snímání polohy objektu pomocí kamer . . . . .	8
3.1	materiál plochy snímáný kamerou: vlevo lesklý, vpravo matný . . . . .	10
5.1	schéma rozložení 3 kamer vzhledem k obrazovce ve stísněných prostorách . . .	17
5.2	princip poloautomatické kalibrace kamery - ořezání výhledu . . . . .	19
5.3	zleva: chybná a korektní aplikace funkce <code>cvWarpPerspective</code> . . . . .	20
5.4	zleva: pohled z levé dolní kamery, transformovaný obraz obou kamer . . . . .	21
5.5	zleva: pohled levé kamery, zkonstruovaná scéna obou kamer při nesprávném použití funkce <code>cvWarpPerspective</code> . . . . .	22
5.6	zleva: test spolehlivosti funkce <code>cvWarpPerspective</code> a jejího nestandardního použití v pod jasnějším bodovým osvětlením . . . . .	22
5.7	ukázka složené transformace kalibrační šachovnice . . . . .	23
5.8	transformace ruky . . . . .	23
6.1	funkce <code>cvRunningAvg</code> knihovny <code>openCV</code> : vlevo originál, vpravo po aplikaci funkce	25
6.2	funkce <code>cvConvertScale</code> knihovny <code>openCV</code> : vlevo parametr intervalu, vpravo ambientní přídavek . . . . .	26
6.3	funkce <code>cvDilate</code> knihovny <code>openCV</code> : vlevo originál, vpravo po aplikaci funkce . .	26
6.4	funkce <code>cvErode</code> knihovny <code>openCV</code> : vlevo originál, vpravo po aplikaci funkce . .	27
6.5	funkce <code>cvGetPerspectiveTransform</code> knihovny <code>openCV</code> : vlevo původní snímek kamery, vpravo namapování na obrazovku při umístění kamery v pravém dolním rohu obrazovky . . . . .	28
6.6	funkce <code>cvCanny</code> knihovny <code>openCV</code> : vlevo transformovaný snímek kamery, vpravo kontura pohybujícího se ukazováčku po aplikaci detekce hran této funkce . . .	28
6.7	objektové schéma programu: Singletony jsou označeny číslovkou 1, hierarchie je shora dolů. . . . .	29
7.1	průběh měření odezvy detekce pozice . . . . .	31
7.2	měření odchylky při detekci dvou bodů . . . . .	31
7.3	tabulka reakcí detekce polohy na běžném osobním počítači pomocí knihovny <code>OpenCV</code> . . . . .	32
7.4	přesnost pozice doteku ve středu obrazovky . . . . .	33
7.5	odchylka pozice doteku v dolní části obrazovky . . . . .	33
7.6	odchylka pozice doteku v levé části obrazovky . . . . .	33
7.7	odchylka pozice doteku v pravé části obrazovky . . . . .	34
7.8	odchylka pozice doteku v horní části obrazovky . . . . .	34
7.9	odchylka pozice bez doteku 25mm nad obrazovkou . . . . .	34
7.10	odchylka doteku dvou bodů ve vzájemné vzdálenosti 120mm . . . . .	35
7.11	stabilita doteku . . . . .	35
8.1	pozadí detekce dvou bodů . . . . .	37



## 1 Úvod

### 1.1 Smysl projektu

Základní myšlenkou tohoto projektu je vytvoření vstupně výstupního rozhraní, jehož výstupem je obraz velkoplošného formátu a vstupem lidská ruka. Požadovaný výsledek by se měl tvářit jako dotyková obrazovka, jež není omezena velikostí.

Nestandardní *HCI* modely podobné tématu můžeme rozdělit na 2 skupiny. Na jedné straně stojí dotykové ovládání simulující ukazatel myši a to včetně starého zaběhnutého modelu v podobě tlačítek, roletových menu a záložek. Na straně druhé jsou alternativní technologie využívající gesta celého těla s omezeným prostorem působnosti, jako je například herní průmysl, prezentace, či sportovní aktivity. Obě skupiny mají svoje uplatnění a obě mají své nevýhody.

První model je příliš statický, často nevyužívá svých výhod v jednoduchosti tahu ruky, mávnutí či zamávání, možnosti více vstupů ( multitouch ) často bývají omezené a uživateli nepřírozené. Druhý model je zase příliš robustní, aby se dal použít na jemné a přesné ovládání. Jeho přesnost a spolehlivost bývá za hranicemi složitějších operací, například kancelářské aplikační využití nepřipadá v úvahu.

Oba modely nejsou a ani nemohou být komplementární. Jeden je ukotven k dotykové obrazovce, druhý využívá prázdný prostor, ve kterém je snímaná osoba uprostřed a zpravidla interaguje nehmotnými virtuálními předměty.

Motivací k této práci je myšlenka tyto technologie spojit a vytvořit systém reagující na dotek i pohyb uživatele.

Cílem je jednoduše využít stávající systém s minimem prostředků, udržet člověku přirozené mechanismy předchozí generace uživatelského rozhraní a přidat funkcionalitu snímání pohybu bez cizího předmětu ( *markerless motion capture* ). Prioritou je flexibilita a široký okruh použití, co se výstupních zařízení týče. Tedy LCD displejů, projektorů se zpětnou projekcí, televizí a jakékoliv jiné technologie interagující s uživatelem na jeho pohyby.

Pozornost bude zaměřena na použití technologií vyžívající bodových zdrojů a přijímačů signálů. Příklad je využitím projektoru a infračervených snímačů namísto LCD displeje s kapacitní fólií. Impulzem k tomuto rozhodnutí je stabilita ceny, velká modularita a relativní přenositelnost.

### 1.2 Terminologie

V dokumentu se vyskytuje několik pojmů, které definují širší okruh informací, či složený termín. Pro tyto pojmy byla vyhrazena klíčová slova.

- obrazovka je jakékoliv výstupní zařízení multimediální instalace generující odezvu uživateli. Typicky projekční plocha projektoru či obrazovka monitoru.
- akce použití ve smyslu ovládání obrazovky je stanovena jako aktivní příkaz v relativní pozici objektu vykonávajícího tento příkaz vzhledem k obrazovce. Příklad je kliknutí tlačítka myši nad aktivním prvkem uživatelského rozhraní.
- uživatel je člověk interagující se systémem. Jeho pohyby jsou zaznamenány a jemu je určena interakce multimediální instalace.
- obraz kamery je čisté netransformované pole pixelů, tak jak byly kamerou sejmuty do obrázku.
- mapa je obraz kamery po aplikaci transformační matice. Tento obraz je namapován do výřezu obrazovky tak, aby byl každý bod obrazu totožný s bodem reálného světa na

povrchu obrazovky. Mapa může být pouze část obrazu. Skládání map je k vytvoření kompletního obrazu.

### 1.3 State of The Art

Témata podobná zadání projektu jsou zpracovávána několika skupinami a společnostmi. Rozsah funkce tohoto projektu je stanoven na okamžitou reakci snímanému uživateli. Tuto funkcionalitu umožňují všechny technologie dotykových displayů na mechanické úrovni.

Pokud se zaměříme na detekci pomocí kamerového systému či podobných bezdrátových senzorů, velkou skupinu tvoří *Motion Capture* senzory určené pro snímání pohybů herce za účelem jejich uložení pro následnou aplikaci v budoucnosti.

#### 1.3.1 Detekce ruky

Studenti Iowa State University řeší v dokumentu *Markerless Motion Tracking*[2] detekci pohybů lidské ruky před kamerou a její virtualizaci do prostorového modelu. Byla zde využita webová kamera umístěna v rovině obrazovky před uživatele snímající dlaň a prsty pohledem pod kolmým úhlem vzhledem k ploše ruky. Pomocí knihovny OpenCV[8] je zde popsáno zpracování obrazu, vymaskování dlaně na základě barvy a detekce hran a určení polohy dlaně a prstů.

#### 1.3.2 Detekce člověka

B. Rosenhahn, T. Brox, U. G. Kersing, A. W. Smith, J. K. Gurnet a R. Klette v práci *A system for marker-less motion capture*[15] analyzují chyby systémů snímání polohy *Motion Capture* a rozšiřují možnosti systému detekcí hran herce v komplexních prostředích. Tento dokument vznikl k projektu detekce člověka několika kamerami, detekci kontury, syntetizování kostry a vytvoření obrazu člověka pomocí úhlů mezi jednotlivými kostmi. K realizaci byly využity funkce *level-set* algoritmů.

#### 1.3.3 Rychlá rekonstrukce pohybu

Společnost Organic Motion v současné době představuje produkt simulující Motion Capture pomocí 6 kamer umístěných okolo snímané scény s hercem detekující jeho přesné pohyby. Dle jejich prezentací je snímání přesné a odezva v člověku téměř nerozenatelném čase. Technologie není popsána v žádném dostupném dokumentu.

#### 1.3.4 Stabilní a flexibilní rekonstrukce pohybu

Společnost Microsoft, Sony a Nintendo paralelně vyvinuly ovladače pro herní konzole založený na technologii snímání člověka jednou či dvěma kamerami ve vzdálenosti 2 až 3 metrů v reálném čase. Microsoft Kinect využívá stereokameru s laserovou detekcí vzálenosti pro detekci kostry celého člověka. Sony a Nintendo technologie kombinují detekci ovladače v uživatelově ruce s gyroskopy.

#### 1.3.5 Obecný nástroj

Existuje nepřeberné množství anonymních projektů zabývajících se zpracováním obrazu, které využívají pomocnou knihovnu OpenCV. Většinou se jedná o částečně funkční implementace algoritmů vytržené z kontextu, některé však obsahují zajímavý postup či myšlenku. Knihovna je vytvořena speciálně pro zpracování obrazu a videa. Knihovna se mimo jiné zabývá implementací algoritmů pro segmentaci obrazu, filtrování, detekci hran či výpočet průměrného snímku v čase



a jeho rozdílového obrazu. Tato knihovna je vytvořena pod licencí volně šiřitelného software ( OpenSource ) pro jazyky rodiny C. V současné době představuje jednu z nejlepších volně dostupných prostředků v oblasti problematiky zpracování obrazu.

## 1.4 Vytyčení cílů projektu

Na základě zadání byly stanoveny cíle projektu:

1. Detekce lidské ruky na obrazovce. Bezchybná detekce polohy ruky a člověku přirozená akce použití.
2. Rychlost odezvy nerozeznatelná člověkem. Využitelnost v reálném čase bez prodlev.
3. Dobrá škálovatelnost využití v mezích fyzických rozměrů obrazovky v poměru k ceně technologie.
4. Spolehlivá detekce ruky uživatele před vlastním dotekem obrazovky. Detekce osoby či několika osob před obrazovkou.
5. Automatická přizpůsobivost technologie při dynamických změnách osvětlení a dalších ruchů.

## 1.5 Zaměření inspirace v existujících projektech

Nejbližší technologie tomuto projektem jsou technologie z oblasti počítačových her. Které jsou realizovány jako kompletní produkty řešící celou problematiku až do aplikační úrovně v cenově dostupném modelu. Vzhledem k nekompletní dokumentaci a částečně uzavřenému knowhow díky komerčnímu využití je studie těchto technologií omezena, z jejich vlastností je však možné se inspirovat. Následující kapitola se zabývá rozbořem těchto technologií a zmapováním jejich vlastností.



## 2 Specifikace problému snímání polohy

Snímání pohybu uživatele za účelem interakce multimediální scény je v současné době rozšířeno hlavně v herním průmyslu. Na rozdíl od standartních technologií detekce pozice, kdy je snímán jeden či několik bodů na rovině, je žádoucí navíc využít informace o pohybu a pozici celého těla.

### 2.1 Příbuzné technologie

Všechny následující technologie využívají kamerového systému. U všech je omezení ve vzdálenosti uživatele od obrazovky a to z obou stran intervalu.

#### 2.1.1 Sony Move

Pro detekci polohy ruky je nutný ovladač v ruce uživatele. Systém využívá jedné kamery umístěné v rovině obrazovky, která snímá bod vyzařující světlo ve viditelné části spektra. Uživatel musí být v dostatečné vzdálenosti od průmětny.

Obor využití se týká pouze herního průmyslu. Uživatelské rozhraní je v jeho spolupráci řešeno velmi okrajově. Nejčastěji je doplněno pouze z nutnosti a možnosti využívat pouze tento ovladač simulující virtuální ukazovátka.

Ovladač podobný válcovému dálkovému ovladači zakončený průsvitnou koulí s vloženou LED svítivou diodou o průměru několika centimetrů musí být vždy viditelný barevnou kamerou, která je vložena pod či nad obrazovkou. Jeden systém umí paralelně zpracovávat několik ovladačů, přičemž snímáný aktivní prvek musí být vždy zcela viditelný.

Systém vyžaduje častou kalibraci, kdy je uživatel nucen postupně umístit ovladač do několika přesně daných relativních poloh vůči svému tělu. K těmto kalibracím dochází řádově každou hodinu. V případě, že uživatel opustí svou pozici, na které stál, systém začne být nestabilní a je nutné znovu kalibrovat.

Obor možností ovládání je celkem široký. Díky zabudovanému gyroskopu je možné využití ovladače jako virtuálního ukazovátka. Díky nutnosti udržovat odstup od 2,5m do 3,5m bez fyzického kontaktu s plochou obrazovky je však velmi těžké využít technickou přesnost detekce.

Reakce systému jsou průměrné. Typická odezva ovladače je od 100 do 200ms[9], což je pro člověka citelné. Není to však tolik, aby se to stalo frustrujícím faktorem. Ovladač obsahuje několik tlačítek pro možnost akcí a příkazů.

Citlivost na světelné podmínky je navzdory použití jedné barevné kamery vynikající a ani dynamické změny nepůsobí problémy.

Hranice mobility jsou omezeny díky faktu, že pro dodržení volnosti pohybu je nutné zajistit prázdný prostor o velikosti cca 3 x 4 metry. Toto je pro osobní použití velmi omezující faktor. Technologie je úzce spjata s herní konzolí Playstation 3, cena celé sestavy se pohybuje do 10000Kč.

#### 2.1.2 Nintendo Wii Motion Controller

I v tomto případě je potřebný ovladač. V tomto případě je kamera umístěna v ovladači a v blízkosti obrazovky je umístěn panel se dvěma LED svítivými diodami. Tyto infračervené diody tvoří úsečku, která podle své velikosti a rotace na průmětně obrazu z kamery stanovuje pozici a natočení ruky. Uživatel musí udržovat minimální odstup od obrazovky, tak jako je tomu u předchozí technologie.

Technické parametry jsou rovněž velmi podobné Sony. Detekce pozice z dvou IR svítivých diod je řešen hardwarově v každém ovladači a tudíž je odezva velmi rychlá, člověku neznatelná. Systém je taktéž úzce spjat s hardwarem. Je kompatibilní s herní konzolí Nintendo Wii. Cena

sestavy je nižší, rychlost počítače se podepsala k celkové odezvě, která se tím stává srovnatelná s technologií Sony. Oproti Sony je kamera uložena v každém ovladači, v blízkosti obrazovky je panel s dvěma diodami.

### 2.1.3 Microsoft Kinect

Jediný systém, který nevyžaduje žádný další předmět napomáhající snímání polohy ruky uživatele. Systém je vybaven dvěma kamerami umístěnými relativně blízko sebe (jednotky centimetrů) snímající prostorový obraz. Pomocí knihovny pro zpracování obrazu je detekováno tělo uživatele, z něhož jsou spočítány souřadnice všech končetin. I v tomto případě je nutné udržovat odstup od obrazovky.

Systém dvou kamer se vzájemným odstupem několik desítek centimetrů v rovině obrazovky snímá a sestavuje kostru uživatele. Všem napomáhá detekce vyzářených paprsků detekující vzdálenost každého bodu scény. Těchto paprsků je vyzářeno velké množství a rozlišení je dostatečné k detekování dětské ruky.

Ovladačem je sám uživatel, není k dispozici žádné další zařízení, kterým by mohl uživatel provádět akce ani příkazy. Možnost ovládání je tedy omezeno pouze na mávnutím ruky. Většina akcí je zprostředkována standartním bezdrátovým gamepadem, jemuž je doplňkem.

Přesnost systému je navzdory použité technologii nízká. Odezva kolem 200ms[9] je znatelná, nutný odstup 3 - 4m je omezující. Tato revoluční technologie je v počátcích a obor využití velmi omezen na dobře viditelná gesta rukou jako je například rozpažení. Systém pozná několik uživatelů před obrazovkou a nevádí mu ani, pokud se uživatelé občas navzájem zastíní. Je tedy připojena podpora detekce identity.

Používání technologie je nezvyklé, jelikož je to jediný masově rozšířený systém bez nutnosti jakéhokoliv ovladače. Pocit z moci vlastního těla je však nenahraditelný a nebýt omezení a prodlevy, ovládání samotné působí velmi kladné pocity.

### 2.1.4 Shrnutí existujících technologií

Všechny výše uvedené technologie jsou výsledkem několikaletého výzkumu, využívají kameru ke snímání polohy a nejsou závislé na velikosti obrazovky. Jejich instalace a kalibrace je snadná. Všechny jsou použity v herním průmyslu a jsou určeny k ovládání uživatelského rozhraní.

Nevýhoda všech technologií je nutnost odstupů, která je pro projekt definovaný v tomto dokumentu nevyhovující. Díky odstupům a nemožnosti kontaktu s pevným objektem je ještě více omezena již tak omezená přesnost a stabilita ruky. Využití v jiném, než herním či prezentačním okruhu je těžko myslitelná a možnost rozvinout technologii do rozsahu dnes známých uživatelských rozhraní, možností a jejich rychlosti je bez radikální změny téměř nemožná.

Všechny výše uvedené způsoby ovládání však prakticky fungují a jejich využitelnost má navzdory nevýhodám smysl. Jejich cena je výborná a kompatibilita s obrazovkou monitoru, či projektorem výborná. Zaslouží si tedy pozornost.

## 2.2 Potenciální Hardware

### 2.2.1 Výstupní rozhraní

Systém by měl být navržen pro co největší škálu typů výstupních zařízení. Pozornost by měla být zaměřena na co největší formáty. Okruh alternativních řešení v této problematice má největší četnost v malých formátech. Tyto technologie, jako jsou například kapacitní a indukční displeje jsou běžně dostupné i na maloobchodním trhu. Další řešení v této oblasti by nepostrádalo smysl, pouze pokud by bylo jednodušší. Pozornost je tedy měla být zaměřena na velkoplošné obrazovky, či projektory.

### 2.2.2 Vstupní rozhraní

Největší pozornost je zaměřena na využití bodových snímačů. Tedy kamer, fotovoltaiické technologie a rádiového snímání polohy. Jednotky radarového snímání polohy dostupné na trhu jsou určeny pro snímání existence cizího objektu v dosahu, nikoliv jeho přesné polohy. Jejich použití tedy můžeme vyloučit.

#### 2.2.2.1 Kamery

Výhoda využití kamer je v hustotě informace na jednotku snímače. Jedna vhodně uložená kamera je schopna snímat celou scénu. Systém není omezen velikostí, jelikož paprsky snímané kamerou jsou omezeny pouze úhlem od osy snímače. Velikost snímané plochy je tedy možné regulovat polohou snímače relativně od snímané plochy. Analytické zpracování signálu je distribuováno vně systému a úzkým hrdlem se může stát výkon počítače, který bude signál přepočítávat na prostorové souřadnice. Další nevýhodou můžou být dynamické světelné podmínky prostředí, jež mohou způsobovat ruchy a artefakty zkreslující měření. Využití kamer způsobuje jisté prostorové omezení, jelikož je nutné je umístit v určité vzdálenosti, jak je uvedeno výše v tomto odstavci. Při umístění kamer je nutné brát v potaz možné rušení způsobené zdrojem světla obrazovky. V neposlední řadě může být nevýhoda prodleva v odezvě.

#### 2.2.2.2 Ultrazvukové snímače

Tyto snímače jsou primárně prostorové, což je velmi nežádoucí. Těchto snímačů je nutné umístit větší množství. Navíc je zde limitní vzdálenost v poměru k přesnosti. Systém je náchylný k ruchům a zároveň může způsobovat rušení.

#### 2.2.2.3 Laserové vysílače a optické brány

Výhoda této technologie je přesnost a rychlost. V porovnání s kamerami je signál téměř zpracován. Při použití této technologie je nutné použít velkého množství snímačů a přijímačů. Detekce pomocí triangulace nepřipadá v úvahu, jelikož disponuje velmi malou detekční vzdáleností. Při snesitelném výkonu jsou to řádově desítky centimetrů[19], přičemž chybovost roste exponenciálně. Zbývá tedy využití soustavy bodových snímačů. Vstupní část by tedy byla závislá na velikosti strany systému, což je největší nevýhoda. Alternativou je použití světelných bran, jež jsou pro toto použití vytvořeny. Rozsahem je možné je využít do velikosti několika metrů, což už může být dostačující. Nevýhodou je cena, jež roste s velikostí snímaného prostoru. Takový snímač pro plochu 3 x 5 metrů od společnosti Honeywell stojí kolem 2000USD[4].

#### 2.2.2.4 Indukční a kapacitní snímače

Tato technologie je relativně neoptimálnější. Je přesná, disponuje vícebodovou detekcí a není závislá na světelných podmínkách a je relativně dostupná. Omezení spočívá ve velikosti snímané plochy, která v dnešní době dosahuje řádově jednotek metrů. Cena prudce roste s velikostí a díky tomu je pro projekt nevhodná.

### 2.2.3 Výběr vhodné technologie

Nejlepší kombinací pro účely tohoto projektu je využití kamerového systému snímajícího také infračervenou část spektra spolu se zpětnou projekcí. Tato kombinace nepůsobí žádné omezení ve velikosti a nezpůsobuje interference. Nevýhodou je mnoho problémů, které z této kombinace plynou. Tyto problémy jsou však řešitelné v programové části projektu.

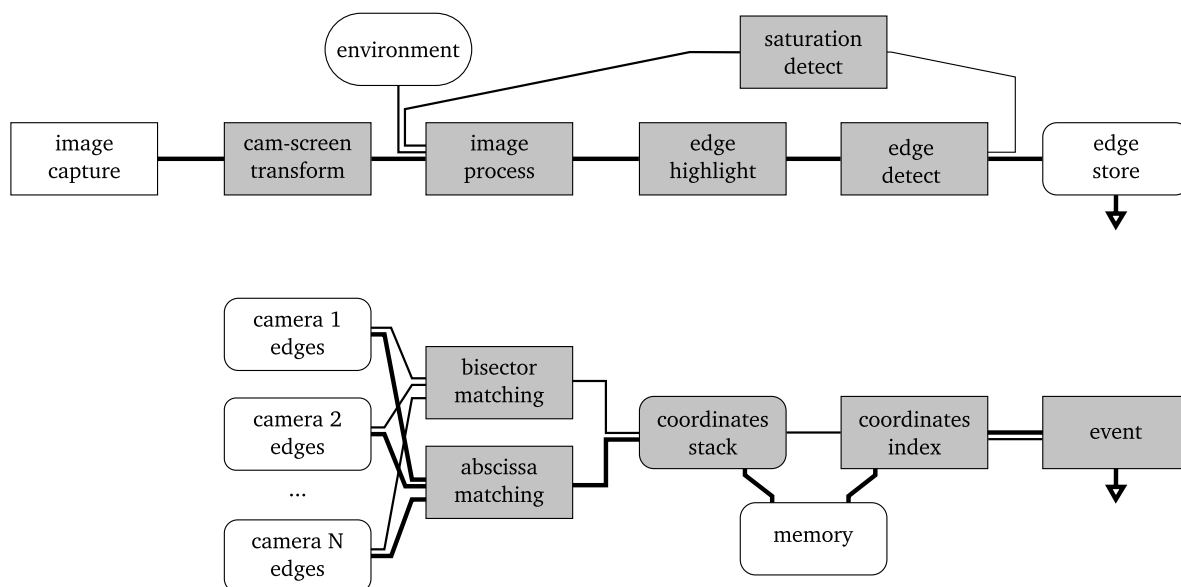
### 2.2.3.1 Kamery

Minimální počet kamer pro zjištění polohy na ploše je jedna. Tato kamera by měla být umístěna kolmo na průmětnu. Při využití zpětné projekce by teoreticky mohla být v prostoru za průmětnou. Kamera by ale musela prozářit průmětnu, aby mohla zabrat polohu ruky člověka stojící před průmětnou. V případě, že by byla kamera před průmětnou, uživatel si může clonit.

Při použití dvou kamer tvořících úsečku mezi uživatelem a průmětnou je problém clonění vyřešen. Největší přesnosti se dosáhne, když budou osy kamer ve vzájemném pravém úhlu. Tímto způsobem se vytvoří ortogonální systém, ve kterém bude každá kamera snímat jednu osu. Kamery tak budou snímat pouze ruce a navíc je možné detekovat vzdálenost ruky od obrazu.

Pro zlepšení přesnosti je možné k tomuto systému přidat třetí kameru, která bude snímat scénu před průmětnou. Při rušení a větších rozměrech obrazu je možné detekovat chyby, existenci uživatele ve větší vzdálenosti od průmětny, gesta paží, nebo třeba velikost postavy uživatele.

## 2.3 Obecné schéma problematiky



Obrázek 2.1: objektové schéma problematiky snímání polohy objektu pomocí kamer

### 3 Analýza scény

Tato kapitola se zabývá environmentálními okolnostmi scény a jejich vlivem na výsledek detekce.

#### 3.1 Vliv prostředí

Metoda pro detekci uživatele je závislá na parametrech prostředí, jako jsou světelné podmínky či materiál průmětny. U dynamických parametrů ( osvětlení ) je důležitá stabilita a schopnost systému se těmto změnám přizpůsobit či je ignorovat. U statických parametrů je důležité využívat výhod i nevýhod ve svůj prospěch.

##### 3.1.1 Materiál průmětný

Kamery zabírají obrazovku či průmětnu, která předává grafický výstup uživateli. Její vlastnosti hrají klíčovou roli v přesné detekci, stanovují minimální počet přijímačů pro věrnou detekci a omezují možnosti detekce doteku.

###### 3.1.1.1 Matný LCD panel, či projekční matnice

Pokud je povrch obrazovky matný, jak jej známe u matných LCD panelů, stín i odraz ruky je rozmělněn a jejich obraz má nižší kontrast, než originál. Tato zdánlivá výhoda se stane problémem při nevhodných světelných podmínkách, například pod velmi ostrým osvětlením s jedním světelným zdrojem pod minimálním úhlem tohoto zdroje v ose kamery. Kontura ruky snímán kamerou je tím protažena a pozice bodu zkreslena, jeho pozice je zdánlivě blíže ke kameře. Tuto vlastnost lze eliminovat vhodným nasvětlením, či přidáním druhé kamery kolmo k ose té první.

###### 3.1.1.2 Lesklý povrch či sklo

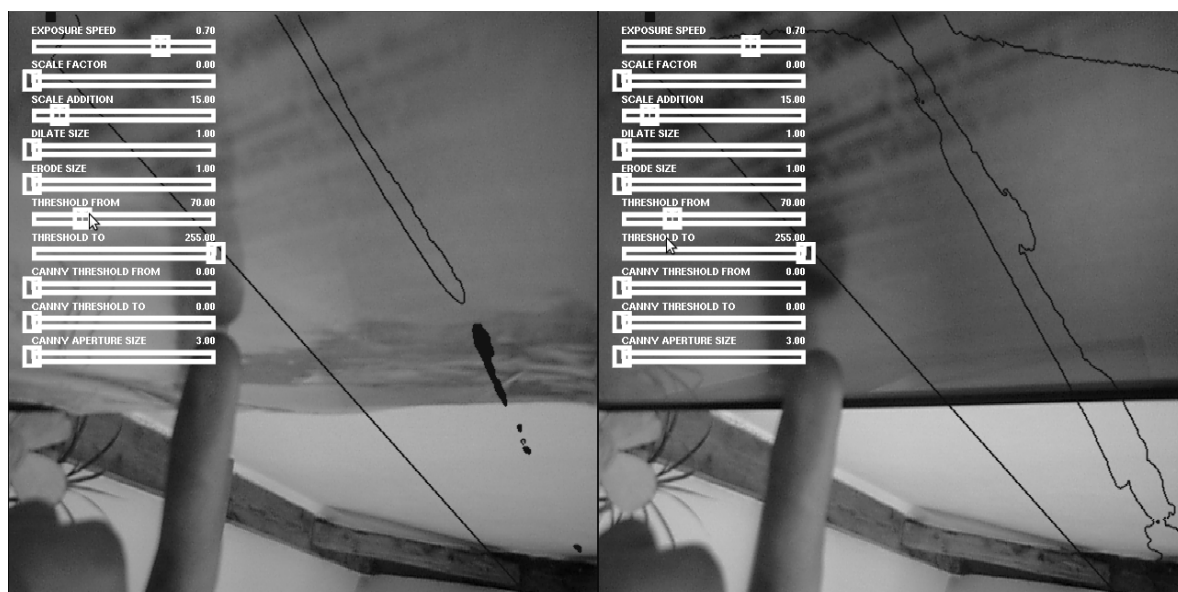
Lesklý povrch odráží objekty v blízkosti průmětny. Tento odraz je ostrý a detekovatelný kamerou. V porovnání s matným displayem je odraz dobře viditelný a změny jsou patrné také na kontuře ruky. Na obrázku 3.1 je vidět rozdíl mezi lesklým a matným povrchem. Černá kontura na obrázku zobrazuje detekci hran. Na matném obrázku jsou ruka i obraz špatně detekovány, odražené světlo je rozprostřeno ambientně a detekce selhává. Lesklý snímek na první pohled vypadá stejně, výsledek operace je však naprosto odlišný a kontura ruky je přesná. Navíc je zde patrná kontura odrazu ruky. Vzhledem k ostrému úhlu kamery vzhledem k obrazovce je mezi rukou a jejím odrazem mezera, která je blízká dvojnásobku vzdálenosti ruky od obrazovky. Díky tomuto efektu je možné detekovat přesnou pozici ruky od obrazovky a dokonce i dotek.

##### 3.1.2 Světelné podmínky a pracovní spektrum světla

Ostré světlo vytváří na materiálu snímáného kamerou ostrý stín, který může být detekovatelný kamerou. Při vhodné pozici světelného zdroje lze tohoto efektu využít stejně, jak je nakládáno s odrazem v kapitole 3.1.1.2. Místnost scény přirozeně obsahuje vedlejší zdroje světla, které nejsou součástí kompozice. Tyto zdroje je vhodné zastínit vlastním vhodně umístěným a silnějším zdrojem, pokud je to možné.

Při použití černobílých kamer, či kamer snímajících infračervenou složku světelného spektra je možné přisvětlení scény člověku neviditelným spektrem.

Výše uvedený scénář může být člověku nebezpečný. Jelikož lidské oko nevidí infračervenou část světelného spektra, duhovka pomocí hybných svalů nezareaguje a na sítnici prochází nadměrné množství světelného výkonu [6]. Člověku nebezpečné je zejména infračervené záření



Obrázek 3.1: materiál plochy snímáný kamerou: vlevo lesklý, vpravo matný

vlnovou délkou blízké viditelnému spektru, tzv. IR-A. Na toto záření je rohovka citlivá a při větších výkonech může způsobovat její popálení, či šedý zákal.

### 3.1.3 Shrnutí - Optimální prostředí

Nejvhodnější kombinací je lesklý povrch před kamerou s možností vlastního přisvětlení scény. Díky tomu odpadá problém s detekcí akce.

#### 3.1.3.1 Modularita

Předpoklady pro systém obrazovky a kamer v blízkosti její roviny umožňují širokou škálu použití. Uživatel může stát přímo u obrazovky, při standartním použití netvoří clonu kamerám, a tím neznemožňuje funkci. Předpoklad je v použití několika kamer zaměřujících se na jeden bod při vhodně nasvětlené scéně. Použití více kamer přesnost a spolehlivost detekce zvýší, byť to není nezbytně nutné.

#### 3.1.3.2 Citlivost detekce

Zběžný test citlivosti jedné VGA kamery umístěné v těsné blízkosti LCD panelu o velikosti 24" umožnil bezproblémovou detekci prstu, přičemž úzkým hrdlem v takto malém měřítku nebylo rozlišení kamery, nýbrž šum obrazu. Odhad limity tloušťky detekovaného objektu v tomto měřítku je kolem 5mm.

#### 3.1.3.3 Rychlost

Podle studie I. Scotta MacKenzie a Colina Ware [16] je minimální frekvence plynulé animace 10Hz, přičemž optimální frekvence pro lidské oko je 24Hz, kdy není rozpoznatelný rozdíl mezi snímky. Stávající široce rozšířené technologie ( Microsoft Kinect, Wii Remote a Sony Move ) mají průměrnou odezvu od 156ms ( Sony Move ) do 218ms ( Microsoft Kinect ) [9]. Odezva u těchto technologií je lidským okem snadno rozpoznatelná a výrazně omezuje využití technologií.

Pro dosažení cíle v oblasti odezvy je tedy vhodné dosáhnout horní hranice odezvy 42ms, což je rovno  $\frac{1}{24}$  vteřiny. Při Použití 60Hz kamery, která má vyhodnotí jeden snímek za 16.67ms, je



nutné segmentovat a zobrazit odezvu za 25.33ms, kamera s frekvencí 30Hz sníží se svou reakční dobou 33.33ms možnou prodlevu mezi sběrem snímku a vrácením vyhodnoceného obrázku na 8.66ms.

Frekvence snímání obrazu tedy hraje klíčovou roli zvláště, pokud je řádově shodná s optimální dobou vyhodnocení.



## 4 Algoritmy pro zpracování obrazu

### 4.1 Metody potlačení šumu a zvýraznění hran

#### 4.1.1 Metoda lokálních extrémů[7]

Tento algoritmus uchopí blízké okolí bodu, vyhledá bod s maximální intenzitou a tento bod uloží do celé plochy okolí.

$$dst(x, y) = \max_{(x', y') : element(x', y') \neq 0} src(x + x', y + y') \quad (4.1)$$

V následujícím kroku se provede operace inverzní:

$$dst(x, y) = \min_{(x', y') : element(x', y') \neq 0} src(x + x', y + y') \quad (4.2)$$

Pokud je tato operace provedena v 1D poli, výsledkem je zvýraznění a propojení jasnějších bodů mezi sebou.

#### 4.1.2 Metoda dvojté konvoluce[13]

Jednoduchý algoritmus zpracování obrazu využívající konvoluce gausiánu, znulování každé druhé řady a sloupce obrazu a druhé konvoluce.

Nejdříve je provedena konvoluce

$$f(x, y) * g(x, y) = \sum_{i=-k}^k \sum_{j=-l}^k f(x - i, y - j) \cdot g(i, j) \quad (4.3)$$

s následujícím jádrem  $g()$ :

$$\frac{1}{16} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \quad (4.4)$$

poté následuje nastavení každého druhého sloupce a řádku na hodnotu 0 a druhá konvoluce s jádrem vynásobeným 4x.

$$\frac{1}{4} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \quad (4.5)$$

### 4.2 Transformace a mapování obrazu[14][12]

Projekce bodů reálného světa do obrazu může být vyjádřeno:

$$m = PM^h \quad (4.6)$$

Bod obrázku v projektivních souřadnicích:

$$m = [x, y, z]^T \quad (4.7)$$

kde  $M^h = [X, Y, Z, 1]^T$  a  $P$  je  $3 \times 4$  projekční matice kamery. Ta jde rozložit jako:

$$P = K[R^T | -R^T t] \quad (4.8)$$

kde rotační matice  $R$  a translační vektor  $t$  reprezentují Euklidovskou transformaci mezi kamerou a světovými souřadnicemi.

Pokud se kamera otáčí okolo své optické osy, vektor  $t$  je nulový a projekce bodů prostoru je rovna  $m = AM$ , kde  $A$  je  $3 \times 3$  matice 3. řádu a  $M$  je bod v prostoru.

Matice  $K$  reprezentuje vlastní parametry kamery:

$$K = \begin{bmatrix} f & s & c_x \\ 0 & a \cdot f & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (4.9)$$

kde  $f$  je ohnisková vzdálenost,  $s$  je parametr zkosení,  $c$  je bod kamery a  $a$  je poměr stran (aspect ratio).

Transformace bodu z 3D světa do 2D obrazu kamery lze zapsat jako:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & s & c_x \\ 0 & a \cdot f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i & t_x \\ j & t_y \\ k & t_z \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (4.10)$$

kde  $i, j, k$  jsou lineárně nezávislé vektory reprezentující orientaci kamery,  $t$  posun.

### 4.3 Canny algoritmus pro detekci hran[3][17]

Jednotlivé kroky algoritmu:

1. Eliminace šumu Například pomocí konvoluce obrazu s odchylkou  $\sigma$  Gaussova normálního rozdělení od hodnoty 1 až 1.4. 4.1.2

$$G(x, y) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (4.11)$$

2. Zjištění gradientu hranovou detekcí Nejvhodnější bude Sobelův operátor[18], který není tak citlivý na šum.

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}, G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \quad (4.12)$$

Intenzita gradientu je vypočtena jako:

$$|\nabla G| = \sqrt{L_x^2 + L_y^2} \quad (4.13)$$

Vektor gradientu:

$$\theta = \text{atan2}(G_x, G_y) \quad (4.14)$$

3. Prahování s hysterezí Je zvolena minimální a maximální intenzita gradientu, který bude v příštím kroku zapsán. Tímto je možné oddělit silné hrany od slabých.
4. Hledání lokálních maxim Na základě směrů gradientů jsou vyhledány všechny hrany. Tyto hrany jsou zaneseny do binárního obrazu.

## 5 Zpracování obrazu

### 5.1 Metody detekce snímání polohy

Existuje několik možností, jak určit polohu ruky uživatele. Obě následující metody jsou citlivé na pohybující se objekty v okolí, jejich dynamické stíny, otřesy kamer a obrazovky a náhlou změnu osvětlení. Se všemi těmito artefakty se však mohou vypořádat, byť bude systém v obou případech za těchto okolností pravděpodobně vyřazen z provozu.

#### 5.1.1 Metoda rozdílového snímku

Metoda předzpracovává obraz tak, aby před detekováním hran zobrazoval pouze objekty k detekci určené.

##### 5.1.1.1 Pseudokód zpracování obrazu

Následující pseudokód je navržen tak, aby byly popřeny statické objekty nehledě na jejich barvu, tvar a kontrast. Tento postup označuje pouze vlastní snímání polohy ruky. Zanedbává úpravu parametrů tak, aby při snímání hran nebyl detekován šum jako falešný objekt. V posledním bodě, kde dochází k vypočtení pozice ukazatele ( například prstu ruky ), není stanovena podmínka určující který extrém bude vybrán. Toto je předem rozhodnuto podle pozice kamery vzhledem k obrazovce.

1. Sejmi vstup z kamery
2. Odečti od něj několik předchozích snímků a vytvoř *HDR* snímek
3. Ořež a přesuň viditelný rozsah *HDR* obrazu tak, aby byl pohybující se objekt kontrastní vzhledem k ambientnímu okolí.
4. Posuň interval barevného spektra tak, aby byly pohybující se objekty kontrastní vzhledem k ambientní barvě obrazu
5. Binarizuj obrázek ( *threshold* )
6. Vynes kontury
7. Ulož pozici coby extrém z konvexní obálky kontury

##### 5.1.1.2 Výhody metody

Tato metoda využívá odečtením několika posledních snímků kamery. Výhodou je, že obraz pro detekci zůstane prost statických hran prostředí v záběru kamery. Tento fakt není až tak důležitý, jelikož při použití perspektivní deformace pro namapování obrazu kamery na plochu obrazovky zůstane ve výhledu pouze část obrazovky, která hrany neobsahuje.

Pomocí detekce pouze pohybujících se objektů lze přesně určit dotek s obrazovkou. Lidská ruka nikdy není naprosto nehybná, pokud se neopírá o pevný bod a naopak vždycky je alespoň na chvíli nehybná, pokud se o pevný bod opře. Zmizení kontury pro detektor na čas v intervalu několika stovek milisekund a její opětovné zobrazení při pohybu opačným směrem může být považováno s téměř 100% pravděpodobností za dotek ( a tudíž aktivní akci substituující klik tlačítka myši atp. ).

### 5.1.1.3 Nevýhody metody

Nevýhodou této metody je nutnost zapamatování si polohy ruky v případě, že je nehybná. Program tak může špatně zareagovat. Pokud uživatel přesune ruku z místa, kde byl dříve nehybný, tak rychle že detekce pohybu nestihne zareagovat, vznikne chyba.

Další nevýhodou je větší prodleva v reakci systému na pohyb. Při detekci rozdílového snímku je kontura zpracována jako rozdíl pohybů ruky a tudíž interval rozdílů poloh dvou bodů v rovině. Výsledkem je průměr z těchto bodů, kde jeden z bodů je aktuální poloha a druhý poloha v minulém čase. Díky tomu je více znatelný dojem, že obraz dohání pohyb v reálném světě.

### 5.1.2 Metoda detekce hran původního snímku

Tato metoda předzpracovává obraz minimálně. K detekci hran je předán obrázek z kamery tak, jak byl sejmут. Před vlastní detekcí jsou pouze úpravy barevné palety, změna kontrastu či desaturace. Transformace perspektivy je také možné vykonat před detekcí hran stejně tak, jako je tomu u metody rozdílového snímku.

#### 5.1.2.1 Pseudokód zpracování obrazu

Postup ve zpracování obrazu touto metodou je o poznání jednodušší. Jeho výhody a nevýhody jsou však srovnatelné s předchozí metodou.

1. Sejmí vstup z kamery
2. Proved' jednoduché úpravy barev
3. Vynes kontury
4. Ulož polohu ruky jako extrém z konvexní obálky kontury

#### 5.1.2.2 Výhody metody

Tato metoda je více závislá na světelných podmínkách a materiálu snímaného povrchu. Při dobré konfiguraci lze artefakty z toho plynoucí využít ve prospěch, jak bylo naznačeno v kapitole 3.1. Vzdálenost stínu od ruky určí vzdálenost ruky od obrazovky. Stín či odraz od ruky lze rozeznat, pokud známe vzájemnou polohu kamery, obrazovky a světla. Pokud je ruka statická, kamera ji stále snímá. Není tedy nutné brát předpoklady v pohybu, jako je tomu u předchozí metody.

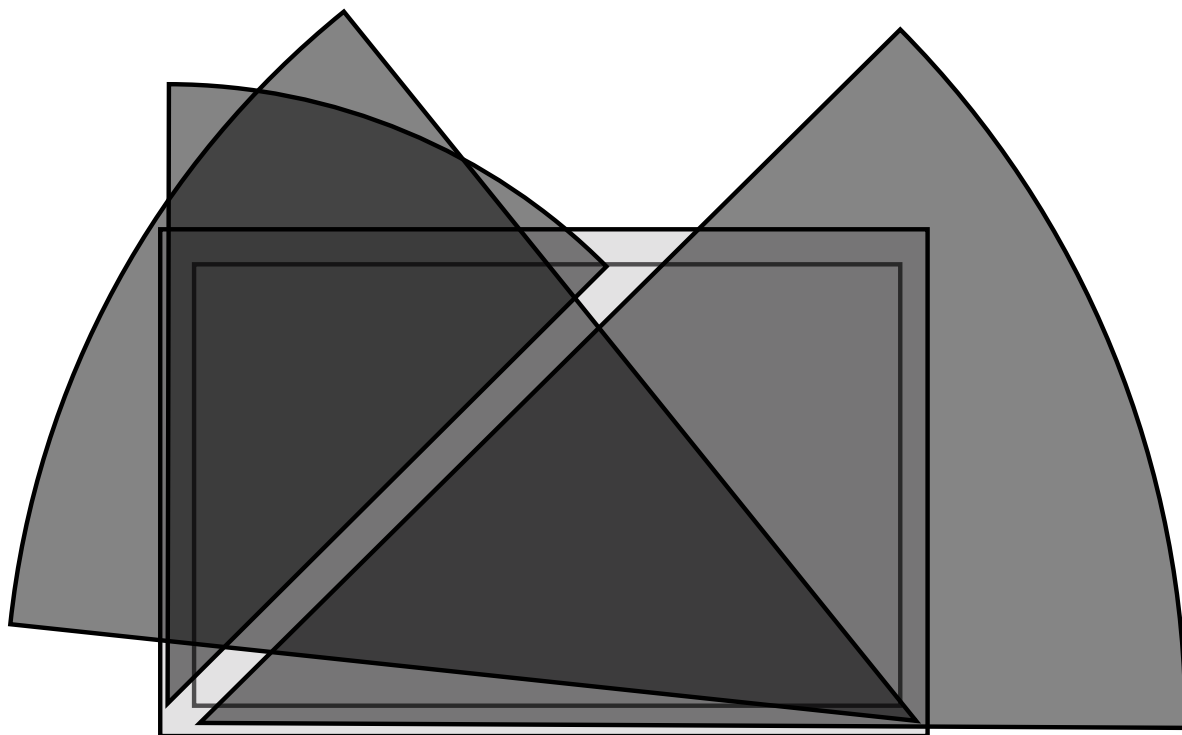
#### 5.1.2.3 Nevýhody metody

Pokud není možné správně nasvětlit scénu, či pokud se dokonce světelné podmínky ve scéně mění, metoda se stává nepoužitelnou. Tato jediná nevýhoda je dostačující k tomu, aby bylo implementováno více metod, které mohou být přepnuty buďto manuálně, či podle nějakého scénáře.

### 5.1.3 Detekce několika disjunktních objektů

Detekce několika objektů je možná v obou případech. Tato vlastnost je však omezena počtem kamer snímajících scénu. Ve stísněných prostorách, kde je nutné, aby byly kamery co nejbližší obrazovce je minimální počet kamer pro detekci celé plochy obrazovky 3 (obrázek 5.1, když uvažujeme standartní čočky kamer o šíři pohledu  $45^\circ$  a širokoúhlý obraz obrazovky o poměru 16:10. V tomto poměru zobrazuje většina světelných projektorů. Při takovémto minimálním počtu kamer je existuje šance, že si bude jeden objekt ve výhledu kamery clonit druhému,

například pokud budou v zákrytu v pravém horním rohu obrazovky. Tento roh je pokryt pouze jednou kamerou a pokud budou oba body v ose jediné kamery, která tuto plochu snímá, dojde k chybě. Z tohoto příkladu je možné odvodit obecné pravidlo:



Obrázek 5.1: schéma rozložení 3 kamer vzhledem k obrazovce ve stísněných prostorách

**Lemma 5.1.1.** *Pro detekci  $N$  objektů je třeba snímat plochu minimálně  $N$  kamerami tak, aby byl každý bod na obrazovce pokryt minimálně  $N$  kamerami.*

V praxi to znamená mít dvojnásobek kamer co počtu disjunktních pohybů. Příklad rozlišení 2 různých pohybů může být realizováno kamerou v každém rohu obrazu. Pokud bude rozsah záběru kamer alespoň  $45^\circ$ , dva páry kamer zaberou celou plochu obrazu 2x a bude možné snímat vždy minimálně 2 body na obrazovce naráz bez možnosti jejich vzájemného překrytí.

## 5.2 Kalibrace a korelace

Tato část se zabývá nastavením programu při instalaci systému a lazením v průběhu používání v případě, že dojde ke změně světelných podmínek.

### 5.2.1 Inicializační parametry

Při instalaci je systému a rozložení kamer je nutná transformace, ve které jsou kamery vzhledem k obrazovce.

Uvažujme situaci, že je systém samostatný a tedy není k dispozici klávesnice, ani myš. Systém je po instalaci slepý, protože kamery ještě nejsou namapovány na obrazovku. Jedinou možností je dát uživateli několik jednoduchých pokynů, které povedou ke správnému namapování perspektivy kamer do výhledu obrazovky, aby se stal systém konfigurovatelný například prostřednictvím menu. Ve chvíli, kdy nebudou kamery seřizeny by mohl být uživatel instruován

vyznačit před kamerou roh obrazovky, či přiložit obrázek se šachovnicí, podle kterého by kamera poznala svou relativní polohu vůči obrazovce.

Při použití černobílých kamer předsazených černým filtrem automatická kalibrace nepřipadá v úvahu, jelikož obrazovka bude kamerám neviditelná stejně tak jakýkoliv kalibrační obrazec na ní zobrazený.

#### 5.2.1.1 Automatická kalibrace

Pokud neplatí omezení uvedená výše, je možné na obrazovce zobrazit kalibrační obrázek, podle kterého se kamery automaticky zkonfigurují. Tato možnost nepřipadá v úvahu. Obraz nemůže být pro kamery viditelný, aby vizualizace nenarušovala detekci pohybu.

#### 5.2.1.2 Poloautomatická kalibrace

Referenční obrázek je nejlepší možnost, jak zjistit polohu kamer. Tento obrázek by mohl být přiložen ( například v papírové podobě ) na obrazovku. Nevhodnost řešení je kompenzována malou četností kalibrace, která je nutná pouze při změně polohy kamer, nikoliv při každém startu systému.

Další možností je existence kalibračních terčíků jako součástí obrazové plochy. Toto řešení může být nevhodné v případech, kde je důležitý vzhled kompozice.

Protože kamera snímá zpravidla pouze výsek obrazovky a celá obrazovka může mít libovolné rozměry, nebude jednoduché vytvořit systém tak, aby uživatel jen přiložil kus papíru o pevném formátu a kamery se zkalibrovaly. Jednou možností je přiložit kalibrační obrázek na několika místech, například v rozích.

Výhled kamery se nejdříve ořízne tak, aby byla viditelná pouze obrazovka. Následující krok bude vypočtení inverzní matice z perspektivní deformace linek. Tato matice se uloží a aplikuje na každý snímek.

Uvažujme příklad, kdy je scéna snímána širším polem kamer. Každá z kamer teoreticky nemusí znát pozici ostatních kamer, tedy alespoň těch, které zabírají disjunktní množinu bodů vzhledem k výřezu obrazu snímaném touto kamerou. Pokud bude detekován pohyb právě touto kamerou, pozice vypočtena pomocí inverzní matice pro tuto kameru bude rozeznána a zaznamenána. Tímto způsobem bude fungovat každá z kamer.

**Lemma 5.2.1.** *V případě poloautomatické kalibrace referenčním obrazcem musí být referenční obrazec přiložen nejvíce  $N$ -krát, kde  $N$  je maximum z hodnot počtu kamer a počtu hran obrazovky větších, než rozměr referenčního obrazce. Extrémní hodnoty  $N$  je dosaženo v případě, že všechny kamery zabírají vzájemně disjunktní plochy.*

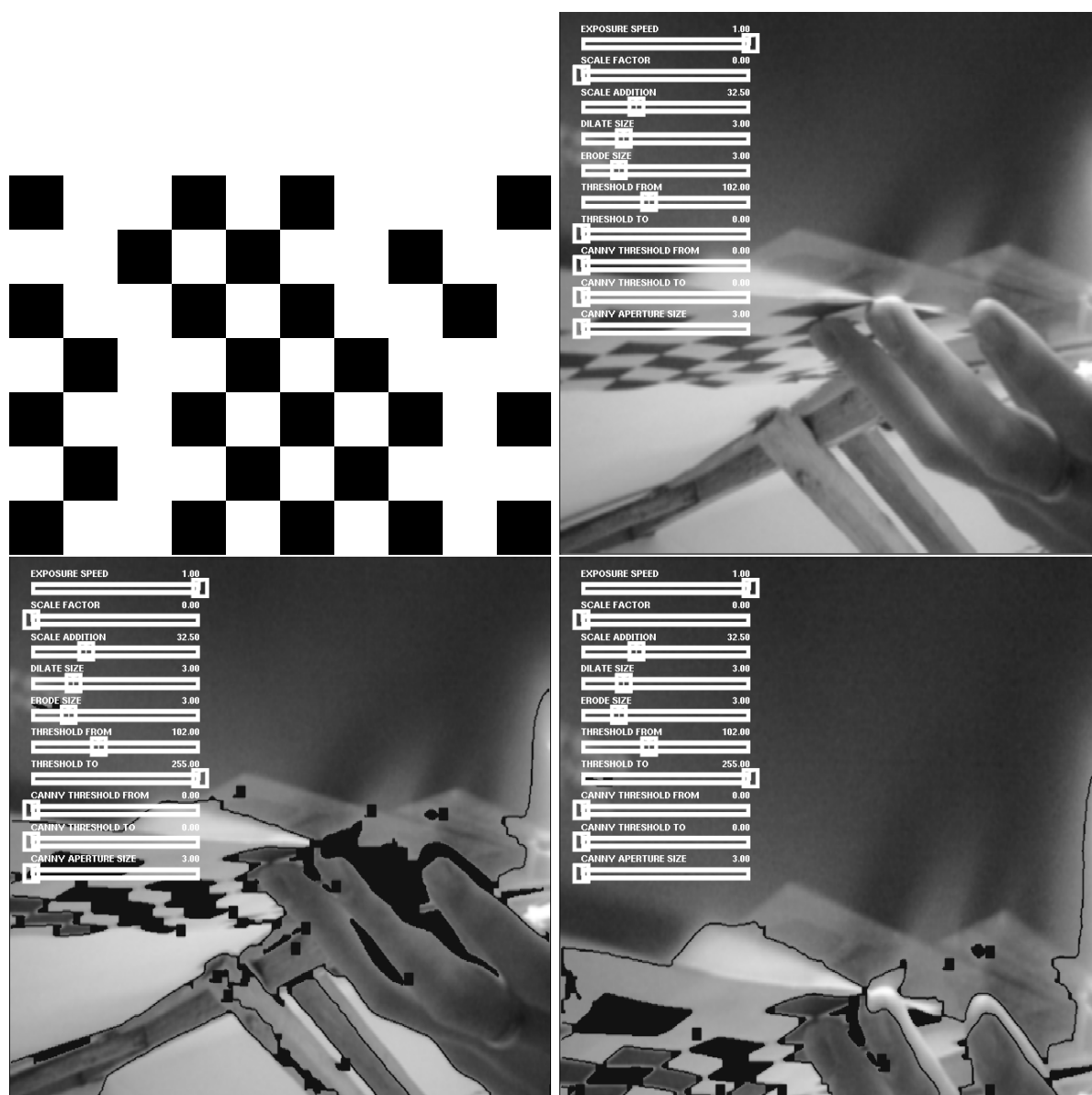
Lemma 5.2.1 říká, že se musí zkalibrovat každá kamera. Není však nutné spojovat kamery vzájemně do jednoho obrazu. Toto však platí pouze za okolností, že bude obrazec přiložen v předem určeném pořadí.

Na obrázku 5.2 je pokus o takovouto kalibraci. Vlevo nahoře je kalibrační obrázek, který byl přiložen na display. Následuje detekce tohoto obrázku a ořezání obrazu o vzdálenější hranu až po kalibrační obrázek.

Způsobů, jak vyznačit kamerám aktivní plochu ke snímání polohy je více. Namísto obrázků může být použita například reflexní páska okolo obrazovky. Všechny tyto principy vždy vytvoří chybu s nenulovou pravděpodobností

Díky faktu, že je před ořezáním scéna plná hran cizích nedefinovatelných objektů, se dostáváme do rekurze problému. Ořezání nemusí být provedeno přesně, což zanechá následky na přesnosti systému. Tento problém svým rozsahem překračuje meze kapitoly diplomové práce, zvláště pak, musí-li být tato kalibrace velmi přesná. Pro kvalitu zpracování bude využita manuální kalibrace.





Obrázek 5.2: princip poloautomatické kalibrace kamery - ořezání výhledu

### 5.2.1.3 Manuální kalibrace

Vzhledem k tomu, že se tato kalibrace provádí při instalaci a ukotvení kamer, výrazně ovlivňuje přesnost měření po celou dobu používání systému. Manuální kalibrace je ze známých možností nejpřesnější.

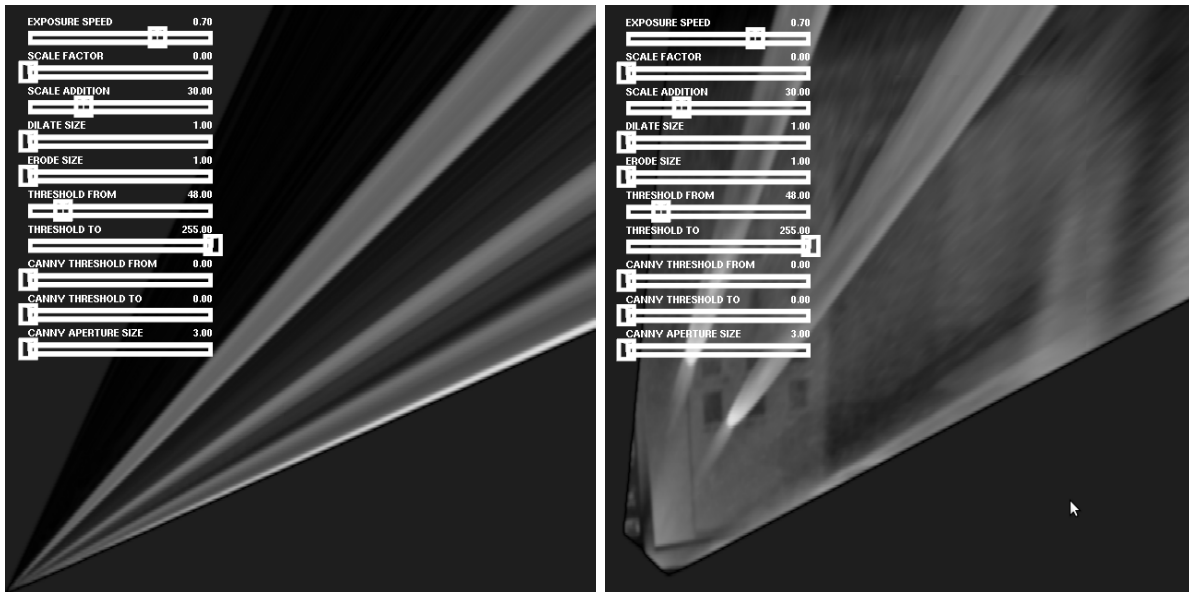
Manuální nastavení bodů a jejich obrazů je uživatelsky velmi složitá funkce. Je třeba připojit další polohovací zařízení, například myš. Při použití operace 5.1 je nutné zjistit obrazy 4 hraničních bodů zdrojového snímku. Projekce obrazu kamery na ploše je lichoběžník. Pro správné zobrazení perspektivy je nutné mapovat jej jako lichoběžník, jak je tomu na obrázku 5.3 vpravo.

Při umístění kamery v ose obrazovky je spodní hrana obrazu ve vzdálenosti 0 a horní v nekonečnu.

I pokud reálně nedosáhne kamera této limitní polohy, operace transformace při pomoci tří bodů ( protože dva 2 body v blízkosti kamery jsou totožné ) zobrazí zborcenou plochu, jelikož transformační matice ztrácí hodnotu. Funkce počítá transformační matici takto:

$$\begin{bmatrix} t_{i'} \cdot x_{-i} \\ t_{i'} \cdot y_{-i} \\ t_{i'} \end{bmatrix} = \text{mapovací matice} \cdot \begin{bmatrix} x_{-i} \\ y_{-i} \\ 1 \end{bmatrix} \quad (5.1)$$

kde  $(x_{i'}, y_{i'})$  je obraz  $(x_{-i}, y_{-i})$



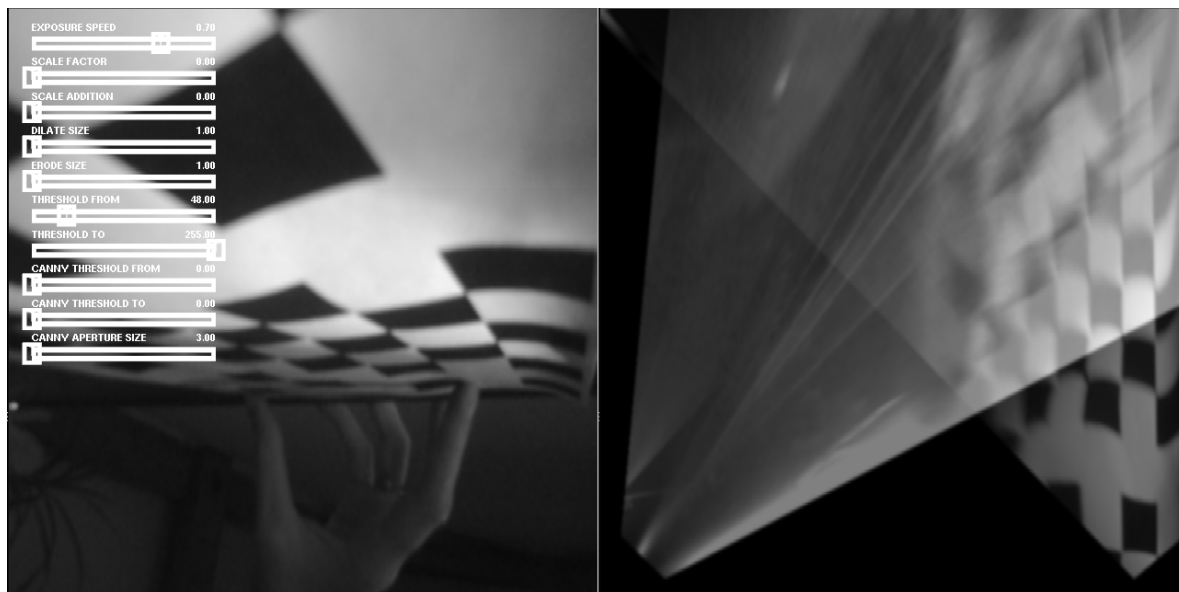
Obrázek 5.3: zleva: chybná a korektní aplikace funkce cvWarpPerspective

Kamera je nyní manuálně namapována do částí obrazovky. Na obrázcích 5.4 a 5.7 je zkouška při přiložení šachovnice. Namapovaná šachovnice by měla být zarovnaná k osám obrázku. Měkký papír, na kterém byla šachovnice vytištěna nebyl dokonale rovný a při rekonstrukci došlo k zesílení chyby. Na obrázku 5.3 vpravo i vlevo jsou vidět 2 prsty levé ruky dotýkající se obrazovky.

Vlevo jako 2 světlejší pruhy, vpravo již o něco lépe rozeznatelné.

### 5.2.1.4 Zajímavý poznatek ohledně funkce cvWarpPerspective

Zajímavý fakt je kontrast ruky při nesprávném použití funkce cvWarpPerspective. Chyba v použití je v tomto případě vytvořena špatným mapováním bodů, kdy 2 z bodů jsou totožné.



Obrázek 5.4: zleva: pohled z levé dolní kamery, transformovaný obraz obou kamer

Matice je tedy vypočtena ze tří bodů namísto čtyř.

Za těchto okolností bylo zjištěno, že při dobrém nasvětlení je detekce polohy cizího objektu ve scéně mnohem kontrastnější, než při správném použití funkce.

Pokud vytvoříme tyto podmínky pro obě kamery a tyto kamery a tyto kamery umístíme v osách vzájemně kolmo, průnik jejich obrazů a tudíž přesná poloha ruky na obrazovce bude snadno a přesně měřitelná.

5.5 ukazuje výsledný obraz obou kamer. Anomálie funkce `cvWarpPerspective` způsobí při osvětlení scény z bodu za kamerami o svítivosti cca 100cd výbornou detekci ruky na obrazovce. V tomto případě byla snímána LCD obrazovka barevnými kamerami, takže k žádnému mechanickému filtrování obrazu nedošlo.

Test hypotézy má výborné výsledky nezávisle na barvě obrazu LCD displaye. Pozice světla hraje velmi významnou roli. Bez světelného zdroje ve směru od kamery k ruce uživatele by nebylo vidět nic. Display pod ostrým úhlem světlo odráží ve směru od kamery a jeho obraz je pro kamery neviditelný. Při druhém světelném zdroji vytvářejícím ambientní osvětlení jsou výsledky nevyhovující (obrázek 5.6).

### 5.2.2 Dynamické parametry

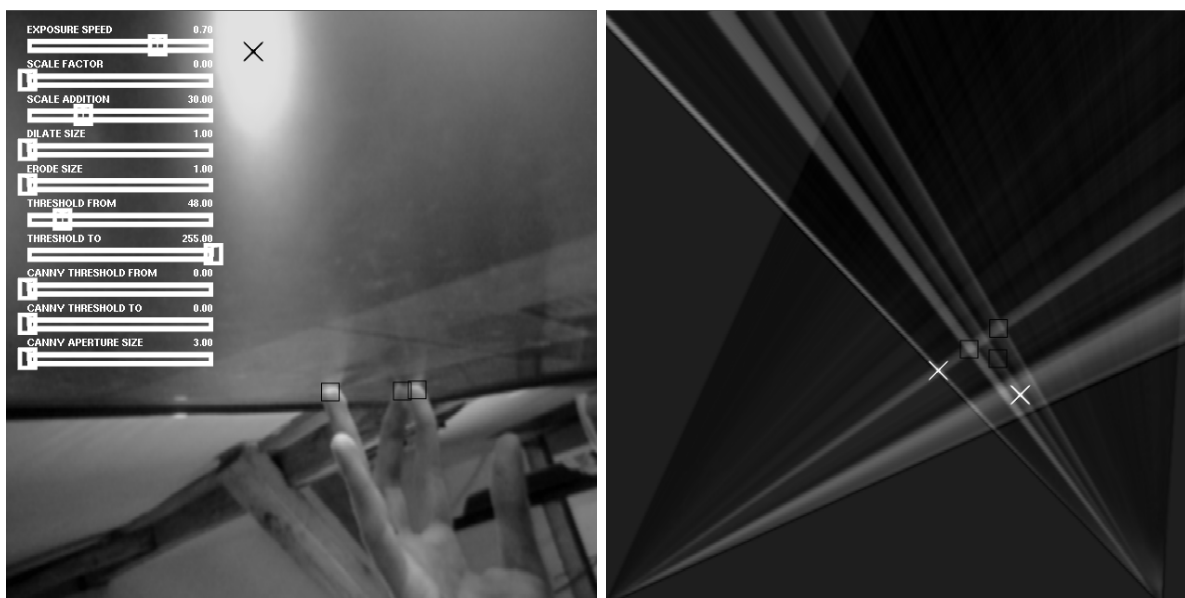
Při spuštění programu je nutné nastavit detekci hran, aby nebyla detekována žádná hrana v klidové poloze, tzn. pokud neprobíhá žádný pohyb před kamerami. Toto lze docílit změnou parametru binarizace obrazu. Šum kamery je tímto potlačen a systém je připraven na detekci pohybu. Problém přichází ve chvíli, kdy je očekáván pohyb uživatele. Potlačením šumu je možné utlumit detekci hran natolik, že nebude detekován ani pohyb uživatele.

#### 5.2.2.1 Optimalizace detekce kontur

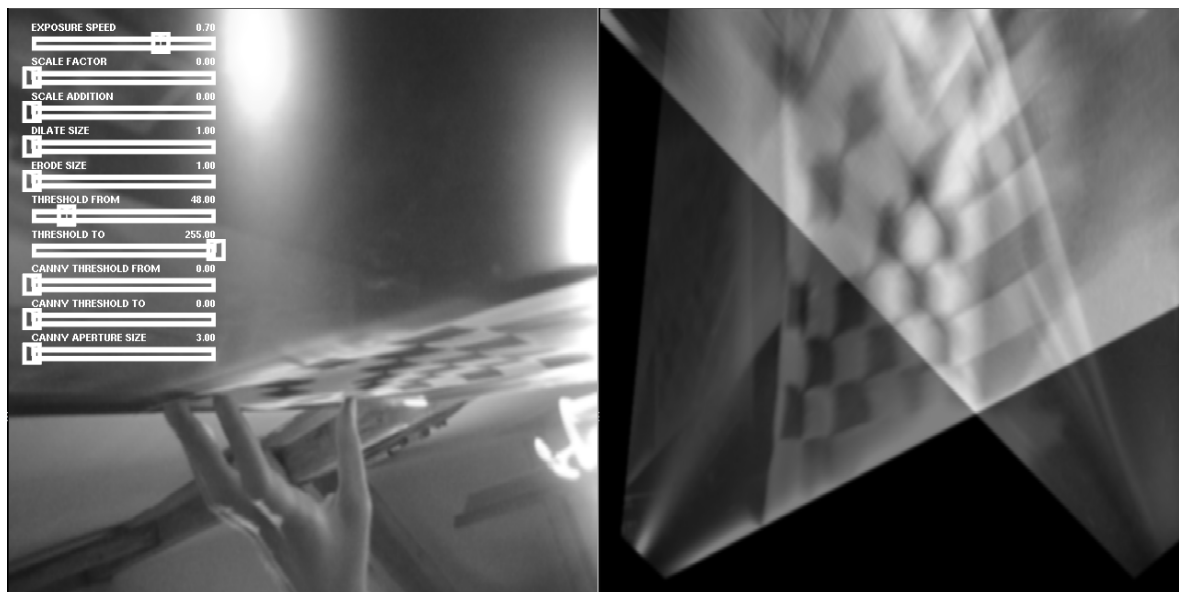
Existuje několik způsobů, jak optimalizovat kontury ruky uživatele za běhu. Všechny však potřebují pohyblivého aktéra před kamerami. Tato procedura může být neviditelná pro uživatele při první detekci pohybu. Potlačení šumu v prvním kroku je nutné nastavit blízko hraniční hodnoty, aby byl tento pohyb vůbec zaznamenán.



Obrázek 5.5: zleva: pohled levé kamery, zkonstruovaná scéna obou kamer při nesprávném použití funkce `cvWarpPerspective`



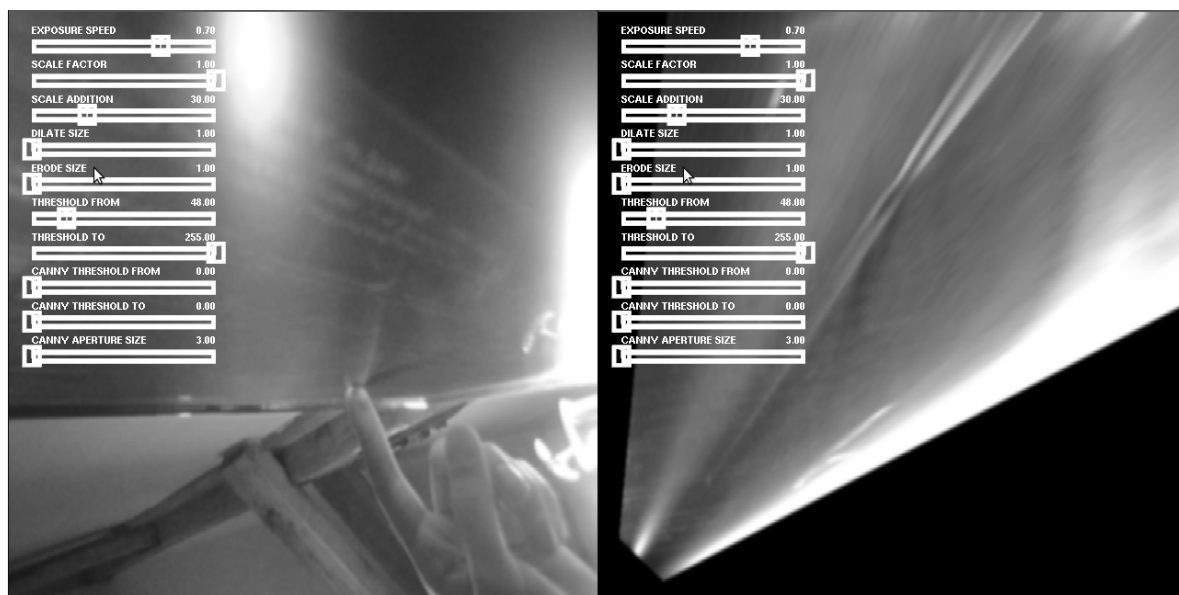
Obrázek 5.6: zleva: test spolehlivosti funkce `cvWarpPerspective` a jejího nestandardního použití v pod jasnějším bodovým osvětlením



Obrázek 5.7: ukázka složené transformace kalibrační šachovnice

### 5.2.2.2 Dynamické změny prostředí

Pokud dojde během snímání pohybu ke změně světelných podmínek, systém se může stát nepoužitelným. Toto lze snadno detekovat faktem, že budou hrany obrazu detekovány v celém snímaném prostoru kamery. Program může spustit inicializační cyklus a automaticky se znovu zprovoznit. Tento scénář je vhodné doplnit o informaci uživateli ve tvaru grafického výstupu ke snížení dezinformace.



Obrázek 5.8: transformace ruky





## 6 Implementace programu

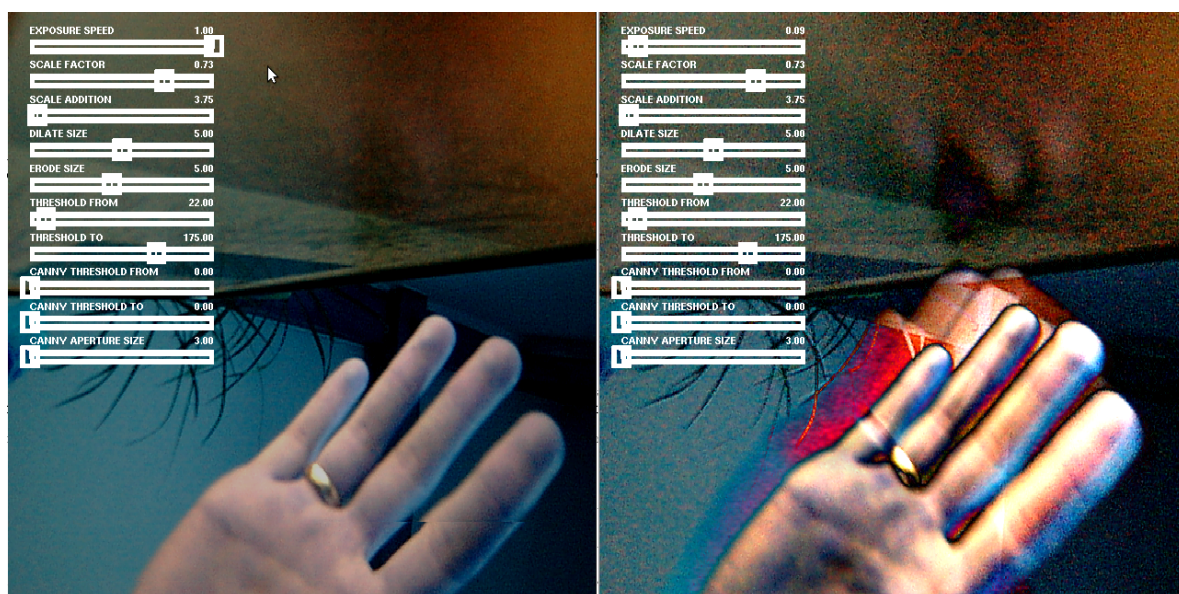
### 6.1 Pomocná knihovna pro segmentaci obrazu

Knihovna OpenCV je určena pro zpracování obrazu za účelem detekce a vyhodnocení aktivních objektů ve statickém obrazu či sekvenci. Obor její funkcionality je pro účely pomocné knihovny tomuto projektu více než dostačující. Pozornost bude zaměřena pouze na její části, které jsou pro tento projekt vhodné. Následuje výčet funkcí pro filtrování, segmentaci a detekci objektů z obrazu.

#### 6.1.1 Pomocné funkce knihovny OpenCV

[8]

**cvRunningAvg()** : Proložení několika posledních snímku do jednoho HDR obrázku. Parametr je doba expozice (6.1).

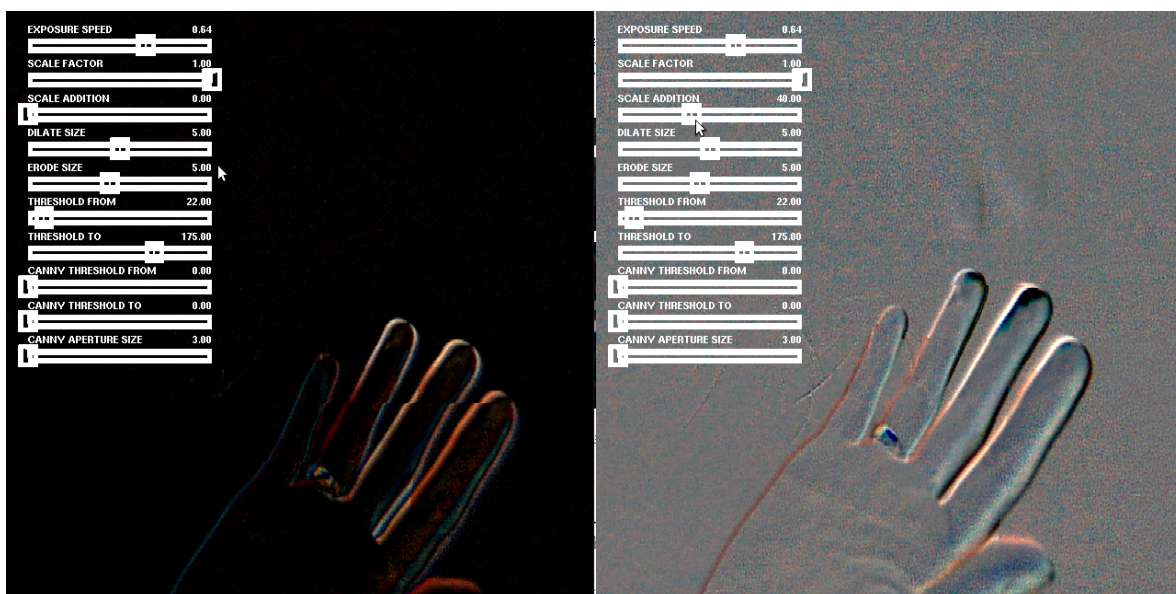


Obrázek 6.1: funkce `cvRunningAvg` knihovny `openCV`: vlevo originál, vpravo po aplikaci funkce

**cvConvertScale()** : Tato funkce transformuje barevnou hloubku obrazu. Například v kombinaci s funkcí `cvRunningAvg()` zpětně převede HDR obrázek do standardní 8-bitové hloubky. Parametry funkce jsou měřítko intervalu výstupního obrazu a přídavek jasu, jak je vidět na obrázku 6.2.

**cvDilate()** : Tato funkce obraz rozostří. Nejedná se však o jednoduchou změnu rozlišení (angl. *downsample*). Rozšířeny jsou pouze kontrastní body a to vždy tak, že původní bod je ve středem obdélníku, který je jeho obrazem. Tyto obrazy jsou skládány aditivně, což má za následek zvýraznění kontrastu, jak je vidět na obrázku 6.3.

**cvErode()** : Funkce `cvErode()` vybere minimum z pixelů v okolí. Jedná se o funkci inverzní k `cvDilate()`. Po aplikaci obou funkcí jsou hrany ruky z původního obrázku zaostřeny a je potlačen šum. Mezi blízkými hranami je vytvořen stín (obrázek 6.4 vlevo), což může být na škodu při přesné detekci hran. Pro zjištění konvexní obálky je to naopak výhodou.



Obrázek 6.2: funkce `cvCvtColorScale` knihovny `openCV`: vlevo parametr intervalu, vpravo ambientní přídavek



Obrázek 6.3: funkce `cvDilate` knihovny `openCV`: vlevo originál, vpravo po aplikaci funkce





Obrázek 6.4: funkce cvErode knihovny openCV: vlevo originál, vpravo po aplikaci funkce

**cvGetPerspectiveTransform()** a **cvWarpPerspective()** : Tato funkce vypočte transformační matici obrazu pomocí pozic 4 bodů. Tato matice může být aplikována na geometrii libovolného obrazu. Obraz kamery libovolně umístěné v prostoru vzhledem k průmětně lze tedy zobrazit tak, aby se souřadnice snímané polohou či pohybem blížily ortogonálnímu prostoru obrazovky. Snímaný bod je možné s tolerovatelnou odchylkou využít bez dalších úprav k detekci polohy vzhledem k průmětně. Na obrázku 6.5 je ukázka aplikace obou funkcí. Kamera je umístěna v pravém dolním rohu obrazovky. Střed čočky kamery je vzdálen 22mm v kolmé ose. Transformace prostoru je nastavena empiricky.

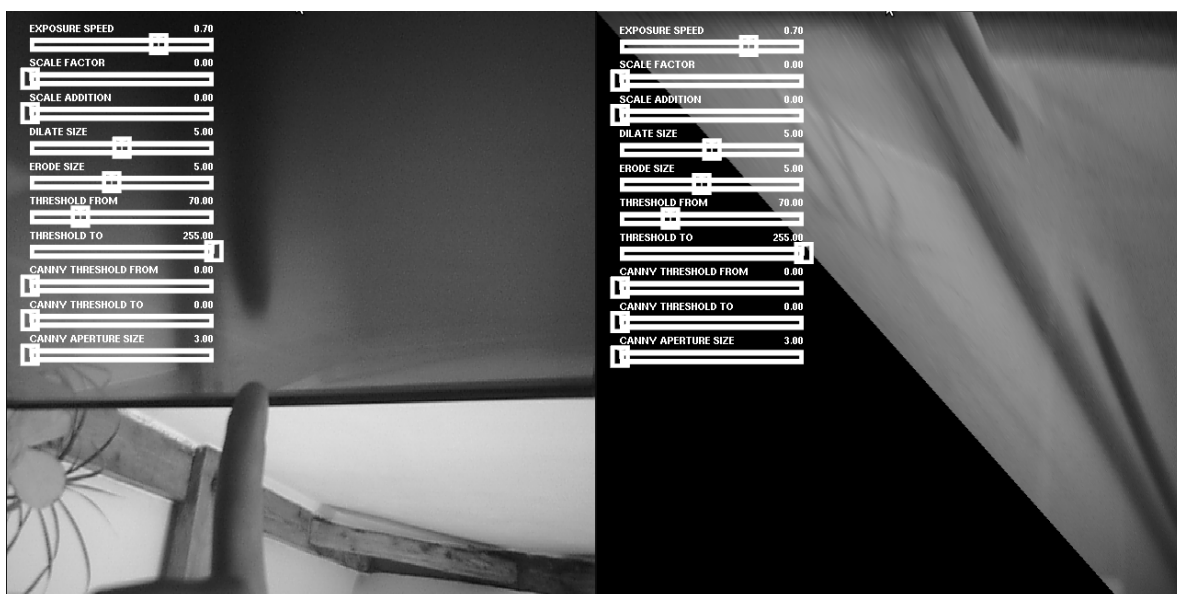
**cvCanny()** : Využívá Canny algoritmus pro vyhledání hran v obrázku. Na obrázku 6.6 je ukázka této funkce. Součástí funkce jsou 2 parametry určující práh zobrazování silných hran a pro propojení těchto hran.

**ostatní funkce** : Další funkce stačí pouze zmínit. Jedná se o funkce běžně používané při zpracování obrazu. Jsou to

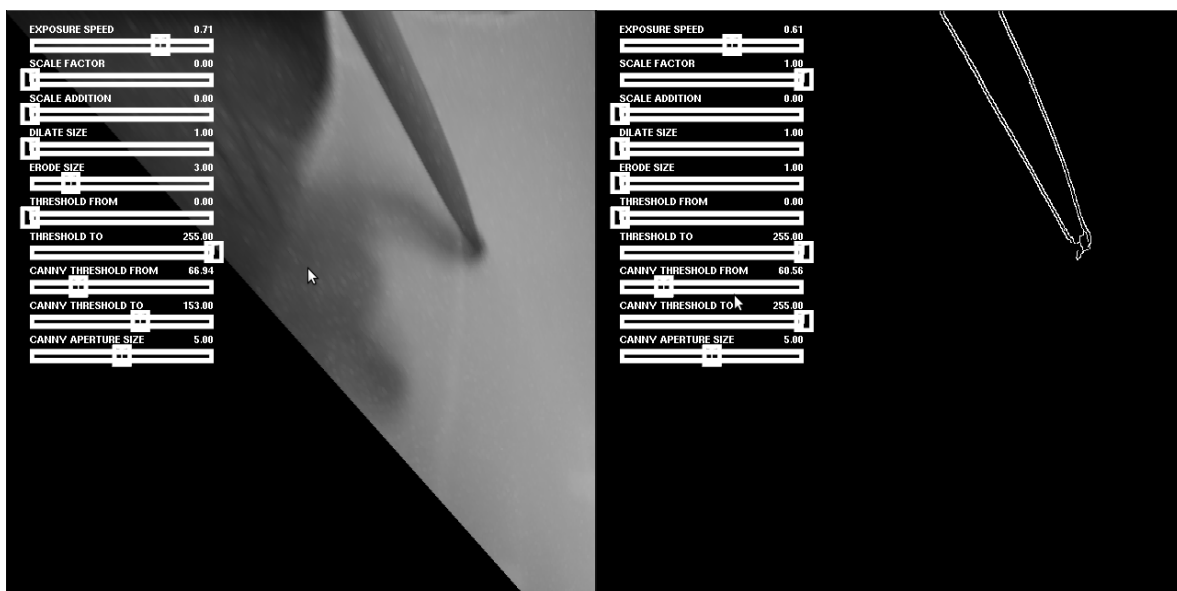
- *cvThreshold()*: tzv. prahování obrazu, neboli binární ořezání
- *cvSmooth()*: rozmazání obrazu
- *cvCvtColor()*: převedení obrazu do jiného barevného prostoru
- *cvAbsDiff()*: rozdílový obrázek

### 6.1.2 Předpokládaná rychlost odezvy

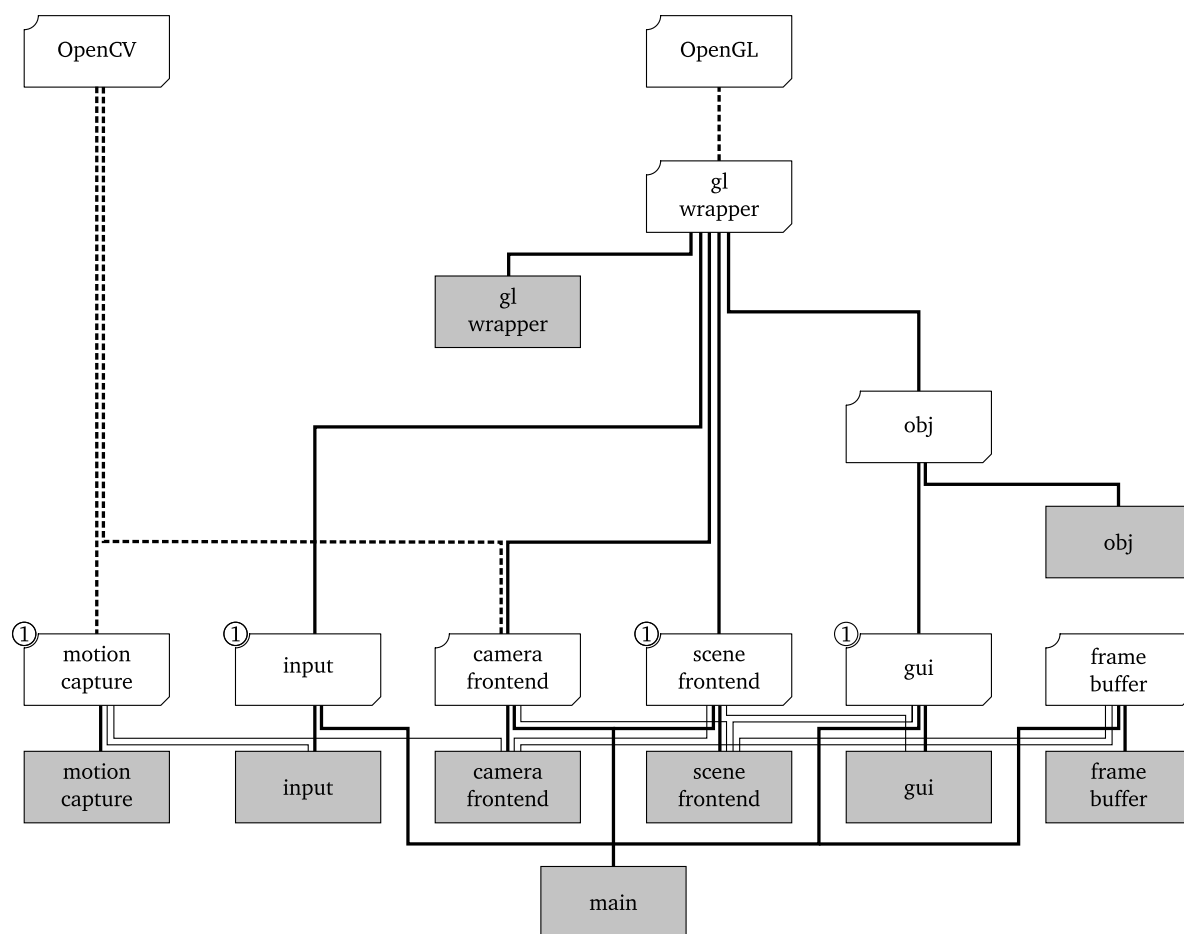
Byl proveden zběžný test na osobním počítači o standartním výkonu s použitím kamery Sony PSEye, která je použita právě u technologie Move. Tato kamera disponuje frekvencí 60Hz. Hardwarová prodleva kamery je tedy 16.6ms. Lidské oko bezpečně pozná prodlevu o délce 100ms ( 10Hz ), ideální prodleva pro iluzi okamžité reakce je 41.6ms ( 24Hz ).



Obrázek 6.5: funkce `cvGetPerspectiveTransform` knihovny `openCV`: vlevo původní snímek kamery, vpravo namapování na obrazovku při umístění kamery v pravém dolním rohu obrazovky



Obrázek 6.6: funkce `cvCanny` knihovny `openCV`: vlevo transformovaný snímek kamery, vpravo kontura pohybujícího se ukazováčku po aplikaci detekce hran této funkce



Obrázek 6.7: objektové schéma programu: Singletons jsou označeny číslovkou 1, hierarchie je shora dolů.

## 6.2 Objektové schéma programu

Schéma 6.7 zobrazuje stavbu programu. Jednotlivé bloky nepředstavují pouze třídy, spíše sdružují funkcionalitu.

- gl wrapper Objekt představující hlavičku glo.h. Obsahuje implementaci pomocných pro inicializaci OpenGL, struktury a definice projektu.
- obj Třída, která načte zdrojový soubor ve formátu Wavefront OBJ a vytvoří z něj vertex buffer object k zobrazení.
- motion capture Singleton, který přijímá kontury a vrací souřadnice uživatelských vstupů.
- input Singleton, který zpracovává vstupy od klávesnice a myši. Použito pro testování.
- camera frontend je třída, jejíž instance představuje wrapper pro každou z kamer připojených k systému.
- scene frontend je hlavičkový soubor obsahující singleton Frontend udržující kontakt a datový prostor mezi vstupem, výstupem a grafickým uživatelským rozhraním. zobrazující OpenGL scénu.
- gui je singleton řídící GUI prvky. Použito pro testování.
- frame buffer jako pomocná funkcionalita pro vytváření OpenGL textur pro všechny objekty, které tyto textury využívají, tedy obraz kamer a scéna.
- main je hlavní vykreslovací smyčka programu obsahující volání instancí tříd a implementující scénáře testů instalace.

## 6.3 Problémy zpracování

Při použití knihovny OpenCV je největší problém v úniku alokované paměti ( leak ). Protože dochází ke zpracování velkého množství dat každý snímek, tyto úniky zaplní operační paměť během několika minut. Během implementace bylo nutné vyhnout se následujícím metodám:

- cvSetCaptureProperty() Funkce pro změnu rozlišení kamery
- cvFindContours() Detekce kontur v obraze. Lze zaměnit za dvojici cvStartFindContours(), cvEndFindContours() s dealokováním paměti, kterou využívají každý snímek.
- cvReleaseCapture() Při deinicializaci snímání kamery se vytvoří leak o velikosti cca 60kB, což se však dá tolerovat vzhledem k tomu, že je tato funkce zavolána pouze jednou pro každou kameru při jeho ukončení.

Mnoho funkcí knihovny OpenCV zpracovávající obraz s výstupem do jiného obrazu se chovají nedeterministicky, pokud je vstupní parametr shodný s výstupním. Tato nekonceptní vlastnost je blízka pouze v několika případech.

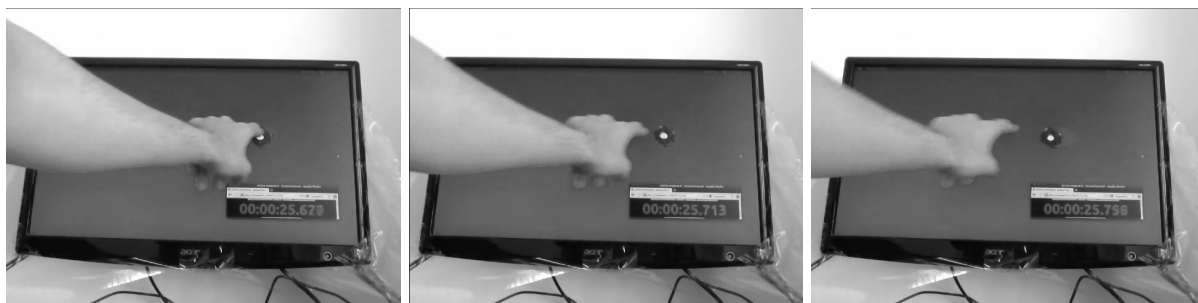
Projekt OpenCV je ve vývoji. Kromě výše uvedených maličkostí je stabilní. Kromě knihovny OpenCV program využívá pouze knihovnu OpenGL pro zobrazení grafiky a Xlib pro zajištění okna. Tyto knihovny jsou stabilní.

## 7 Testování

### 7.1 Metody Testů

Jelikož je systém zaměřen na fyzický vstup uživatele do systému měřeném kamerami pod ostrým úhlem k obrazu, chyba měření je variabilní ve všech následujících testech. Chyba při měření odchylek je cca 1mm. Barevné VGA Kamery o frekvenci 60Hz jsou při testu umístěny 11cm od hrany 24" LCD obrazovky a namířeny do středu obrazovky. Scéna je zobrazena v rozlišení 640x480. Rozteč bodu je tedy  $0.8328125mm \times 0.629167mm$ . Scéna byla nasvětlena nepřímým denním světlem z pravé strany.

Na obrázku 7.1 je zachycen průběh měření odezvy na příkladu posunu při doteku. Obraz je zobrazen 60 snímků za vteřinu, kamera, která obraz snímá má frekvenci 60Hz. Chyba v měření je tedy půl snímku, tedy 8.33ms. Přesnost stopek lze zanedbat, jelikož je jejich přesnost nesrovnatelně vyšší a protože je snímán rozdíl, nikoliv absolutní hodnota. Největší chyba v tomto měření je v rozdílu pozice objektu a jeho obrazu. Do tohoto rozdílu se promítají všechny ostatní chyby projektu.



Obrázek 7.1: průběh měření odezvy detekce pozice

Při testování přesnosti polohy byl k detekci pozice využit hrot tužky namířen do ukazatele myši, jehož poloha je referencí k odchylce. Detekovaná pozice byla zaznamenána.

Při testování přesnosti při doteku více bodů je měřena poloha středu detekované souřadnice obrazu dvou tužek dotýkajících se obrazovky ve vzájemné vzdálenosti 120mm ( obrázek 7.2 ).



Obrázek 7.2: měření odchylky při detekci dvou bodů

Test stability je proveden spojitým pohybem prstu dotýkajícím se obrazovky v celém prostoru detekovaným kamerami. Výpadky detekce jsou zaznamenány programem.

## 7.2 Výsledky Testů

### 7.2.1 Latence

Měření bylo provedeno metodou snímku obrazovky se stopkami, kterou zabírala kamera a následně detekovala jejich konturu. Tabulka obsahuje rozdíly časů v ms.

rychlost odezvy [ms]												
	doba expozice			chyba měření			počet aktivních hran			chyba měření		
#	čas	obraz	delta	čas	obraz	delta	čas	obraz	delta	čas	obraz	delta
1	7546	7588	42	876	919	43	383	469	86	732	818	86
2	2985	3034	49	371	455	84	862	905	43	521	613	92
3	7028	7070	42	841	927	86	225	311	86	712	755	43
4	971	1013	42	927	970	43	97	184	87	711	797	86
5	983	1068	85	581	667	86	417	503	86	498	571	73
6	278	315	37	268	354	86	845	931	86	32	118	86
7	5679	5765	86	490	576	86	521	613	92			
8	190	237	47	578	621	43	438	524	86			
9	311	398	87									
10	62	105	43									

Obrázek 7.3: tabulka reakcí detekce polohy na běžném osobním počítači pomocí knihovny OpenCV

Z tabulky 7.3 je patrné, že chyba měření definuje interval hodnot až na jejich dvojnásobek. Toto je dané stylem měření. I přes tento nedostatek lze stanovit průměrnou odezvu v intervalu 42 až 120 ms. Způsob měření je zachycen na obrázku 7.1.

Tabulka dále zobrazuje prodlevy při detekování rostoucího počtu hran ( řádově do stovek ) a rostoucí doby expozice rozdílového snímku. Při zohlednění rozdílu v měření jedné hodnoty, který je průměrně 41ms, je možné prohlásit libovolné nastavení 10 po sobě jdoucích funkcí OpenCV určených pro detekci kontur pohyblivého bodu jako funkce s konstantní časovou složitostí nezávisle na použitých parametrech. Funkce pro vykreslení HDR obrázku složeného z několika po sobě jdoucích snímcích je tedy nezávislá na době expozice, jelikož nečeká na jejich vykreslení, ale počítá snímky uložené v minulosti.

### 7.2.2 Měření odchylky pozice

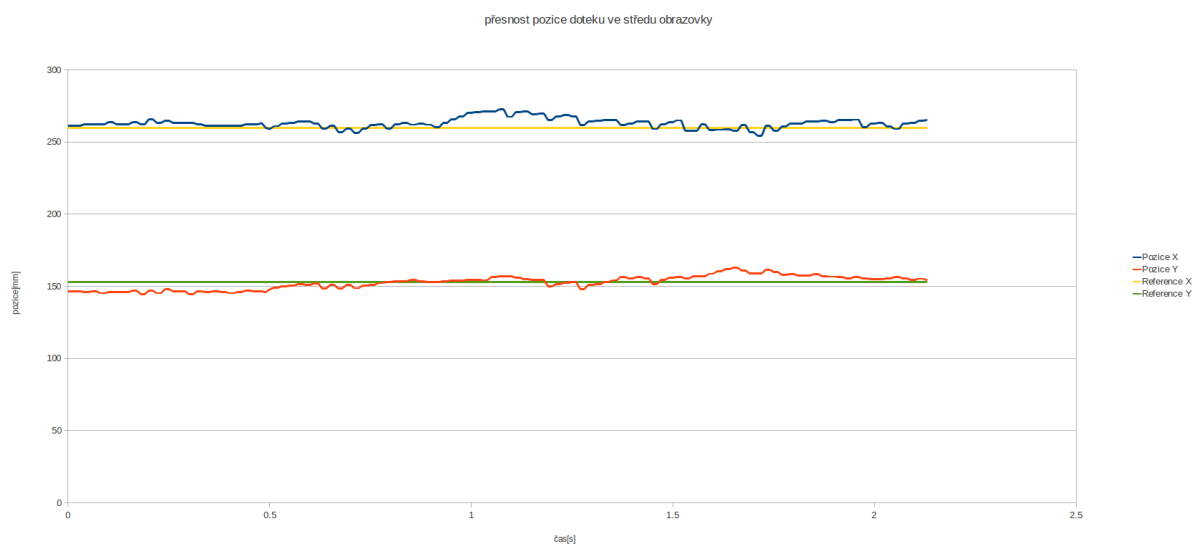
Při doteku ve středu obrazovky ( graf 7.4 ) byla naměřená průměrná odchylka  $3.243mm$  ve vodorovné ose X, resp.  $0.163mm$  v ose Y. Maximální hodnota rozptylu byla naměřena  $12.817mm$  pro X,  $10.255mm$  pro Y. Naměřeno bylo 241 hodnot po dobu 2.2s.

Měření odchylek při doteku v různých částech obrazovky zaznamenalo tyto fakta: Měření odchylky pozice ve svislé ose ve středu obrazovky ( 7.8 a 7.5 ) vykazuje obdobnou přesnost. Měření na levé a pravé straně má viditelně vyšší odchylku.

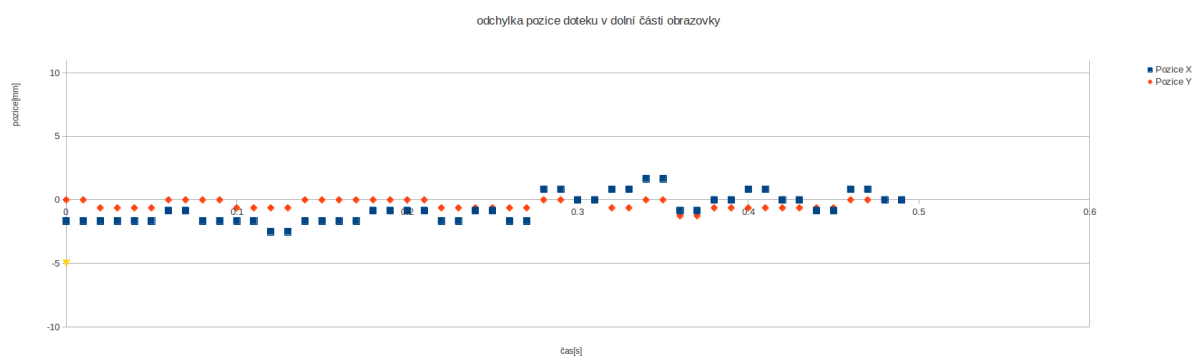
Průměrná odchylka vlevo ( 7.6 ) byla  $2.698mm$  (X),  $8.17mm$  (Y). Maximální odchylka byla  $4.996mm$  (X),  $9.438mm$  (Y). Na pravé straně obrazovky ( 7.7 ) byla průměrná odchylka  $0.200mm$  (X),  $4.253mm$  (Y), maxima  $2.498mm$  (X),  $5.033mm$  (Y). Naměřeno bylo 50 hodnot v každé oblasti.

### 7.2.3 Měření odchylky pozice bez doteku

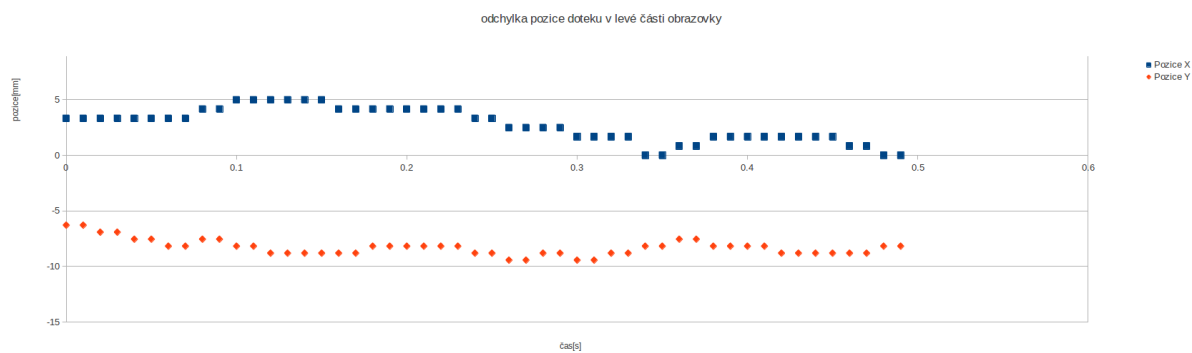
Maximální odchylka v ose X je  $10.31mm$ , v ose Y  $12.1mm$ . Naměřeno bylo 241 hodnot. Měření bylo provedeno ve horizontálním i vertikálním středu obrazovky. V rozích obrazovky je tato hodnota o řád vyšší, což je vzhledem k povaze detekce očekávané chování. Naměřené hodnoty jsou vyneseny v grafu 7.9.



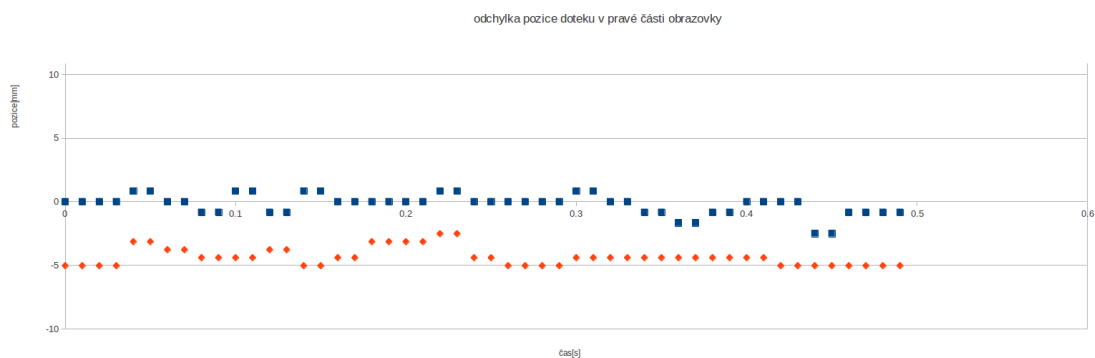
Obrázek 7.4: přesnost pozice doteku ve středu obrazovky



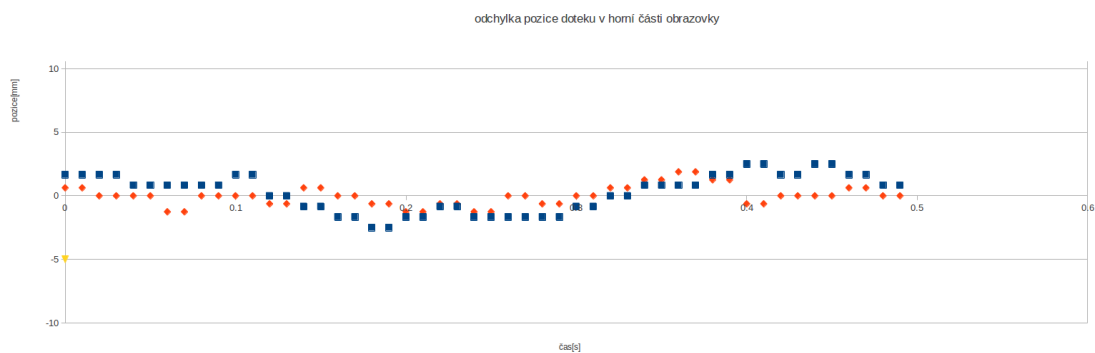
Obrázek 7.5: odchylka pozice doteku v dolní části obrazovky



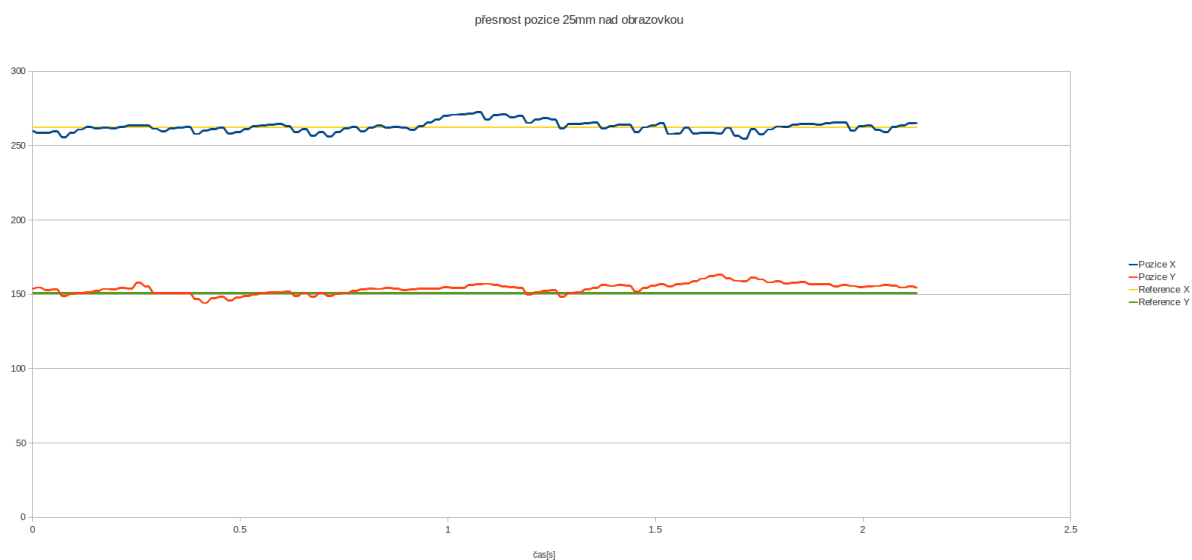
Obrázek 7.6: odchylka pozice doteku v levé části obrazovky



Obrázek 7.7: odchylka pozice doteku v pravé části obrazovky



Obrázek 7.8: odchylka pozice doteku v horní části obrazovky

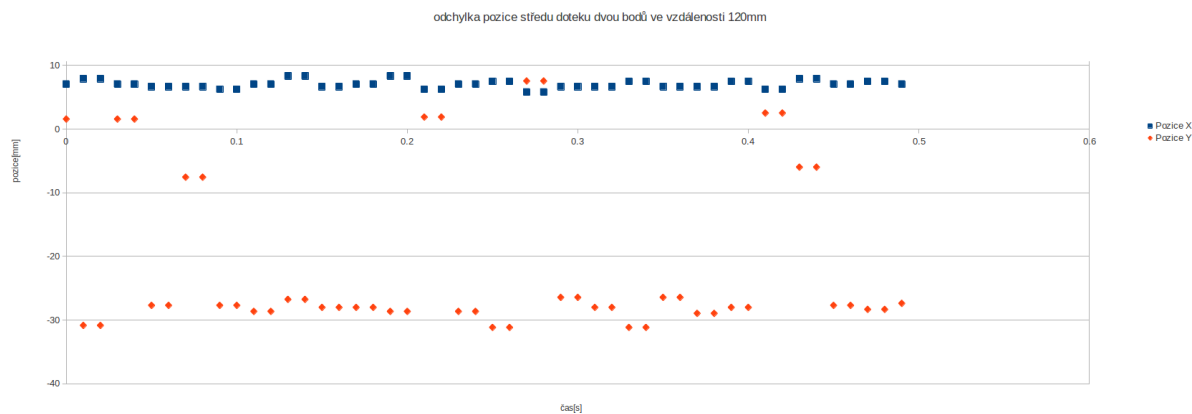


Obrázek 7.9: odchylka pozice bez doteku 25mm nad obrazovkou



### 7.2.4 Měření odchylky pozice dvou bodů

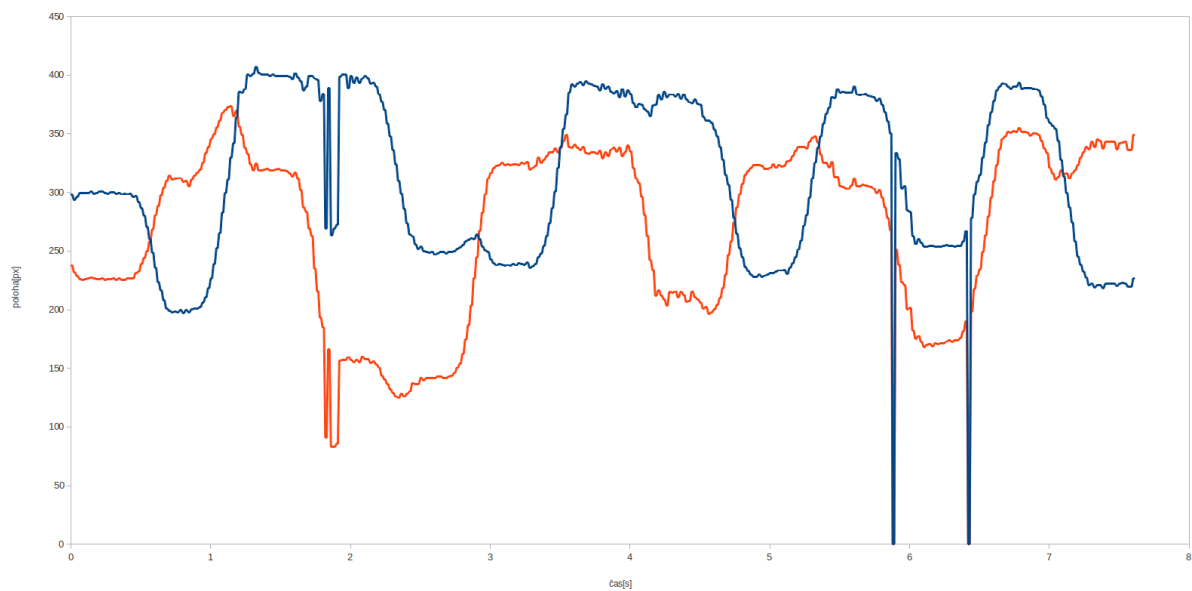
Měření více bodů s výsledkem v 7.10 vykazovalo průměrnou odchylku  $6.904\text{mm}$  v ose X a  $20.983\text{mm}$  v ose Y. Maximální odchylka v X je  $8.328\text{mm}$ , v Y pak  $31.144\text{mm}$ .



Obrázek 7.10: odchylka doteku dvou bodů ve vzájemné vzdálenosti 120mm

### 7.2.5 Měření stability

V průběhu testu 7.11 trvajícího  $7610\text{ms}$  byl zaměřeny dva chybné úseky náhle po sobě o délce 20ms a 50ms před 2. vteřinou testování a dva výpadky o délce 20ms. Program byl po tuto dobu funkční na 98.55%.



Obrázek 7.11: stabilita doteku



## 8 Diskuze

### 8.1 Výsledky testů

#### 8.1.1 Latence

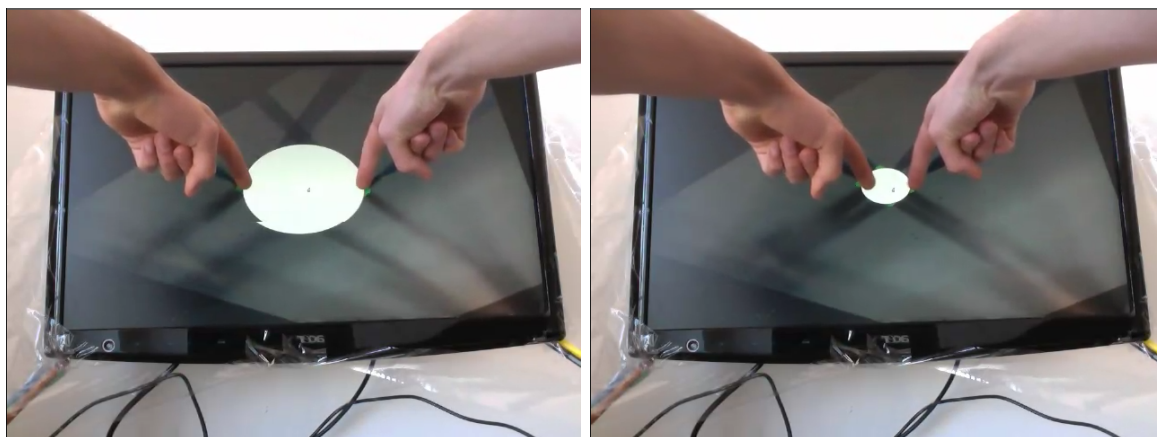
Při využití většiny funkcionality toolkitu lze výsledky rozdílů odezvy zanedbat. Dotek ruky je člověku nerozeznatelný, framerate 24Hz - 11Hz je k tomu dostačující.

Co však zanedbat nelze je pohyb s dotekem ( drag ). Člověk rozezná prodlevu v pohybu snadněji, nežli rozdíl odstínů, zvláště pak, pokud má k dispozici jednoduchou pomůcku, jak ji zjistit. Tou je vzdálenost mezi pohybujícím se objektem a jeho obrazem na obrazovce.

#### 8.1.2 Přesnost doteku

Výsledky přesnosti doteku jsou vynikající. Program není opatřen žádným pomocným algoritmem, který by dotek zpřesňoval, či průměroval. Test měří přesnou hodnotu z detekce pohybu. Tato hodnota není posunuta v závislosti polohy kamer, což by bylo v jistém ohledu logické.

V grafu 7.10 je vidět, že při detekci dvou bodů je většina bodů posunutých ve směru od kamer v ose kolmé na jejich spojnici Odpovědí na tuto otázku je na obrázku 8.1



Obrázek 8.1: pozadí detekce dvou bodů

V pravé části obrázku je detekce přesná. Stíny pod bílou plochou označující naměřený okruh jsou projekce prstů obou kamer po namapování obrazů kamer na obrazovku. Jejich 4 průniky detekující dotyk jsou ve vodorovné ose správně. Ve svislé je však horní průsečík výše. Rozlišení v takové vzdálenosti od kamery je velmi nízké.

Absence korekce zakřivení čočky pro detekci jednoho bodu není důležitá, protože je snímán průsečík celých vektorů v ose kamery. Pokud je bodů více, zvláště pak, pokud se fyzicky na obraze kamer protínají ve větší vzdálenosti od obrazovky, toto zkreslení pocítí. V takovémto případě je vhodné se zamyslet, je-li na úkor rychlosti vhodné aplikovat algoritmus na korelaci zakřivení čočky kamery, nebo alespoň upravit tuto situaci nějakou rychlejší náhražkou algoritmu.

Dalším faktem, který výrazně ovlivňuje přesnost je styl držení ruky při doteku. Pokud z pěsti pod ukazovákem odstává palec, kamera jej zaznamená a v případě protnutí s libovolným objektem na obraze druhé kamery vyjádří jako dotyk. V tomto případě je detekován chybný bod.

Tento artefakt lze minimalizovat co nejbližším přiblížením os kamer k rovině obrazovky. Nevhodný důsledek této korekce je ztráta rozlišení obrazu v ose kamery. V limitním případě,

kdy je osa kamery na rovině obrazovky, je mapa obrazovky přímka a detekce je ořezána na jeden rozměr. Úplně to eliminuje artefakt držení ruky a přesněji to detekuje vzdálenost ruky od obrazovky. Realita se však stane složitější přidáním dalších problémů. Například hrany okolí, které se dostanou nad detekovanou obrazovku, zhoršená možnost detekce objektů před obrazovkou apod.

Výsledky detekce bodů mimo osu kamery ( tedy v levé a pravé části obrazovky ) mají menší přesnost, než body ve svislé ose uprostřed. To je způsobeno nižším rozlišením kamery, která je k detekovanému průsečíku dále. Chyba je způsobena také větší chybou kamery, která je průsečíku map blíže, jelikož je obraz objektu zabírá větší plochu a sebemenší odchylka způsobí nepřesnost.

### 8.1.3 Odchylka pozice nad obrazovkou bez doteku

Detekce pozice doteku je zpracována pomocí průsečíků úseček totožných s obrazem objektů na mapě obrazovky. Pro detekci pozice objektu, který se nedotýká jsou tyto úsečky protaženy do polopřímek. Průsečíkem těchto polopřímek na obrazovce je definovaná pozice objektu nad ní.

Vzhledem k ostrému úhlu kamer nad obrazovkou je se použitá metoda nabízí. Je jednoduchá a rychlá. Artefakt držení ruky se však v tomto případě projevuje ve větším rozsahu, zvláště pak, je-li v záběru celá ruka uživatele. Prsty, které jsou navíc jsou zaznamenány do mapy, jejich hrany vyznačeny a úsečky hran protaženy do polopřímek. Díky tomu je zde mnohem více potenciálních bodů k určení pozice.

Pozice má tendenci inklinovat ve směru ke kamerám. Kromě toho jsou kontury sevřených prstů snadno náchylné k dynamickým změnám obrazu, stínům a natočení ke světlu. Díky tomu se v mezích bodech obrazovky odchylka dynamicky mění ve velkém rozsahu.

Čím blíže je ruka obrazovce, tím lépe se dá určit její pozice. Řešení této problematiky je buďto omezení detekce do určité vzdálenosti, nebo oddálení kamer od obrazovky. Limitní vzdálenost ruky se snadno pozná tím, že se rozptýl bodů překročí hranici násobku velikosti množiny. Oddálení kamer od obrazovky přesune mezní body obrazů kamer mimo obrazovku. K tomu je potřeba vyšší rozlišení kamer a více prostoru instalace.

Detekce předmětů nad obrazovkou je upravena. Pole bodů je seřazeno, je vypočten medián, který je určen jako středová poloha. Tento bod je rozšířen poloměrem, který v grafickém rozhraní zobrazuje odchylku uživateli. Navíc je průměrováno několik posledních hodnot měření. Bez těchto optimalizací by díky nepřesnostem detekce popsaným výše nebylo možné detekci bodů nad obrazovkou využít.

### 8.1.4 Stabilita

Při optimálních podmínkách, ve kterých bylo měření byla stabilita detekce vynikající. Výpadky naměřené v grafu 7.11 jsou člověku nerozeznatelné.

Problém nastává v případě, pokud se podmínky prostředí změny v neprospěch instalace. Mezi takovéto okolnosti patří například silné difúzní světlo namířené do obrazovky. Odlesk tohoto světla zastíní všechny kontury zvláště pak, jedná-li se o obrazovku s matným displayem.

Lesklý display se stabilitou osvědčil více jelikož kontury odrazů jsou ostré a algoritmus tento odraz využije ve svůj prospěch. Empiricky bylo prokázáno, že vlastnost lesklých povrchů odrážet všechny hrany okolí včetně těch nežádoucích není nebezpečná pro vlastní detekci objektů před obrazovkou. Transformací prostoru do mapy jsou hrany ve větší vzdálenosti od scény rozptýleny.

Stabilita při nevhodných podmínkách je takřka skoková od plně funkčního systému po plně nefunkční. Při použití je třeba dbát na správné nasvětlení. Pro stabilitu je klíčová.

Přes všechny nevýhody byl systém na jedné konfiguraci testován celý den od rána pod přirozeným osvětlením a za celou dobu nezměnil výsledky detekce.

## 8.2 Naplnění vytyčených cílů

### 8.2.1 Bezchybná detekce polohy ruky

Dá se říci, že se tento cíl podařilo dosáhnout. Při doteku je detekce přesná. Při malých rozměrech obrazovky je v jednotkách milimetrů, což je dostačující například i na práci s operačním systémem. Pro větší rozměry obrazovky minimální detekovaná velikost bodu i nepřesnost lineárně roste. V těchto situacích se však nepředpokládá, že bude nutné ovládání na tak přesné úrovni.

### 8.2.2 Rychlost Odezvy

I když je rychlost odezvy aplikace nízká ( 42ms až 84ms ), rozdíl mezi vstupem a výstupem je velmi patrný. Podle článku [10] je detekovatelná odezva pohybem člověkem 10ms, což je rovno frekvenci 100Hz. Při profesionálním použití toolkitu je vhodné využít kamery o frekvenci vyšší. Nejlevnější způsob je koupit ovladače pro Nintendo Wii, každý v hodnotě 40USD a využít 100Hz IR kamery, který každý ovladač obsahuje.

K nerozpoznatelné prodlevě to však stále nestačí. 70% prodlevy je způsobeno zpracováním obrazu. 32ms v nejlepším případě k jeho zpracování při 300Hz kamerách, které z 10ms využijí 3.33ms na snímek je třeba softwarové zpracování zrychlit 5x.

Protože systém zpracovával pouze 2 kamery a vykreslování probíhalo téměř bez účasti CPU, pro získání potřebného času by měl být obraz rozdělen do několika částí, které budou zpracovány zvlášť a následně složeny. Ani to však není dostatečné. Propustnost sběrnice v tomto případě omezí výhody paralelismu.

Knihovna OpenCV se díky těmto faktům na rychlé zpracování obrazu na současných počítačích nehodí. Řešení touto cestou by bylo velmi náročné. K těmto potřebám vznikla knihovna GpuCV[5]. Vektorový procesor v grafické kartě a paralelní řešení na úrovni shader jazyku má ke zpracování obrazu lepší předpoklady. Na základě [1] dosahuje GpuCV nersrovnatelně rychlejších výsledků. Pro srovnání funkce Erode() a Dilate() testované v tomto dokumentu ze 7.45ms při využití dosahují více než 8 násobného zrychlení při CpuCV-GLSL ( 860 $\mu$ s ), při použití GpuCV-CUDA dokonce téměř 20-násobku nárůstu výkonu ( 390 $\mu$ s na sestavě Core2Duo s grafickou kartou NVIDIA GTX280 ). Proces zvýraznění kontur Sobelovým operátorem je přes GpuCV-CUDA 9x rychlejší. Pro větší obrázky je tento nárůst ještě propastnější. Pro malé obrázky ( například 128x128 ) je OpenCV rychlejší z důvodu inicializace CUDA jazyka ( 100 $\mu$ s ), dalších 180 $\mu$ s pro OpenGL a GLSL ).

Omezení, které knihovnu GpuCV v tomto ohledu staví do výhodnější role na kandidáta pro projekt byly zjištěny v pokročilé fázi projektu.

### 8.2.3 Dobrá škálovatelnost technologie

Tento systém byl navržen pro detekci polohy před zpětnou projekcí s LCD či DLP projektořem. Prvotní myšlenka pro vznik tohoto projektu byl objev, který jsem učinil s Jakubem Hyblerem z Institutu Intermédií katedry Počítačové grafiky a interakce ČVUT FEL v Praze.

**Lemma 8.2.1.** *Promítaný obraz LCD či DLP projektoru je neviditelný černobílé kameře, která je předložena černým filtrem z vyvolaného diapozitivu a zároveň červeným filtrem stejného odstínu, který je použit u anaglyfické stereoskopie.*

Černobílé kamery nejsou osazeny filtrem IR světla, který je nutný pro barevné kamery, aby odstranil stíny v kanálech obrazu. Černý a červený filtr odfiltruje složku obrazu viditelného světla, kterou vyzařují LCD a DLP projektoři. Světlo projektoru je studené a nezasahuje do IR spektra.

Kamera v těchto situacích snímá obraz, jako by na něm obraz projektoru nebyl. Tento poznatek byl impulzem ke snímání obrazu před projektorem z bezprostřední blízkosti a detekce objektů na průmětně. Hlavní předpoklad v možnosti odfiltrovat vlastní obraz a tím zrušit veškeré artefakty v nechtěné detekci tento problém zjednodušuje.

Projekt multimediální instalace škálovatelnost posunul ještě o kousek dále při uložení kamer v ostrém úhlu k rovině obrazovky. Vlastní obraz je díky útlumu světla odraženého od obrazovky potlačen a netvoří artefakty stejně jako by nebyl viditelný tak jako obraz projektoru před kamerou osazenou černým a červeným filtrem.

Kamery navíc nemusí být monochromatické. Testování tohoto projektu bylo provedeno pomocí barevných webových kamer. Vzhledem k použití pouze bodových zdrojů je možné scénu sestavit v téměř libovolném měřítku ve velkém rozsahu kvality. Jediným omezením v kvalitě detekce je cena obrazovky a kamer. Tyto komponenty jsou však běžně dostupné a často bývají běžnou výbavou domácností, kanceláří i institucí.

#### 8.2.4 Spolehlivá detekce bez doteku

Tento cíl byl částečně splněn. Velikou roli hraje rozložení a kvalita kamer. Vzájemná poloha vzhledem k obrazovce a nasvětlení scény. Systém tyto možnosti podporuje. Závisí tedy na okolnostech, s jakými bude scéna sestavena. I za těchto okolností je však spolehlivá detekce vzdálena realitě.

#### 8.2.5 Automatická přizpůsobivost technologie dynamickým změnám prostředí

Původní myšlenka brala v úvahu fakt, že bude implementována metoda rozdílového snímku v kapitole 5.1.1, která snímá pouze pohyblivé předměty. V praxi se ukázalo, že problém s detekcí statických hran se vyřeší namapováním obrazu kamer do obrazovky. Po namapování a ořezání obrazu je záběr omezen pouze na výřez obrazovky, který žádné hrany neobsahuje.

Od rozdílového snímku se navíc ustoupilo kvůli času potřebnému k jeho zrekonstruování, i když je na testovaném systému úzké hrdlo prodleva kamery mezi sejmutím dvou snímků a ne zpracování procesoru. Toto úzké hrdlo je na testovaném systému partikulární záležitost, protože jsou testované kamery relativně pomalé ( 60Hz, 16ms na snímek ).

Poslední nevýhoda metody rozdílového snímku je latence o velikosti  $\frac{t-t_0}{2}$ , kde  $t$  je aktuální čas, tedy kdy je snímán poslední snímek pro proces rozdílu.  $t_0$  je čas prvního snímku v minulosti, který je určen jako šablona pro rozdíl. Na obrázku vznikne pruh pohybu mezi záběrem v minulosti a aktuálním snímkem kamery. Nejmenší prodleva, která tímto způsobem vznikne je  $\frac{1}{2}$  času, pokud detekovaný bod označíme uprostřed intervalu.

Ke zrušení prodlevy způsobené sejmutím několika snímků a jejich průměrováním by bylo třeba zaznamenávat směr pohybu a na jeho základě snímat pohyblivý bod na levé či pravé straně hrany.

Dynamické změny se projevují pouze při zpracování obrazu rozdílovým snímkem. Při akutní změně osvětlení rozdílový snímek nezobrazí pouze pohyblivé hrany, ale také ambientní rozdíl světelné intenzity mezi začátkem a koncem měření. Systém je tedy v tuto chvíli slepý, jelikož prahovací funkce pracuje jiným intervalu.

Systém nyní pracuje v režimu detekce hran z každého snímku zvlášť. Při aktuální konfiguraci je pracovní rozsah při vhodných inicializačních parametrech ( nepřímé nasvětlení ) dostatečně široký, aby mohl systém pohodlně pracovat po celý den.

V zájmu dalšího zvýšení stability je možné detekovat počet a rozsah detekovaných hran na mapě a v případě rovnoměrného výskytu v celé ploše zvýšit hodnotu ambientního násobku kanálu jasu. Kontrast scény se potlačí a dále nebudou detekovány přebytké falešné hrany. Tento způsob generuje potenciální problémy, například ztrátu detekce hran žádaných. Při nevhodné implementaci by se stabilita naopak zhoršila.

## 9 Závěr

### 9.1 Zhodnocení projektu

Objem problému tohoto projektu je velmi rozsáhlý a s ním i okruh jeho řešení.

Největší výhoda řešení vybraného v tomto dokumentu je cenová dostupnost a široký rozsah použitelných komponent, rozsah možností instalace a přesnost detekovaného bodu.

Mezi nedostatky je rychlost zpracování pro některé scénáře využití řešitelná vhodnějším výběrem knihovny pro zpracování obrazu. Dále pak nutnost poprvé kalibrovat obraz.

### 9.2 Budoucí rozšíření tohoto projektu

Diskutabilní je řešení mapování obrazu kamer do prostoru obrazovky. Ve finální podobě projektu je nutná prezence polohovacího zařízení a klávesnice. Uživatel při prvním spuštění označí klikem myši nad kurzorem hranice obrazovky po nebo proti směru hodinových ručiček nad kurzorem a to pro každou kameru. Z pozic je vytvořena konvexní obálka, ze které je vypočtena transformační matice. Tato matice je uložena při opětovném spuštění programu. Uživatel Tedy tuto proceduru musí minimálně jednou provést. Další okruh, který je třeba rozšířit se týká optimalizace detekce před nebo bez dotyku, filtrování pravých hran od těch matoucích. Za uvážení stojí zapojení algoritmu pro detekci předem definované kontury pro zvýšení přesnosti. Potenciální rozšíření tohoto projektu může být zapojení systému několika disjunktních způsobů detekce naráz s výběrem vhodného metody pro aktuální dynamické podmínky. Tento průzkum rozšíří pracovní rozsah a potlačí nutnost udržení světelných podmínek. V neposlední řadě je možnost vytvořit uživatelské rozhraní na míru spojení detekce bezdotekového pohybu a doteku, či zapojení do operačního systému ve formě polohového ovladače.





## A Slovník pojmů

- HCI Human-Computer Interogation neboli styk člověka s počítačem.
- Multitouch Režim dotykového ovládání, kde je využito několik doteků naráz.
- Motion Capture Detekce pohybu nejčastěji osob, nebo jejich částí za účelem zaznamenání jejich pohybu a následné rekonstrukce v digitální podobě.
- Markerless Motion Capture Detekce pohybu bez nutnosti použít cizí předmět pro vlastní detekci.
- level-set je numerická metod zabývající se segmentací objektů v obraze na základě detekce hran[11].
- HDR je Hight Dynamic Range Image, neboli obrázek s vyšším rozsahem barevné složky zasahující do neviditelného spektra. Tento obrázek je možné dále zpracovávat bez ořezání barev.
- Threshold obrázku do češtiny překládaná jako "prahování" označuje funkci pro změnu hodnot pixelů na 1 nebo 0 podle funkce prahování jejich originální hodnoty.



## B Uživatelská příručka

Program pracuje v několika režimech. Po spuštění je v režimu prezentačním. Je zobrazena modrá obrazovka s rotujícím objektem uprostřed. Pokud byl v minulosti program spuštěn a byla provedena kalibrace, je znovunačtena kalibrační matice ze souboru *backup* a program začne detekovat polohu.

Režim zobrazení výhledu kamery pomocí kláves *q* a/nebo *w* zobrazí výhled jedné a nebo několika kamer. V tomto režimu je možné prohlédnout si reálný výhled kamer(y) a kalibrovat, pokud je zobrazena pouze jedna kamera.

### B.1 Kalibrace

Kalibrace je realizována levým tlačítkem myši. Nejprve je nutné kameru fyzicky přesunout tak, aby zabírala všechny čtyři strany obrazovky. V záběru nemusí být celá obrazovka. Některé ( například čtyři ) rohy mohou být za hranicí obrazu.

1. Levým tlačítkem myši 1x klepněte do rohu obrazovky v záběru kamery. V případě, že není žádný roh viditelný, klepněte do místa, kde se hrana obrazovky v záběru kamery stýká s hranou snímku.
2. Krok 1 opakujte po a nebo proti směru hodinových ručiček do té doby, až bude každý roh či vizuální konec hrany obrazovky označen. Označení není nijak vizualizováno. Poslední bod nesmí být totožný s prvním, žádný bod nesmí být označen dvakrát. Pokud se procedura
3. Pokud máte jakýkoliv problém stiskněte tlačítko *f* na klávesnici, čímž vyčistíte zásobník klíčových bodů. A opakujte proceduru od kroku 1.
4. Po označení všech bodů na viditelné kameře stiskněte klávesu *f*. Pokud byly body označeny správně, zobrazí se transformovaný obraz ( maska ). Pokud byly body označeny špatně, vymaže se zásobník bodů a je nutné kalibraci provést znovu. Opakujte od kroku 1.
5. Pokud se kalibrace zdařila, na obrazovce je vidět transformovaný obraz.  
Pro opakování kalibrace stiskněte přepínač transformovaného pohledu ( klávesa *a* pro první kameru, klávesa *s* pro druhou kameru ), kter přepíná mezi maskou a obrazem. Můžete pokračovat od bodu 1.
6. Dotechem ruky zjistíte, jestli je nutné masku otočit ( klávesa *r*, či transponovat ( klávesa *t* ). V případě potřeby proveďte otočení vícekrát.
7. Obraz by měl být ve stavu, kdy se shodují pozice objektu dotýkajícího se obrazovky a jeho obrazu v celé ploše zobrazovaného výseku masky.
8. Stiskněte klávesy *a* a *s*. Přepněte tím výhled na další kameru. Proveďte kalibraci od kroku 1 pro další kameru.

Po provedení kalibrace je možné si prohlédnout detekovaný bod ve formě bílé elipsy. Při záběru první kamery ( *a* ) jsou zobrazeny navíc pomocné body znázorňující průniky hran ( světle zelená ) a průniky přímek totožných s hranami ( tmavě zelená ).

### B.2 Korelace

Pokud je zobrazen obraz nebo maska pouze jedné z kamer, je zobrazeno několik posuvníků v levém horním rohu obrazovky. Jedná se o parametry funkcí, kterými je obraz zpracován. Pomocí kláves *1*, *2*, *3*, *4* a *8* je možné prohlédnout si obraz v různých fázích zpracování.

Proces jde v uživatelském rozhraní shora dolů vyjma funkcí *Erode*, *Dilate* a *Threshold*. *Erode* a *Dilate* jsou využívány ke korekci detekovaných hran nakonec a *Threshold* je použit pouze v případě, že jsou jeho hodnoty nastaveny na jiný interval než 0 až 255. V tomto případě je vyjmut z procesu a je nahrazen prahováním přímo v algoritmu *Cannyho* detektoru hran.

### B.3 Shrnutí

Obecné ovládání:

- klávesy *q* a *w* : přepínač zobrazení obrazu každé kamery; V případě, že je zapnuto více kamer, obrazy se zobrazí přes sebe s 50% průhledností.
- klávesy *a* a *s* : přepínač zobrazení transformovaného ( masky ) a netransformovaného obrazu každé kamery; Fungují pouze v případě, že byl obraz dříve alespoň jednou zkalibrován.
- klávesa *g* : přepínač zobrazení korelačního GUI
- klávesa *Esc* : ukončení programu

Kalibrace:

- *Levé tlačítko myši* v režimu zobrazení jedné kamery : přidání pozice kurzoru do zásobníku konvexní obálky obrazovky pro danou kameru
- Klávesa *f* : při dostatečném počtu bodů v zásobníku konvexní obálky vypočtení a aplikace transformační matice z obrazu aktuálně zobrazené kamery do masky obrazovky
- Klávesa *r* : rotace vypočtené transformační matice
- Klávesa *t* : transpozice vypočtené transformační matice

Korelace:

- Klávesa *1* nenumernická : zdrojový obraz kamery
- Klávesa *2* nenumernická : transformovaný obraz
- Klávesa *3* nenumernická : intervalový filtr
- Klávesa *4* nenumernická : chromatický filtr
- Klávesa *8* nenumernická : prahovací filtr

## C Aplikační příručka

Pro správný chod programu je třeba nainstalovat tyto balíky:

*linux-libc-dev, freeglut3-dev, libc-dev-bin, libc6-dev, libcanberra-pulse, libgl1-mesa-dev, libglew1.5-dev, libgstreamer-plugins-base0.10-dev, libjpeg62-dev, libpng12-dev, libsm-dev, libstdc++6-4.5-dev, libswscale-dev, libtiff4-dev, libvorbis-dev, libx11-dev, libau-dev, libxcb-render-util0, libxcb-render0-dev, libxcb-shm0-dev, libxcb1-dev, libxcomposite-dev, libxdamage-dev, libxdmcp-dev, libxext-dev, libxfixes-dev, libxfg-dev, libxi-dev, libxine-dev, libxrandr-dev, libxrender-dev, libhighgui-dev, libv4l-dev*

Dále jsou to:

*CMake, Intel TBB, Qt, IPP, pkg-config, gtk+2.x, ffmpeg, gstreamer, unicap, OpenCV-2.2.0,*

Pro zvýšení výkonu na kartách NVIDIA:

*CUDA Toolkit, Developer Drivers for Linux (270.40)*

Program byl testován na OS Ubuntu Linux 9.04, Ubuntu 10.04 TLS, Ubuntu 10.10 a Ubuntu 11.04



## D Přiložené DVD

DVD obsahuje program zkompilovaný pro Linux OS x64 včetně zdrojového kódu v C++ a tohoto dokumentu ve formě PDF včetně zdrojového textu pro LaTeX.

- bin
  - miit2011
- doc
  - miit2011.pdf
- resSrc
  - algoritmy.tex
  - analyza.tex
  - diplbakmacro.tex
  - diskuze.tex
  - implementace.tex
  - prirucky.tex
  - slovník.tex
  - specifikace.tex
  - testovani.tex
  - uvod.tex
  - zaver.tex
  - zpracovani obrazu.tex
  - bak.tex
  - bak.bib
  - dediacritics
  - diacritics
  - make
  - mr.cls
  - Figures
  - Tables
  - Resources
- src
  - cvo.cpp
  - cvo.h
  - fb.cpp
  - fb.h
  - glo.cpp
  - glo.h
  - gui.cpp

- gui.h
- input.cpp
- input.h
- main.cpp
- Makefile
- motionCapture.cpp
- motionCapture.h
- obj.cpp
- obj.h
- scene.cpp
- scene.h
- obj
- blend
- tex



## E Literatura

- [1] Y. Alluse, P. Horain, A. Agarwal, and C. Saipriyadarshan. Gpucv: A gpu-accelerated framework for image processing and computer vision. [www-public.it-sudparis.eu/horain/Publications/.../ISCV08Allusse.pdf](http://www-public.it-sudparis.eu/horain/Publications/.../ISCV08Allusse.pdf), 2006. Institut TELECOM; TELECOM & Management SudParis.
- [2] K. Byron, L. Pham, and J. Situka. Markerless motion tracking, 2009. Graduate Program of Human Computer Interaction, Virtual Reality Application Center, Iowa State University.
- [3] J. Canny. A computation approach to edge detection. *PAMI-8(6):679-698*, 1986. Pattern Analysis and Machine Intelligence, IEEE Transactions on.
- [4] Honeywell. Honeywell light curtains product catalog. [http://sensing.honeywell.com/index.cfm/ci\\_id/95950/la\\_id/1/series\\_id/136793.htm](http://sensing.honeywell.com/index.cfm/ci_id/95950/la_id/1/series_id/136793.htm), 2011.
- [5] P. Horain and Y. Allusse. Gpu-accelerated computer vision. <https://picoforge.int-evry.fr/cgi-bin/twiki/view/Gpucv/Web/WebHome>, 2010.
- [6] J. Hughes. Velká obrazová všeobecná encyklopedie. [s.l.] : Svojtka & co., 1999.
- [7] Intel. Algoritmus erode pro zostření hran. [http://opencv.willowgarage.com/documentation/cpp/imgproc\\_image\\_filtering.html#erode](http://opencv.willowgarage.com/documentation/cpp/imgproc_image_filtering.html#erode), 2010. [Online; 1. květen 2011].
- [8] Intel. Seznam funkcí knihovny opencv. [http://opencv.willowgarage.com/documentation/cpp/image\\_filtering.html](http://opencv.willowgarage.com/documentation/cpp/image_filtering.html), 2010. [Online; 1. duben 2011].
- [9] Kabbo. Microsoft kinect vs. playstation move vs. nintendo wii. <http://thetechjournal.com/electronics/gaming-electronics/microsoft-kinect-vs-playstation-move-vs-nintendo-wii.xhtml>, 2010.
- [10] O. Kawakami, Y. Kaneoke, K. Maruyama, R. Kakigi, T. Okada, N. Sadato, and Y. Yonekura. Visual detection of motion speed in humans: spatiotemporal analysis by fmri and meg, 2002.
- [11] A. Kratochvílová. Parciální diferenciální rovnice ve zpracování obrazu. page 51, 2009. [geraldine.fjfi.cvut.cz/oberhuber/.../07-kratochvilova-vyzkumak.pdf](http://geraldine.fjfi.cvut.cz/oberhuber/.../07-kratochvilova-vyzkumak.pdf).
- [12] L. Lang. Rekonstrukce objektu a pozice pozorovatele z 2d pohledů. pages 19,20, 2004. [herakles.zcu.cz/skala/MSc/Diploma`Data/DP`2004`Lang`Ladislav.pdf](http://herakles.zcu.cz/skala/MSc/Diploma`Data/DP`2004`Lang`Ladislav.pdf).
- [13] W. Nistico, U. Schweigelshohn, M. Hebbel, and I. Dahm. Real-time structure preserving image noise reduction for computer vision on embedded platforms, 2004.
- [14] M. Pollefeys. Projective 2d geometry (cont'), course 3. Multiple View Geometry, Comp 290-089.
- [15] B. Rosenhahn, T. Brox, U. G. Kersing, A. W. Smith, J. K. Gurney, and R. Klette. A system for marker-less motion capture. 1, pages 45 – 51, 2006. Kunstliche Intelligenz (KI).
- [16] M. Scott. Lag as a determinant of human performance in interactive systems. <http://www.yorku.ca/mack/CHI93b.html>, 1993. [Online; 17. duben 2011].
- [17] Wikipedie. Canny edge detector. [http://en.wikipedia.org/wiki/Canny\\_edge\\_detector](http://en.wikipedia.org/wiki/Canny_edge_detector), 2011.

- [18] Wikipedie. Sobel operator. [http://en.wikipedia.org/wiki/Sobel\\_operator](http://en.wikipedia.org/wiki/Sobel_operator), 2011.
- [19] Y. Xing and R. C. Liu. Displacement sensor using laser triangulation in pavement inspection.