Měření výšky osob pomocí kamerového systému

Measurement of person stature using camera system

Petr Žáček

Bakalářská práce
2012

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2011/2012

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Petr ŽÁČEK**
Osobní číslo:	**A09279**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Bezpečnostní technologie, systémy a management**
Téma práce:	**Měření výšky osob pomocí kamerového systému**

Zásady pro vypracování:

1. **Seznamte se a prostudujte doporučenou literaturu.**
2. **Vypracujte literární rešerši na dané téma.**
3. **Navrhněte a sestavte experimentální zařízení pro měření výšky osob pomocí kamerového systému.**
4. **Vytvořte patřičné algoritmy ve vhodném programovacím prostředí (Matlab, VEE pro, Visual studio, C++).**
5. **Vyhodnoťte výsledky experimentů, odhadněte meze použitelnosti zařízení a navrhněte možnosti pokračování v oblasti bezpečnostního průmyslu.**

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **GONZALES, Rafael C; WOODS, Richard E; EDDINS, Steven L. Digital image processing using MATLAB. Upper Saddle River : Pearson/Prentice Hall, 2004. 609 s. ISBN 0-13-008519-7.**
2. **JAIN, Anil K. Fundamentals of digital image processing. Upper Saddle River : Prentice Hall, 1989. 569 s. ISBN 0-13-336165-9.**
3. **CAPPELLINI, V. Time-varying image processing and moving object recognition, 4 : proceedings of the 5th international workshop, Florence, Italy, September 5-6, 1996. Amsterdam : Elsevier, 1997. 332 s. ISBN 0-444-82307-7.**
4. **SEDLÁČEK, Miloš; ŠMÍD, Radislav. MATLAB v měření. Vyd. 2. přeprac. Praha : Nakladatelství ČVUT, 2007. 210 s. ISBN 978-80-01-03781-2.**
5. **ZAPLATÍLEK, Karel; DOŇAR, Bohuslav. MATLAB : tvorba uživatelských aplikací. 1. vyd. Praha : BEN - technická literatura, 2004. 215 s. ISBN 80-7300-133-0.**

Vedoucí bakalářské práce: **Ing. Milan Navrátil, Ph.D.**
Ústav elektroniky a měření

Datum zadání bakalářské práce: **24. února 2012**

Termín odevzdání bakalářské práce: **25. května 2012**

Ve Zlíně dne 24. února 2012

L.S.

prof. Ing. Vladimír Vašek, CSc.
děkan

doc. Mgr. Milan Adámek, Ph.D.
ředitel ústavu

ABSTRAKT

Tato bakalářská práce se zabývá možnostmi měření velikosti osob za využití kamerového systému a návrhem experimentálního zařízení pro měření spolu s aplikací v patřičném programovacím prostředí. V teoretické části jsou popsány možnosti detekce osob v obraze a principy měření velikosti osob. V praktické části jsou odvozeny hlavní vztahy pro výpočet velikosti osob. Dále je na základě odvozených vztahů implementována aplikace v programovacím jazyce MATLAB, která využívá k detekci a měření velikosti osob systému dvou kamer. Následně byl systém dvou kamer sestaven a otestován navrženou aplikací. Všechny výsledky experimentu byly shrnuty a zhodnoceny. Na závěr jsou popsány možnosti pokračování a zlepšení této práce.

Klíčová slova: MATLAB, kamerový systém, měření výšky osob, rektifikace, kamera, detekce pohybujících se objektů.

ABSTRACT

This bachelor work deals with methods of measurement of person stature using camera systém and construction of experimental device fo masurement with application designed in an appropriate programming language. In the theoretical part the options of detection person in image and principes of measurement of person stature are described Firstly, the practical part derives main formulas for measurement of person stature. The application using derived formulas and two cameras for detection and measuring of moving person were implemented in a computing language MATLAB. Camera system of two cameras was constructed and than tested with created application. All results from experiment are summarized and at the end of this work options of continuation or improvement of this work are described.

Keywords: MATLAB, camera system, measurement of person stature, rectification, camera, detection of moving objects

Touto cestou bych chtěl moc poděkovat zejména vedoucímu mé bakalářské práce Ing. Milanu Navrátilovi, Ph.D. za cenné rady a konzultace ohledně bakalářské práce. Dále bych chtěl poděkovat své přítelkyni a své rodině za toleranci a psychickou podporu při vypracovávání.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně ……

podpis diplomanta

OBSAH

ÚVOD

Žijeme ve 21. století, v digitálním světě. Technika je všude kolem nás a denně na ni narážíme. Troufám si tvrdit, že minimálně 80 procent lidí v České republice vlastní nebo pracuje s výpočetní technikou, která nám každý den ulehčuje práci, a stává se všední záležitost. Výpočetní techniky se využívá i v bezpečnostní technologii, proto nás při procházce městem již nepřekvapí přítomnost kamer skoro na každém kroku.

Jak už jsem se zmínil, použití kamerového systému nám může ulehčit mnoho práce. Vyhodnocení obrazu z kamer ovládáme v dnešní době televizi, řídíme a vyhodnocujeme dopravu na silničních komunikacích na, hlídáme přestupky na silnicích, detekujeme požáry a mnoho jiných funkcí. Pro všechny tyto činnosti je nutné kvalitní zpracování a vyhodnocení obrazové informace z kamer. Pro člověka jsou to jednoduché úkoly, ale naučit počítač rozeznat v obraze například člověka, je velmi složitá záležitost. Člověk dokáže automaticky odfiltrovat jevy na pozadí, nevadí mu změna osvětlení, ani jiné rušivé elementy, které mohou být při rozpoznávání obrazu člověka pro počítač nepřekonatelné. Rozhodovací logika počítače je velmi jednoduchá, protože si musí vystačit se stavy Ano či Ne (true nebo false). Například: Nachází se v obraze pohyb, či není? Celkem jednoduchá otázka, řekli bychom, ale pro realizaci na počítači poměrně složitá. Někdo by mohl namítnout, že se to dá zjistit jednoduše tak, že pokud se změní obraz, je v něm pohyb, ale to prakticky ve sto procentech případů neplatí.

Kamera není schopná snímat scénu, i když se jedná o statickou scénu, pořád stejně. Do obrazu se dostává šum z elektronických součástek a okolí a ani samotný snímací čip vždycky nepracuje stejně. Není možné při každém pořizování snímku docílit stejné doby osvícení snímané části čipu.

Toto bakalářské téma jsem si zvolil proto, protože mám rád programování a počítačové zpracování problémů všeho druhu.

Samotný problém počítačového vidění má dle mého názoru v dnešní době budoucnost. S neustálým rozvojem počítačové techniky rostou i možnosti využití rozpoznávání obrazu. I když nejsem zastáncem monitorovací techniky, protože je ze strany lidského faktoru snadno zneužitelná, idea světa, kde bezpečnost hlídají autonomní kamerové systémy vyhodnocující a řešící všechna nebezpečí a rizika, je dobrým nápadem, bohužel zatím utopickým.

Podobným problémem se zabývá tato bakalářská práce. Věnuje se konkrétně možnostem, jak využít kamerový systém k měření výšky osob, popřípadě objektů. Rozebírá možné metody detekce pohybu osob v obraze a možnosti následného měření výšky osob na základě této detekce. Zabývá se návrhem aplikace pro experimentální měření za využití dvou kamer. Probírá zhodnocení výsledků a návrh možných vylepšení a úprav do budoucna.

I. TEORETICKÁ ČÁST

1 MOŽNOSTI PRO MĚŘENÍ VÝŠKY OSOB KAMEROVÝM SYSTÉMEM

Existuje mnoho způsobů, jak lze měřit výšku osob (objektů) pomocí kamerového systému. Některé využívají jednu kameru, jiné dvě kamery, další například kameru spolu s osvětlujícím zařízením a referenčními plochami. V teoretické části této bakalářské práce se věnuji principům metod, které lze v praxi využít, a můžeme se s nimi setkat.

Je třeba říci, že důležitým předpokladem k měření osob nebo objektů pomoci kamerového systému, je nutná detekce osob nebo objektů ve snímané scéně, tedy v obraze z kamery. To může být největším problémem v samotném měření, kdy algoritmus pro měření je už oproti detekci triviálním úkolem. Takže se dá říci, že problém měření výšky osob se dá rozdělit na dvě základní části:

- problém detekce osob ve snímané scéně
- problém výpočtu výšky detekovaných osob ve snímané scéně

V dalších částech rozebírám výše zmíněné problémy. Popisuji principy, jak lze detekovat pohybující se osobu (objekt) v obraze, a jak ji lze následně změřit.

1.1 Detekce pohybujících se objektů (osob) v obraze

Jak už jsem uvedl výše, detekce pohybujících se objektů či osob je základním stavebním kamenem v měření výšky osob pomocí kamerového systému a je to i zároveň ten nesložitější krok v této problematice. Detekce objektu spočívá v porovnávání dvou následujících snímků z kamer, zda k pohybu (změně) ve snímané scéně došlo, a následně ve vypreparování a vyhodnocení těchto změn. V tom nejjednodušším případě se porovnává aktuální snímek s pozadím snímané scény. Potom přichází detekce pozadí, jak a kdy toto pozadí detekovat. Jsou principy, kdy se využívá pouze vyfocení pozadí - ve scéně nenachází žádný pohybující objekt (osoba) nebo zde pohyb je a pozadí se generuje při běhu na základě snímků z kamer a je proměnné v čase. Dále budou tyto možnosti podrobněji vysvětleny.

1.1.1 Thresholding (prahování)

Pro detekci pohybu a změn v obraze je důležité vysvětlit, co znamená pojem thresholding, česky prahování. Thresholding je metoda převodu snímku v RGB nebo ve

stupních šedi na tzv. binární snímek, kdy takový binární obrázek je složen pouze z hodnot 0 a 1, a hodnota 0 znamená, že dané pixely na odpovídajícím místě jsou nezměněné a hodnota 1 znamená změnu v obraze. Při této metodě dochází k absolutnímu rozdílu dvou snímků. Jednotlivé pixely se odečtou a rozdíl je vyjádřen absolutní hodnotou. Zmíněnou operaci nazýváme rozdílování. Následně je tato hodnota porovnávána s tzv. prahem, který se obvykle používá v rozmezí 10-60, pokud bereme hodnoty ve stupních šedi, tj. rozmezí od 0 do 255. Pokud je hodnota menší než práh, je výsledná hodnota 0, naopak pokud je hodnota rozdílu větší než práh, je výsledná hodnota 1. Rovnice prahování má následující tvar

$$f(x) = \begin{cases} 0; & |i_1(x,y) - i_2(x,y)| < práh \\ 1; & |i_1(x,y) - i_2(x,y)| \geq práh \end{cases} \quad (1)$$

Hodnota prahu je závislá hlavně na úrovni šumu v obraze, protože i pozadí okem nezměněné se celkem hodně mění díky šumu. V praxi to znamená, že hodnota nezměněného pixelu může hodně kolísat. Zejména nepříjemný šum se dá odfiltrovat v závislosti na hodnotě prahu. Obecně můžeme říci, že čím více je obraz zašuměný, tím větší je volena hodnota prahu.

1.1.2 Porovnání aktuálního snímku se známým a předem nasnímaným pozadím

Nejjednodušší metodou, jak zachytit a oddělit pohybující se objekt, je porovnání aktuálního snímku z kamery s již známým a předem nasnímaným pozadím. Pozadí je nutné pořídit v době, kdy se ve snímané scéně nic nepohybuje a jsou zde pouze objekty pozadí. Pozadí je nutné po nějaké době obnovovat. K obnově musí dojít z následujících důvodů:

- změna osvětlení pozadí – zhasnuté světla, nebo naopak nový světelný zdroj
- se změnou osvětlení i změna denního času – pohyb slunce a stínů
- změna statických předmětů v pozadí – odnesení stolů, židlí, obrazů, apod.
- změna pozice kamery

V praxi se obnova pozadí provádí většinou v pravidelných cyklech. Například každých 10 minut se nasnímá nové pozadí. Zde může nastat problém, že při ukládání pozadí, nemusí být pozadí bez pohybu.

1.1.2.1 Výhody a nevýhody

Mezi výhody by se dalo zařadit to, že tato metoda je nenáročná na prostředky. Z toho vyplývá, že je lehce implementovatelná. Pokud nastane změna v pozadí, jednoduše se pořídí snímek aktuálního pozadí a to je všechno.

Na druhou stranu to má i své nevýhody, protože pozadí se musí pořídit jen v době, kdy ve scéně není pohyb. Další poměrně značnou nevýhodou je, že je těžké automaticky rozlišit, kdy se pozadí změnilo. V praxi se obnova pozadí provádí v pravidelných cyklech.

1.1.3 Metoda porovnávání snímku s automaticky generovaným pozadím

Druhá metoda, kterou zde uvádím, je už trochu složitější na realizaci. Spočívá v autonomním generování pozadí za chodu. Další kroky jsou srovnatelné s předchozí metodou, protože i zde dochází k vzájemnému rozdílování a prahování snímků s pozadím a mezi dvěma snímky, pro zjištění změn.

Generování pozadí by se dalo zkráceně popsat následujícím způsobem: (Podrobněji spolu s blokovým schématem chodu algoritmu, je metoda popsána v kapitole 3.3.1 v praktické části bakalářské práce, protože jí je využíváno v měření.)

První se vyhodnotí, zda je ve snímku pohyb. Místa bez pohybu jsou zaregistrované a na daném místě se v pomocné matici velikosti snímku navýší hodnota o 1, pokud hodnota v pomocné matici dosáhne předem navolené hodnoty, tak je daný pixel na dané pozici zahrnut jako pozadí. Z toho vyplývá, že pozadí se nemusí vygenerovat celé, ale jen jeho část, ale zároveň lze i jeho část použít k detekci pohybu v obraze.

1.1.3.1 Výhody a nevýhody

Jako každá metoda, tak i tato má bohužel své výhody a nevýhody. Podrobně se jim budu taky věnovat až v praktické práci, kde budou popsány jako poznatky z měření, ale ve zkratce by se dalo říct, že výhody jsou následující

- Není potřebná režie pozadí
- Malé změny se automaticky zaregistrují – přidání předmětu do pozadí
- Možnost vyhodnocování pohybu i bez vygenerovaného pozadí
- Pozadí, i když částečné, lze generovat při pohybu osob

Jako hlavní nevýhody lze uvést následující

- Složitost implementace
- Náhlé změny budou vyhodnoceny jako pohyb
- Do pozadí mohou být započítány chvíli stojící osoby
- Pozadí nemusí být nikdy vygenerované nebo na druhou stranu hodně proměnlivé

1.2 Principy měření výšky osob ve snímané scéně

I zde, je více možností, jak danou problematiku řešit. V této části popisuji hlavní metody, jak lze výpočtu výšky osoby pomocí kamerového systému dosáhnout.

1.2.1 Porovnání osoby s již známým objektem

První metodou, jak je možné měřit výšku osob pomoci kamerového systému, je metoda, která snímá scénu s již předem známým objektem s předem známou velikostí, kdy je předmět postaven do přesné vzdálenosti od kamery. Tato metoda je principiálně nejjednodušší, protože využívá jen jedné kamery.

Předpoklady výpočtu výšky osoby jsou následující

- Znalost výšky předmětu v cm a pixelech a jeho přesné umístění ve snímané scéně
- Zachycení osoby ve snímané scéně v pravý okamžik – ve vzdálenosti umístěného předmětu
- Výška osoby ve snímané scéně v pixelech

Pomoci těchto znalostí a faktů jsme schopni dopočítat z následující jednoduché rovnice podobnosti velikost osoby v cm.

$$\frac{výška\ předmětu_{pixel}}{výška\ předmětu_{cm}} = \frac{výška\ osoby_{pixel}}{výška\ osoby_{cm}} \rightarrow výška\ osoby_{cm} = \frac{výška\ osoby_{pixel} \cdot výška\ předmětu_{cm}}{výška\ předmětu_{pixel}} \quad (2)$$

Zjištění velikosti porovnávaného předmětu je jednoduché, stačí ho změřit. Jeho umístění je podobně jednoduchým úkolem. Trochu složitější je zachycení osoby v dané

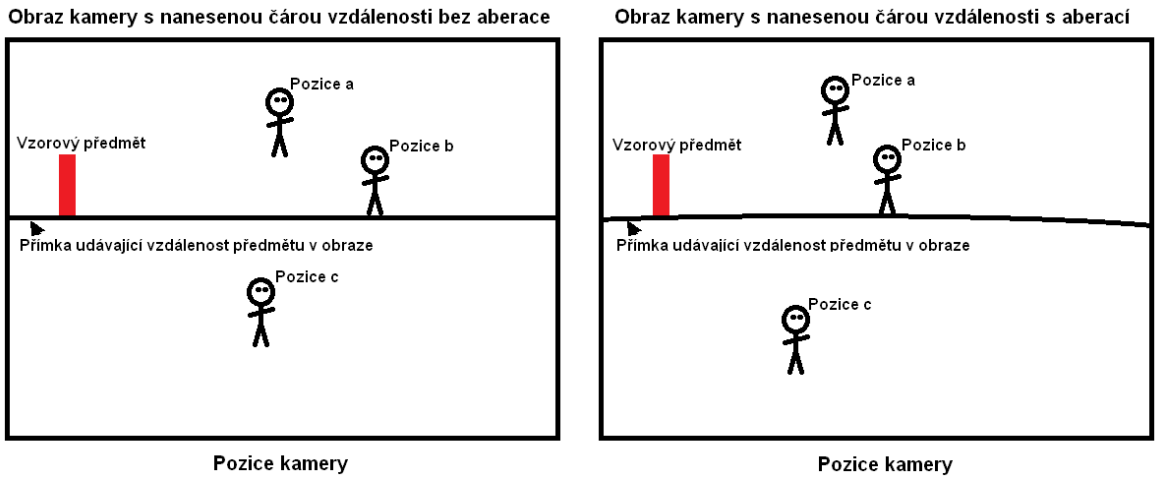
vzdálenosti, kdy musíme pořídit snímek, pro zjištění výšky osoby v pixelech, a následně vypočítat její výšku. Toho docílíme pomocí následujících možností:

- Digitálně vyhodnocením snímku obrazové scény
- Pomocí PIR závory
- Metoda využívající aktivního měření vzdálenosti osoby od kamery – například laserovým dálkoměrem

1.2.1.1 Digitální vyhodnocení snímku obrazové scény

Jak už naznačuje nadpis kapitoly, tak zachycení snímku s osobou ve vzdálenosti předmětu bude sloužit pouze digitální vyhodnocení obrazu z kamery. Protože známe, kde se porovnávaný předmět nachází, tak známe i jeho vzdálenost v obraze. Pokud je porovnávaný předmět, například tyč, umístěná kolmo k podlaze, tak vzdálenost můžeme digitálně do scény vynést pomocí přímky, která bude kolmá k předmětu a rovnoběžná se spodním okrajem obrazu. Tato přímka nám nyní bude udávat pomyslnou kolmou rovinu, která udává vzdálenost ve snímané scéně. Pokud se bude osoba blížit směrem ke kameře, bude se shora blížit k nanesené čáře a k pořízení snímku s osobou v námi požadované vzdálenosti bude stačit pořídit snímek, kdy osoba protne tuto vynesenou čáru.

Pokud bude snímaná scéna zakřivená v důsledku aberace způsobené čočkami kamery, musíme i tuto aberaci zahrnout do tvaru této čáry. Vše lze vidět v následujícím obrázku.



Obrázek 1 Znázornění digitální aplikace přímky vzdálenosti předmětu s a bez případné aberace

Pozice a znamená, že se osoba ve scéně nachází ve větší vzdálenosti od kamery než vzorový předmět. Pozice b znamená, že se osoba nachází ve vzdálenosti jako vzorový předmět a pozice c znamená, že se osoba nachází blíže kameře, než vzorový předmět. Pro výpočet výšky osoby je důležitá právě pozice b, kdy v daném snímku zjistíme velikost osoby v pixelech a pomoci výše uvedené rovnice vypočteme její výšku.

Velkou nevýhodou této metody je fakt, že osoba někdy nemusí pomyslnou čáru protnout, protože se může pohybovat pouze v prostoru za ní nebo před ní. To by se dalo ošetřit například tím, že bychom předmět v pixelech naměřili v různých vzdálenostech a potom bychom vynesli těchto pomyslných čar více v závislosti na vzdálenostech předmětů. Každá pomyslná čára by přitom odpovídala vzdálenosti předmětu. Pak by stačilo hlídat protnutí jakékoliv vynesené čáry a dosazení do rovnice korespondující velikosti předmětu v pixelech, dle protnuté čáry. Další nevýhodou je nepřesnost, která vzniká v důsledku rozlišení kamery a digitalizovaní pomyslné hranice, protože i kdybychom vykreslili čáru o tloušťce 1 pixelu, tak 1 pixel může ve větších vzdálenostech od kamery v obraze znamenat i rozmezí vzdálenosti v řádu 10 centimetrů.

Výhoda této metody spočívá hlavně v jednoduchosti její implementace, protože pracujeme jen s jedním zařízením pro měření - kamerou.

1.2.1.2 Metoda využívající PIR závory

Další metodou, jak pořídit snímek osoby v potřebné vzdálenosti od kamery je, propejení kamery s PIR závorou, kdy pořízení snímku scény řídíme impulzem z PIR závory. Impulz pro pořízení snímku kamery bude vyslán, když dojde k přerušením paprsků PIR závory. PIR závoru umístíme do vzdálenosti odpovídající vzorovému předmětu.

Zde se setkáváme s velkou nevýhodou v podobě možnosti, že osoba nemusí nikdy PIR závoru protnout. Tato nevýhoda se odstranit pomocí sítě PIR závor, ale bylo by to nákladné a přibylo by hlídání více impulzů. Další nevýhodou může být seřízení a koordinace kamery s PIR závorou v programovacím prostředí.

Tato metoda má i nesporné výhody. Hlavní výhodou je vyhodnocování prakticky v reálném čase. Scéna se nemusí neustále snímkovat a vyhodnocovat, ale dochází jen k vyhodnocování signálu z PIR závory. Je to nesrovnatelně rychlejší, v porovnání s

neustálým snímkováním a vyhodnocováním. Dále nemusíme zaznamenávat všechny snímky, ale stačí nám pouze snímek v okamžiku protnutí PIR závory osobou. Tento snímek je následně pomocí rozdílování a prahování vyhodnocený, abychom v něm zjistili osobu. Další výhodou je větší přesnost, než u předchozí metody - porovnávání s předmětem a použití digitální hranice v obraze.

1.2.1.3 Metoda využívající aktivního měření vzdálenosti osoby od kamery

V realizaci a rychlost vyhodnocování je tato metoda srovnatelná s metodou předchozí. Také zde, za cenu složitější implementace součinnosti zařízení pro měření vzdálenosti, je zajištěno rychlejší vyhodnocování snímané scény, které je opět prakticky v reálném čase.

Metoda spočívá v aktivním zjišťování vzdálenosti osoby od kamery. Tuto vzdálenost můžeme zjistit například využitím laserového dálkoměru nasměrovaného rovnoběžně s osou kamery.

Bohužel, realizace této měřicí soustavy je složitější než u metody využívající čistě digitálního zpracování, protože musíme implementovat součinnost zařízení v programovacím prostředí. Tato metoda má další velkou nevýhodu. Z důvodu namíření laserového dalkoměru rovnoběžně s osou kamery, by opět mohlo dojít k situaci, kdy osoba neprotne osu kamery a nedojde k změření vzdálenosti a tím pádem ani k vyhodnocení výšky osoby. Nevýhoda by se dala odstranit použitím více laserových dálkoměrů. To opět vede k složitější realizaci a prodražení.

1.2.2 Stereoskopické snímání scény pomoci dvou kamer

Tuto metodu zde popíši stručně, protože se jí budu věnovat prodrobněji v praktické části této práce, v kapitole 3.4, kde metodu aplikuji pro výpočet výšky osob.

Metoda využívá snímání scény dvěma kamerami. V podstatě jde o realizaci principu lidského vidění pomocí dvou kamer, kdy kamery mají funkci lidských očí. Kamery jsou umístěné v neměnné vzdálenosti. Například lidské oči jsou v průměru ve vzdálenosti 6,5 cm od sebe. Každá kamera snímá svůj vlastní obraz a na základě korelace odpovídajících bodů na obraze první a druhé kamery lze spočítat polohu osoby a následně její výšku. Kamery musí mít rektifikované (seřízené) osy, tzn. musí být ve stejné výšce a rovnoběžně. Metoda

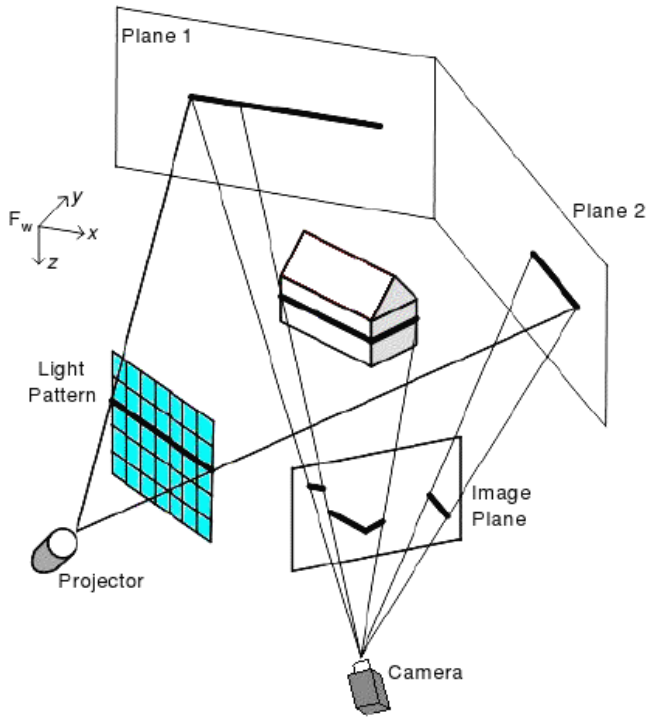
patří do tzv. pasivní triangulace, kdy k odvození vztahů pro výpočet použijeme podobnosti trojúhelníků.

1.2.3 Měření pomocí kamery a osvícení scény pomocí světelného zdroje

Nejnáročnější na realizaci, na výpočet a na princip je následující metoda, která využívá soustavy složené z níže uvedených částí, viz. také obrázek 2.

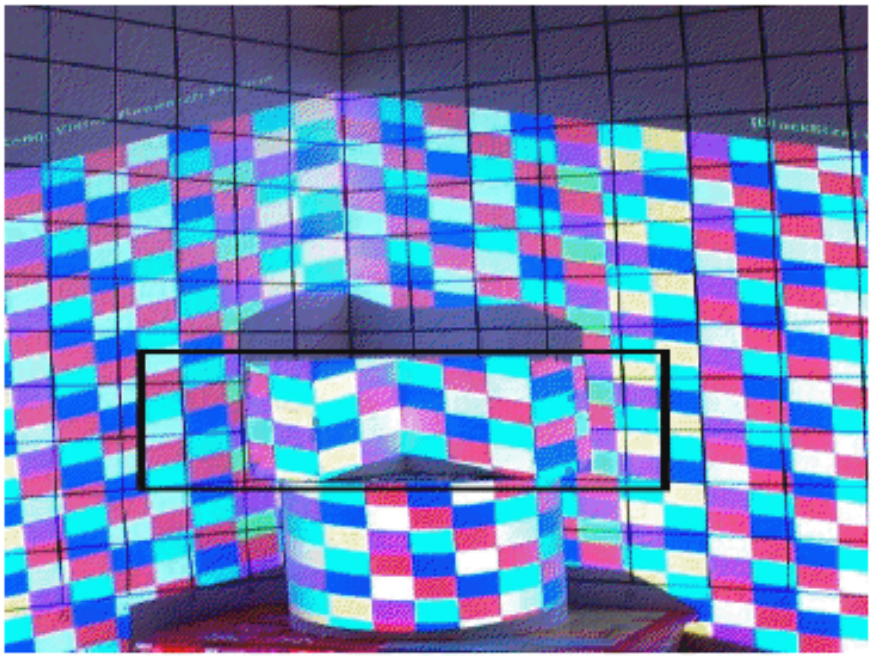
- Kamera (Camera)
- Projektor nebo zdroj osvětlení (Projector)
- Vzor pro nasvícení (Light Pattern)
- Dvě plátna (Plane 1 & Plane 2)

Metoda spočívá v nasvícení objektu přes vzor (například mřížku), který vytvoří na pozadí dvou pláten vzor, ten je následně nasnímám kamerou a vyhodnocen. Pomocí výsledného vzoru v závislosti na vzájemných polohách ploch obrazu, pláten a vzoru, můžeme vypočítat rozměry osvětleného objektu. Soustava pro toto měření vypadá následovně.



Obrázek 2 Soustava pro měření rozměrů objektu [1]

Výsledný obrázek na kameře, který zachycuje osvětlenou scénu přes mřížku, může vypadat následovně.



Obrázek 3 Snímek z kamery osvětlené scény projektorem přes mřížku [1]

K výpočtu rozměrů dochází na základě vyhodnocení vzorů, které jsou vrženy osvětlením projektoru na promítané roviny pláten. Výpočet je složitý, a proto zde uvádím jen existenci této metody.

II. PRAKTICKÁ ČÁST

2 PROSTŘEDKY VYUŽITÉ K VÝVOJI A NÁVRHU APLIKACE PRO MĚŘENÍ VÝŠKY OSOB POMOCÍ KAMEROVÉHO SYSTÉMU

V praktické části mé bakalářské práce se věnuji použitým hardwarovým a softwarovým prostředkům k realizaci a návrhu aplikace a experimentálnímu zařízení pro měření výšky osob pomocí kamerovým systémem. Popisuji zde chod programu a samotnou aplikace, která je navržena v programovacím prostředí MATLAB od massachusettské firmy MathWorks v USA. Popisuji a hodnotím průběh a poznatky z měření. Uvádím nástin možnosti vylepšení a pokračování v této práci. Nakonec rozebírám možnosti použití v bezpečnostní problematice.

2.1 Hardwarové prostředky

Tato část je věnována použitým hardwarovým prostředkům v rámci měření. Při vlastním měření jsem kromě samotného výpočetního stroje použil dvou webových kamer pro snímání scény, které byly připojeny k počítači pomocí rozhraní USB.

2.1.1 Microsoft WebCam: LiveCam Studio

Pro snímání scény a pořizování snímku k následnému vyhodnocení jsem použil dvě webové kamery LiveCam Studio od firmy Microsoft. Tyto kamery umožňují pořizování videa v kvalitě 720p HD. K samotnému experimentu a měření bohužel této možnosti použito nebylo z důvodu hardwarové náročnosti a tím pádem prodloužení vyhodnocování jednotlivých snímků. Jedná se o Full HD kameru, která dokáže zaznamenávat až 30 snímků za sekundu. S tím, že ke snímání bude využito oficiálního programu. Při vyhodnocování pomocí aplikace vytvořené v programovacím jazyku MATLAB, je dosáhnutí této snímací frekvence nereálná.

Obrázek 4 Microsoft WebCam: LiveCam Studio [3]

Zde je tabulka se základními hardwarovými požadavky na funkci jedné kamery. Můžeme si všimnout, že už jen pro práci s jednou kamerou jsou to poměrně veliké nároky.

Tabulka 1 Hardwarové a softwarové požadavky pro chod jedné webové kamery

minimální		optimální - pro video 720p HD	
Procesor	Intel Dual-Core 1.6 GHz a víc	Procesor	Intel Dual-Core 3.0 GHz a víc
Pamět	1 GB RAM	Pamět	2 GB RAM
HDD	1.5 GB	HDD	1.5 GB
Operační systém	Windows XP service pack 2, Windows Vista, Windows 7	Operační systém	Windows XP service pack 2, Windows Vista, Windows 7
USB	2.0	USB	2.0

2.2 Softwarové prostředky

V této části popisuji využité softwarové prostředky, které jsem využil pro vývoj aplikace k měření osob a k otestování funkčnosti navrženého experimentálního zařizení.

Pro naprogramování programové části práce bylo použito programovací prostředí MATLAB, verze 2011a, a jeho součásti. Pro otestování funkcí kamer a jejích možností jsem použil oficiální software od firmy Microsoft LifeCam verze 3.6, pomocí kterého lze nastavovat všechny vlastnosti kamer (rozlišení, korekce osvětlení, barvy, jas, apod.). Podobné možnosti nastavení kamer nabízí i níže popsané prostředí programu MATLAB 2011a skrze funkci imaqtool, kterou popisuji podrobněji níže. Jednu funkci ovšem nenabízí a to je vypnutí speciální korekční funkce obrazu kamery TrueCOLOR, které je pro vyhodnocování nutné vypnout, aby neovlivňovalo velikou proměnlivost obrazu při

vyhodnocování pozadí. Vypnutí provedeme pomocí výše zmíněného oficiálního programu LifeCam od společnosti Microsoft.

2.2.1 MATLAB 2011a

Programovací prostředí MATLAB jsem zvolil proto, že obsahuje už předem zpracované prostředky pro práci s obrazem, technikou a přístroji, aniž by bylo potřeba je ručně programovat. V našem případě pro vývoj aplikace byly použity prostředky pro nahrávání obrázků z webkamery a pro jejich následné zpracování. Tyto prostředky jsou v programovacím prostředí dělené, kromě základních vestavěných funkcí, do tzv. Toolboxů, které by bylo možno nazýváme moduly nebo knihovny. Jednotlivé Toolboxy obsahují funkce (podprogramy) podle svého významu a funkce. Takovými prostředky (použité funkce z těchto toolboxů uvedu a popíši níže) jsou například:

- **Image Processing Toolbox** – pro práci, úpravu a zobrazování obrázků
- **Image Acquisition Toolbox** – pro nahrávání, manipulaci a správu obrazových dat

Nejenom funkce v toolboxech, ale i samotné prostředí MATLAB podporuje základní funkce, které jsou pro tento programovací (skriptovací) jazyk důležité. Jsou například základní matematické operace, operace nad maticemi, operace pro vykreslování GUI prostředí, operaci pro práci s grafy, atd. Níže uvádím funkce, které jsem k realizaci programové části práce použil.

2.2.1.1 Vestavěné prostředky

Vestavěných prostředků v MATLABu je mnoho a mnoho, proto zde nebudu uvádět a popisovat všechny. Samy o sobě by stačily na celou bakalářskou práci. Uvedu zde jen ty, kterých jsem použil při programování aplikace.

- **set, get, findobj** – jsou to vestavěné funkce sloužící k práci s objekty GUI rozhraní. Set slouží k nastavování hodnot těchto objektů. Get slouží k získávání hodnot těchto objektů a funkce findobj, jak už z anglického názvu vyplývá, slouží k hledání objektů. Konkrétně funkce vrací hodnotu, cestu, k danému objektu, pod jakou je uložen v paměti.
- **zeros** – tato funkce slouží pro vytvoření matice řádu *n* x *n*, jejíž všechny prvky budou nulové.

- **int2str, str2int, str2double, uint8(16, 32), logical** – toto jsou funkce sloužící pro převod mezi datovými typy proměnných. Funkce str2int převádí proměnnou datového typu string na proměnnou datového typu integer. Dále funkce logical nastavuje proměnné datový typ, kde bude obsahovat pouze 1 nebo 0.
- **rectangle** a **text** – rectangle slouží pro vykreslení, v našem případě, obdélníku na obrazovku podle zadaných parametrů. V práci tuto funkci používám k označení pohybujících se osob (objektů, změn) ve scéně. Funkce text slouží k vypsání textu na obrazovku. V programu tuto funkci využívám k vykreslení těžiští postav a středů zobrazované scény.

2.2.1.2 Image Processing Toolbox

I tento toolbox obsahuje nepřeberné množství funkcí, tak i tento jich obsahuje mnoho. Uvedu jen ty, které jsem použil k vývoji a naprogramování aplikace na měření výšky osob. Jak už vyplývá z názvu toolboxu a jak už jsem uvedl výše, tak tento toolbox obsahuje funkce důležité pro práci, úpravu a manipulaci s obrázky. Obsahuje nejen funkce pro čtení a zápis obrázku z médií (harddisk, CD, flash paměť a jiné), ale také nástroje pro převod obrázků mezi formáty (RGB, odstíny šedi, černobílé (binární) - logické obrázky), komprimaci, zobrazování na obrazovku, složitější operace jako (thresholding, hledání hran, filtrace), a mnoho dalších.

V mé bakalářské práci jsem použil zejména následujících vestavěných funkcí z výše zmíněného toolboxu:

- **imshow** – zobrazení obrázku na obrazovku z paměťové proměnné nebo z jiného média
- **imwrite, imread** – funkce sloužící k zápisu nebo načtení z HDD.
- **im2gray** – převod formátu obrázku z RGB (červená, zelená, modrá) na formát grayscale (stupně šedi)
- **im2bw** – funkce sloužící pro převod obrázku z formátu stupně šedi, který vznikl odečtením dvou obrázků ve stupních šedi, do tzv. logického (binárního) obrázku, kdy jsou hodnoty nastaveny buďto na 1 nebo 0 v závislosti na hodnotě jasu obrázku (proběhne tzv. thresholding, popsaný v kapitole … teoretické části)

- **bwmorph** – jsou to tzv. filtry, sloužící k úpravě logických obrázků, například k ošetření šumu nebo smazání osamocených bodů (hodnoty 1 obklopeny samými hodnotami 0), mezi jednotlivými filtry se volí tak, že jejich název slouží jako parametr funkce bwmorph, který se zapisuje v apostrofech, a jednotlivé parametry mohou být
  - **dilate** – nastavuje hodnotu výstupního pixelu na maximální hodnotu z hodnot pixelů jeho nejbližšího okolí.

    1 0 0 1 0 0
    1 **0** 0 -> 1 **1** 0
    0 0 0 0 0 0

  - **erode** – opačná funkce k funkci dilate, která nastaví hodnoty výstupního pixelu na minimální hodnotu z hodnot pixelů jeho nejbližšího okolí.

    1 0 0 1 0 0
    1 **1** 0 -> 1 **0** 0
    0 0 0 0 0 0

  - **clean** – odfiltruje osamocené pixely hodnoty 1, tj. pixely obklopeny samými pixely hodnoty 0.

    0 0 0 0 0 0
    0 **1** 0 -> 0 **0** 0
    0 0 0 0 0 0

  - **majority** – tento filtr funguje tak, že pokud je v jeho okolí 5 a více pixelů hodnoty 1, nastaví se jeho výstupní hodnota na 1, v opačném případě se nastaví na 0.

    1 1 1 1 1 1 1 1 0 1 1 0
    1 **0** 1 -> 1 **1** 1 0 **1** 1 -> 0 **0** 1
    0 0 0 0 0 0 0 0 0 0 0 0

  - **close** – tento filtr je kombinací filtrů erode a dilate, kdy se první provede dilate a následně erode.
  - **fill** – tento filtr je opakem filtru clean, kdy vyhledá pixely hodnoty 0, které jsou obklopeny pixely hodnoty 1, a nastaví jeho výstupní hodnotu na 1.

- **bridge** – tento filtr nastavuje výstupní hodnotu pixelu na 1 v případě, že má nespojené sousedy s hodnotou 0.

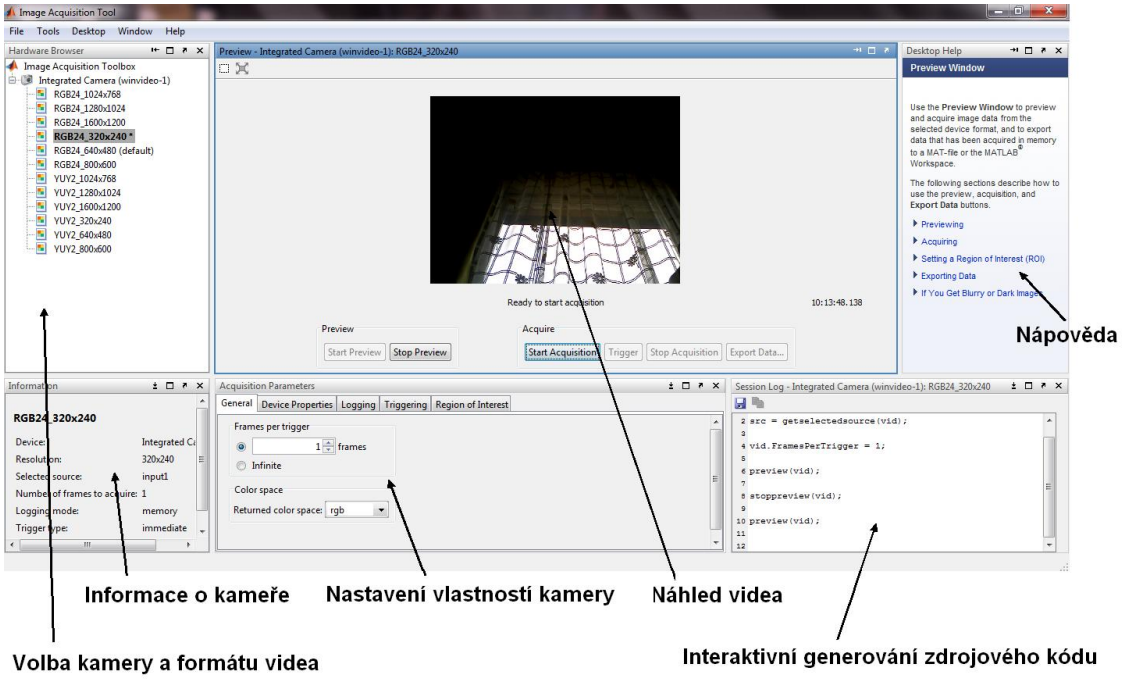
```
1 0 1          1 1 1   1 0 0          1 1 0
1 0 1    ->    1 1 1   1 0 1    ->    1 1 1
0 0 0          0 0 0   0 0 1          0 1 1
```

Z výše uvedených filtrů jsou dle mého názoru pro úpravu binárních obrázků nejdůležitější filtry erode a dilate, nebo jejich kombinace v podobě filtru close, neboť tyto filtry plně nebo částečně nahradí funkci zbývajících filtrů clean, fill, majority a bridge. Použití každého filtru navíc, znamená zpomalení v rychlosti provádění programu, a tím pádem i v zaznamenávání pohybu osob a následného vyhodnocení výšky osob. V samotné aplikaci skrze grafické uživatelské prostředí ale můžeme v sekci „Filter Option“ za chodu měnit, které filtry budou použité pro vyhodnocování pohybu.

2.2.1.3 Image Acquisition Toolbox

Tento toolbox obsahuje prostředky, které jsou určené zejména pro správu a získávání videa z kamer, v podobě snímků, připojitelných k počítači. Obsahuje funkce a prostředky pro získání videa z kamery, pro jeho správu (ukládání do paměti, HDD), ale i prostředky (objekty) pro ovládání a nastavování parametrů kamer (jas, barvy, zoomování, apod.).

- **imaqtool** – je jednou ze základních funkcí tohoto toolboxu. Slouží k vyzkoušení ovládání a nastavování funkcí kamery pomocí uživatelského rozhraní. Uživatel si může zvolit, kterou kameru chce použit a nastavit její příslušné rozlišení. Potom si může zobrazit náhled videa, pořídit snímek a data uložit jako proměnnou do prostředí MATLABu, kde s ní můžeme následně pracovat. V neposlední řadě toto rozhraní umožňuje vygenerovat kód (skript), který může uživatel následně použít ve svém zdrojovém kódu k práci s kamerou. Následující obrázek naznačuje rozložení funkcí, které toto rozhraní nabízí.



Obrázek 5 MATLAB – rozhraní imaqtool

- **videoinput, getselectedsource, delete** – funkce videoinput vytvoří video objekt, do kterého se dle nastavených parametrů ukládají jednotlivé snímky videa. S tímto objektem je možné dále pracovat pomocí dalších funkcí. Funkce getselectedsource se vstupním parametrem video objektu, vytvořeným funkcí videoinput, vytvoří objekt, pomoci kterého je možno nastavovat vlastnosti kamer. V práci jsou pomocí tohoto objektu nastavovány vlastnosti kamery jako kompenzace světla na pozadí, zaostřování kamery, délka expozice. Funkce delete slouží k smazání vytvořeného video objektu.

- **start, stop, preview** – tyto funkce slouží ke spuštění, zastavení a náhledu dat z kamer. Funkce start se vstupním parametrem ve formě video objektu zajistí, že je video (data) z kamer přístupné prostředí MATLAB a video je postupně ukládáno do paměti k možnosti jeho dalšího zpracování pomocí jiné funkce (trigger). Funkce stop zastaví ukládání do paměti a funkce preview slouží k náhledu videa, které je ukládáno do paměti.

- **triggerconfig** – slouží k volbě, zda bude video načítáno manuálně za pomocí funkce trigger, která slouží k pořízení několika za sebou jdoucích snímků, se vstupním

parametrem jména video objektu a s parametrem udávajícím počet snímků, nebo automaticky po funkci start. Volba se zadává ve formě vstupního parametru. V mé práci jsem zvolil možnost manuálního ukládání snímků, kdy se získává jen jeden snímek, vždy před jeho vyhodnocením.

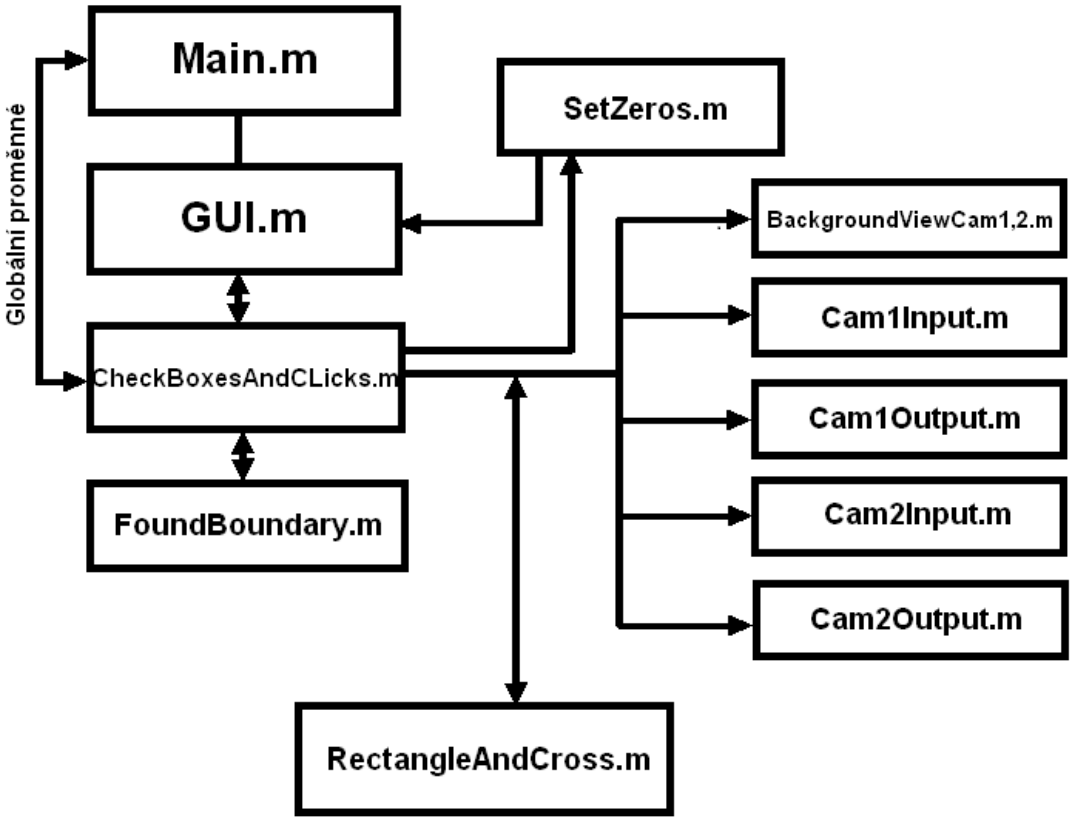
- **getshapshot, getdata** – funkce getshapshot slouží k pořízení jednoho snímku z kamery. Funkce getdata slouží k získání dat, která byla načtena pomocí funkce trigger. Data jsou pak uložena do vektoru obrázků.

3 POPIS A FUNKCE VLASTNÍ APLIKACE

Jak jsem uvedl výše, k naprogramování aplikace jsem využil prostředí MATLAB a jeho součástí ve formě patřičných toolboxů. Samotný program je rozdělený do několika vzájemně závislých .m souborů. Některé fungují jako funkce, jsou jim předávány parametry a samy parametry vrací. Takové soubory mají první řádek s označením „function“, který v prostředí MATLAB znamená, že se jedná o funkci. Další fungují jen jako skripty za pomoci globálních parametrů. Celý zdrojový kód, v podobě všech souborů s připonou .m, je v digitální formě na přiloženém CD.

3.1 Struktura celého programu a popis jednotlivých částí

Struktura celého programu je složena z více souborů s příponou .m. Konkrétně je rozdělen do 12 samostatných souborů obsahujících zdrojový kód, ale jejich celá funkce by se bez žádného neobešla. Dohromady tvoří jeden celek. Jejich vzájemná propojenost lze vidět na následujícím obrázku.



Obrázek 6 Struktura celého programu a návaznost jeho jednotlivých částí

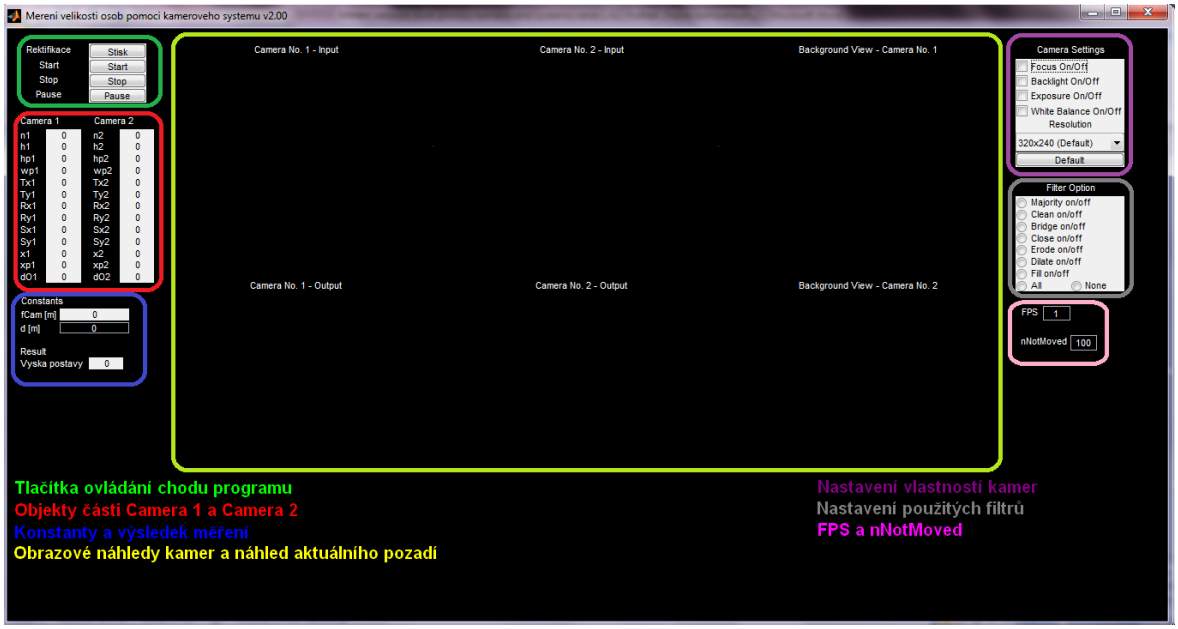
Celé by to šlo samozřejmě naprogramovat do jednoho velkého .m souboru, ale z programátorského hlediska by to bylo hodně nepřehledné a zmatené. Popis jednotlivých .m souborů je uvedený v následujících podkapitolách.

3.1.1 Main.m

Soubor Main.m je hlavní soubor. Jedná se o skript, který se spouští na samotném začátku. Definuje zejména globální proměnné, se kterými pracují ostatní funkce aplikace. Kromě proměnných pro vlastní výpočet se tu definují objekty pro práci s kamerou a pro nahrávání videa.

3.1.2 GUI.m

GUI.m soubor také plní roli skriptového souboru a zajišťuje definici a vytvoření grafického uživatelské rozhraní (dále jen „GUI“). Dochází zde k inicializaci všech ovládacích prvků (objektů) aplikace, ať už se jedná o kolonky, kde se zobrazují hodnoty proměnných, nebo prvky, pomocí kterých se dají měnit vlastnosti kamer. Rozdělení GUI je znázorněno na následujícím obrázku. Samotné GUI je plně přizpůsobitelné rozměrům okna a rozlišení monitoru. Všechny hodnoty, jako rozměry jednotlivých objektů a jejich poloha, jsou počítány relativně (normovaně) v závislosti na velikosti rozlišení obrazovky. Jsou v rozmezí od 0 do 1, kdy hodnota 1 v horizontálním směru znamená hodnotu počtu pixelů v jednom řádku na obrazovce. GUI se zvýrazněním jednotlivých částí je vidět na obrázku 7.



Obrázek 7 GUI

3.1.2.1 Tlačítka ovládání chodu programu

Jedná se objekty typu uicontrol, které reaguji na kliknutí levého tlačítka myši a vyvolají tzv. „callback“, kterým se provede část kódu v rámci souboru CheckBoxesAndClicks.m. Jsou to tyto čtyři tlačítka:

- **Rektifikace** – tlačítko slouží pro proceduru rektifikace os kamer
- **Start** – tlačítko slouží ke spuštění celého algoritmu měření, samotný algoritmus bude popsán v kapitole 4.3.
- **Stop** – slouží k zastavení průběhu měření a smazání objektů videa. Po stisku tlačítka se nedá vrátit k aktuálnímu stavu měření, ale musí se znovu zmáčknout tlačítko start. Výsledky aktuálního měření zůstanou zobrazeny.
- **Pause** – slouží k pozastavení průběhu měření. Po jeho opětovném stisku měření pokračuje v tom místě, kde bylo přerušeno. Ne však od stavu snímané scény, která se mohla změnit během pozastavení.

3.1.2.2 Objekty části Camera 1 a Camera 2

Jedná se o objekty typu text, které pouze zobrazují hodnoty, s kterými algoritmus počítá. Podrobně budou popsány v části 3.2. Jsou to proměnné parametrů snímané scény

(výška objektu v pixelech, těžiště objektu, aktuální číslo snímku a jiné). První sloupec ukazuje proměnné první kamery a druhý sloupec ukazuje proměnné druhé kamery.

3.1.2.3 Konstanty fCam a d, a výsledek měření

V části GUI Constants jsou zobrazeny konstanty používané k výpočtu výšky zjištěných osob (objektů). Jedná se o hodnoty označené:

- **fCam [m] –** vypisuje hodnotu ohniskové vzdálenosti kamer v metrech.
- **d [m]** – tento objekt se trochu liší od předešlých, a to tím, že je to objekt typu „edit" a tím pádem můžeme jeho obsah měnit za chodu nebo před začátkem měření. Jedná se o vzdálenost mezi kamerami v metrech a tento typ jsem zvolil z toho důvodu, že z důvodu instalace nemusí být vzdálenost kamer konstantní. Aby se nemuselo zasahovat do zdrojového kódu programu, tak se hodnota d zadá vždy před začátkem chodu programu.

Kolonka Result zobrazuje vypočtenou výšku zjištěné osoby (objektu) ve snímané scéně. Hodnota je vypsána v metrech.

3.1.2.4 Obrazové náhledy kamer a náhled aktuálního pozadí

Tady se zobrazují náhledy aktuálních neupravených a upravených snímků jednotlivých kamer. Dále je zde zobrazeno aktuální vygenerované porovnávací pozadí. Tato část rozhraní GUI je rozdělena na šest stejně velkých částí, ve kterých se snímky zobrazují. Dělí se na následující objekty:

- **Camera No. 1 – Input** – zde se zobrazuje aktuální vstupní snímek z první kamery s vykresleným obdélníkem kolem pohybujícího se objektu, v našem případě osoby.
- **Camera No. 2 – Input** – zde se analogicky, jako u objektu Camera No. 2 – Input zobrazuje aktuální vstupní snímek, ale druhé kamery.
- **Camera No. 1 – Output** – zde se zobrazuje aktuální výstupní snímek z první kamery s vykresleným obdélníkem kolem pohybujícího se objektu. Tento snímek je ve formě binárního obrázku, který je výstupem použitého algoritmu pro rozpoznávání pohybu ve scéně.

- **Camera No. 2 – Output** – zde se zobrazuje aktuální výstupní snímek z druhé kamery s vykresleným obdélníkem kolem pohybujícího se objektu. Tento snímek je ve formě binárního obrázku, který je výstupem použitého algoritmu pro rozpoznávání pohybu ve scéně.
- **Background View – Camera No. 1** – zde se zobrazuje náhled aktuálního vygenerovaného pozadí algoritmem pro detekci pohybu ve snímané scéně z první kamery. Podrobný popis algoritmu pro generování pozadí bude popsán v kapitole 3.3.
- **Background View – Camera No. 2** – zde se zobrazuje náhled aktuálního vygenerovaného pozadí algoritmem pro detekci pohybu ve snímané scéně z druhé kamery.

3.1.2.5 Nastavení vlastností kamer

Zde se dají pomocí zaškrtávacích kolonek (checkboxů), nebo v případě nastavení rozlišení pomocí seznamu (pop up okna), nastavit základní vlastnosti kamer, jako jsou

- **Focus** – zaostřování
- **Exposure** – délka uzávěrky
- **White Balance** – kompenzace bílých barev
- **Backlight** – kompenzace osvětlení
- **Resolution** – změna rozlišení snímaných obrázků kamerou

Tlačítkem Default, můžeme nastavit všechny vlastnosti kamer na původní hodnoty, které byly nadefinovány na začátku měření. Je nutné poznamenat, že každá změna nastavení vlastnosti kamery musí být provedena před spuštěním měření pomocí tlačítka start. Jinak by nastalo chybové hlášení a pád aplikace. Prostředí MATLAB nedovoluje měnit vlastnosti kamer za chodu.

Implicitně jsou všechny vlastnosti, které se týkají softwarové kompenzace obrazu, vypnuté. Vypnuté jsou proto, aby nedocházelo k velkým změnám ve snímané scéně, které mohou být způsobeny zaostřením kamery, nebo redukcí osvětlení, apod. Tyto změny by mohly být, a s největší pravděpodobností by taky byly, vyhodnoceny jako pohyb.

3.1.2.6 Nastavení použitých filtrů

V této části GUI za chodu měnit, použití filtrů pro vyhodnocování pohybu ve snímané scéně. Jsou to objekty typu „radio columns“, které fungují obdobně jako zaškrtávací kolonky. Princip funkce jednotlivých filtrů byl popsán v kapitole 2.2.1.2 v části věnované funkcím Image Processing Toolboxu. Implicitně není použit žádný filtr, ale minimálně použití filtru erode a dilate bych označil jako významné. Proč bude popsáno dále. Pomocí kliknutí na All nebo None, můžeme povolit nebo zakázat všechny filtry najednou.

3.1.2.7 FPS a nNotMoved

Tyto dva poslední objekty jsou typu „edit“, tzn. jejich hodnotu můžeme měnit za běhu programu.

První kolonka FPS slouží ke změně frekvence zobrazování snímků na obrazovku. To znamená, že pokud bude nastavena hodnota FPS na 10, tak se bude zobrazovat a vykreslovat v části obrazových náhledů kamer pouze každý 10 snímek. Slouží nám to ke zrychlení provádění programu, protože zobrazování snímků na obrazovku je hardwarově nejnáročnější a nejdéle trvá.

Kolonka nNotMoved slouží jako mez pro generování pozadí za běhu. Konkrétně udává počet snímků, ve kterých nedošlo v rámci jednoho pixelu ke změně, aby mohl být tento pixel vyhodnocený jako pozadí a být zařazen do vygenerovaného pozadí.

3.1.3 CheckBoxesAndClicks.m

Jedná se o funkci, která ošetřuje volání tzv. „callbacků“, které vyvolávají objekty rozhraní GUI. Tyto objekty předávají tzv, identifikátor daného „callbacku“ jako vstupní parametry této funkci, který je následně pomocí větvení programu „case“ vyhodnocen a je provedena odpovídající operace (část kódu). Například objekty vyvolávající zmíněné „callbacky“ jsou stisknutí tlačítka, nebo zatržení zaškrtávacího pole (checkboxu), dále políčko pro volbu filtrů (anglicky radio columns) a v neposlední řadě i objekty typu „edit“, kde je možno během chodu zadat hodnoty pro běh programu (proměnná d, FPS a nNotMoved). Funkce těchto „callbacků“ jsem z části popsal výše, v kapitole 3.1.2, kde jsou popsané i jednotlivé části.

Dále se tu nachází hlavní část programu, která se spouští klepnutím na tlačítko Start. Běhu a principu algoritmu na počítání výšky, a detekci pohybu v obraze bude věnována kapitola 3.3.

3.1.4 FoundBoundary.m

Jedná se o funkci se vstupním parametrem obrázku reprezentovaným ve formě dvourozměrné matice. Tato matice je typu logical, což je binární obrázek a to znamená, že obsahuje pouze hodnoty 0 nebo 1, kde 0 reprezentuje místo bez pohybu a 1 reprezentuje místo, kde je pohyb. Tato funkce vrací vektor pozic nejbližšího nenulového sloupce matice zleva a zprava a nejbližšího nenulového řádku shora a zdola a příznakovou proměnnou Found. Pokud je objekt při běhu algoritmu nalezen, tak se nastaví hodnota proměnné Found na 1 a pokud ne, tak je hodnota proměnné Found nastavena na 0. Hodnota Found se dále nastavuje na 0, pokud by poloha nenulových pixelů byla stejná zprava i zleva nebo shora a zdola, to znamená vyhledání samostatného pixelu nebo skupiny pixelů v jednom řádku nebo sloupci. Tyto parametry jsou předané zpátky do hlavní části programu nacházející se ve funkci CheckBoxesAndClicks.m a slouží jako parametry pro funkci rectangle v rámci funkce RectangleAndCross.m, která ohraničuje zeleným obdélníkem pohybující se objekty - osoby ve snímané scéně.

Algoritmus se skládá ze čtyř cyklů „for“. Každý cyklus slouží k nalezení jedné hrany (levé, spodní, horní, pravé) pohybujícího se objektu - osoby. Cyklus probíhá do té doby, dokud nenajde nenulový součet hodnot řádku či sloupce matice. Skončí také, pokud bylo dosaženo okraje obrázku, tzn. že byly prozkoumány všechny řádky nebo sloupce matice a žádný neobsahoval ani jeden pixel hodnoty 1.

3.1.5 BackgroundViewCam1.m a BackgroundViewCam2.m

Opět se jedná o funkci, kdy je vstupním parametrem obrázek ve formě dvourozměrné matice, která obsahuje hodnoty pixelů pozadí. Funkce slouží pro vykreslení tohoto obrázku do rozhraní GUI na danou pozici. Konkrétně zobrazuje matici obsahující hodnoty aktuálně generovaného pozadí pro danou kameru.

3.1.6 Cam1Input.m, Cam1Output.m, Cam2Input.m a Cam2Output.m

Jedná se o funkce s jedním vstupním parametrem ve formě snímku z kamer, reprezentovaným opět dvourozměrnou maticí. Všechny tyto funkce slouží k vykreslení snímku na obrazovku. Liší se v tom, že každá funkce slouží k zobrazení jiného snímku na jínou pozici v rámci GUI. Co jednotlivé funkce zobrazují, bylo popsáno v kapitole 3.1.2.4.

3.1.7 SetZeros.m

Skript slouží k vypsání nulových hodnot pro rozměry nalezených objektů v rámci funkce FoundBoundary.m do rozhraní aplikace GUI. Provádí se, pokud nebyly nalezeny relevantní pohybující se osoby ve snímané scéně.

3.1.8 RectangleAndCross.m

Funkce slouží k provedení následujících operací:

1. Výpočet těžiště nalezeného pohybujícího se objektu (osoby) a vykreslení tohoto těžiště do části rozhraní GUI do obrazu z dané kamery.
2. Výpočet výšky objektu v pixelech a vypsání tohoto rozměru do GUI.
3. Výpočet šířky objektu v pixelech a vypsání tohoto rozměru do GUI.
4. Vykreslení obdélníků do obrazů z kamer v rámci GUI za použití hodnot vstupních parametrů, které byly vypočítány funkcí FoundBoundary.m.
5. Vypsání do rozhraní GUI všech ostatních vstupních parametrů – Rx1 a Rx2
6. Výpočet vzdálenosti těžiště od středu v pixelech a předání této hodnoty zpět pomoci výstupního parametru do funkce CheckBoxAndClicks.m.

3.2 Použité proměnné

Popisuji zde hlavní použité proměnné pro počítání výšky osob. Jsou to zároveň proměnné, které jsou zobrazované v rozhraní GUI. Všechny proměnné nejsou přímo důležité pro vlastní výpočet, ale z důvodu testování a případné je bylo vhodné vypisovat.

- **n1, n2** – udávají číslo snímku první a druhé kamery, který je zpracováván.
- **hp1, hp2** – udávají vypočtenou výšku osoby (objektu) na obrazu z první a druhé kamery v pixelech.

Výsledná výška objektů z obou kamer je vypočítáná jako průměr jednotlivých výšek hp1 a hp2, která je potom dosazená do výpočtu skutečné výšky osoby. Rovnice výpočtu má následující tvar

$$hp = \frac{hp_1 + hp_2}{2} \tag{3}$$

- **wp1, wp2** – udávají vypočtenou šířku osoby (objektu) na první a druhé kameře v pixelech.
- **Tx1, Tx2** – udávají x-ovou souřadnici těžiště osoby (objektu) na snímku první a druhé kamery, zároveň slouží jako vstupní parametr pro funkci text, kterou se vykresluje těžiště objektu do snímků z kamer.
- **Ty1, Ty2** - udávají y-ovou souřadnici těžiště osoby (objektu) na snímku první a druhé kamery, použití viz. Tx1 a Tx2.
- **Rx1, Rx2** – udávají x-ovou souřadnici levého spodního rohu vykreslovaného obdélníku kolem osoby (objektu) (slouží jako vstupní parametr pro funkci rectangle) do snímku první i druhé kamery.
- **Ry1, Ry2** – udávají y-ovou souřadnici levého spodního rohu vykreslovaného obdélníku kolem osoby (objektu) do snímků z první a druhé kamery, použití viz. Rx1 a Rx2.
- **Sx1, Sx2** – udávají x-ovou souřadnici středu snímků z kamer v závislosti na rozlišení jejich snímání. Slouží také pro vykreslení křížků do středů snímků za použití funkce text.
- **Sy1, Sy2** – udávají y-ovou souřadnici středu snímků z kamer v závislosti na rozlišení jejich snímání. Slouží také pro vykreslení křížků do středů snímků za použití funkce text.

- **x1, x2** – udávají horizontální reálnou vzdálenost osoby (objektu) od jednotlivých kamer v metrech.
- **xp1, xp2** – udávají horizontální vzdálenost těžiště osoby (objektu) od středu snímané scény v pixelech pro první a druhou kameru. Jedná se o absolutní hodnotu rozdílu x-ové souřadnice těžiště objektu a x-ové souřadnice středu snímku.

$$xp_i = |Tx_i - Sx_i| \tag{4}$$

- **dO1, dO2** – udávají vypočtenou reálnou vzdálenost osoby (objektu) od první a druhé kamery v metrech.

3.3 Princip funkce hlavních algoritmů

V této části popisuji dva hlavní algoritmy aplikace na měření výšky osob pomocí kamerového systému. V první části popisuji algoritmus pro detekci pohybu ve snímané scéně spolu s autonomním generováním pozadí a v druhé algoritmus nebo spíše princip pro výpočet výšky detekované osoby (objektu) pomocí předešlého algoritmu.

3.3.1 Algoritmus pro detekci pohybujících se objektů a generování pozadí

Stisknutím tlačítka start nejprve dojde k inicializaci všech potřebných proměnných (matic), které budou požité v průběhu následujícího cyklu „while“. tento cyklus běží do té doby, dokud nestiskneme tlačítko stop. Během inicializace se vytvoří následující proměnné (názvy jsou dle zdrojového kódu), všechny jsou vytvořené s hodnotami 0.

- **IOMCam1, IOMCam2** – jsou to dvourozměrné matice, vytvořené funkcí zeros, proto obsahují samé nuly. Jsou typu logical, tedy binární matice (obrázek). Rozměry matice jsou odvozené od velikosti rozlišení kamer. Matice slouží k uchování pozic indexů pixelů, v kterých došlo ke změně (pohybu). Slouží pro finální filtrování a označení pohybujícího se osoby (objektu) ve snímané scéně funkcí rectangle v souboru RectangleAndCross.m. Zároveň jsou to vstupní parametry pro funkce Cam1Output.m a Cam2Output.m a budou zobrazeny v rozhraní GUI na obrazovku.

- **PreviousBackgroundCam1, PreviousBackgroundCam2** – znovu se jedná o dvourozměrné matice. Jejich velikost je závislá na rozlišení obou kamer a jsou vytvořené funkcí zeros s hodnotami 0. Tyto matice slouží k uložení stavu předchozího vygenerovaného pozadí. Hodnoty matice se budou pohybovat v intervalu <0, 255>, protože je v nich uchovávána informace o obrázku ve stupních šedi.
- **CurrentBackgroundCam1, CurrentBackgroundCam2** – fungují stejně jako PreviousBackgroundCam1 a PreviousBackgroundCam2 s tím rozdílem, že uchovávají aktuální stav vygenerovaného pozadí.
- **PreStatIndexCam1, PreStatIndexCam2** – jedná se opět o dvourozměrné matice pro uchování kladných čísel v intervalu <0, nNotMoved> s počátečními hodnotami 0 o velikosti v závislosti na rozlišení kamer. Slouží k uchování předchozí pravděpodobnosti, že na dané pozici ve snímané scéně nedošlo k pohybu, a že se v závislosti na hodnotě může jednat o pozadí na snímané scéně.
- **CurStatIndexCam1, CurStatIndexCam2** – principiálně jsou stejné jako PreStatIndexCam1 a PreStatIndexCam2 s tím rozdílem, že uchovávají aktuální stav pravděpodobností.
- **PreBackgroundIndicatorCam1, PreBackgroundIndicatorCam2** – stejně jako v předcházejících případech se jedná o dvourozměrné matice o velikosti v závislosti na rozlišení kamer, tvořené samými nulami. Jedná se o matice s hodnotami typu logical, takže jsou hodnoty 0 nebo 1. Ukládají se do nich informace o předchozím stavu, zda bylo na dané pozici nalezené pozadí nebo ne. Hodnota 0 znamená, že na dané pozici nebylo vygenerované pozadí a hodnota 1 znamená, že je na dané pozici bod udávající polohu bodu z pozadí.
- **CurBackgroundIndicatorCam1, CurBackgroundIndicatorCam2** – funkčně i hodnotami jsou stejné jako předchozí matice PreBackgroundIndicatorCam1 a PreBackgroundIndicatorCam2 s tím rozdílem, že uchovávají aktuální stav. Pokud je zde 1, tak se na této nachází bod pozadí. Hodnota 0 znamená, že na této pozici není bod pozadí. To značí dvě věci. Za prvé, že se zde nachází objekt, který se nehýbe jenom chvíli, nebo za druhé že se zde nachází bod pohybujícího se objektu.

Po inicializaci proměnných pro chod cyklu „while“ se dostáváme na začátek průběhu samotného cyklu. Celý cyklus se dá rozdělit do dvou částí:

- Začátek cyklu
- Tělo cyklu

Začátek průběhu cyklu je možné rozdělit na dva případy:

a) První případ je, když se jedná o první procházení cyklem, tady jsou kamerami pořízeny dva snímky a uloženy podle pořadí jejich pořízení do proměnných

- PreviousImageCam1 první snímek z první kamery

- PreviousImageCam2 první snímek z druhé kamery

- CurrentImageCam1druhý snímek z první kamery

- CurrentImageCam2 druhý snímek z druhé kamery

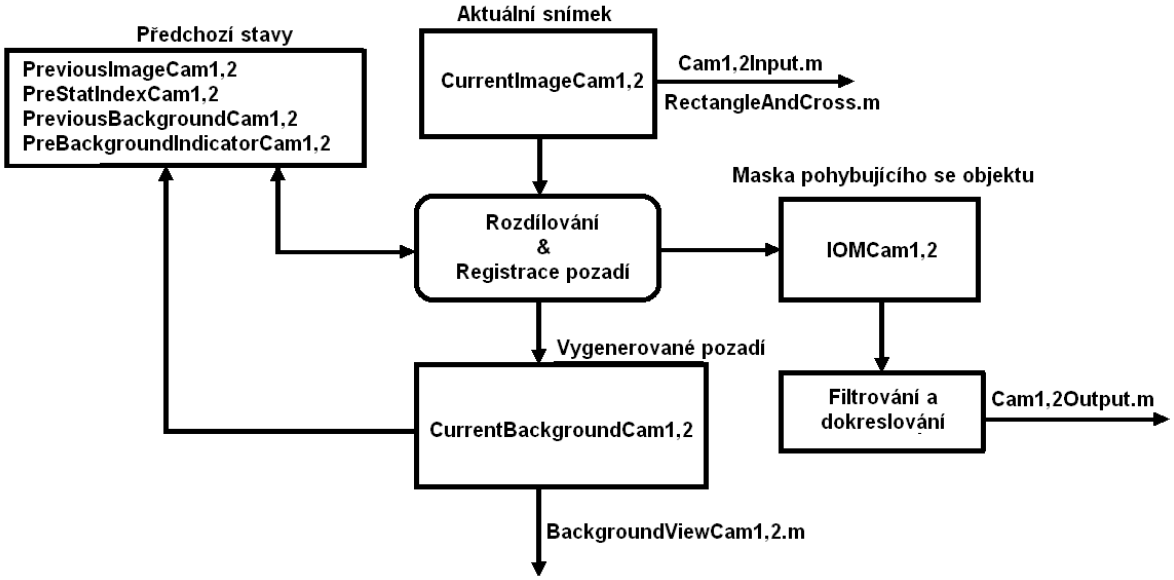
b) Druhý případ je, když se jedná o průchod cyklem jindy než poprvé. To probíhá uložení obsahu CurrentImageCam1 a CurrentImageCam2 do proměnných PreviousImageCam1 a PreviousImageCam2 a do proměnných CurrentImageCam1 a CurrentImageCam2 jsou uloženy aktuální snímky z první a druhé kamery. Podobně se ukládají aktuální obsahy následujících proměnných do proměnných uchovávajících předchozí stavy těchto proměnných.

- Obsahy CurStatIndexCam1 a CurStatIndexCam2 se ukládají do PreStatIndexCam1 a PreStatIndexCam2.
- Obsahy CurBackgroundIndicatorCam1 a CurBackgroundIndicatorCam2 se ukládají do PreBackgroundIndicatorCam1 a PreBackgroundIndicatorCam2.
- Obsahy CurrentBackgroundCam1 a CurrentBackgroundCam2 se ukládají do PrreviousBackgroundCam1 a PreviousBackgroundCam2.

Dá se říci, že proměnné označené předponou Pre (Previous) a Cur (Current) plní dohromady roli „bufferů“ (vyrovnávacích pamětí). Je v nich uchován aktuální (current) a předchozí (previous) stav.

Další část cyklu „while“ je samotná detekce pohybu ve snímané scéně s generováním pozadí. Průběh celé registrace pozadí s rozpoznáním pohybujících (rozdílování) se objektů a

následným filtrování a vykreslením objektů (obdélník kolem objektu, těžiště a střed snímané scény) je možné vidět v následujícím obrázku, který je udělaný jako blokové schéma



Obrázek 8 Blokové schéma hlavní části cyklu „while“ znázorňující průběh registrace pozadí a detekce pohybu ve snímané scéně

Průběh by se dal shrnout do čtyř bodů

- Rozdílování a prahování (anglicky thresholding)
- Registrace pozadí
- Filtrování
- Vykreslování objektů do výsledného IOMCam1,2

3.3.1.1 Rozdílování a prahování

V této části algoritmu probíhá rozdílování mezi aktuálním a předchozím snímkem a mezi aktuálním snímkem a aktuálním pozadím. Výstupem jsou jejich rozdíly, které jsou následně pomocí prahování (funkce Image Processing toolboxu im2bw s použitým prahem T = 0.2) převedené na jejich masky. Tyto operace by se dají vyjádřit pomocí následujících vztahů. Všechny uvedené názvy korespondují s názvy ve zdrojovém kódu jazyka MATLAB.

- **Rozdílování a prahování aktuálního a předchozího snímku kamer**

$$FrameDiffCam_i(x,y) = |PreviousImageCam_i(x,y) - CurrentImageCam_i(x,y)| \quad (5)$$

$$FrameDiffMaskCam_i(x,y) = \begin{cases} 1 \; if \; FrameDiffCam_i(x,y) \geq T \\ 0 \; if \; FrameDiffCam_i(x,y) < T \end{cases} \quad (6)$$

- **Rozdílování a prahování aktuálního snímku a aktuálního pozadí**

$$BackgroundDiffCam_i(x,y) = \\ = |CurrentImageCam_i(x,y) - CurrentBackgroundCam_i(x,y)| \quad (7)$$

$$BackgroundDiffMaskCam_i(x,y) = \begin{cases} 1 \; if \; BackgroundDiffCam_i(x,y) \geq T \\ 0 \; if \; BackgroundDiffCam_i(x,y) < T \end{cases} \quad (8)$$

Hodnota T = 0.2, kterou používám i v programu, znamená, že použitá hodnota pro prahování funkcí im2bw je brán 0.2 násobek maximální hodnoty obrázku ve stupních šedi. Konkrétně jde o hodnotu 0.2*255 a to je 51.

3.3.1.2 Registrace pozadí

Princip registrace pozadí je následující: vyhodnocuje se, v kolika následujících snímcích byl pixel nezměněn, to znamená, jakou dobu se jeho hodnota nezměnila tak, že by jeho pozice byla označena v rámci prahování hodnotou 1. Čím déle se pixel nemění, tím větší je pravděpodobnost, že patří do pozadí. Hodnota této doby (pravděpodobnosti) je reprezentována výše zmíněnou proměnnou nNotMoved, která je implicitně nastavena na 100. Pokud se v průběhu 100 snímků daný pixel nezmění, tak je tento pixel zařazený do pozadí. Tato hodnota se dá za chodu měnit. Čím je hodnota menší, tím dochází k rychlejšímu obnovování pozadí. A s menší hodnotou nNotMoved může být do pozadí zaregistrovaná i chvíli stojící osoba (v praxi to při měření znamenalo, že když osoba při hodnotě nNotMoved = 100 stála zhruba 3 sekundy, tak byla zahrnuta do pozadí). Naopak s větší hodnotou nNotMoved může dojít k tomu, že vůbec žádný pixel nebude zahrnut do pozadí. Zvláště pokud dojde k náhlým změnám osvětlení, apod. Registrace pozadí se dá popsat pomocí následujících tří vztáhů.

- **Vztahy pro generování matice pozadí**

$$CurStatIndexCam_i(x,y) = \begin{cases} PreStatIndexCam_i(x,y) + 1 \; if \; FrameDiffMaskCam1 = 0 \\ 0 \; if \; FrameDiffMaskCam1 <> 0 \end{cases}$$

$$CurrentBackgroundCam_i(x,y) = \begin{cases} CurrentImageCam_i(x,y) \; if \; CurStatIndexCam_i(x,y) = nNotMoved \\ PreviousBackgroundCam_i(x,y) \; if \; CurStatIndexCam_i(x,y) <> nNotMoved \end{cases} \quad (9)$$

$$CurBackgroundIndicatorCam_i(x,y) = \begin{cases} 1 \; if \; CurStatIndexCam_i(x,y) = nNotMoved \\ PreBackgroundIndicatorCam_i(x,y) \; if \; CurStatIndexCam_i(x,y) <> nNotMoved \end{cases}$$

První rovnice popisuje ukládání hodnoty do matice o velikosti obrázku, která odpovídá hodnotě počtu nezměněných snímků v rámci pozice jednoho pixelu. Pokud pixel na pozici (x,y), tj. v x-tém řádku a y-tém sloupci matice, má hodnotu 30, znamená to, že v rámci 30 snímků nebyl pixel na pozici (x,y) změněný. Druhá rovnice vyhodnocuje předešlou matici, jestli nějaký pixel nedosáhl hodnoty nNotMoved a jestli má být zařazen do pozadí nebo ne. Poslední rovnice ukládá pozice pixelů do matice, které byly zahrnuté do pozadí.

V poslední části následující algoritmu je vyhodnocení výsledného logického obrázku (IOM – Image of Moving), v kterém jsou hodnotou 1 označené pixely s pohybujícím se objektem a hodnotou 0 pixely, které neznamenávají pohybující se objekt, v našem případě osobu. Toto vyhodnocení lze popsat pomocí následující rovnice

- **Vztah pro výpočet matice s pozicemi pixelů s pohybem**

$$IOMCam_i(x,y) = \begin{cases} BackgroundDiffMaskCam_i(x,y) \; if \; CurBackgroundIndicatorCam1 = 1 \\ FrameDiffMaskCam_i(x,y) \; if \; FrameDiffMaskCam = 0 \end{cases} \quad (10)$$

Rozhoduje se v závislosti na matici CurBackgroundIndicator, což je matice obsahující pozice pixelu, kde bylo vygenerováno pozadí, pokud se na dané pozici nachází pozadí, tak je IOM přiřazená hodnota z rozdílování a prahování v rámci pozadí (Rovnice 6), pokud zde nebylo vygenerováno pozadí, tak se použije hodnota z rozdílování a prahování v rámci dvou následujících snímků (Rovnice 5).

3.4 Popis principu měření výšky pomocí dvou kamer (stereoskopické vidění)

V této kapitole se věnuji principu, jak vypočítat výšku osoby (předmětu) z obrazu snímaného dvěma kamerami. Popisuji zde rovnice výpočtu spolu s jejich odvozením. Princip zobrazuji i za pomoci obrázků.

Základem jsou dvě kamery s rektifikovanými osami snímání, umístěné ve výšce okolo 1 metru. Vycházímet z vyhodnocení obrazů kamer, kdy pomocí algoritmu pro detekci osob v obraze známe pozici těžiště osoby.

Celý princip se dá rozdělit do dvou základních kroků. V prvním kroku odvodíme výpočet vzdálenosti objektu od roviny obrazu kamer. Pro odvození budeme uvažovat pohled na scénu shora. V druhém kroku budeme odvozovat výpočet samotné výšky osoby (objektu) v metrech, kdy budeme pro odvození uvažovat pohled na scénu z boku.

3.4.1 Odvození rovnic pro výpočet vzdálenosti od kamer v závislosti na pozici osoby

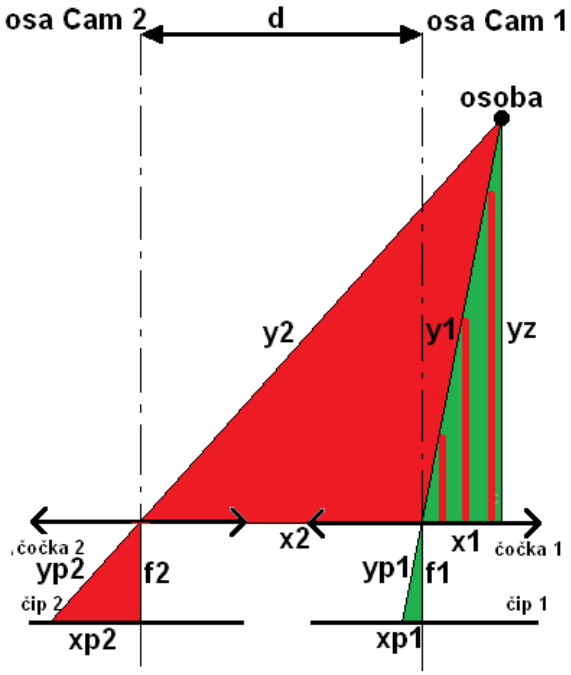
Jak už jsem uvedl výše, tak pro odvození rovnice výpočtu vzdálenosti osoby od kamer vycházíme z pohledu na scénu shora. Celkově budeme muset odvodit 5 rovnic, pro každou pozici, kde se může osoba ve snímané scéně nacházet a bude proto platit jiná rovnice výpočtu. Osobu bereme jako její těžiště promítnuté do půdorysny, tedy jako bod, a ten se může nacházet v následujících pozicích.

- **1. pozice** - Vpravo od obou kamer (vpravo na obraze z první i druhé kamery)
- **2. pozice** - Přímo před první kamerou (těžiště osoby a střed obrazu první kamery jsou totožné)
- **3. pozice** - Mezi kamerami (vlevo na obraze z první kamery a vpravo na obraze druhé kamery)
- **4. pozice** - Přímo před druhou kamerou (těžiště osoby a střed obrazu druhé kamery jsou totožné)
- **5. pozice** - Vlevo od obou kamer (vlevo na obraze z první i druhé kamery)

Každé odvození je odvozené z podobnosti dvou pravoúhlých trojúhelníků. Všechny možné pozice osoby před kamerami spolu se znázorněním trojúhelníků pro dané pozice,

vidíme na následujících obrázcích v rámci podkapitol 3.4.1.1 – 3.4.1.5. Popisuji zde odvození rovnic výpočtu vzdálenosti osoby od kamer pro každou pozici.

3.4.1.1 Pozice 1 – vpravo od obou kamer



Obrázek 9 První pozice osoby při pohledu shora

Seznam proměnných:

y_z – kolmá vzdálenost těžiště objektu od kamer

y_1 – vzdálenost od první kamery

y_2 – vzdálenost od druhé kamery

x_1 – horizontální vzdálenost od první kamery

x_2 – horizontální vzdálenost od druhé kamery

f_1 – vzdálenost čipu (roviny obrazu) od čočky první kamery

f_2 – vzdálenost čipu (roviny obrazu) od čočky druhé kamery

xp_1 – vzdálenost těžiště objektu v obraze první kamery v pixelech

xp_2 – vzdálenost těžiště objektu v obraze druhé kamery v pixelech

yp_1 – vzdálenost od středu čočky k bodu pixelu těžiště na čipu první kamery

yp_2 – vzdálenost od středu čočky k bodu pixelu těžiště na čipu druhé kamery

Z podobnosti trojúhelníků pro pozici 1 vyplývá následující soustava dvou rovnic o dvou neznámých

$$I.\ \frac{y_z}{x_1} = \frac{f_1}{xp_1}$$
$$II.\ \frac{y_z}{x_2} = \frac{f_2}{xp_2} \tag{11}$$

Protože uvažujeme dvě totožné kamery, tak můžeme říct, že $f_1 = f_2 = f$, dále můžeme říct, že platí vztah $x_2 = x_1 + d$. Pokud tyto dva vztahy dosadíme do rovnic a z první rovnice vyjádříme x_1

$$x_1 = \frac{y_z \cdot xp_1}{f} \tag{12}$$

a dosadíme vyjádřené x_1 do druhé rovnice, tak dostaneme rovnici o jedné neznámé, kde neznámou je y_z. Po vyjádření y_z dostaneme následující rovnici

$$\frac{y_z}{\frac{y_z \cdot xp_1}{f} + d} = \frac{f}{xp_2} \rightarrow y_z = \frac{f}{xp_2} \cdot \left(\frac{y_z \cdot xp_1}{f} + d\right) \rightarrow y_z = \frac{y_z \cdot xp_1}{xp_2} + \frac{f \cdot d}{xp_2} \rightarrow y_z - \frac{y_z \cdot xp_1}{xp_2} =$$
$$= \frac{f \cdot d}{xp_2} \rightarrow y_z \cdot \left(1 - \frac{xp_1}{xp_2}\right) = \frac{f \cdot d}{xp_2} \rightarrow y_z = \frac{d \cdot f}{xp_2 - xp_1} \tag{13}$$

po dosazení y_z do Rovnice 12 dostaneme následující rovnici pro výpočet x_1

$$x_1 = \frac{y_z \cdot xp_1}{f} \rightarrow x_1 = \frac{\frac{d \cdot f}{xp_2 - xp_1} \cdot xp_1}{f} \rightarrow x_1 = \frac{d \cdot xp_1}{xp_2 - xp_1} \tag{14}$$

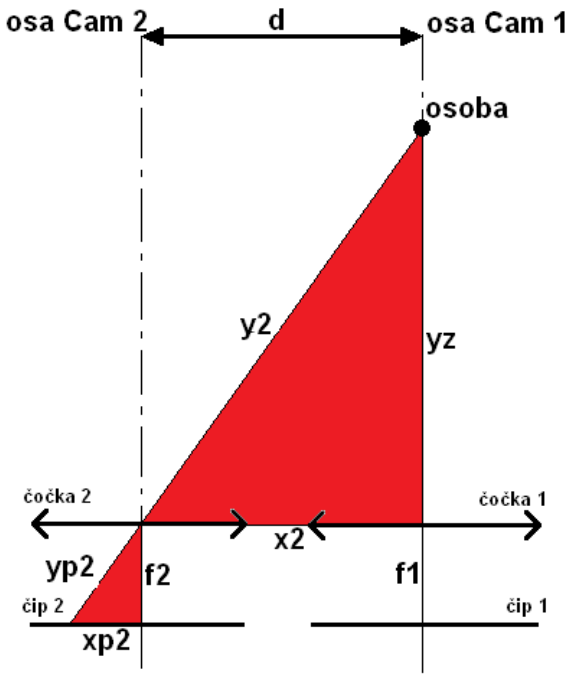
Vztah pro výpočet vzdálenosti osoby od první kamery, kterou budeme potřebovat pro výpočet výsledné výšky osoby (objektu) je následující

$$y_1 = \sqrt{{x_1}^2 + {y_z}^2} = \sqrt{\left(\frac{d \cdot xp_1}{xp_2 - xp_1}\right)^2 + \left(\frac{d \cdot f}{xp_2 - xp_1}\right)^2} = \sqrt{\frac{d^2}{(xp_2 - xp_1)^2} \cdot \left(f^2 + {xp_1}^2\right)} = \\ = \frac{d}{(xp_2 - xp_1)} \cdot \sqrt{f^2 + {xp_1}^2} \tag{15}$$

Rovnice pro výpočet vzdálenosti středu čočky od bodu těžiště na čipu první kamery yp_1 je následující

$$yp_1 = \sqrt{f^2 + {xp_1}^2} \tag{16}$$

3.4.1.2 Pozice 2 – přímo před první kamerou

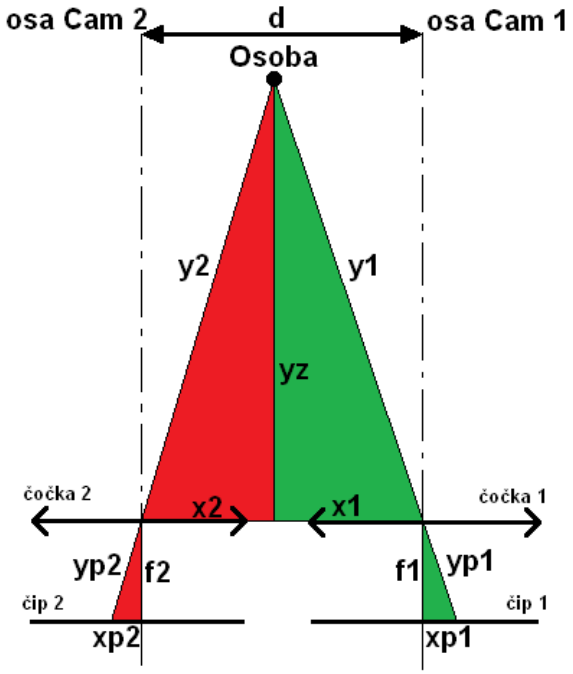


Obrázek 10 Druhá pozice osoby při pohledu shora

Vztah pro výpočet vzdálenosti objektu od druhé kamery y_2 lze odvodit z následující podobnosti v rámci trojúhelníků pro pozici 2

$$\frac{y_2}{d} = \frac{yp_2}{xp_2} \rightarrow y_2 = \frac{d \cdot yp_2}{xp_2} \tag{17}$$

3.4.1.3 Pozice 3 – mezi kamerami



Obrázek 11 Třetí pozice osoby při pohledu shora

Vzdálenosti obrazové roviny obou kamer se opět rovnají, proto budeme uvažovat vzdálenosti obrazové roviny první kamery f_1 a vzdálenosti obrazové roviny druhé kamery f_2 pouze jako jednotnou vzdálenost obrazové roviny f. Horizontální vzdálenost objektu od druhé kamery můžeme vyjádřit následovně $x_2 = d - x_1$, po dosazení tohoto vyjádření x_1 do soustavy rovnic vyplývající z podobnosti trojúhelníků pro třetí pozici dostaneme následující soustavu rovnic o dvou neznámých

$$I.\ \frac{y_z}{x_1} = \frac{f}{xp_1}$$
$$II.\ \frac{y_z}{d - x_1} = \frac{f}{xp_2} \tag{18}$$

Po vyjádření horizontální vzdálenosti od první kamery x_1 z první rovnice soustavy a po dosazení do druhé rovnice soustavy a po upravení dostaneme následující vztah pro výpočet kolmé vzdálenosti objektu od kamer y_z

$$y_z = \frac{d \cdot f}{xp_1 + xp_2} \tag{19}$$

Po dosazení y_z do vyjádřeného vztahu z první rovnice soustavy pro výpočet x_1 a následných úpravách, dostaneme následující rovnici vztahu pro výpočet x_1

$$x_1 = \frac{d \cdot xp_1}{xp_1 + xp_2} \tag{20}$$

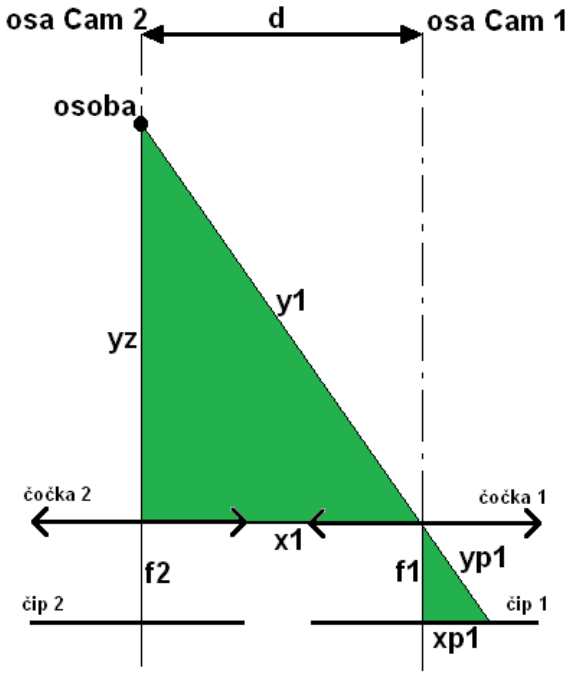
Z Pythagorovy věty a dosazením y_z a x_1 dostaneme následující rovnici vztahu pro výpočet vzdálenosti od druhé kamery y_2 pro pozici 3

$$y_2 = \frac{d}{(xp_1 + xp_2)} \cdot \sqrt{f^2 + {xp_2}^2} \tag{21}$$

Z Pythagorovy věty a dosazením f a xp_2 dostaneme následující rovnici vztahu pro výpočet vzdálenosti středu čočky druhé od bodu těžiště na čipu druhé kamery yp_2 v rámci pozice 3

$$yp_2 = \sqrt{f^2 + {xp_2}^2} \tag{22}$$

3.4.1.4 Pozice 4 – přímo před druhou kamerou

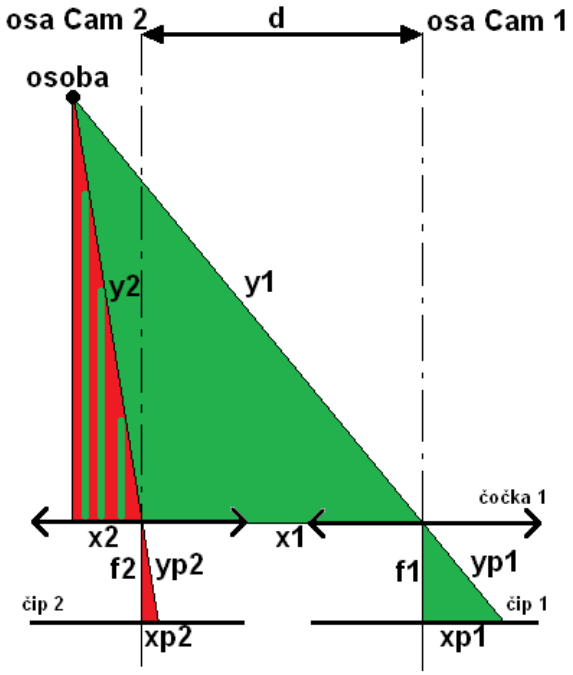


Obrázek 12 Čtvrtá pozice osoby při pohledu shora

Vztah pro výpočet vzdálenosti objektu od první kamery y_1 lze odvodit z následující podobnosti v rámci trojúhelníků pro pozici 4

$$\frac{y_1}{d} = \frac{yp_1}{xp_1} \rightarrow y_1 = \frac{d \cdot yp_1}{xp_1} \tag{23}$$

3.4.1.5 Pozice 5 – vlevo od obou kamer



Obrázek 13 Pátá pozice osoby při pohledu shora

Soustava rovnic vyplývající z podobnosti trojúhelníků v rámci páté pozice osoby před kamerami vypadá následovně

$$I.\ \frac{y_z}{x_2} = \frac{f}{xp_2}$$
$$II.\ \frac{y_z}{x_2 + d} = \frac{f}{xp_1} \tag{24}$$

Po vyjádření horizontální vzdálenosti od druhé kamery x_2 z první rovnice soustavy a po dosazení do druhé rovnice soustavy a po upravení dostaneme následující vztah pro výpočet kolmé vzdálenosti objektu od kamer y_z

$$y_z = \frac{d \cdot f}{xp_1 - xp_2} \tag{25}$$

Po dosazení y_z do vyjádřeného vztahu z první rovnice soustavy pro výpočet x_2 a následných úpravách, dostaneme následující rovnici vztahu pro výpočet x_2

$$x_2 = \frac{d \cdot xp_2}{xp_1 - xp_2} \tag{26}$$

Z Pythagorovy věty a dosazením y_z a x_2 dostaneme následující rovnici vztahu pro výpočet vzdálenosti od druhé kamery y_2 pro pozici 5

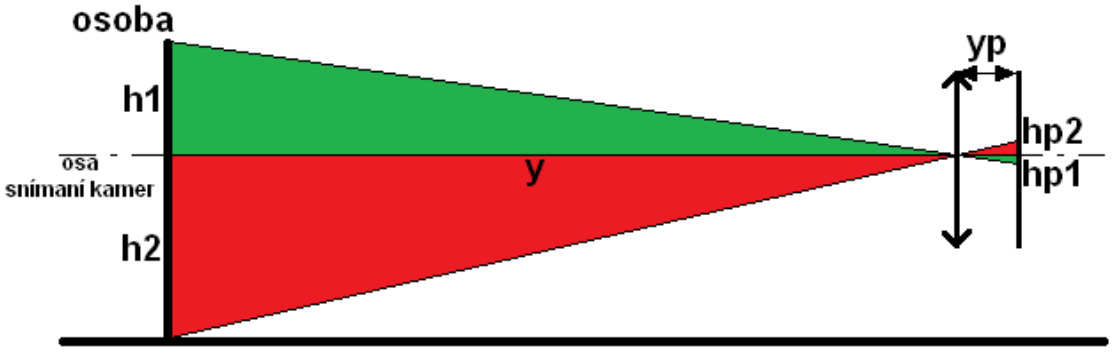
$$y_2 = \frac{d}{(xp_1 - xp_2)} \cdot \sqrt{f^2 + {xp_2}^2} \tag{27}$$

Z Pythagorovy věty a dosazením f a xp_2 dostaneme následující rovnici vztahu pro výpočet vzdálenosti středu čočky druhé od bodu těžiště na čipu druhé kamery yp_2 v rámci pozice 5

$$yp_2 = \sqrt{f^2 + {xp_2}^2} \tag{28}$$

3.4.2 Odvození rovnic pro výpočet výšky osoby v závislosti na poloze osoby

Pro odvození vztahu pro výpočet výšky osoby musíme opět vzít v potaz všechny pozice osob před kamerami. Jak bylo uvedeno na začátku kapitoly 3.4, tak k odvození vztahů využíváme pohled z boku. Jako osobu bereme úsečku procházející těžištěm objektu. Tento pohled s nanesením potřebných veličin je vidět na následujícím obrázku



Obrázek 14 Pohled z boku

Seznam proměnných:

y – udává vzdálenost objektu od kamery (první nebo druhé)

yp – udává vzdálenost od středu čočky kamery (první nebo druhé) od bodu na čipu udávajícím těžiště objektu (stejná se vzdáleností yp_2 nebo yp_1 z předchozího odvozování)

h_1 – udává výšku osoby nad osou snímání v metrech

h_2 – udává výšku osoby pod osou snímání v metrech

hp_1 – udává výšku osoby v obraze v pixelech korespondující s výškou h_1

hp_2 – udává výšku osoby v obraze v pixelech korespondující s výškou h_2

Nejprve vyjádříme z podobnosti trojúhelníků obecný vztah pro výpočet výšky osoby, dle obrázku 14. Předpokládejme, že $h = h_1 + h_2$ a *hp* (které zjistíme v rámci detekce osoby v obraze) se bude rovnat $hp_1 + hp_2$. Vztahy pro výpočet h_1 *a* h_2 z podobnosti trojúhelníků mají následující tvar.

$$\frac{y}{h_1} = \frac{yp}{hp_1} \rightarrow h_1 = \frac{y \cdot hp_1}{yp} \tag{29}$$

$$\frac{y}{h_2} = \frac{yp}{hp_2} \rightarrow h_2 = \frac{y \cdot hp_2}{yp} \tag{30}$$

Pro získání vztahu pro výpočet *h* předchozí vztahy vzájemně sečteme. Po úpravě získáme následující vztah pro výpočet *h*.

$$h = h_1 + h_2 = \frac{y \cdot hp_1}{yp} + \frac{y \cdot hp_2}{yp} = \frac{y}{yp} \cdot (hp_1 + hp_2) = \frac{y \cdot hp}{yp} \tag{31}$$

Nyní dosadíme postupně pro každou pozici objektu do předešlého vztahu pro výpočet výšky *h*. Za *y* a *yp* budeme dosazovat vztahy odvozené z kapitoly 3.4.1. Po dosazení budou, vztahy pro výpočet výšky osoby v rámci každé z pěti možných pozic, vypadat následovně.

1. Pro první pozici osoby (vpravo od obou kamer)

   Do Rovnice 31 dosadíme za *y* vyjádření y_1 z Rovnice 15 a za *yp* dosadíme vyjádření yp_1 z rovnice 16. Po úpravách bude vypadat vztah pro výpočet výšky osoby v rámci první pozice následovně

$$h = \frac{y \cdot hp}{yp} = \frac{\frac{d \cdot hp}{(xp_2 - xp_1)} \cdot \sqrt{f^2 + {xp_1}^2}}{\sqrt{f^2 + {xp_1}^2}} = \frac{d \cdot hp}{(xp_2 - xp_1)} \tag{32}$$

2. Pro druhou pozici osoby (přímo před první kamerou)

   Do Rovnice 31 dosadíme za *y* vyjádření y_2 z Rovnice 17 a za *yp* dosadíme yp_2. Po úpravách bude vypadat vztah pro výpočet výšky osoby v rámci druhé pozice následovně

$$h = \frac{y \cdot hp}{yp} = \frac{\frac{d \cdot yp_2}{xp_2}}{yp_2} = \frac{d}{xp_2} \tag{33}$$

3. Pro třetí pozici osoby (mezi oběma kamerami, tj. vlevo od první a vpravo od druhé)

   Do Rovnice 31 dosadíme za *y* vyjádření y_2 z Rovnice 21 a za *yp* dosadíme vyjádření yp_2 z rovnice 22. Po úpravách bude vypadat vztah pro výpočet výšky osoby v rámci třetí pozice následovně

$$h = \frac{y \cdot hp}{yp} = \frac{\frac{d \cdot hp}{(xp_1 + xp_2)} \cdot \sqrt{f^2 + {xp_2}^2}}{\sqrt{f^2 + {xp_2}^2}} = \frac{d \cdot hp}{(xp_1 + xp_2)} \tag{34}$$

4. Pro čtvrtou pozici osoby (přímo před druhou kamerou)

   Do Rovnice 31 dosadíme za *y* vyjádření y_1 z Rovnice 23 a za *yp* dosadíme yp_1. Po úpravách bude vypadat vztah pro výpočet výšky osoby v rámci čtvrté pozice následovně

$$h = \frac{y \cdot hp}{yp} = \frac{\frac{d \cdot yp_1}{xp_1}}{yp_1} = \frac{d}{xp_1} \tag{35}$$

5. Pro pátou pozici osoby (vlevo od obou kamer)

   Do Rovnice 31 dosadíme za *y* vyjádření y_2 z Rovnice 27 a za *yp* dosadíme vyjádření yp_2 z rovnice 28. Po úpravách bude vypadat vztah pro výpočet výšky osoby v rámci pátou pozice následovně

$$h = \frac{y \cdot hp}{yp} = \frac{\frac{d \cdot hp}{(xp_1 - xp_2)} \cdot \sqrt{f^2 + {xp_2}^2}}{\sqrt{f^2 + {xp_2}^2}} = \frac{d \cdot hp}{(xp_1 - xp_2)} \tag{33}$$

Pokud se na odvozené vztahy pro výpočet podíváme podrobněji, tak zjistíme, že vztah pro výpočet výšky je převodním vztahem pro míru velikosti v určité vzdálenosti dané v pixelech. Proto bychom mohli tímto vztahem počítat jakýkoliv rozměr v pixelech (šířku i výšku), pokud bychom si byli jisti, že všechny body v obraze jsou ve stejné vzdálenosti.

3.5 Rektifikace os kamer

V této části se věnuji a popisuji průběh rektifikace os kamer a výrobě konstrukce pro držení kamer k měření. Rektifikace os kamer je důležitá z důvodů měření výšky osob, protože se při počítání vychází z předpokladu, že jsou osy kamer rovnoběžné. Základem rektifikace bylo sestrojení konstrukce pro držení kamer.

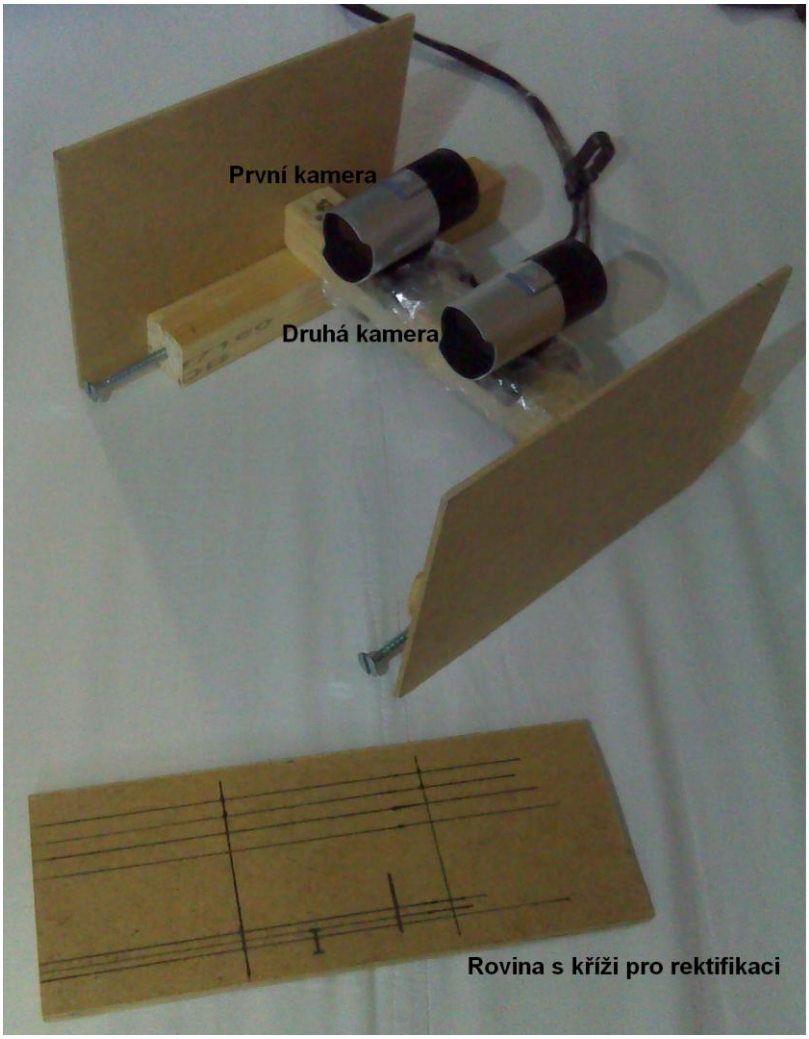
3.5.1 Konstrukce pro držení kamer

Konstrukce pro držení kamer sestává z následujících částí

- Dvě webové kamery
- Dva samořezné šrouby
- Překližka
- Dřevěný hranol
- Spojovací materiál (šrouby, hřebíky, lepicí páska)

3.5.1.1 Postup výroby konstrukce pro držení kamer

Z dřevěného hranolu jsem jeho rozřezáním na tři části a složením do písmene H vytvořil podstavec pro připevnění kamer. Podstavec jsem pomocí vodováhy vyrovnal do roviny tak, aby se obě dvě kamery nacházely ve stejné výšce. Potom jsem obě dvě kamery na tento podstavec zafixoval pomocí lepicí pásky ve vzdálenosti 6,5 cm od sebe. V předu jsem do podstavce našrouboval dva samořezné šrouby tak, aby jejich hlavičky byly ve stejné vzdálenosti od podstavy kamer, protože budou sloužit pro umístění roviny s kříži pro rektifikaci středů obrazů z kamer, tj. zaměření středů obrazů kamer na odpovídající kříž, kdy oba dva kříže budou od sebe vzdáleny stejně, jako kamery, tj. 6,5 cm. Z překližky jsem vyřezal dva a dva stejné kusy. První dva slouží k zajištění roviny s dvěma kříži a další dva slouží k nakreslení křížů. Roviny s kříži jsem byly k sobě slepeny z důvodu zajištění jejich rovnosti. První kříž je na plochu nakreslený ve vzdálenosti první kamery od plochy na straně vedle kamery. Druhý kříž je nakreslený od prvního kříže ve vzdálenosti 6,5 cm, stejně jako je vzdálenost kamer.



Obrázek 15 Konstrukce pro držení kamer spolu s rovinou s kříži

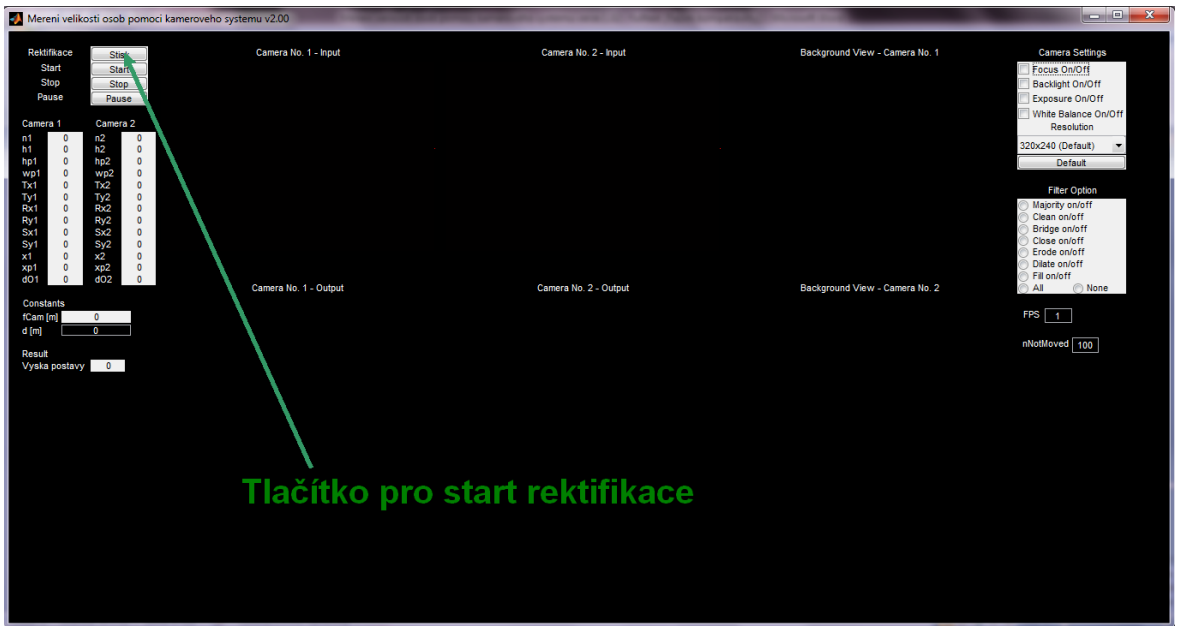
3.5.1.2 Příprava na průběh rektifikace v rámci aplikace pro měření

Rovina s kříži pro rektifikaci se umístí do konstrukce pro měření tak, že její spodní hrana umístí těsně před hlavičky šroubů a její hrany se vyrovnají s hranami bočních překližek. Tím se docílí toho, že kříže a plocha budou ve stejné vzdálenosti od obou kamer. Nastavení je znázorněno na obrázku 16.

Obrázek 16 Zarovnání roviny s kříži

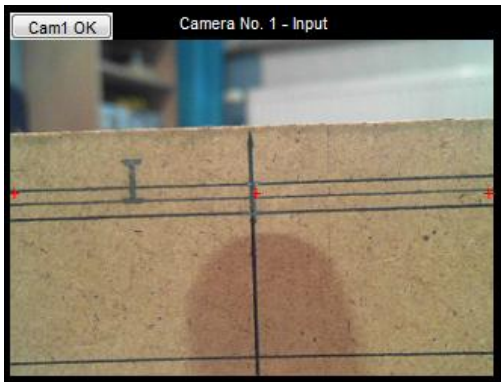
3.5.1.3 Průběh rektifikace v rámci aplikace

Pro začátek rektifikace je nutné v aplikaci stisknout v levém horním rohu tlačítko Rektifikace, viz. obrázek 17. Po stisknutí tlačítka Rektifikace se spustí část zdrojového kódu z funkce CheckBoxesAndClicks.m, která je obsluhou „callbacku“ stisku tohoto tlačítka. Rektifikace probíhá pro každou kameru zvlášť. Nastavení každé kamery se odklepne daným tlačítkem, Cam1 OK a Cam2 OK. Tlačítka pro potvrzení rektifikace jednotlivých kamer jsou viditelné pouze při rektifikaci dané kamery. Postup je viditelný z následujících obrázků 10, 11, 12, 13 a 14.

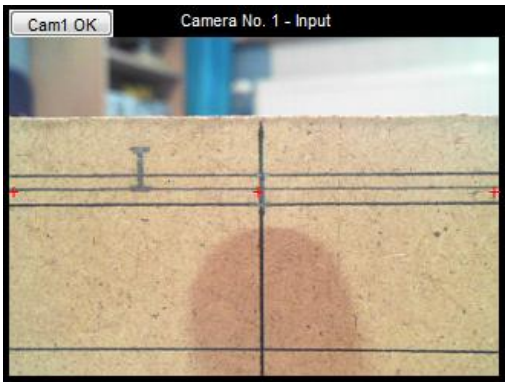


Obrázek 17 Rektifikace – 1. krok

Následuje vyrovnání první kamery, kde obraz z první kamery je vykreslován v části aplikace „Camera No. 1 – Input“ pomocí funkce Cam1Input.m. V rámci vyhodnocení „callbacku“ dojde k zaostření kamery na hodnotu 32. Hodnota 32 odpovídá zaostření ve vzdálenosti roviny s kříži. Jakmile střed obrazu z první kamery zaměříme na první kříž, tak se klikne na tlačítko Cam1 OK, které bude vykreslené nad obrazem z první kamery. Vše můžeme vidět na následujících obrázcích 18 a 19. Pro usnadnění rektifikace jsou do obrazu vykreslené tři červené křížky, které jsou umístěné v jedné přímce na středu a okrajích obrazu z kamery. Právě tyto křížky se snažíme manuálním nastavováním kamery vyrovnat do jedné roviny a střední křížek nasměrovat na kříž na rovině.

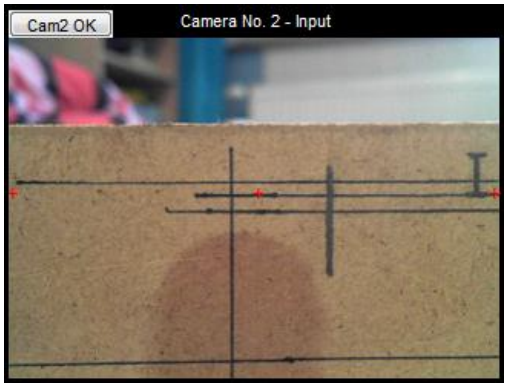


Obrázek 18 Kamera 1 před rektifikací

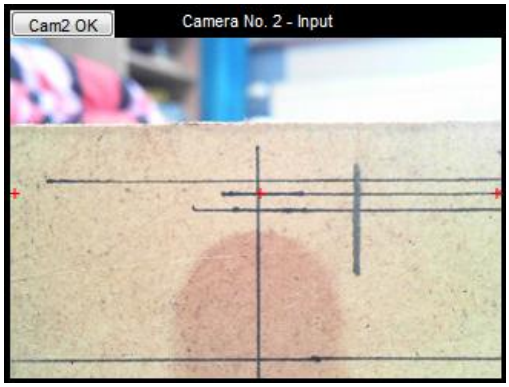


Obrázek 19 Kamera 1 po rektifikaci

Po stisknutí Cam1 OK dojde k vyobrazení obrazu z druhé kamery v části GUI určené pro zobrazování snímků z druhé kamery označeném jako „Camera No. 2 – Input“. Opět dojde k vykreslení pomocných tří červených křížků. Kameru stejně jako předtím vyrovnáme a klikneme na tlačítko Cam2 OK. Tím je rektifikace ukončena a můžeme začít měřit. Důležité je zarovnání křížků druhé kamery na stejnou přímku na rovině s kříži. Rektifikace druhé kamery je znázorněna na obrázcích 20 a 21.



Obrázek 20 Kamera 2 před rektifikací



Obrázek 21 Kamera 2 po rektifikaci

4 ZHODNOCENÍ PRŮBĚHU REALIZACE A VÝSLEDKŮ EXPERIMENTU, NÁVRH ÚPRAV

V této kapitole praktické části bakalářské práce píši o poznatcích a problémech, s kterými jsem se při realizaci experimentu setkal. Dále shrnu výsledky samotného měření. Nakonec navrhnu možné úpravy a vylepšení do budoucna.

4.1 Problémy při realizaci aplikace

Při realizaci aplikace pro měření výšky osob jsem narazil na spoustu problémů a překážek, s kterými jsme se musel vypořádat. Ve většině případů se mi to podařilo pomocí prostudování doporučené literatury nebo webových stránek http://www.mathworks.com/help, kde byly uvedené rady ohledně jednotlivých částí prostředí MATLAB a vzorové příklady fungování jednotlivých funkcí, apod.

Zdrojové kódy navržené aplikace pro měření výšky osob (soubory s příponou .m) jsou ke shlédnutí na přiloženém CD.

4.1.1 Použití webových kamer

Webové kamery od společnosti Microsoft, které jsem využil k experimentu, jsou po technické stránce velice vyspělé. Kamera disponuje speciálními funkcemi, jako například automatické ostření obrazu, automatická úprava barev, automatická korekce osvětlení a jasu. To je ale pro samotné měření spíše překážkou než výhodou, protože všechny tyto funkce vedou k náhlým změnám v obraze a to vede k falešným detekcím pohybujících se objektů. Proto bylo nutné nejprve všechny tyto speciální funkce na začátku běhu programu zakázat.

Kamera je schopná snímat ve velmi vysokém rozlišení, ale za předpokladu připojení k výkonnému počítači. Samotná kamera je schopná snímat obraz ve Full HD rychlostí 30 snímků za sekundu, při připojení jedné kamery a bez úpravy obrazu bylo toto možné, ale po připojení druhé kamery a aplikaci vyhodnocování obrazu pomocí prostředí MATLAB, bylo této rychlosti snímání prakticky nemožné dosáhnout. Při testování doma, jsem narazil i na systémovou chybu v podobě nedostatku prostředků pro vyhodnocení operace.

4.1.2 Prostředí MATLAB

Realizace aplikace pro měření výšky osob šla v prostředí MATLAB poměrně dobře. Místy jsem ovšem narazil na problém, kdy prostředí MATLAB bylo nečekaně ukončeno. Běh záznamu videa pomocí oficiálního softwarového prostředku k webové kameře od firmy Microsoft byl plynulý a nepřerušovaný. S použitím prostředí MATLAB ke zpracování videa z webových kamer docházelo k chybám v obrazu (špatné barevné spektrum).

Protože MATLAB je navržen jako skriptovací jazyk, tak jsem narazil na problém ve strukturalizaci programu (rozdělení do funkcí). Problémy díky strukturalizaci jsem měl zejména v uchovávání a zacházení s globálními proměnnými, ale tyto problémy se mi, při podrobnějším prostudování programovacího jazyka v rámci prostředí MATLAB, podařilo odstranit.

Návrh prostředí GUI bylo celkem přímočaré a intuitivní. S jeho návrhem a programováním, jsem díky doporučené literatuře, neměl potíže.

4.2 Podmínky experimentu

Pro samotný experiment bylo zvoleno co nejmenší rozlišení (320x240 bodů) snímaného obrazu kamerami, aby bylo docíleno co největší rychlosti snímání. Musíme totiž brát ohled na to, že nezobrazujeme pouze jeden snímek, ale celkově při každém průchodu hlavním cyklem aplikace, musíme zobrazit 4 snímky z kamer (dva neupravené, dva upravené). K tomu se aktualizuje vygenerované pozadí. Navíc každý průchod cyklem znamená vyhodnocování obrazu z obou kamer, registraci pozadí a to celý běh zpomaluje. Z těchto důvodu bylo nastaveno vykreslování obrazu z kamer na obrazu tak, aby se vykresloval pouze každý 10. zpracovaný snímek pro urychlení vykonávání programu, protože vykreslování snímků z kamer na obrazovku bylo nejdéle trvající operací při běhu programu. Pro experiment bylo použito prostor laboratoře C304 v budově U5 na Jižních Svazích. Soustava kamer byla umístěna ve výšce 1 m a osy byly rektifikovány pomoci navržené části aplikace pro rektifikaci a roviny s kříži.

4.3 Nastavení algoritmu detekce osob v obraze

Při testování algoritmu pro detekci osoby v obraze a následnému výpočtu její výšky bylo nejprve správně navrhnout hodnotu proměnné nNotMoved pro registraci pozadí. Pokud jsme zvolili hodnotu od 20 do 75, tak do pozadí byla zahrnuta velmi rychle i osoba samotná. Pokud jsme nastavili hodnotu příliš vysokou, 200 a více, tak k zahrnutí pixelů do pozadí nikdy nedošlo. Po několika pokusech se jako optimální ukázala hodnota 125, kdy došlo poměrně rychle k registraci pozadí a už bylo těžší, aby byla osoba zahrnuta do pozadí, i když to nebylo nemožné, protože kdybychom brali rychlost snímkování kamerou 30 snímků za sekundu, tak by k registraci osoby do pozadí došlo zhruba za 4-5 sekund.

Dále bylo důležité rozhodnout použití filtrů obrazu v rámci detekce pohybu. Ukázalo se vhodné používat alespoň filtry erode a dilate, i když se mírně prodloužila prodleva mezi jednotlivými zpracováními snímků z kamer. Použití dalších filtrů se ukázalo být zbytečným, vedlo to pouze k prodloužení chodu cyklu s vyhodnocením obrazu z kamer.

Při delším chodu měření se počet vyhodnocených snímků za sekundu zmenšoval, pravděpodobně to bylo způsobeno chodem prostředí MATLAB a neuvolňování paměti s pořízeními snímky.

4.3.1 Výsledky detekce osob a registrace pozadí

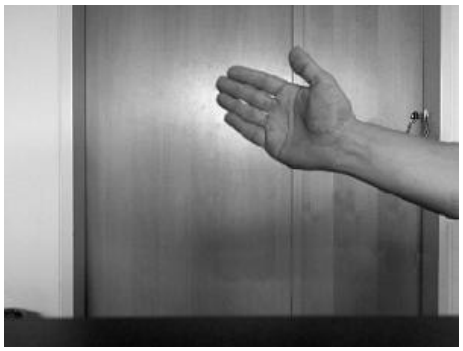
Registrace, jak je možné vidět na následujících obrázcích, probíhala bez pohybu. Uvádím zde obrázky registrace pozadí s pohybem a bez pohybu ve scéně po 100 snímích.

- **Scéna bez pohybu**

Obrázek 22 Scéna bez pohybu

Obrázek 23 Registrované pozadí

- **Scéna s pohybem**

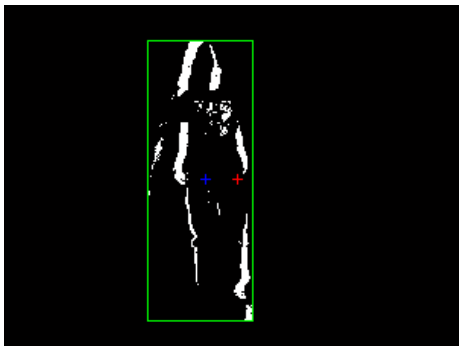


Obrázek 24 Scéna s pohybem

Obrázek 25 Registrované pozadí

Rozpoznání osoby ze snímků kamer byl těžší úkol, protože ne vždy byla osoba nalezena správně, zvlášť pokud k měření bylo důležité mít co nejpřesnější detekci, kdy v měření hráli roli opravdu pixely. Ukázka detekce navrženým programem

- **Detekce osoby**



Obrázek 26 Maska detekce pohybu

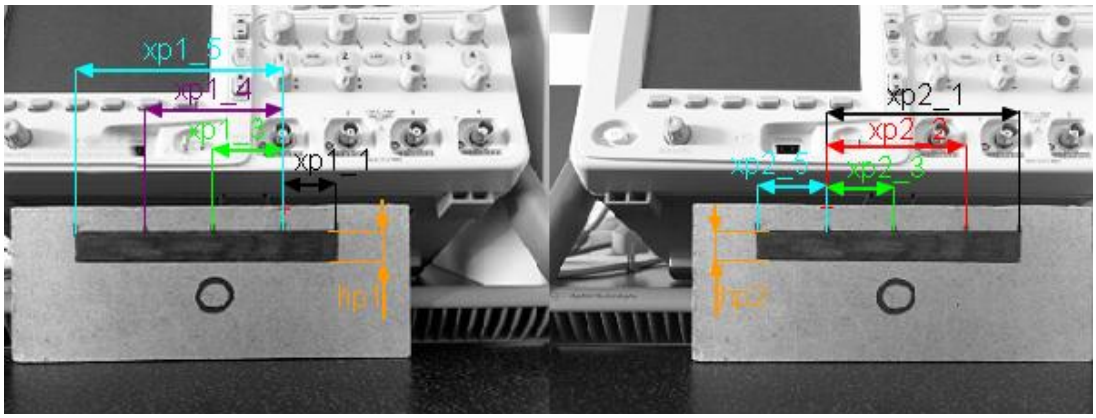
Obrázek 27 Výsledek v obraze

4.4 Výsledky experimentu měření výšky osoby (objektu)

Při měření jsme narazili na jeden velký problém, díky kterému docházelo k velkým chybám v měření pohybující se osoby. Zmíněný největší problém při měření výšky pohybujících se osob byl ten, že detekce osoby nebyla moc korektní. Docházelo k velkým odchylkám v detekci osob a výsledné pozici těžiště detekované osoby. Při testování měření na statickém objektu fungovala aplikace o poznání lépe, kdy odchylky od skutečné výšky objektu nebyly tak veliké.

4.4.1 Testování měření výšky statického objektu

Při testování měření výšky statického objektu, kdy bylo prevedeno určování výšky a rozměrů v pixelech manuálně, byla ověřena správnost odvození rovnic pro výpočet výšky osoby (objektu) za použití dvou kamer. Pro tuto kontrolu byly využity následující snímky z kamer, viz. následující obrázek



Obrázek 28 Snímky pro kontrolu správnosti odvozených rovnic

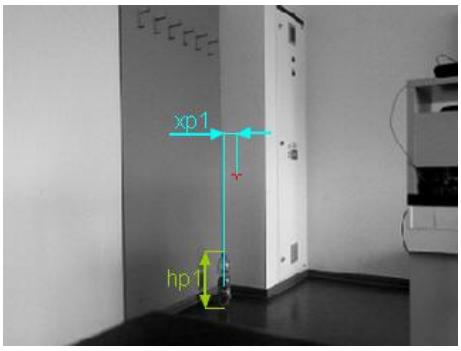
Obrázek je spojením snímku z první kamery (levá polovina) a snímku z druhé kamery (prvá polovina). Pomocí kamer byl zaznamenán obdélník o pevné výšce 1,5cm. Nad obdélník byly nakreslené čáry, kde každá je v jedné z pěti možných pozic. Podle každé pozice můžeme otestovat jednu rovnici.

Každý zakreslený rozměr v obrázku znamená jeden referenční rozměr pro otestování rovnice, kdy číslice za středníkem znamená číslo udávající pozici, kterému rozměr odpovídá. Například *xp1_1* a *xp2_1* jsou hodnoty pro rovnici objektu v pozici 1 před kamerami (objekt je vlevo od obou kamer). Z tabulky si můžeme všimnout, že všechny výpočty pro pozici předmětu (osoby) byly stanovené správně a z vypočtené nejistoty měření můžeme říci, že přesnost měření byla v řádu desetin milimetru, kdy změna xp_1 nebo xp_2 v řádu 1 pixelu znamená změnu právě o 0,0002 m.

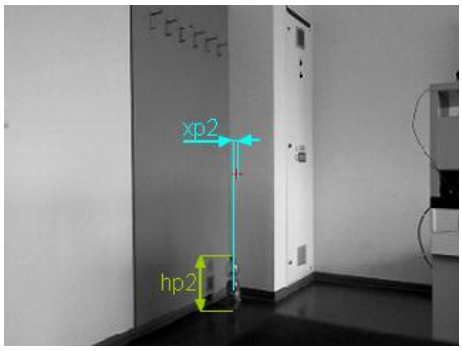
Tabulka 2 Výpočet výšky statického objektu pro všechny pozice

pozice	xp_1 [px]	xp_2 [px]	hp_1 [px]	hp_2 [px]	hp [px]	d [m]	$h_{reálná}$ [m]	$h_{naměřená}$ [m]
1	30	107	18	17	17,5	0,065	0,015	0,0148
2	0	78	18	17	17,5	0,065	0,015	0,0146
3	40	38	18	17	17,5	0,065	0,015	0,0146
4	77	0	18	17	17,5	0,065	0,015	0,0148
5	115	39	18	17	17,5	0,065	0,015	0,0150
$h_{naměřená\ průměrná}$								0,0148
nejistota měření								0,000150

Další test probíhal měřením výšky plastové láhvi, která byla umístěna do pěti různých vzdáleností. Láhev byla následně nasnímána kamerami a programem na úpravu obrázků byla změřena jednotlivých velikostí láhve v pixelech. Postup zjištění potřebných rozměrů, které probíhalo pomoci programu PhotoFiltre, je znázorněn na následujících obrázcích. Rozměry snímků byly přesně 320x240 pixelů jako snímací rozlišení kamer.



Obrázek 29 Rozměry láhve – Kamera 1



Obrázek 30 Rozměry láhve – Kamera 2

Následně byly zjištěné rozměry objektu (láhve) hp_1 a hp_2 zprůměrovány a spolu s xp_1 a xp_2 dosazeny do Rovnice 33, jelikož byla umístěna do pozice 5 (vlevo od obou kamer). Výsledky jsou měření zapsány v následující tabulce.

Tabulka 3 Výpočet rozměrů láhve

vzdálenost [cm]	xp_1 [px]	xp_2 [px]	hp_1 [px]	hp_2 [px]	hp [px]	$h_{skutečná}$ [m]	$h_{naměřená}$ [m]
60	93	54	209	209	209	0,34	0,348
120	87	70	91	91	91	0,34	0,347
180	64	53	60	60	60	0,34	0,354
240	43	34	48	48	48	0,34	0,348
300	10	3	39	39	38,5	0,34	0,357
$h_{průměrná\ naměřená}$ [m]							0,351
nejistota měření [m]							0,00397

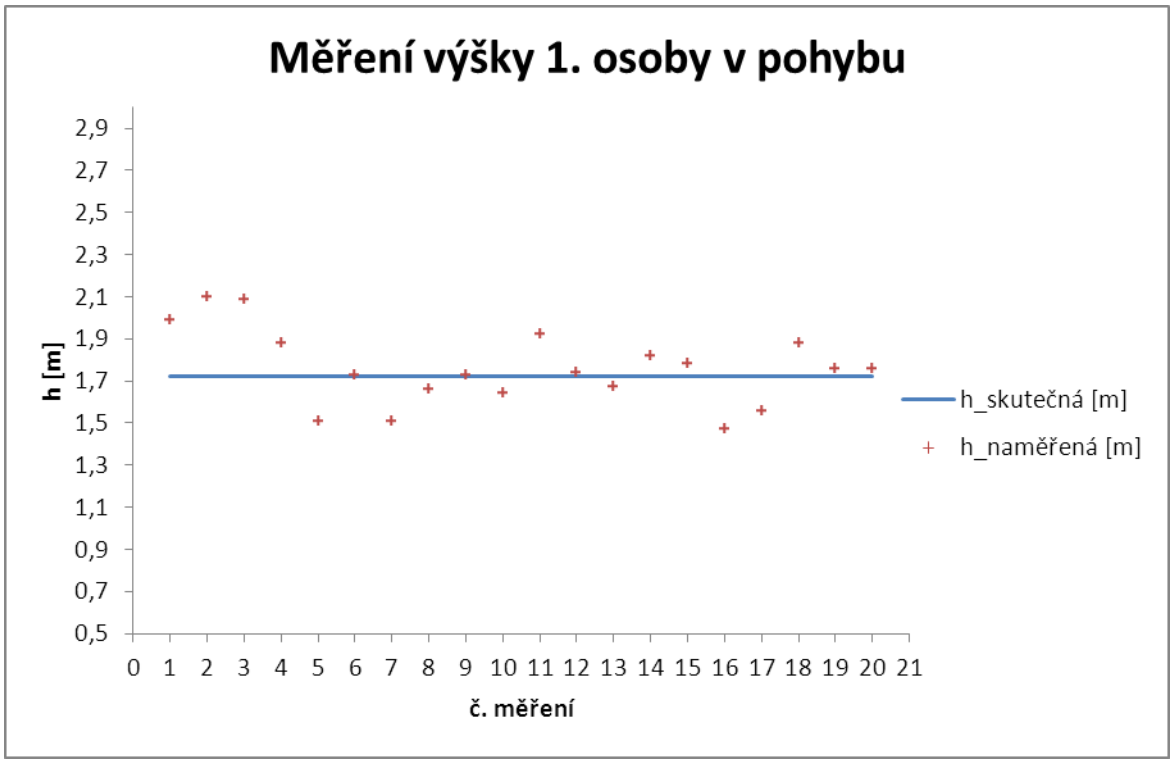
V tabulce výsledků si můžeme všimnout, že se výsledná velikost v závislosti na vzdálenosti závratně nemění, spíše jde o kolísání z důvodu odečítání velikosti v pixelech z obrázku. Z měření můžeme říci, že nám velikost láhve vyšla 0,351 ± 0,00397 m. Samozřejmě vše záleží na přesnosti stanovení rozměru v pixelech. Další kritérium od kterého se přesnost měření odvíjí je rozlišení. Troufám si říci, že čím bude větší rozlišení, tím bude měření přesnější. Bohužel, měření za použití většího rozlišení nebylo možné, z důvodu hardwarové náročnosti, otestovat.

4.4.2 Testování měření výšky pohybujících se osob

Jak už jsem nastínil na začátku kapitoly, tak s měřením osoby za pohybu byl trochu problém. To hlavně z důvodu detekce osoby a výpočtu rozměrů v pixelech potřebných pro výpočet výšky. Vyplývá to z toho, že ověření výpočtu pro statické objekty vyšlo krásně. Naměřené výsledky byly následující viz. tabulka 3 a graf za ní, postava byla měřená 20 krát a testování proběhlo na mé osobě. Moje výška je 1,72 m. První měření popíši podrobněji odvození výsledku měření, včetně výpočtu s uvedením rovnic. Další měření, na jiných osobách, budou sepsány ve formě tabulky s výsledky v kapitole 4.4.2.1.

Tabulka 4 Výsledky měření výšky 1. osoby v pohybu

č. měření	$h_{skutečná}$ [m]	$h_{naměřená}$ [m]
1	1,72	1,99
2	1,72	2,10
3	1,72	2,09
4	1,72	1,88
5	1,72	1,51
6	1,72	1,73
7	1,72	1,51
8	1,72	1,66
9	1,72	1,73
10	1,72	1,64
11	1,72	1,92
12	1,72	1,74
13	1,72	1,67
14	1,72	1,82
15	1,72	1,78
16	1,72	1,47
17	1,72	1,56
18	1,72	1,88
19	1,72	1,76
20	1,72	1,76
$h_{naměřená\ průměrná}$ [m]		1,76
nejistota měření [m]		0,18



Obrázek 31 Graf s výsledky měření výšky 1. osoby v pohybu

Z tabulky č. 4 si můžeme všimnout, že výsledky nevycházeji nijak přivětivě. Pro stanovení výsledku měření musíme vypočítat jejich aritmetický průměr spolu s nejistotou měření. Aritmetický průměr vypočítáme následovně, kdy *n* je počet měření, tedy 20

$$h_{naměřená\ průměrná} = \frac{\sum_{i=1}^{n} h_{naměřená_i}}{n} = 1{,}76\ m \tag{34}$$

Nejistotu měření vypočítáme následovně, kdy *n* je počet měření, tedy 20

$$nejistota\ měření = \sqrt{\frac{1}{n\cdot(n-1)} \cdot \sum_{i=1}^{n}\left(h_{naměřená_i} - h_{naměřená\ průměrná}\right)^2} = 0{,}18\ m \tag{35}$$

Z výpočtu aritmetického průměru naměřené výšky a směrodatné odchylky měření můžeme říci, že naměřená výška pohybující se osoby byla 1,76 ± 0,18 m. Výška osoby

byla určena s relativní nejistotou měření 14 procent. Relativní nejistota měření se vypočítála následovně.

$$relativní\ nejistota\ měření = \frac{nejistota\ měření}{h_{naměřená\ průměrná}} \cdot 100 = \frac{0{,}18}{1{,}76} \cdot 100 \cong 11\ \% \qquad (36)$$

Můžeme říci, že to není moc optimistický výsledek. Nejistoty v měření jsou způsobeny hlavně nepřestnostmi v detekci osob a následném určení těžiště. Což je způsobeno nepříznivými vlivy okolí (osvětlení, stíny), splynutím osoby s okolím (barva oblečení), šumem a dalšími faktory. Rozlišení kamer má na měření také svůj vliv. Vylepšení detekce osob je tak hlavním podnětem pro úpravu do budoucna. Taková nepřesnost výpočtu těžiště osoby je vidět i na obrázku 27, kdy do detekované osoby byl zahrnutý i stín.

4.4.2.1 Shrnutí výsledků měření výšky osob

Celkově experimentální měření proběhlo na 10 osobách. Výsledky z měření jsou shrnuté v následující tabulce, kdy průměrné naměřené výšky byly vypočítané z rovnice 34, vypočítané nejistoty měření byly z rovnice 35 a relativní nejistoty měření byly vypočítané z rovnice 36.

Tabulka 5 Výsledky měření výšky osob v pohybu

osoba č.	$h_{skutečná}$ [m]	$h_{průměrná\ naměřená}$ [m]	nejistota měření [m]	relativní nejistota měření [%]
1	1,72	1,76	0,18	10,0
2	1,83	1,87	0,28	14,8
3	1,54	1,60	0,18	11,3
4	1,64	1,64	0,16	9,8
5	1,83	1,78	0,19	10,7
6	1,80	1,75	0,18	10,3
7	1,75	1,74	0,19	10,9
8	1,72	1,76	0,22	12,5
9	1,92	1,91	0,20	10,5
10	1,68	1,68	0,18	10,7
relativní nejistota $měření_{průměrná}$ [%]				11,2

Z tabulky můžeme říci, že mnou navržená aplikace na měření výšky osob pomocí sestrojeného kamerového systému je schopná průměrně vyhodnotit výšku osoby s relativní chybou měření jedenáct procent.

Použité hodnoty k vypočítání hodnot v tabulce č. 5 jsou k nahlédnutí v příloze P I.

4.5 Návrh úprav a vylepšení

Žádný program není dokonalý, tak i mnou navržená aplikace pro měření výšky osob má své problémy a dá se v mnoha směrech upravit nebo vylepšit. Hlavně z důvodu nedostatku času nebylo možné aplikovat všechny nápady, které mě napadly.

Nejprve je důležité ukázat negativní faktory, které ovlivnily průbeh měření a které vedly k větší nejistotě měření. Tyto faktory jsou i podnětem k vylepšení a úpravám.

Negativní faktory ovlivňující průběh měření:

- Prodleva mezi snímáním první a druhé kamery – dochází k pohybu osoby mezi snímáním 1. a 2. kamery (způsobuje nejistoty v důsledku určení těžiště osoby)
- Stíny, osvětlení, pozadí, šum – dochází k nejistotem v důsledku určení těžiště osoby, možnost extrémních nejistost
- Nepřesnost v rektifikaci os – hlavně nejistoty při měření ve větších vzdálenostech

Seznam možných úprav a vylepšení:

- Vylepšení struktury programu – zrychlení chodu programu a vyhodnocování výšky
- Použití automatické rektifikace os – osy budou dopočítávány v závislosti na polohách kamer, nebude nutné je ručně vyrovnávát
- Rozvinutí výpočtu výšky - osoby půjde měřit z jakékoliv pozice a úhlu
- Implementace rozpoznání obličejů
- Implementace prvků pro identifikaci v rámci bipedální lokomoce
- Vylepšení algoritmu pro detekci osob v obraze
    - Možnost detekce více osob najednou
    - Odstranění šumu

  - Přesnější vyhledání osoby – vylepšení v rámci funkce FoundBoundary.m

- Rozšíření o možnost měření dalších veličin osoby – šířka, rychlost chůze, směr pohybu, apod.
- Podrobné posouzení vlivů na přesnost měření – osvětlení, rozlišení a jiné

Ze seznamu vidíme, že je mnoho možností jak program vylepšit. Tím pádem by mohla být tato práce dobrým pokladem k diplomové práci. Osobně bych se o to sám v budoucnu rád pokusil.

4.6 Možnosti využití systému v bezpečnostní problematice

Jako možnost využití výsledků této bakalářské práce a algoritmu pro měření výšky osob za pomoci kamerového systému v problematice bezpečnostního průmyslu, mě napadá zejména využití jako doplněk při vyhodnocování obrazu z bezpečnostních kamer. Například využití v prostorách bank by mohlo být následující. Kamery by nepřetržitě zaznamenávaly výšku osob nacházejících se v budově, a pokud by došlo k vloupání, tak by se ze záznamu rovnou zjistila výška pachatele. Na základě vyhodnocení výšky z kamer mohlo dojít ke korekci výpovědí svědků, která je mnohdy nepřesná a výška je pouhým odhadem svědka. Pokud by byl systém implementován i do městského kamerového systému, tak by mohlo probíhat hromadné tipování pachatele dle jeho výšky anebo i jiných rozměrů jako je šířka. Samozřejmě by to bylo možné po vylepšení stávajícího programu.

ZÁVĚR

Cílem této bakalářské práce bylo navrhnout experimentální zařízení pro měření výšky osob pomocí kamerového systému a vytvořit algoritmy v patřičném programovacím prostředí.

V teoretické části byly zváženy možnosti, jak můžeme postupovat při detekci osob v obraze a možnosti jak můžeme pomocí kamerového systému měřit výšku detekované osoby. Pro návrh experimentálního zařízení pro měření výšky osob pomocí kamerového systému byl zvolen princip stereoskopického vidění. Odvození vztahů pro výpočet výšky osob se věnuje praktická část. Kamerový systém byl sestaven ze dvou kamer umístěných s vzájemně rektifikovanými osami. Pro rektifikaci os a držení kamer byla sestrojena speciální konstrukce. Pro detekci osob v obraze byl zvolen algoritmus vyhodnocující změny v obraze z kamer v závislosti na automaticky generovaném pozadí scény.

Na programování aplikace a implementaci zvolených algoritmů pro měření výšky a detekci pohybu bylo použito programovacího jazyka MATLAB. Ten byl zvolen, protože obsahuje prostředky pro zpracování obrazu z kamer. Aplikace byla navržena tak, aby zajišťovala nahrávání snímků z kamer, vyhodnocení pohybu a generování pozadí a nakonec samotné měření výšky osob na základě předchozího vyhodnocení. Aplikace je ovládána pomocí navrženého grafického uživatelského rozhraní, které zobrazuje výsledky měření a obraz z kamer. Dále navržená aplikace umožňuje měnit vlastnosti kamer, použití filtrů k vyhodnocení pohybu a provází uživatele rektifikací os kamer snímáním rektifikační roviny s kříži.

Měření probíhalo v laboratoři C304 v budově U5. Nejprve bylo navržené zařízení z důvodu ověření správnosti odvození vztahů pro výpočet výšky osob otestováno na statickém objektu. Dále pro otestování samotné aplikace spolu s fungováním algoritmu pro detekci osob v obraze proběhlo testování na pohybujícím se objektu (mé osobě). Na konci praktické části je popsán průběh samotného měření a všechny naměřené výsledky jsou sepsány a vyhodnoceny, spolu s možnostmi úprav a vylepšeními do budoucna. Dále jsou uvedeny možnosti navrženého systému a aplikace využití v bezpečnostní praxi.

ZÁVĚR V ANGLIČTINĚ

Objective of this bachelor work was to design experimental device for measurement of person stature using camera system and also to create algortihms in appropriate programming language.

In the theoretical part, options were considered how we can do detection of person in image and options how we can measure stature of detected person stature by using camera system. The principle of stereoscopic vision was chosen for design of experimental device for measurement of person stature. Derivation of formulas measurement of person stature is wrote in the practical part of this work. Camera system consists of two cameras with rectificated axes. For rectification and holding cameras the special construction was constructed. For person detection in image the algorithm which evaluates changes in image from cameras with automatically generated background was chosen.

For programming application and implementation of chosen algorithms for measurement of person stature and detection of person in image programming language MATLAB was used. It was chosen, because it includes resources for processing of image data from camera. Application was designed for saving images from cameras, evaluation of moving in scene and generation of background and also for measuring of person stature from previous evaluation. Application is controlled by designed graphical user interface, which also shows results of measurement of person stature. The designed application also allows changing of cameras properties, using the filters for evaluation of move and also leads the user throught rectification within showing rectification plane with crosses.

Measuring was done in laboratory C304 at U5 building. Correctness of derivated formulas for measurement of person stature was firstly tested on static object at designed device. The good working of algorithm for detection of moving person was tested on moving object (on my person). At the end of the practical part the process of measuring is described and all measured results are summarized and evaluated within options of possible changes and improvements to the future. Furthermore, some possibilities of designed measuring system are mentioned as well as application in a Security Industry.

SEZNAM POUŽITÉ LITERATURY

[1] LI, Y F a B ZHANG. *A method for 3D measurement and reconstruction for active vision* [online]. 2004 [cit. 2012-05-13]. DOI: 10.1088/0957-0233/15/11/007. Dostupné z: http://iopscience.iop.org/0957-0233/15/11/007/

[2] CHEN, Tung-Chien. *Video Segmentation Based on Image Change Detection for Surveillance Systems*. Santa Cruz, California USA, 2007. Dostupné z: http://classes.soe.ucsc.edu/ee264/Spring07/ChenReport.pdf. Final Project. University of California Santa Cruz.

[3] *Microsoft Hardware LifeCam Studio*. [online]. [cit. 2012-05-14]. Dostupné z: http://www.microsoft.com/hardware/en-us/p/lifecam-studio

[4] ZAPLATÍLEK, Karel a Bohuslav DOŇAR. *MATLAB pro začátečníky*. 2. vyd. Praha: BEN - technická literatura, 2005, 151 s. ISBN 80-7300-175-6.

[5] ZAPLATÍLEK, Karel a Bohuslav DOŇAR. *MATLAB: tvorba uživatelských aplikací*. 1. vyd. Praha: BEN - technická literatura, 2004, 215 s. ISBN 80-7300-133-0.

[6] GONZALES, Rafael C, Richard E WOODS a Steven L EDDINS. *Digital image processing using MATLAB*. Upper Saddle River: Pearson/Prentice Hall, c2004, xiv, 609 s. ISBN 0-13-008519-7.

[7] CAPPELLINI, V. *Time-varying image processing and moving object recognition, 4: proceedings of the 5th international workshop, Florence, Italy, September 5-6, 1996.* Amsterdam: Elsevier, 1997, xiii, 332 s. ISBN 0-444-82307-7.

[8] JAIN, Anil K. *Fundamentals of digital image processing*. Upper Saddle River: Prentice Hall, c1989, xxi, 569 s. ISBN 0-13-336165-9.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CD Compact Disc

GUI Graphical User Interface

HD High-Definition

HDD Hard Disc Drive

PIR Pasiv Infra Red detector

RAM Random Access Memory

RGB Red-Green-Blue

USB Universal Serial Bus

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: NAMĚŘENÉ HODNOTY Z MĚŘENÍ VÝŠKY OSOB

	Osoba č.1	Osoba č.2	Osoba č.3	Osoba č.4	Osoba č.5
index měření	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]
1	1,99	1,85	1,13	1,65	1,96
2	2,10	1,85	1,61	1,46	1,96
3	2,09	1,7	1,82	1,72	1,91
4	1,88	1,85	1,78	1,55	1,48
5	1,51	2,45	1,72	1,46	1,96
6	1,73	1,52	1,47	1,76	1,99
7	1,51	1,51	1,61	1,50	1,53
8	1,66	1,78	1,75	1,50	1,32
9	1,73	1,39	1,24	1,55	1,55
10	1,64	1,94	1,56	1,81	1,97
11	1,92	1,87	1,36	1,37	1,75
12	1,74	2,40	1,71	1,85	1,85
13	1,67	1,67	1,73	1,44	1,68
14	1,82	2,34	1,55	1,85	1,87
15	1,78	1,76	1,59	1,73	1,80
16	1,47	1,98	1,57	1,66	1,90
17	1,56	2,11	1,53	1,54	1,62
18	1,88	1,67	1,71	1,82	1,82
19	1,76	1,98	1,60	1,72	1,72
20	1,76	1,87	1,35	1,78	1,87
$h_{naměřená\ průměrná}$ [m]	1,76	1,87	1,60	1,64	1,78
nejistota měření [m]	0,18	0,28	0,18	0,16	0,19
relativní nejistota [%]	10,0	14,8	11,3	9,8	10,7

	Osoba č.6	Osoba č.7	Osoba č.8	Osoba č.9	Osoba č.10
index měření	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]	$h_{naměřená}$ [m]
1	1,30	1,30	1,51	2,04	1,88
2	1,96	1,96	1,53	1,49	1,49
3	1,71	1,71	2,05	2,08	1,75
4	1,75	1,76	2,07	1,88	1,55
5	1,43	1,44	1,99	1,50	1,60
6	1,90	1,86	1,79	1,79	1,43
7	1,98	1,97	1,87	2,10	1,77
8	1,86	1,65	1,79	2,01	1,76
9	1,97	1,87	1,86	1,84	1,59
10	1,65	1,58	1,77	1,86	1,80
11	1,76	1,71	1,95	2,03	1,54
12	1,87	1,67	1,64	2,09	1,34
13	1,58	1,83	1,84	2,17	1,71
14	1,72	1,54	1,70	1,76	1,40
15	1,67	1,91	2,17	1,80	2,01
16	1,83	1,74	1,56	2,22	1,90
17	1,54	1,66	1,48	2,08	1,78
18	1,91	1,70	1,53	2,03	1,69
19	1,74	1,95	1,69	1,82	1,80
20	1,95	2,07	1,34	1,67	1,90
$h_{naměřená\ průměrná}$ [m]	1,75	1,74	1,76	1,91	1,68
nejistota měření [m]	0,18	0,19	0,22	0,20	0,18
relativní nejistota [%]	10,3	10,9	12,5	10,5	10,7