VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

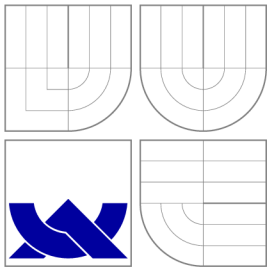
DETEKCE POHYBUJÍCÍCH SE OBJEKTŮ VE VIDEO SEKVENCI

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

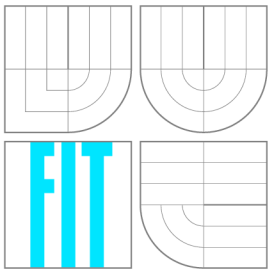
AUTOR PRÁCE
AUTHOR

JAKUB MUSIL

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE POHYBUJÍCÍCH SE OBJEKTŮ VE VIDEO SEKVENCI

MOVING OBJECTS DETECTION IN VIDEO SEQUENCES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAKUB MUSIL
AUTHOR

VEDOUCÍ PRÁCE — Ing. MICHAL ŠPANĚL
SUPERVISOR

BRNO 2008

Zadání bakalářské práce/6303/2007/xmusil29

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Musil Jakub**
Obor: Informační technologie
Téma: **Detekce pohybujících se objektů ve video sekvenci**
Kategorie: Počítačová grafika

Pokyny:

1. Prostudujte základy zpracování obrazu. Zaměřte se na detekci pohybu v obrazové sekvenci.
2. Zorientujte se v metodách analýzy pohybu v obraze a detekce pohybujících se objektů.
3. Vyberte vhodnou metodu a navrhněte postup detekce.
4. Experimentujte s vaší implementací.
5. Případně navrhněte vlastní modifikace metod.
6. Diskutujte dosažené výsledky a možnosti budoucího vývoje.
7. Vytvořte stručný plakát prezentující vaši bakalářskou práci, její cíle a výsledky.

Literatura:

- Dle pokynů vedoucího

Při obhajobě semestrální části projektu je požadováno:

- Splnění prvních tří bodů zadaní.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Španěl Michal, Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2007
Datum odevzdání: 14. května 2008

VYSOKÉ UČENÍ TECHNICKÉ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Pavel Zemčík
vedoucí ústavu

Licenční smlouva

Licenční smlouva je uložena v archivu Fakulty informačních technologií Vysokého učení technického v Brně.

Abstrakt

Práce se věnuje tvorbě aplikaci pro detekci pohybujících se objektů ve vstupní video sekvenci. Jsou v ní detailně popsány metody, které se při jeho realizaci využívají, jejich kladné a záporné vlastnosti. Detailně jsou zde popsány metody porovnání na úrovni histogramů a rozdílu jednotlivých bodů. Teoretická část přibližuje možné postupy a optimalizace, pomocí kterých lze detektor přizpůsobit prostředí, pro které je vytvářen. Součástí práce je aplikace pro detekci a vizuální označení pohybujících se objektů. Aplikace je testovaná na několika typově odlišných video sekvencí. Dosažené výsledky jsou blíže diskutovány a rozebrány, včetně ukázek dosažených výstupů detektoru.

Klíčová slova

Detekce pohybu, porovnání obrazů, Local Binary Patterns, zpracování obrazu, zpracování video sekvence, detektor pohybu, C++, Visual Studio.

Abstract

This publication is dedicated to creating appliaction for motion detection in video sequence. There is summary of several possible methods to solve this matter in theoretical and practical point of view with postive and negative properties. This methods are histograms subtraction and subtraction of each pixel in the tested frame. Fundamental part of this publication is application for motion detection. There are many of tests for created detector for several type moving object included in input video sequence. Results are intimately describe.

Keywords

Motion detection, compare picture, Local Binary Patterns, picture processing, movie processing, motion detector, C++. Visual Studio.

Citace

Jakub Musil: Detekce pohybujících se objektů ve video sekvenci, bakalářská práce, Brno, FIT VUT v Brně, 2008

Detekce pohybujících se objektů ve video sekvenci

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Michala Španěla

.....
Jakub Musil
12. května 2008

Poděkování

Děkuji Ing. Michalovi Španělovi za jeho cenné rady, konstruktivní kritiku a připomínky při tvorbě praktické a teoretické části práce.

Obsah

Kapitola 1

Úvod

Tvorba detektoru, nebo-li aplikace pro detekci pohybujících se objektů ve video sekvenci je téma z oblasti počítačové grafiky, které umožňuje využít nejrůznějších přístupu a algoritmů. Ty volíme především dle typu použiti výsledné aplikace.

V praxi naleznou využití detektory typu „alarm“, které pouze rozhodují o případném výskytu pohybu a spustí definovaný proces, kterým může být například archivace této video sekvence, upozornění patřičného orgánu a podobně. Dalším typem detektorů s kterými se můžeme setkat jsou ty, které kladou důraz na vyznačení místa, v kterém k pohybu dochází. Pro jejich tvorbu využíváme odlišné posloupnosti algoritmů a metod než u předchozího typu.

Detektory lze rozdělit také dle vstupu se kterým pracují. Můžeme odlišit detektory, které pracují v reálném čase a komunikují v online režimu s video kamerou. V tomto případě musíme brát zřetel především na rychlost zpracování, která by neměla být delší, než je doba platnosti jednotlivých snímků. Což je při použití standardní rychlosti 25 snímků za sekundu poměrně krátká doba na jejich zpracování. A proto se snažíme použité metody optimalizovat na úkor kvality dosaženého výsledku. Druhou skupinou jsou detektory pracující s již vytvořeným video záznamem, který je detektoru předložen. Při realizaci tohoto typu si můžeme dovolit použít metody, které jsou časově náročnější a klást tak důraz především na správnost našeho výpočtu, než na rychlost jeho zpracování.

Jedna vlastnost zůstává pro všechny typy aplikací sloužící k rozeznání pohybu ve video sekvenci stejnou. A tím je přístup k testované video sekvenci přes jednotlivé snímky, které musíme ze sekvence extrahovat. A to ať pracujeme s připojenou kamerou nebo s uloženým video záznamem.

Po této úvodní kapitole se čtenář seznámí se základními pojmy počítačové grafiky a pojmy souvisejícími s obrazem, včetně jeho převodu z reálného světa do podoby srozumitelné číslicovým systémům, nejčastěji počítačům.

Třetí kapitola popisuje nejvýznamnější metody, které jsou používány při tvorbě aplikace detekující pohyb. Jsou zde uvedeny příklady použití, výhody a naopak nevýhody se kterými je nutno počítat při výběru těchto metod. Při zjednodušeném pohledu na popisované metody je můžeme rozdělit na metody pracující s histogramy, které vyjadřují charakteristiku porovnávaných snímků a metody, které porovnávají hodnoty odpovídajících si bodů testovaných snímků.

Následující kapitola je o návrhu vytvářeného detektoru, přiblížení jeho základních bloků, struktury použitých metodách, o optimalizacích a rozšířeních základních metod.

V kapitole páté je čtenáři přiblížená výsledná struktura, jsou zde uvedeny knihovny, které nalezly uplatnění při implementaci vytvářeného detektoru pohybu.

V šesté kapitole jsou obsaženy ukázky použití vytvořeného detektoru. Pro tyto testy byly použity různé typy vstupních video sekvencí, včetně těch problematických, které jsou například způsobeny změnou denní doby a světelných podmínek. Nebo také mírným chvěním se zařízením starajícím se o pořízení samotného záznamu.

Poslední kapitolou je závěr, kde shrnuji získané poznatky a diskutuji možné rozšíření detektoru.

Cílem této teoretické části je podat čtenáři přehled o problematice tvorby detektoru pohybu a o přiblížení základních pojmů počítačové grafiky a teorie obrazu.

V praktické části jsem se věnoval tvorbě samotného detektoru. Při jeho tvorbě byl kladen důraz především na rozlišení samotného pohybu a na jeho vyznačení. Na možnost přizpůsobení programu pomocí uživatelského nastavení a také na možnost volby použití vstupu z připojené online kamery, ale stejně tak již z vytvořeného a ukončeného video záznamu. K praktické části jsou přiloženy ukázkové video záznamy, které nejlépe vystihují chování detektoru.

Kapitola 2

Teoretická část

Počítačová grafika je z technického hlediska obor informatiky. Zabývá se tvorbou a analýzou prostorových informací, které vznikou snímáním, nebo-li digitalizací obrazu reálného světa.

Obraz je v počítači uložen jako sled jedniček a nul pomocí dvou hlavních způsobů. Jsou jimi **rastrová grafika**, ve které jsou uloženy hodnoty o barvě a jasu jednotlivých bodů obrazu v pomyslné mřížce. A **vektorová grafika**, která reprezentuje grafické objekty pomocí matematického popisu.

Při práci s běžnými snímky reálného světa je využíváno rastrové grafiky. Mezi hlavní charakteristiky takto uložených obrazů patří počet zobrazovaných bodů, nebo-li rozlišení a barevná hloubka. Ta představuje množství informací, které můžeme o každém pixelu zadat a udává se v bitech.

2.1 Obraz a jeho zpracování

Prvním krokem pro zpracování a rozpoznání obrazu na počítači je získání obrazu reálného světa v číslicové podobě, která je vhodná pro uložení a další zpracování na počítači či jiném číslicovém systému.

Při tvorbě digitálního obrazu musíme nejdříve provést převod optické veličiny, vyjadřující podobu skutečného světa na elektrický signál. Ten je spojitý a v každém čase obsahuje určitou hodnotu své úrovně. Druhým krokem je transformace spojitého analogového signálu na signál digitální. Tento proces se nazývá digitalizace. Digitální obraz je ekvivalentem spojité obrazové funkce $f(i, j)$, kde i a j jsou souřadnice v prostoru. Je získán pomocí vzorkování obrazu do matice určitého rozměru $M * N$. Vzorkování se řídí obecně známou Shanonovou větou. Z které vyplývá, že nejmenší detail v digitálním obraze musí být minimálně dvojnásobkem vzorkovaného intervalu.

Digitalizace má své parametry. Jedním z nejdůležitějších je volba správného rozlišení výsledného obrazu. To nám přímo úměrně udává kolik informací obraz bude obsahovat. Pro každý typ zpracování digitálního obrazu je vhodné jiné rozlišení. Jsou aplikace, které potřebují pro svou správnou funkci co nejvíce informací, a proto použiji vysoké rozlišení s vědomím, že výpočetní nároky na zpracování obrazu stoupnou. Stejně tak se i setkáme s aplikacemi, které kladou důraz na co nejrychlejší zpracování na úkor ztráty části informací. Např. programy, které zpracovávají obraz v reálném čase. Mezi ně patří také nejrůznější detektory pohybu. Rozměry obrazu se nejčastěji udávají v obrazových bodech označovaných jako pixel.

2.2 Obrazové transformace

V této kapitole budou přiblíženy základní transformace, které lze aplikovat na jednotlivé snímky testované video sekvence. Srovnáme-li snímky jednoho a totéž prostředí, které je neměnné dojdeme ke zjištění, že ne všechny body obrazu jsou zcela identické. To způsobuje především šum, který je v procesu digitalizace grafických objektů všudypřítomným problémem. K jeho potlačením jeho důsledků využíváme různě složité obrazové transformace.

Grayscale

Nebo-li převod barevného snímku, který je reprezentován pomocí tří složek na jemu odpovídající snímek reprezentovaný pouze jednobajtovou hodnotou. Ta obsahuje 256 odstínu šedé barvy a je vypočtena pro každý pixel převáděného obrazu samostatně. Pro převod se využívá následujícího vzorce

$$I = 0,299R + 0,587G + 0,114B \tag{2.1}$$

Kde R značí barvu červenou, G barvu zelenou, B modrou transformovaného snímku a I výslednou intenzitu ve stupni šedi.

Tyto poměry jsou stanoveny z odlišné citovosti lidského zraku na základní barvy. Nejvíce citlivý je člověk na barvu zelenou, proto se podílí nejvíce na nově získané hodnotě.

Obrázek 2.1: Grayscale

Stanovíme-li nový černobílý snímek právě tímto způsobem, nově vytvořený obraz se nám jeví jako původní snímek barevný. (Čerpáno z [1])

Prahování

Je nejjednodušší metodou pro omezení barevného prostoru. V anglickém jazyce je označována jako thresholding. Převede nám vstupní matici pixelů na odpovídající matici v monochromatické podobě. Tedy v barvě černé a bíle.

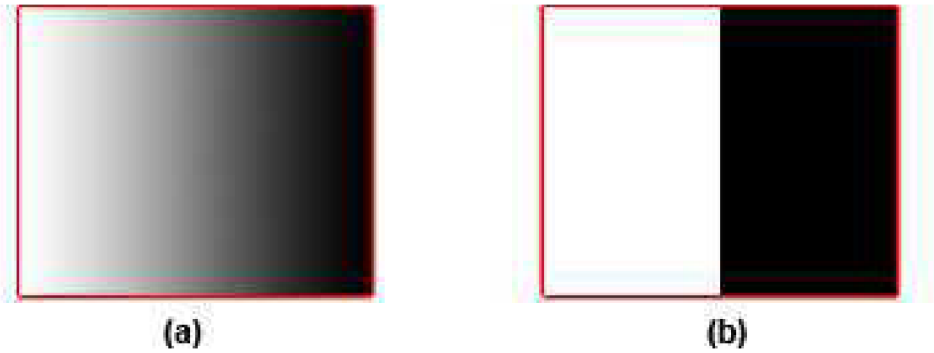
Proces rozhodování o výsledné hodnotě pixelu je určen barevným prostorem vstupního snímku. Je-li vstupem obrázek v odstínu šedi, stačí přistupovat ke každému pixelu pouze jednou, aby jsme přečetli jeho intenzitu. Tu následně porovnáváme s prahem, který je dalším vstupním atributem této metody. Přidělení nové hodnoty x se řídí níže uvedeným předpisem.

$$f(n) = \begin{cases} 0 & \text{je-li } x > Threshold \\ 255 & \text{je-li } x <= Threshold \end{cases} \tag{2.2}$$

Je-li hodnota menší než určený prah, hodnota nového bodu má bílou barvu. V opačném případě barvu černou, což je v RGB schématu hodnota 0.

Tato metoda dosahuje uspokojujících výsledků pro vstupní obraz vysokého kontrastu. Pro dosažení kvalitního výsledku je potřeba zvolit správnou hodnotu prahu pro konkrétní vstup. Určení správné hodnoty vychází z analýzy histogramu a může být zvolena například pomocí střední hodnoty nebo mediánu histogramu vstupního obrazu. Pro vyvolání efektu „hrubého síta“ simulujícího staré fotografie zvolíme náhodnou hodnotu prahu pro každý převáděný pixel.

Příklad tohoto filtru ukazuje obrázek 2.2, kdy vstupním snímkem je levý obrázek (a) a novým snímkem, vzniklý aplikací tohoto filtru je obrázek druhý, označený (b). Jako hodnota prahu byla v této ukázce použita konstanta 128.



Obrázek 2.2: Prahování

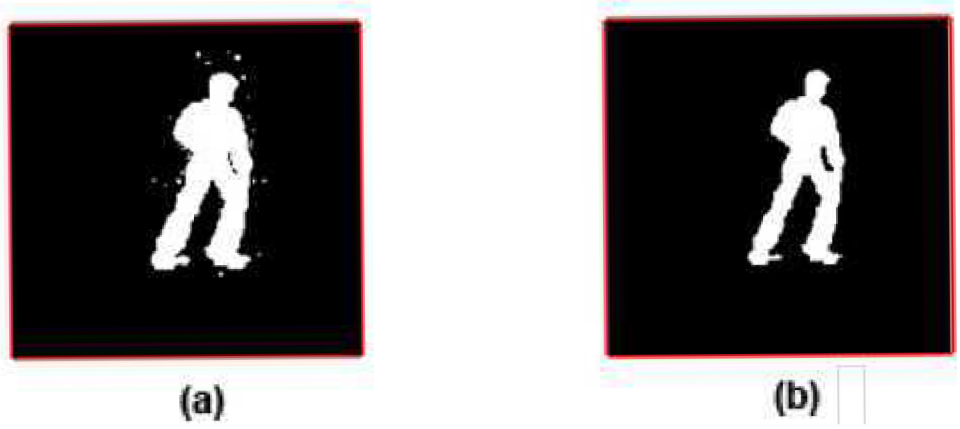
Prahování je v problematice detekce pohybu ve video sekvenci používáno především v metodě, které odečítá intenzity pixelů ležících na stejné pozici. Pomocí prahovaní můžeme „vynulovat“ pixely, které mají pouze malou intenzitu. Menší než námi určený prah.

Erosion

Pracuje s jedním vstupním snímkem, který je v monochromatické podobě. Jeho princip je takový, že pro každý obrazový bod vstupního snímku prochází jeho okolí. Za okolí bývá nejčastěji volena matice o rozměrech 3x3. V této matici se vyhledá hodnota nejmenší, což je barva bílá v barevném modelu RGB a přiřadí se jako hodnota nová pro právě testovaný pixel.

Aplikováním filtru dosáhneme odstranění osamocených pixelů, které vnikají například vlivem šumu při odečítání odpovídajících si pixelů od sebe. A zároveň zůstanou zachovány celistvé oblasti, pouze jejich hranice se zmenší, což lze pro použití v detektorech pohybu tolerovat.

Obrázek 2.3 ukazuje použití popisovaného metody na obrázku, který byl pomocí prahovaní převeden do monochromatické podoby, odstranil nám osamocené pixely a zároveň zachoval informaci o detekovaném pohybu.



Obrázek 2.3: Filtr Erosion

Kapitola 3

Metody detekce pohybu v obraze

Tato kapitola přibližuje čtenáři základní metody, které jsou využívány při tvorbě detektoru pohybu. Je zde popsán jejich princip, klady, zápory a možnosti využití.

Standardní cestou při rozpoznání pohybu ve video sekvenci je získání jednotlivých snímků, ze kterých je testovaná video sekvence tvořena. Detekci pohybu provádíme nad takto získanou sekvenci snímků. Zpravidla porovnáváme změny mezi dvěma po sobě jdoucími snímky, nebo případně průměrem několika snímků. V optimálním případě jsou snímky, mezi kterými nedošlo k pohybu zcela identické. Ale dosažení této skutečnosti není reálné, protože obraz je ovlivněn vlivem denního světla a především také vlivem všudypřítomného šumu. Rozsah šumu je dán kvalitou snímací techniky a prostředí, ve kterém se snímaný objekt nachází. Částečně potlačit tyto nepříjemné jevy lze použitím průměrování několika po sobě jdoucích snímků, nebo použitím filtrů.

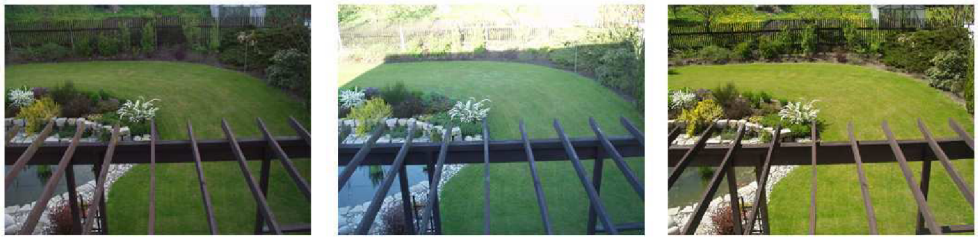
Při volbě vhodné metody musíme zohlednit požadovanou podobu vytvářeného detektoru. Chceme-li vytvořit aplikaci, která pohyb v obraze pouze detekuje, ale dále ho již nezpracovává(nevyhledává konkrétní místo pohybu) použijeme jinou metodu, nebo kombinaci metod, než pro aplikaci, jejíž prioritním cílem je pohyb pokud možno co nejpřesněji vizuálně označit.

Porovnávání provádíme nad snímky, které označujeme pozadí a popředí. Popředí bývá zpravidla snímek aktuální. Ale pozadí může být:

- Pevné: V tomto případě je za pozadí považován stálý snímek, který se v průběhu zpracování nikterak nemění, nebo pouze v předem určených časových intervalech. Výhodou této volby je ušetření výpočetního výkonu, protože referenční snímek pozadí se zpracuje vybranými algoritmy pouze jednou a dále jej jenom porovnáváme se snímkem aktuálním. Můžeme využít náročné filtry, které nám snímek zbaví šumu, protože se jedná o operaci prováděnou pouze jedenkrát za čas. Jako nevýhodu tohoto typu pozadí zmíníme náchylnost na změny v prostředí, které se dopředu nedají vždy přesně určit. Jedná se například o změnu světelných podmínek, ke kterým dochází vlivem okolního počasí. Tyto změny v předem vytvořeném pozadí nejsou zahrnuty a při porovnání s aktuálním snímkem, který změny obsahuje může dojít k mylné detekci pohybu. Obrázek 3.1 ukazuje stejné prostředí, které bylo zachyceno v různou denní dobu.

- Proměnné: Pozadí je neustále aktualizováno. Může jim být snímek popředí, který byl nahrazen aktuálním obrazem. V tomto případě porovnáváme snímek aktuální(popředí) se snímkem předposledním(pozadí). Tato volba dovoluje použít výsledky algoritmů prováděné v předchozím kroku nad snímkem popředí, které přechází ve snímek pozadí

a je nahrazeno snímkem novým. Ale častější variantou použití je průměrování několika posledních snímků, které nám částečně eliminuje šum v obraze a jeho drobné chyby způsobené nekvalitním snímáním a reprezentací. Nevýhodou ovšem jsou již značné výpočetní nároky, protože nemůžeme využít výpočty provedené v předchozím kroku a musíme tedy neustále provádět algoritmy nad dvěma snímky. Nad popředím a nad průměrovaným pozadím.



Obrázek 3.1: Shodné prostředí snímané za různých světelných podmínek

3.1 Porovnání histogramů

Tato metoda porovnává histogramy určitých vlastností, kterými nejčastěji bývají jas a LBP koeficienty(kapitola 3.3) pozadí a popředí. Histogramy vyjadřují charakteristiku obrazu.

Hlavní nasazení je především v detektorech typu „alarm“, které mají za úkol pouze rozhodnou o přítomnosti pohybu, ale není po nich vyžadováno jeho nalezení. Pozadí může být pevné, což využijeme při snímání prostoru, které zachovává konstantní světelné podmínky. Takovým prostorem může být vnitřní prostor(skladiště, hala apod.), kde jediným proměnlivým parametrem je šum. Snímáme-li naopak prostory ovlivněné světelnými podmínky využijeme proměnné pozadí, které je neustále aktualizováno.

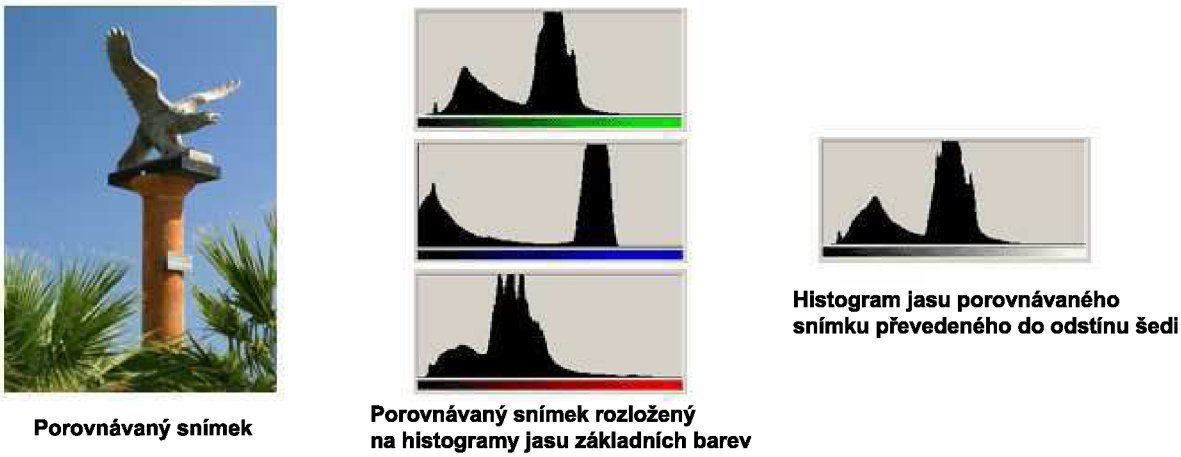
Problémem této metody jsou objekty, které se sice pohybují, ale neustále se nachází ve snímaném prostředí(viz. obrázek 4.1). Takto pohybující objekty mění minimálně barevnou charakteristiku obrazu a metoda není schopna detekovat pohyb, ke kterému dochází. Pro odstranění této vlastnosti můžeme porovnávané obrazy rozdělí na více částí a porovnávání tak provádět pouze mezi odpovídajícími částmi. Výpočetní nároky při tomto rozdělení zůstávají takřka zachovány. Rozdělíme-li obraz na dostatečně jemné části získáme tím možnost detekovat ve které částí obrazu k pohybu došlo a můžeme jej vizualizovat.

Jako hlavní výhodou můžeme uvést malou změnu histogramů při mírném chvění zaznamenávajícího zařízení, které nám správně nevyvolá detekci pohybu.

Histogramy jasu

Použití histogramu jasu je nejjednodušší volbou. V tomto případě histogram vyjadřuje jasovou charakteristiku všech pixelů testovaného obrazu. Histogram je v praxi realizován jako jednorozměrné pole. Jeho prvky reprezentují určitý jas a hodnota tohoto prvku symbolizuje počet pixelů, které tomuto jasu náleží. Abychom takovýto histogram mohli spočítat, musíme projít všechny jednotlivé body snímku, určit jeho jasovou hodnotu dle které inkrementujeme prvek v poli na patřičném indexu, reprezentující určenou hodnotu jasu.

Jasovou hodnotu můžeme brát přímo z jednotlivých základních barevných složek snímku a můžeme tak sestrojit histogramů více, jak ukazuje obrázek 3.2, nebo častější volbou je převod do odstínu šedi a nad takto převedeným obrázkem spočteme histogram pouze jeden.



Obrázek 3.2: Histogramy jasu

Tímto způsobem vypočteme histogramy pozadí a popředí. V dalším kroku od sebe odečteme histogramy na odpovídajících si oddílech. Jsou-li porovnávané snímky takřka shodné, budou hodnoty na jednotlivých oddílech histogramu velice podobné a v opačném případě, kdy snímky jsou rozdílné bude hodnota značně odlišná a tento rozdíl nám vyjadřuje míru odlišnosti.

Jako vhodnou optimalizaci můžeme pole zastupující reálný histogram zmenšit s tím, že jeden jeho oddíl nebude zastupovat počet pixelů s jednou konkrétní jasovou hodnotu, ale bude zastupovat určitý rozsah.

Histogramy LBP koeficientů

Jedná se o metodu, která je využívána především pro rozpoznávání objektů v obraze(viz.[3]). A to především lidského obličeje. Můžeme se s ní setkat pod označením Local Binary Patterns. Tato metoda využívá podobně jako metoda předchozí histogramy porovnávaných snímků, avšak ty již nereprezentují jasovou charakteristiku obrazu, ale LBP koeficientů.

LBP koeficienty jsou vypočítány pro každý pixel obrazu odděleně. Jako základ se bere jasová hodnota troj okolí pixelu, pro který je koeficient počítán. Možný stav ukazuje první matice (a) v obrázku 3.3, počítaný pixel je v naši matici prostřední s hodnotou 6.

8	4	12
3	6	6
1	5	9

(a)

1	0	1
0		1
0	0	1

(b)

1	2	4
8		16
32	64	128

(c)

1	0	4
0		16
0	0	128

(d)

Obrázek 3.3: Výpočet LBP koeficientu

Druhým krokem je transformace této matice na matici novou, která vznikne porovnáním

hodnoty testovaného pixelu s hodnotou okolních osmi pixelů. Je-li hodnota okolního pixelu menší než prostředního, nová matice na tomto místě obsahuje hodnotu nula. V opačném případě je na patřičné místo vložena hodnota jedna. Vzniklá matice je v našem příkladě označena jako matice (b). Třetím krokem je součin hodnot této matice s hodnotou n^2, kde n je pořadí násobeného indexu. Pro názornost je na obrázku uvedena pod označením (c) matice, jejíž body vzniknou vyplněním právě n^2. Po dokončení tohoto součinu získáme výslednou matici (d). Posledním krokem je již pouze součet hodnot výsledné matice a výsledek toho součtu prohlásíme za LBP koeficient prostředního bodu.

LBP operátorem, nebo-li koeficientem klasifikujeme všechny pixely snímku. Při jeho tvorbě musíme zohlednit krajní pixely, které nemají všechny své okolní body a vypustit jej, nebo tuto skutečnost nějakým jiným způsobem zohlednit.

Máme-li vypočteny koeficienty všech testovaných pixelů vytvoříme klasickým způsobem, popsaným v předchozí kapitole 3.1 histogramy reprezentující vypočtené koeficienty. S těmito histogramy pak pracujeme stejným způsobem, jako kdyby reprezentovaly pouze jasovou charakteristiku. Platí pro ně také stejné možnosti dalšího rozšíření, jako je například rozdělení obrazu na více částí pro detekci předmětu pohybujícího se uvnitř obrazu, případně pro vyznačení místa pohybu.

Obraz popsaný koeficienty LBP má tu výhodu, že je odolný vůči celkové změně jasu obrazu. Z toho vyplývá, že je vhodné jej použít v prostorech, kde vzniká např. vlivem změny denní doby ke změně světelných podmínek.

3.2 Porovnání rozdílných bodů

Tato metoda, které porovnává hodnoty jednotlivých pixelů pozadí a popředí je ze všech popisovaných metod nejbližší postupu, který využívá člověk při hledání odlišností mezi několika snímky. Avšak s jedním významným rozdílem. A tím je inteligence lidského mozku, který je schopen abstrahovat určité rozdíly mezi snímky, které do porovnání nechceme zohlednit. Můžou jimi být různé chyby při tisku, chyby způsobené ztrátou barevné kvality fotografie a podobně. To bohužel neplatí při porovnání testovaných snímků výpočetním zařízením, které neumí odlišit, zda jsou jednotlivé body odlišné z důvodu špatné kvality jednoho snímku či druhého snímku, nebo zda vyjadřují „opravdovou“ změnu. Jako je například příjezd automobilu na křižovatku, kterou znázorňuje obrázek 3.4.

Obrázek 3.4: Rozdíl snímků

Prostřední obrázek reprezentuje snímek pozadí. Může jim být snímek neměnný(konstantní), ale také snímek, který je neustále aktualizován. Dalším snímkem prvnímv pořadí, je naše

popředí, na kterém chceme detekovat pohyb. Tento snímek obsahuje ve srovnání s prvním snímkem vozidlo, které by měl náš algoritmus odhalit a detekovat tak pohyb.

Porovnávat snímky na úrovni rozdílu hodnot jednotlivých pixelů můžeme nad barevnými snímky, v tom případě provádíme rozdíl jednotlivých pixelů hned třikrát(červená, zelená a modrá barva). Ale častěji nejdříve snímky zbavíme barevné informace převedením do odstínu šedi, které nám ušetří počet přístupů k hodnotám jednotlivých pixelů a zároveň nám stejně dobře uchovává informaci o podobě snímku. Pro převod můžeme využít postup, který je blíže popsán v kapitole 2.2. Odečteme-li hodnoty pixelů převedené do stupnice šedé barvy obrázku popředí a pozadí získáme výsledek označený jako rozdíl(obrázek 3.4). Ten v sobě nese informaci o míře rozdílnosti jednotlivých bodů. Na souřadnicích, kde byly hodnoty bodů shodné je barva černá. Čím více se odlišují odečítané pixely tím více se výsledná hodnota blíží barvě bíle.

Na takto získaný snímek, vyjadřující odlišnosti jednotlivých pixelů můžeme aplikovat hned několik filtrů. Zpravidla prvním z nich je prahování (viz 2.2), které porovnává intenzitu pixelu s předem určenou hranicí, nebo-li prahem a dle tohoto porovnávání je pixel nahrazen barvou bílou, nebo černou. Získáme tak dvou barevný snímek, jako je uvedený na obrázku 3.5 pod označením (b), který vznikl právě touto metodou z předchozího obrázku (a). Další využívanou optimalizací je použití filtru erosion (kapitola 2.2), který odstraní osamocené pixely.

Obrázek 3.5: Rozdíl snímků

Náročnost na porovnávání snímku dle rozdílu hodnot jednotlivých pixelů je přímo úměrná rozlišení testovaných snímků, které nám udává počet přístupů k pixelům. Z tohoto důvodu je možno provést optimalizaci, kdy rozdíl provádíme pouze na určitých souřadnicích. Například odečítáme pouze pixely ležící na sudém řádku, sudém sloupci. Zdvojením takto získaných hodnot zachováme původní rozměr obrázku.

Tato metoda je vhodná pro vyznačování místa pohybu. Při standardním použití se po získání rozdílového snímku(po aplikování prahovaní) spočte počet bílých pixelů a ta se porovná s hranicí, jejíž překročení znamená detekci pohybu.

3.3 Ostatní metody

Existuje mnoho dalších možných přístupu k detekci pohybu ve video sekvenci. Pro ukázku uveďme metodu sledování optického toku, s kterou se můžeme setkat pod názvem Optical Flow. Její princip je založen na sledování toku-pohybu klíčových pixelů. Na základě poznatků z těchto pohybů můžeme určit směr pohybu, jeho rychlost a také předpovědět budoucí polohu. Metoda není primárně určena pro použití při tvorbě detektoru pohybu,

ale jestliže by jsme použili správně rozvrženou síť sledovaných pixelů, mohli by jsme na základě sledování jejich polohy určit pohyb objektu.

Kapitola 4

Návrh řešení

Hlavním cílem praktické části této práce bylo vytvořit detektor pohybu, který vizuálně identifikuje tu část obrazu, ve které k pohybu dochází [2].

Po prozkoumání a ověření kladů a záporů jednotlivých metod využívaných při tvorbě detektoru pohybu, které jsou blíže popsány v kapitole 3 jsem vytvořil finální strukturu detektoru. Právě ta bude hlavním předmětem této kapitoly.

Základní koncepce detektorů pohybu je následující:

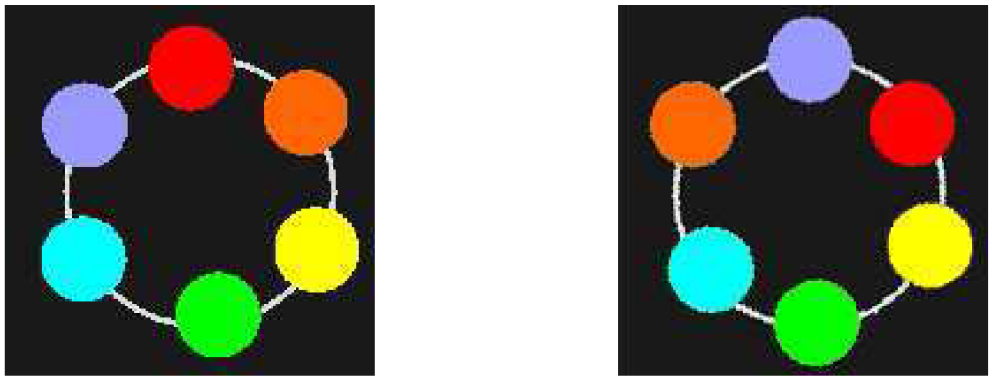
- Zpracování vstupní video sekvence a její rozložení na jednotlivé snímky, z kterých je sekvence tvořena
- Aplikace filtrů na takto získané filtry
- Provedení samotného algoritmu detekující pohybující se objekty
- Vyhodnocení a zpracování výsledku získaného v předchozím bodě.

První část zpracování vstupu nám dává volbu použití vstupní video sekvence získané z připojené kamery, která v online režimu zpracovává snímané prostředí. Druhým přístupem je možnost vyhledávat pohyb v předloženém video záznamu, který byl pořízený v libovolné době. Má aplikace podporuje použití obou dvou typů vstupu. Použití vstupu, který je získáván a ihned zpracováván v reálném čase nachází využití především v nejrůznějších zabezpečovacích zařízení. Jako je například snímání vstupních prostorů, parkoviště, vlakového nástupiště atd. V těchto případech požaduje okamžitou reakci při indikování pohybu, kterou může být spuštění bezpečnostního alarmu, upozornění strojvůdce a zabránění blížícímu se neštěstí. Naopak možnost práce s existujícím ukončeným video záznamem využijeme pro testovací a porovnávací účely. Lze tak porovnávat reakce jednotlivých metod nad stejnými daty, jejich časovou a výpočetní náročnost. Tohoto jsem využil v kapitole 6, kde se zabývám testováním vytvořené aplikace pro detekci a rozpoznání pohybujících se objektů.

Protože všechny metody a algoritmy, které v této práci byly zmíněny pracují s obrazovými body(ať už s jejich jednotlivými body, nebo celkovou charakteristikou) musíme video sekvenci rozdělit na jednotlivé snímky. To není nikterak náročná operace, protože video sekvence je ve své podstatě sekvence snímků, které mezi sebou přecházejí určitou rychlostí. Tato rychlost bývá označována jako FPS(frame per second).

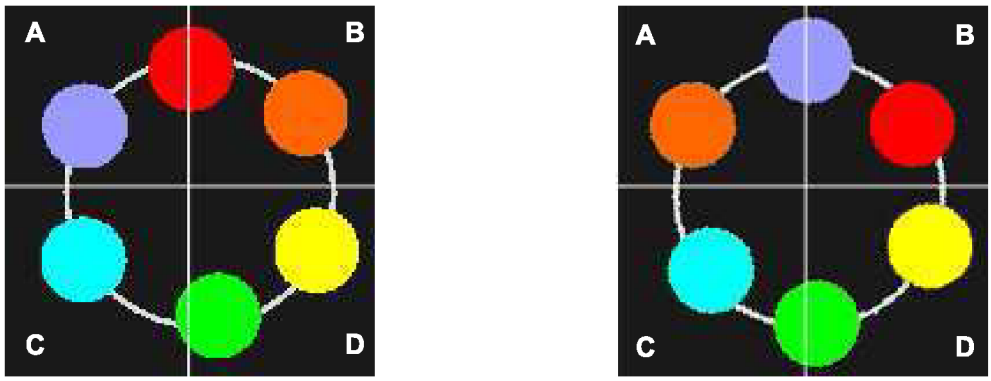
Nyní můžeme přejít ke stěžejní části, kterou je samotné rozpoznání a následné označení pohybu. Chceme-li dosáhnout co nejlepších výsledků, je vhodné využít a zkloubit pozitivní vlastnosti hned několika metod, které mezi sebou budou spolupracovat. Já jsem zvolil

metodu práce s histogramy pro spuštění alarmu, nebo-li prvotní indikaci detekce pohybu. Tyto histogramy vyjadřují celkovou charakteristiku obrazu(viz. kapitola 3.1). Pohybuje-li se objekt uvnitř sledovaného prostoru neumí tato metoda ve své základní podobě rozeznat tento pohyb. Po sobě jdoucí snímky se sice mění, ale celková charakteristika obrazu zůstává zachována. Tento problém ukazuje následují obrázek 4.1 na kterém můžeme porovnat dva obrázky. Je nesporné, že identické nejsou, ale přesto porovnáme jejich barevnou charakteristiku pomocí histogramů, získáme dva identické histogramy. A metoda nám tedy chybně neidentifikuje rozdíl mezi snímky.



Obrázek 4.1: Obrázky nejsou shodné, ale přesto jsou jejich histogramy stejné

Z tohoto důvodu jsem použil pro metodu optimalizaci. Jedná se o rozdělení snímků na více částí. V mém případě na šestnáct stejně velkých částí. Sestavení a porovnání histogramů provádíme pouze mezi odpovídajícími částmi testovaných snímků. Pro příklad uveďme rozdělení na čtyři části našeho problematického obrazce.



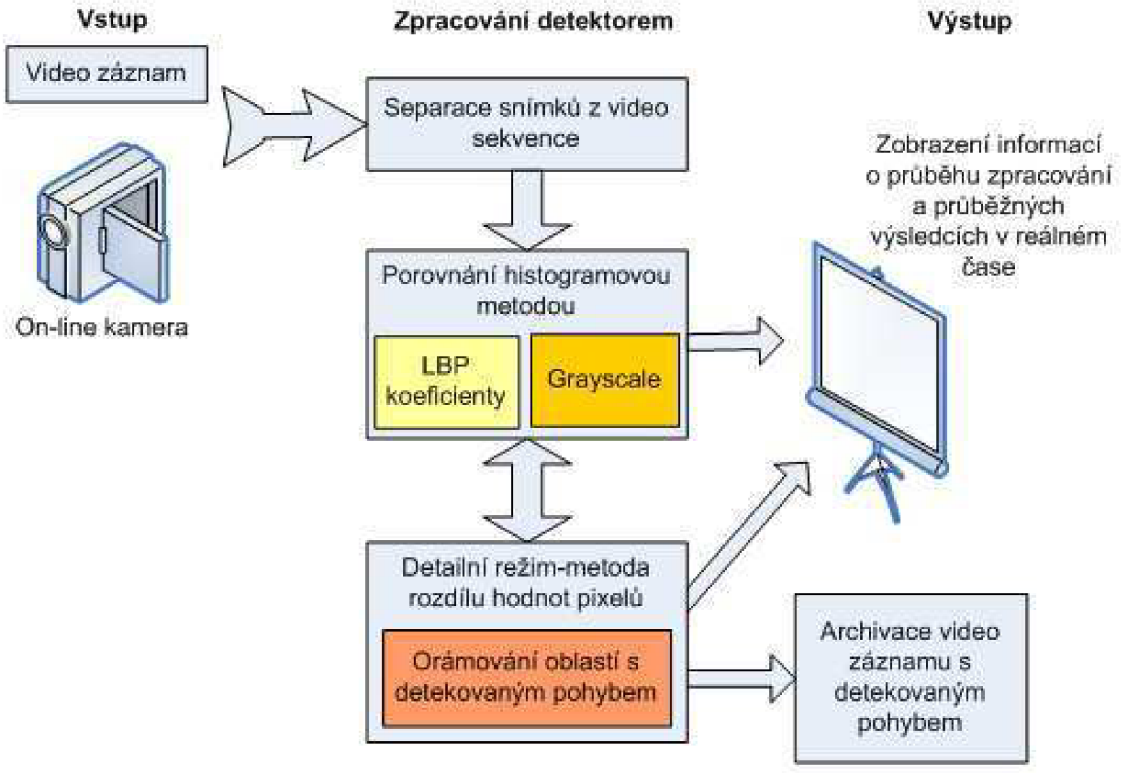
Obrázek 4.2: Rozdělení snímku na části

Pro uvedený příklad sestrojíme histogramy pro všechny čtyři části a porovnáváme je s odpovídajícími histogramy následujícího obrázku. Oblasti označené (c) a (d) generují shodné histogramy v obou uvedených obrázcích. Ovšem oblasti (a) a (b) mají odlišnou barevnou charakteristiku než na vedlejším snímku, což nám umožní detekovat pohyb a spustit alarm. Výsledek porovnání odpovídajících si histogramů(celočíselná hodnota) se porovná s nastaveným prahem pro spuštění alarmu. Je-li větší v libovolné části než hodnota

prahu je spuštěn alarm. Tato realizace nám v optimálním případě, kdy jsou dostatečně odlišné histogramy již v první porovnávané části ušetří výpočetní výkon, protože tato část rovnou spustí alarm a porovnávání výpočtu a testování následujících částí je přeskočeno.

Rozdělení na více oblastí není jediná optimalizace této metody, kterou jsem využil. Toto prvotní porovnávání spouští alarm, kterým je v mém případě pokyn pro start detailního rozpoznání pohybujících se objektů metodou rozdílu jednotlivých snímků. A následného zvýraznění rozdílných oblastí. Při překročení prahu v libovolné části snímku není ihned spouštěn alarm, jak je očekáváno, ale ten je spuštěn až při výskytu určitého počtu po sobě jdoucích snímků, ve kterých histogramová metoda detekovala pohyb. Určení počtu potřebných snímků pro start detailního režimu je dostupné přes nastavení programu. Histogramová metoda neustále běží, je ukončena(respektive přerušena) až přechodem do detailního režimu, nebo samozřejmě také ukončením celé aplikace.

Jak již bylo zmíněno v kapitole 3.1, histogramy mohou reprezentovat v nejjednodušším případě intenzity hodnot jednotlivých bodů snímku převedeného do odstínu šedi. Stejně tak můžou být použity LBP koeficienty, které jsou méně náchylné na změnu celkových světelných podmínek ve sledovaném prostředí. Vytvořený detektor umí pracovat s oběma typy a dává na výběr jeho uživateli, který typ se rozhodne zvolit. Změna této volby je dostupná přes nastavení programu. To je blíže popsáno v příloze.



Obrázek 4.3: Základní schéma vytvářeného detektoru

Druhou metodou, kterou označuji jako detailní režim je porovnávání hodnot jednotlivých pixelů. Pro tuto operaci jsou vstupem obrázky pozadí a popředí, tj. ty, které mezi sebou odečítáme, převedeny do odstínu šedi. Následně je proveden samotný algoritmus, jehož výsledek je obrázek nový, který je shodného rozměru jako vstupní snímky a vyjadřuje

odlišnost sobě odpovídajících si obrazových bodů. Čím více jsou odlišné, tím větší(bližší bíle barvě v modelu RGB) je hodnota nového pixelu. Pro další zpracování potřebujeme obrázek nikoliv ve stupních šedi, ale v binární podobě. Binární podoba snímku znamená, že se v něm vyskytuje pouze barva černá, reprezentovaná hodnotou nula a barva bílá, zastoupena hodnotou jedna. K tomuto převodu byly použito prahování, blíže popsano v 2.2. Takto upravený obrázek je maskou, které v sobě nese pohyb, který je vyjadřován oblastmi v binárním snímku. Vstupní video sekvence a tedy i jednotlivé jeho snímky obsahují bohužel šum, a proto binární maska může obsahovat osamocené pixely, které vyjadřují pohyb i na místech, kde by jsme je nepředpokládali. Na jejich odstranění jsem použil další filtr a tím je filtr zvaný erosion (viz. kapitola 2.2).

Tato upravený snímek, nebo-li maska je využita k označení pohybu. I u této funkce detektoru lze využít několik přístupů. Mohou jimi být obarvení bílých pixelů například na červenou barvu a jejich přiložení na snímek, z kterého byla získána. Nebo lze jednotlivé oblasti nacházející se v masce vyhledat a orámovat. Toto řešení jsem použil při tvorbě detektoru.

Jediným výstupem detailního režimu však není pouze zobrazení vstupní video sekvence s orámovanými částmi, ve kterých dochází k pohybu, ale také archivace, uložení této části záznamu, ve kterém byl detekován pohyb na pevný disk počítače. Vytvořený záznam je opatřen názvem vyjadřující časový okamžik výskytu pohybu. Tato funkce nám nabízí využití vytvořeného detektoru bez nutnosti jeho sledování v reálném čase. Můžeme jej nechat zpracovávat vstupní video sekvenci a on automaticky archivuje intervaly, ve kterých došlo k pohybu a my jej můžeme zpětně přehrát a analyzovat.

Podobně, jako je ošetřen přechod z prvotního porovnávání popředí s pozadím metodou sestavení histogramů, kdy požadujeme pro přechod do detailního režimu výskyt určitého počtu snímků s detekovaným pohybem za sebou, je potřeba pro přechod z detailního režimu do režimu základního opět několika po sobě jdoucích snímků bez pohybu. Tato úprava nám částečně eliminuje nestabilní pohyb v obraze, kdy může být pohyb v krátkém časovém okamžiku omezen, případně pozastaven.

Při opuštění detailního režimu a návrat do základního testování je případný vytvářený video záznam uzavřen a je k dispozici pro další použití.

Práce s video záznamem, nad jehož snímky aplikujeme řadu metod a funkcí je výpočetně náročné. Nároky jsou přímoúměrně počtu zpracovávaných bodů a ty jsou dány rozlišením jednotlivých snímků a rychlostí jejich přehrávání. Z tohoto důvodu detektor umožňuje pracovat pouze s určitými body obrazů. Například z matice 2x2,3x3 atd. se využije pouze jeden pixel a je vytvořen obraz redukovaný. S touto podobou pracují jednotlivé vnitřní algoritmy detektoru a před výsledným výstupem je obraz(v našem případě binární maska vyjadřující pohyb) roztažen na svou původní velikost. Toto vypouštění nám značně urychlí běž celé aplikace.

Kapitola 5

Implementace detektoru

V této kapitole bude popsána implementace detektoru pohybu. Budou zde vysvětleny použité postupy a algoritmy.

Detektor je tvořen jedním programem, respektive jedním spustitelným souborem **detektor.exe**. Ten zpracovává vstupní data podle svého nastavení, které umožňuje přizpůsobení detektoru pro konkrétní situace. Pro jeho vytvoření byl použit jazyk C++ a vývojové prostředí Microsoft Visual Studio 2008.

5.1 Knihovna OpenCV

OpenCV, nebo-li Open Computer Vision Library je volně dostupná knihovna původně vyvíjená firmou Intel s motivací využití vysokých výkonů nových procesorů. Její hlavní zaměření je podpora počítačového vidění pro interakci mezi počítačem a člověkem. Nalezne využití v monitorovacích a bezpečnostních systémech, robotice atd.

Nabízí řadu funkcí pro práci s obrazovými daty, implementuje v sobě nejrůznější filtry. Umožňuje snadnou práci s obrazovými snímky, jejich kopírování, nebo vytváření snímků nových. V projektu byla využita pro zpracování video sekvence, kterou umí rozdělit na posloupnost jednotlivých snímků. Umí pracovat s existujícími video soubory, ale také zpracovávat data přicházející z připojené kamery v reálném čase. Jednotlivé mnou vytvořené funkce implementující metody vedoucí k detekci pohybu knihovnu využívají pro přístup k hodnotám jednotlivých bodů obrazu.

Umožňuje nejenom snímky a video sekvence načítat, ale také vytvářet. Tato skutečnost dovoluje vytvoření záznamu s detekovaným pohybem.

5.2 Knihovna cvBlobsLib

Tato knihovna slouží ke zpracování binárních obrazů, které obsahují pouze dvě barvy. Implementuje algoritmy pro vyhledání spojitých oblastí v obraze. Tyto oblasti bývají nazývány jako „bloby“. Bloby můžeme filtrovat, například pomocí jejich velikosti, kdy můžeme eliminovat bloby nedostatečné velikosti či nevyhovujícího poměru stran.

Při tvorbě detektoru nalezla uplatnění při vyhledání spojitých oblastí a jejího orámování. Výsledkem detailního režimu 3.2 je právě binární obraz, který v sobě obsahuje informace o pohybujících se tělesech. Tento obraz je zpracován pomocí knihovny cvBlobsLib a ta se již postará o vyhledání oblastí vyhovující rozsahu, který je určen v nastavení programu.

5.3 Obecná struktura detektoru

Výkonná část detektoru je obsažena v jediném zdrojovém souboru. Ten obsahuje jednotlivé funkce, které zapouzdřují metody a postupy z kapitoly 4. Implementovány byly také funkce, které se přímo na určení pohybu nepodílejí. Jedná se například o načtení nastavení ze souboru, nebo o upravený převod číselného typu na typ řetězcový.

Hlavním komunikačním rozhraním jednotlivých procedur a funkcí jsou snímky uložené v paměti. Přístup k nim zajišťuje knihovna OpenCV a umožňuje tak snadnou výměnu snímků mezi jednotlivými bloky programu pomocí předávaných referencí a parametrů.

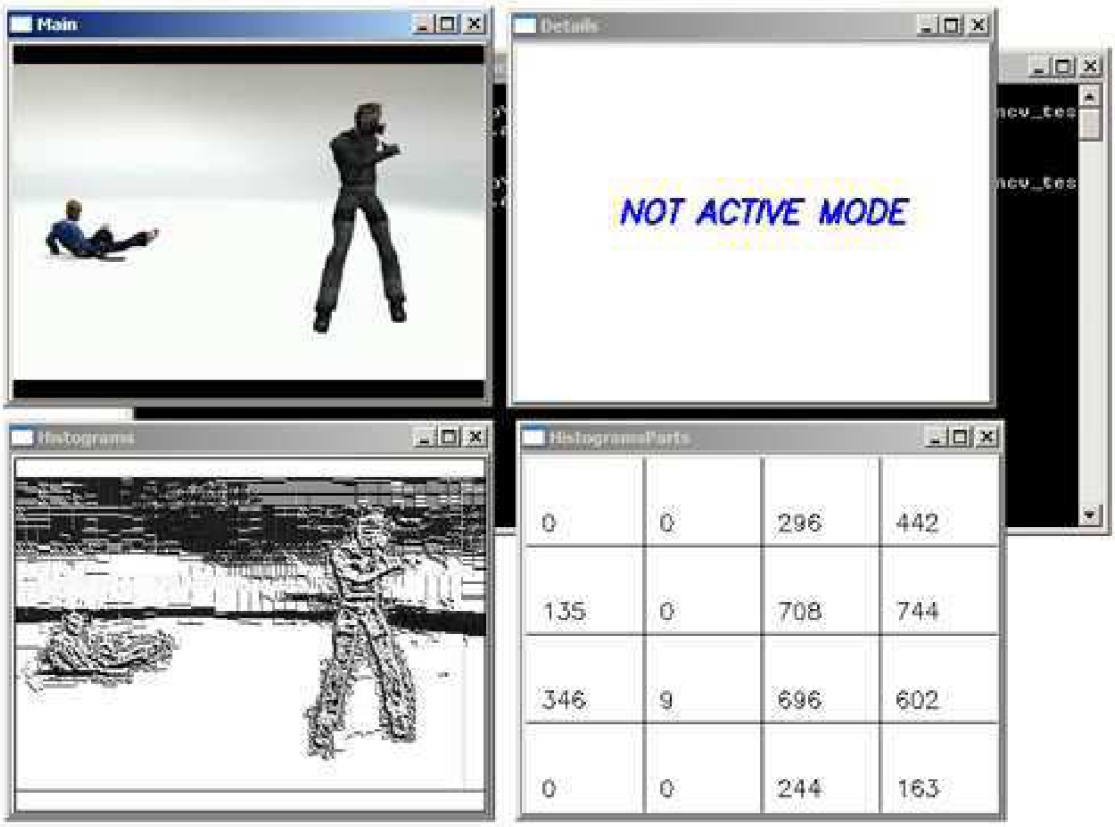
Histogramová metoda je tvořena pomocí funkce pro výpočet histogramu, která pomocí svých parametrů přijímá snímky popředí, pozadí a také informaci o části, pro kterou se budou histogramy tvořit a následně porovnávat. Funkce je volána cyklicky z nadřazeného bloku postupně pro jednotlivé části snímku do té doby, než je výsledek porovnání histogramů v některé z částí větší než nastavená konstanta, nebo je vyčerpáno všech šestnáct částí obrazu.

Základem detailního režimu realizující metodu odčítání jednotlivých snímků vedoucí k označení pohybu jsou funkce grayscale, thresholding a erosion. Ty přijímají vstupní snímek a vracejí odpovídající upravený snímek. Tímto postupem je vytvořena maska pohybu, která je pomocí knihovny cvBlobsLib zpracována. Knihovna vyhledá všechny spojité oblasti v masce, umožní jejich filtraci dle zadané velikosti a umožňuje přístup k souřadnicím jednotlivých oblastí. Ty jsou podkladem pro výsledné orámování.

5.4 Ukázka a práce s programem

Na následujícím obrázku je zobrazen vytvořený detektor, který zpracovává vstupní video sekvenci a nachází se v základním režimu. Program zobrazuje okna znázorňující výsledky jednotlivých využívaných metod detekce pohybu. Jejich zobrazení lze za běhu programu měnit stisknutím klávesy **w**.

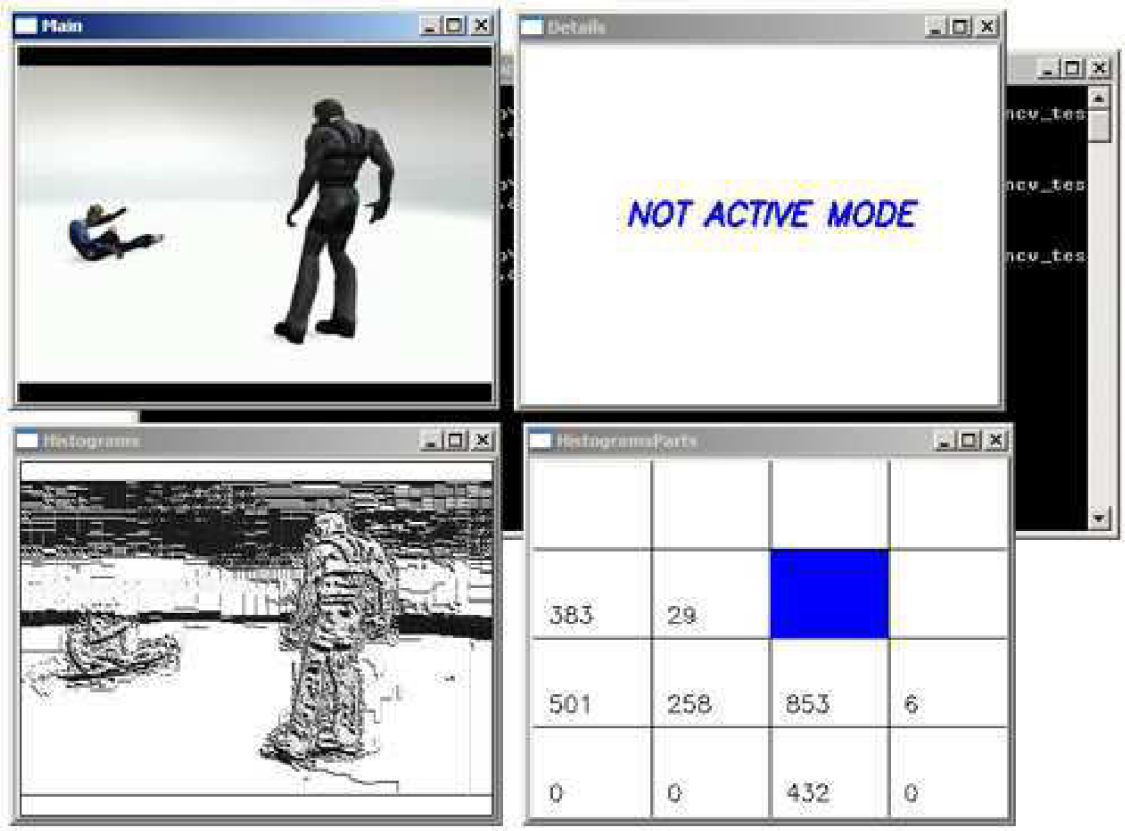
Okno s titulkem **main** je základním výstupem. Přehrává zpracovávanou video sekvenci a v případě detekce pohybu je zde tento pohyb označen a celý obraz je opatřen červeným orámováním. Okno s titulkem **details** ukazuje masku detailního režimu. Protože v příkladu zobrazeném na obrázku 5.1 se detektor nachází v základním, histogramovém režimu okno detailního režimu nezobrazuje žádné informace. Je v něm zobrazen text „not active mode“. Další dvě okna jsou již věnovány základnímu režimu. První z nich **histograms** zobrazuje připravené pozadí, které je testováno s popředím. Jedná se o průměr posledních tří snímků v odstínu šedi, nebo v LBP koeficientech. Ty jsou použity v naší ukázce. Okno s titulkem **histogramsParts** vizuálně ukazuje míru odlišnosti v jednotlivých částech obrazu, která byla vypočtena hitogramovou metodou. Této volby doporučuji využít pro určení správného nastavení alarmu, který je testován právě s těmito hodnotami zobrazovanými v každé z části snímku. Jeho překročení vede ke startu detailního režimu.



Obrázek 5.1: Detektor v základním režimu

Obrázek 5.2 zaznamenává detektor opět v základním režimu, kde v prostřední části testovaného snímku byla hodnota vzniklá porovnáváním jeho histogramu s pozadím větší než nastavený prah, to vede ke startu detailního režimu. O této skutečnosti je uživatel infor-

mován vybarvením patřičného obdélníku. Histogramy pro další části obrazu není potřeba počítat, protože základní režim není využíván k přesné detekci místa pohybu, ale pouze k určení zda k němu v jeho libovolné části došlo.

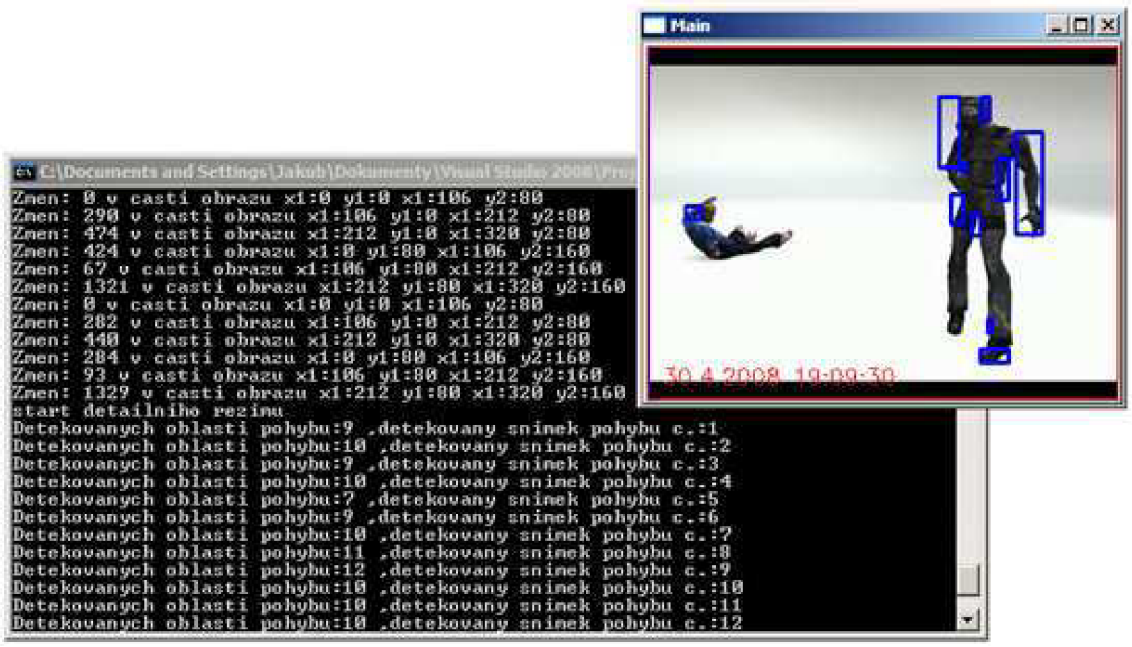


Obrázek 5.2: Detektor v základním režimu, který právě detekoval pohyb

Třetí ukázka zachycuje detektor v detailním režimu se zapnutým zobrazením informačních oken. Protože se nacházíme v detailním režimu okna **histograms** a **histogramsParts** nejsou využity. Okno **details** ukazuje aktuální binární masku, která je využita pro vyznačení místa pohybu. Výsledné označení vidíme v hlavním okně **main**. O tom, že se nacházíme v detailním režimu nás informuje také hlavní okno **main** a to pomocí celkového červeného orámování snímku.

Obrázek 5.3: Detektor v detailním režimu

Další možností zobrazení informací o průběhu zpracování je textový výpis pomocí konzole. Aktivovat jej lze přes nastavení programu, nebo stisknutím klávesy **i** za běhu programu. Jak může takový výpis vypadat ukazuje obrázek 5.4. Pomocí textového výpisu jsme informování o počtu oblastí a počtu zpracovaného snímku v detailním režimu, nebo o míře rozdílu jednotlivých částí obrazu v režimu základním.



Obrázek 5.4: Detektor a jeho textový výpis

Kapitola 6

Testování detektoru pohybu

Tato kapitola je věnována testování vytvořeného detektoru pohybujících se objektů. Jsou v ní popsány typově různé vstupní video sekvence, které nejlépe charakterizují možnosti využití implementovaného detektoru. Všechny zde ukázané příklady jsou obsaženy na přiloženém CD nosiči v archivu se jménem **ukazka.zip**. Tento archiv je potřeba rozbalit na libovolné místo na disku, kde bude mít detektor práva k zápisu nových souborů.

6.1 Pohybující se postava

Vstupní video sekvence obsahuje pohyb postavy, ta se na začátku video sekvence ve snímaném prostředí nevyskytuje a vstupuje do ní později a následně ji opustí. Tato akce se opakuje dvakrát. Video sekvence je v rozlišení 352x240 pixelů a rychlost snímků je 15 snímků za sekundu. Na přiloženém CD nosiči tento příklad nalezneme v adresáři **pohybPostavy**, který obsahuje soubory **runClassic.bat** a **runInformationMode.bat** pro spuštění detektoru ve standardním zobrazení a v zobrazení s rozšiřujícími okny.

Do okamžiku vstupu postavy do snímaného prostředí základní histogramová metoda, pracující v tomto případě s charakteristikou intenzit pixelů v jednotlivých šestnáctinách obrazu, žádný pohyb nedetekuje. Detekuje jej až při vstupu postavy do prostředí a předává tímto zpracování detailnímu režimu, který pohybující se objekty graficky vyznačí.

Pohybující se postava projde ze strany na stranu, aby následně prostředí opustila. Po opuštění detailní režim nenajde žádné pohybující se předměty, a proto vrací řízení zpět režimu základnímu. Postava po krátké přestávce, po kterou se v testovaném prostoru nevyskytuje žádný pohyb, opět vstupuje do záběru. Detektor tento okamžik odhalí a opět pohybující se objekt graficky vyznačí.

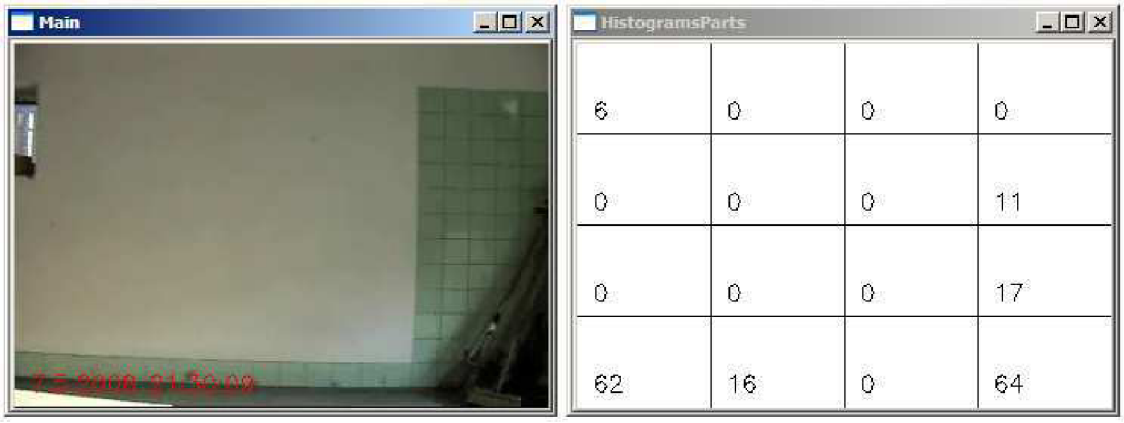
Následující tabulka 6.1 charakterizuje klíčové okamžiky v testované video sekvenci.

Časové okamžiky [sekund]							
0	4,3	7,5	11,8	16,5	20	24	28
začátek video sekvence	vstup postavy do snímaného prostředí	změna pohybu postavy	opuštění snímaného prostoru	vstup postavy do prostoru	změna pohybu postavy	opuštění prostoru postavou	konec

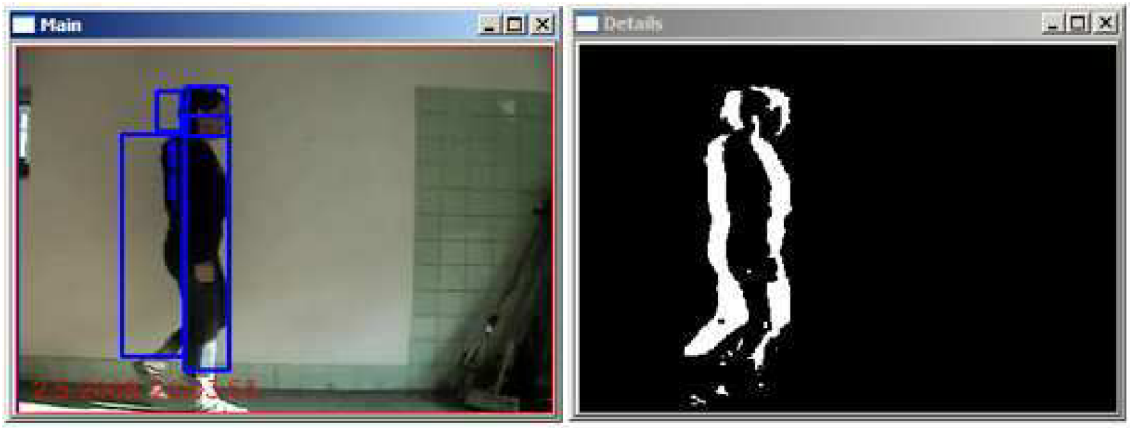
Tabulka 6.1: Klíčové okamžiky diskutované sekvence

Obrázek 6.1 zachycuje detektor zpracovávající diskutovanou video sekvenci krátce po spuštění. Před vstupem postavy. V této situaci není žádný pohyb detekován. Obrázek ukazuje informační okno, které zobrazuje výsledky zpracování základní histogramovou metodou. Můžeme na něm vidět, že nejvíce změn detekuje v pravém spodním rohu a to na úrovni 64.

Prah pro puštění alarmu byl stanoven pomocí nastavení programu na hodnotu 130. To je dostatečně velká rezerva pohlcující šum a jiné zkreslení, které by mohlo vyvolat nechtěnou detekci pohybu. Ale zároveň ihned po vstupu postavy do snímaného prostředí je tato hodnota překročena a následně pomocí detailního režimu ohraničena. Tuto skutečnost ukazuje obrázek 6.2. Pravé okno ukazuje masku, která byla vytvořena jako podklad pro vizualizaci pohybujícího se objektu.



Obrázek 6.1: Snímek bez výskytu pohybující se postavy



Obrázek 6.2: Detekovaný pohyb pohybující se postavy

Detektor pro pro tento příklad produkuje očekávaný výstup. Pohybující se objekt se pohybuje dostatečnou rychlostí a zabírá téměř celou výšku obrazu. To, s přispěním poměrně velkého kontrastu mezi pozadím a pohybující se postavou, jsou dobré podmínky pro detektor pohybu.

6.2 Prostředí snímané chvějícím se zařízením

O tom, že vytvořený detektor nalezne využití v podmínkách, které nenabízí stabilní snímaný obraz nás přesvědčí v pořadí druhý test. Vstupní video sekvence zachycuje silnici po které projede vozidlo. Silnice je snímaná dosti nestabilní kamerou, což může simulovat bezpečnostní kameru, která se vlivem nejrůznějších otřesů chvěje. Detektor chvění správně ignoruje a jako pohyb vyhodnotí až samotný průjezd automobilu.

Příklad se nachází na přiloženém CD v adresáři **nestabilniKamera** a spustíme jej pomocí **runClassic.bat**. Vstupní videosekvence je 15 sekund dlouhá a opět byla vytvořena v rozlišení 352x240 obrazových bodů s rychlostí 15fps.

Obrázek 6.3 ukazuje vybrané časové okamžiky testované vstupní sekvence. Jednotlivé snímky byly pořízeny v chronologickém pořadí zleva doprava. Obrázky nedokážou přesně vystihnout podstatu tohoto testu(snímající zařízení se chvěje), ale zaměříme-li se na pravý spodní okraj tří zobrazených snímků nalezneme v něm odlišnosti, které jsou důsledkem pohybu snímacího zařízení.

Obrázek 6.3: Detekovaný pohyb pohybující se postavy

Příklad využívá výhody histogramových metod(blíže v kapitole 3.1). Vytvořený detektor tuto metodu aplikuje na jednotlivé části porovnávaných snímků. Částí je 16 a všechny jsou shodných rozměrů, pro rozlišení 352x240 je to rozměr právě 44x30. Třepe-li se kamera v rozsahu $\pm$ 10 pixelů jedná se zhruba o posuv na úrovni do 20%, což neprodukuje příliš velkou změnu jasové charakteristiky této části, a proto kmitání nevyvolá překročení alarmového prahu v žádné své části. Čím menší jednotlivé části vytvoříme, tím více budou náchylné na změny svého jasového histogramu vlivem třepání snímacího zařízení.

Soubor s nastavením, který je s tímto příkladem dodán je určeno, že základní histogramový režim bude vytvářet histogramy pouze z odstínu šedi. Alarmový prah pro přepnutí do detailního režimu byl stanoven na konstantu 160. Detailní režim je aktivován v případě, že režim základní detekuje v libovolné šestnáctině obrazu rozdíl větší než alarmový prah u dvou po sobě jdoucích snímků. Tento počet je opět určen v souboru nastavení pod položkou *startDetailDelay*, která může nabývat celočíselnou hodnotu větší nebo rovno nule.

6.3 Přibližující/vzdalující se objekt

Tento test ukazuje reakci detektoru na objekt, který je při zahájení video sekvence součástí pozadí a je umístěn v blízkosti snímacího zařízení. Od tohoto zařízení se začíná vzdalovat. Vzdaluje se až do té chvíle, než dorazí na krajní polohu snímaného prostoru, kde se zastaví, aby se následně opět začal pohybovat cestou zpět, po kterou se k video kameře přibližuje.

Následující obrázky 6.4 zaznamenávají výsledný výstup detektoru. Na prvním obrázku zleva je zachycen první snímek vstupního video záznamu.

Obrázek 6.4: Objekt se oddaluje od kamery

Při zahájení zpracování video záznamu detektor správně žádný pohyb nedetekuje, protože k žádnému ani nedochází. Dojde k němu až posléze, kdy se model automobilu rozjede směrem od kamery. Protože v tuto chvíli objekt zabírá podstatnou část snímané výseče, není pro základní histogramovou metodu problém tento pohyb detekovat a předat řízení detailnímu režimu, který se snaží o vyznačení pohybu metodou odečítání po sobě jdoucích snímků pozadí a popředí. Tento stav ukazuje prostřední obrázek. Model se postupně přibližuje ke konci snímané výseče, kde na okamžik zastaví. Ve chvíli, kdy se pohybující auto přibližuje k tomuto okraji a vzdaluje se od snímajícího zařízení je již podstatně malé a zároveň jeho trajektorie je konstantní, což vede k předčasnému ukončení detekovaného pohybu. Tato nepříjemná skutečnost je způsobena malým rozdílem po sobě jdoucích obrazů video sekvence a detailní metoda, která je v tuto chvíli aktivní po odečtení popředí a pozadí vytvoří masku pohybu, která neobsahuje dostatečně velké bloky vyjadřující rozpoznaný pohyb, a proto je ukončen. Řízení je předáno zpět režimu základnímu.

Model se po krátké přestávce rozjede zpět směrem ke kameře. Pohyb není detekován ihned, ale až po krátké chvíli. To je opět způsobené nedostatečným pohybem. Vozidlo se v začátcích zpětného pohybu vyskytuje pouze v jedné z šestnácti částí, ve kterých histogramová metoda testuje pohyb, a proto pohyb nemění charakteristiku v žádné z částí. Až vozidlo vstoupí do další z části snímku je pohyb detekován a opět vyznačen.

Vozidlo svůj pohyb opakuje, opět se vzdálí od snímané kamery, ale nyní u okraje zatočí a pohybuje se horizontálně vůči snímanému zařízení. Detektor tento pohyb správně odhalí a vyznačí, jak ukazuje následující příklad na obrázku 6.5.

Obrázek 6.5: Objekt měnící trajektorii pohybu

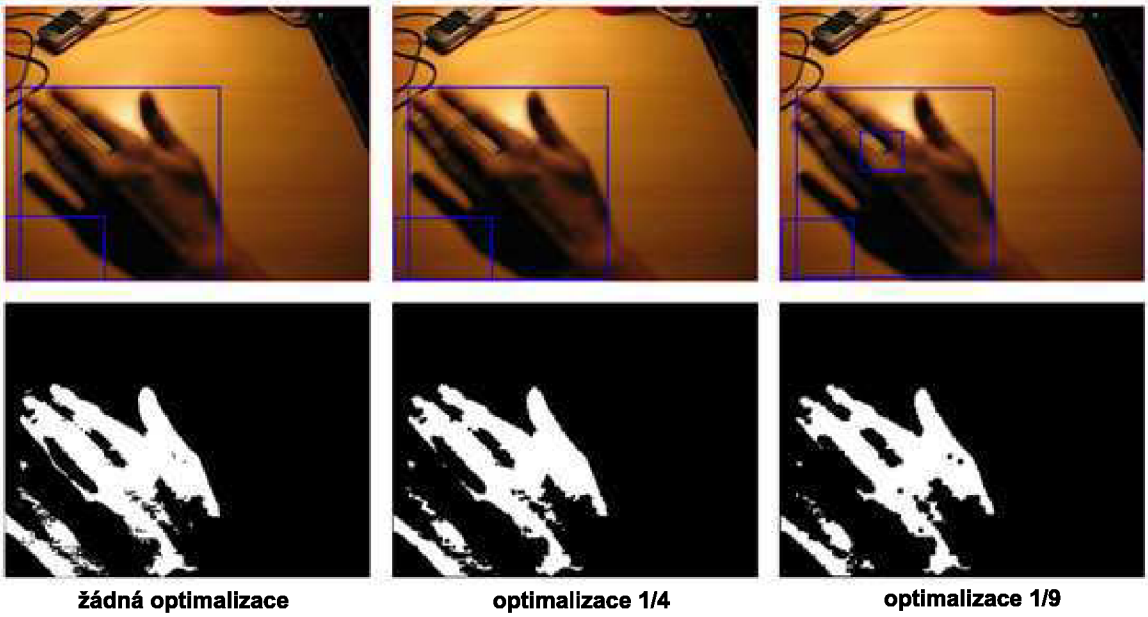
Příklad je umístěn na přiloženém CD v adresáři **priblizeniOddaleni**. Vstupní video sekvence je rozlišení 320x240 pixelů a rychlost snímků je 15fps. Spustit jej můžeme standardně souborem **runClassic.bat**, respektive **runInformationMode.bat** pro zobrazení výsledků interních algoritmů detektoru.

6.4 Optimalizace a výpočetní nároky

Optimalizace a její využití je hlavním tématem tohoto testu. Skutečnost, že vstupní video záznam může být různé velikosti, snímkové frekvence a také výpočetní výkony počítačů jsou značně odlišné mě vedla k vytvoření optimalizace, která zrychlí samotnou práci detektoru a zároveň zachová dostatečnou kvalitu svých výstupů. Optimalizace funguje na principu vypouštění určitých bodů. Tato vlastnost je volitelná přes nastavení programu. Nastavená hodnota optimalizace 1 znamená, že implementované algoritmy a metody budou pracovat s hodnotou každého z pixelů vstupní video sekvence. Je-li nastavena tato konstanta na hodnotu 2 znamená to, že bude pracováno pouze s pixelem na každém lichém řádku a lichém sloupci. To znamená, že z matice 2x2 pixelů se bude opracovat pouze s jedním pixelem, což nám značně urychlí běh detektoru.

Pro ukázku jsem použil video jednou tolik větších rozměrů(640x480 pixelů) ve srovnání s ostatními uvedenými testy. Nároky na zpracování videa těchto parametrů značně stoupají. Video bylo testováno s postupným nastavením optimalizace na hodnoty 1(žádná optimalizace), 2(1 pixel z 4) a 3(1 pixel 9) a byly sledovány výsledky detekce pohybujících se objektů a také čas, který detektoru zabralo zpracování vstupu. Testované příklady v adresáři **optimalizace** na CD, které je přílohou této práce. Soubory **runNoneOptimalization.bat**, **runFrstOptimalizazion.bat** a **runScndOptimalizazion.bat** spustí detektor se shodným vstupním souborem postupně s žádnou optimalizací, 1 pixel z 4 respektive 1 pixel z 9. Pro každou z těchto optimalizací byl upraven soubor nastavení tak, aby detektor dosahoval co nejlepších výsledků. Jednalo se především o úpravu alarmového prahu pro základní metodu a minimální velikost akceptovatelných oblastí v masce reprezentující pohybující se objekty.

Obrázek 6.6 zaznamenává výstup detektoru pro shodný snímek vstupní sekvence s různým nastavením optimalizace. Ve spodní řadě jsou zobrazeny masky s detekovaným pohybem. Maska vytvořená detektorem bez optimalizace je nejjemnější. Je to způsobeno tím, že např. u optimalizace uvedené v prostředním sloupci byla maska vytvořena z redukovaného snímku na jednu čtvrtinu původního rozlišení. Takto redukovaný snímek projde algoritmy a po dokončení je roztažen na původní velikost, pomocí kopírování pixelů na své sousedy.



Obrázek 6.6: Porovnání optimalizací

Tabulka 6.2 ukazuje dobu zpracování testované sekvence detektorem s různou volbou optimalizace.

	Doba zpracování [s]
Přehrání videa(skutečná délka)	2,3 (35 snímků)
Detektor s žádnou optimalizací	18
Detektor s optimalizací 1z4	8
Detektor s optimalizací 1z9	3,6

Tabulka 6.2: Porovnání optimalizací detektoru

Uvedené hodnoty byly naměřeny na notebooku s procesorem Intel Core Duo 1,73Ghz, 1024MB Ram a operačním systémem MS Windows XP.

6.5 Souvislý vstup a archivace detekovaných pohybů

Poslední uvedený příklad slouží k prezentaci automatické archivace detekovaných pohybů ve vstupním souvislém video záznamu. Pro tyto testovací účely jsem vytvořil přibližně pěti minutové video, na kterém je zachycena silnice s postupně deseti pojíždějícími vozidly. Tento video záznam odpovídá záznamům pořízených z různých bezpečnostních a průmyslových kamer, které neustále snímají daný prostor.

Detektor správně označí projíždějící vozidla(i jednoho chodce) a vytvoří video záznamy obsahující tento pohyb. V souboru s nastavením, které je přichystáno pro tento příklad je určeno, aby detektor archivoval, nebo-li vytvářel nové video sekvence, které obsahují pouze detekovaný pohyb. Video sekvence jsou ukládány do adresáře **output** a jsou opatřeny názvem, který reprezentuje datum a čas, kdy k pohybu došlo. Po dokončení zpracování vstupu detektorem nalezneme v adresáři jedenáct souborů, které zachycují všechny průjezdy automobilu včetně jednoho průchodu chodce.

Obrázek 6.7 zachycuje zleva prostředí bez detekovaného pozadí, první a druhý obrázek průjezd automobilu.

Obrázek 6.7: Prostředí snímané průmyslovou kamerou

Všechny potřebné soubory k provedení tohoto testu nalezneme ve složce **prumyslovaKamera** na CD, jež je přiloženo k této práci. Samotný test spustíme souborem **runNoneOptimalization.bat** pro zpracování bez optimalizace. Jestliže disponujeme nižším výpočetním výkonem a detektor nestíhá v reálném čase zpracovávat video můžeme spustit detektor s optimalizací 1 pixel ze 4 souborem **runFrstOptimalization.bat**. Výstupní video soubory budou vytvořeny v adresáři **output**.

Kapitola 7

Závěr

Testování vytvořeného detektoru pohybu je věnována šestá kapitola. Ta ukazuje, že vytvořený detektor správně detekuje pohybující se objekty ve vstupní video sekvenci. Je poměrně odolný na nedokonalosti jeho vstupu, ktré jsou způsobené šumem, chvěním zaznamenávajícího zařízení atd. Ovšem možnosti jeho dalších optimalizací a rozšíření zůstávají otevřené.

Detektor využívá dvě základní metody. Metodu histogramovou a metodu rozdílnosti jednotlivých bodů, jako podklad pro vyznačení pohybu. Pro metodu histogramovou byl testovaný snímek rozdělen na 16 rozměrově stejně velkých částí. Protože v praktickém využití většinou očekáváme pohybující se objekty pouze v určité části snímané výseče, bylo by vhodné vytvořit rozdělení snímku na jednotlivé části, které by zohledňovaly tuto skutečnost. V místě, ve kterém pohyb očekáváme by části byly menší a hustější, než ve zbytku snímku, kde pohyb neočekáváme.

V detailním režimu lze experimentovat s vytvořenou binární maskou, která obsahuje oblasti pohybu. Ty jsou jednotlivě orámovány. Pro dosažení lepšího vizuálního výsledku orámování by bylo vhodné některé oblasti sjednotit a až ty orámovat. Případně označit konkrétní objekty, které se v testované video sekvenci pohybují. Sledovat jejich pohyb apod.

Během tvorby detektoru jsem si vyzkoušel práci s obrazovými daty získanými z video sekvence. Jejich zpracování, chování jednotlivých obrazových transformací a detekčních metod, které jsem také teoreticky popsal v této textové práci. Problematika detekce pohybu je značně rozmanité téma, které nabízí možnosti různých přístupů, kdy nelze přesně rozlišit, který přístup je správný a naopak. Při jeho tvorbě musíme zohlednit nasazení, pro které je detektor konstruován, ale také výpočetní výkon, který máme k dispozici a počet informací obsažených ve vstupní video sekvenci.

Literatura

[1] Wikipedie: Otevřená encyklopedie. Grayscale.
http://en.wikipedia.org/wiki/Grayscale.

[2] Andrew Kirillov. Motion detection algorithms.
http://www.codeproject.com/KB/audio-video/Motion_Detection.aspx.

[3] Matti Pietikäinen Marko Heikkilä. A texture-based method for modeling the background and detecting moving objects.
http://www.ee.oulu.fi/mvg/publications/show_pdf.php?ID=662.

Seznam příloh

Příloha 1. CD

Struktura přiloženého CD je tvořena následujícími adresáři a soubory:

Ukázky.zip: Všechny testovací příklady, které jsou obsaženy v kapitole 6 a několik dalších. Archiv je potřeba rozbalit na pevný disk z toho důvodu, že při spuštění detektoru pomocí přiložených automatizovaných skriptů dochází ke kopírování příslušného souboru s nastavením, což na CD nelze realizovat.

Teoretická část: Obsahuje tuto textovou část práce a kopii zadání.

Plakát Obsahuje stručný plakát reprezentující cíle a výsledky bakalářské práce.

Zdrojové kódy: Všechny zdrojové kódy, včetně použitých knihoven. Kompletní detektor je obsažen v projektu MS Visual Studio včetně nastavení kompilace, který se otevře pomocí souboru **detektor.sln**

Programová dokumentace: Programová dokumentace obsahující komentáře k jednotlivým funkcím a proměnným vyskytujících se ve zdrojových kódech. Byla vygenerována pomocí nástroje Doxygen, je ve formátu html a hlavním souborem je index.html.

Příloha 2. Ovládání programu

Program je realizován jako konzolová aplikace, která využívá pro zobrazení svého výstupu systémových oken. Spustit jej můžeme z příkazové řádky zadáním názvu jeho spustitelného souboru. Prvním paremetrem je cesta k video záznamu, který si přejeme zpracovat. Tento parametr je nepovinný a není-li vyplněn detektor předpokládá práci s připojenou on-line kamerou.

```
detektor.exe [input video file]
```

Nastavení, podle kterého bude detektor vykonávat svou činnost je uvedeno v souboru **settings.ini**. Ten se nachází ve shodném adresáři, ze kterého je detektor spuštěn.

Určité nastavení lze změnit přímo za běhu programu pomocí stisknutí patřičných kláves, jsou jimi

k: Ukončení programu

p: Pauza, nebo-li pozastavení zpracování vstupu. Jestliže je program přerušen způsobí jeho pokračování od místa ve kterém byl pozastaven.

w: Přenutí mezi zobrazním a skrytím informačních oken

i: Zobrazení/skrytí textového výpisu do konzole, ve které byl detektor spuštěn

Příloha 3. Nastavení programu

Je zapsáno a při startu detektoru načítáno ze souboru **settings.ini**. Struktura tohoto souboru je následující:

```
[motionDetectionSettings]
alarmThreshold=1200
outputVideo=true
outputWindows=false
outputText=false
typeDetection=1
startDetailDelay=3
endDetailDelay=3
optimalization=1
minimumSizeBlobs=500
```

Uvedené řádky slouží k nastavení:

alarmThreshold: Celočíselná hodnota. Udává alarmový práh. Základní histogramová metoda porovnává výsledek jednotlivých částí právě s touto hodnotou. Je-li větší než alarmový práh je snímek vyhodnocen jako snímek obsahující pohyb. Jestliže takto dojde k detekci určitého počtu po sobě jdoucích snímku předá se řízení detailnímu režimu.

outputVideo: Chceme-li, aby detailní režim ukládal tu část video sekvence v které detekoval pohyb nastavíme hodnotu na *true*. Uložené video soubory se nachází v adresáři *output* detektoru s názvem, který je tvořen z datumu a času, kdy došlo k zahájení ukládaného pohybu.

outputVindows: Urřuje zda budou po spuštění detektoru otevřeny informační okna. Hodnota *true* říká ano a hodnota *false* zase ne.

outputText: Zobrazení textových informací o průběhu zpracování detektoru do systémové konzole. Hodnota *true* říká ano a hodnota *false* zase ne.

typeDetection: Celočíselná hodnota. Nastavuje typ histogramové metody. Číslo 1 říká, že histogramy budou tvořeny z LBP koeficientů popředí a pozadí a jiná hodnota, že budou počítány pouze z intenzit odstínu šedi těchto dvou porovnávaných snímků.

startDetailDelay: Číselná hodnota, která udává kolik je potřeba výskytu po sobě jdoucích snímků v základním režimu s detekovaným pohybem pro přechod do režimu detailního.

endDetailDelay: Obdoba předchozí položky nastavení, ale s tím rozdílem, že udává počet po sobě jdoucích snímků, ve kterých nebyl detekován pohyb pro opuštění detailní režimu a návrat do režimu základního-histogramového.

optimalization: Celočíselná nezáporná hodnota udávající použitou optimalizaci. Hodnota 1 udává, že se budou procházet všechny obrazové body, při nastavené hodnotě 2 se z matice 2x2 pracuje pouze s jedním bodem, při hodnotě 3 se v matici 3x3 využívá jediný pixel atd.

minimumSizeBlobs: Udává v pixelech minimální velikost oblastí binární masky detailního režimu, které budou akceptovány pro následné orámování.