VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMEDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SMART CAR: AUTOMATICKÁ DETEKCE VOZIDEL

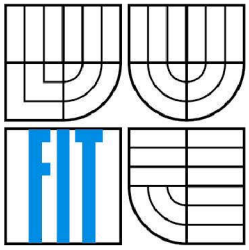
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MARTIN BURKOT

BRNO 2009

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

SMART CAR: AUTOMATICKÁ DETEKCE VOZIDEL

SMART CAR: AUTOMATIC CAR DETECTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MARTIN BURKOT
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÍTĚZSLAV BERAN
SUPERVISOR

BRNO 2009

Abstrakt

Tato bakalářská práce se věnuje detekci pohybujících se vozidel v sekvenci obrázků. V úvodu je proveden stručný rozbor současných metod pro detekci vozidel a pohybu ve scéně obecně. V dalších kapitolách je navržena a popsána implementace detektoru pohybujících se vozidel v obraze založeném na určování optického toku. V závěru je provedeno zhodnocení daného řešení.

Abstract

This bachelor thesis deals with the detection of moving vehicles in image sequence. In the introduction is made a brief analysis of current methods for detecting the movement of vehicles and the scene in general. In subsequent chapters is designed and described the implementation of the detector moving vehicles in an image based on the determination of optical flow. At the end there review of proposed solution.

Klíčová slova

detekce vozidel, detekce pohybu, optický tok, OpenCV

Keywords

vehicle detection, motion detection, optical flow, OpenCV

Citace

Příjmení Jméno: Název práce v jazyce práce, bakalářská práce, Brno, FIT VUT v Brně, rok

SMART CAR: Automatická detekce vozidel

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vítězslava Berana. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Martin Burkot
18. květen 2009

Poděkování

Děkuji Ing. Vítězslavovi Beranovi za ochotu a odborné vedení při psaní této práce.

Obsah

1 Úvod

Jednou ze základních vlastností člověka je vidění. K tomuto smyslu přistupujeme se stejnou samozřejmostí, jako k rozeznávání jednotlivých objektů, které vidíme.

Pro počítačové vidění je nejen rozeznání, ale i detekce různých objektů obrovským problémem.

Ve své práci se pokouším v sekvenci snímků o detekci vozidel. Ne však všech vozidel, ale pouze vozidel, která jeví pohyb. Takováto aplikace fungující v reálném čase by mohla do budoucna pomoci vyvarovat se mnoha dopravním nehodám.

V první části své práce se zaměřuji na již známé metody detekce vozidel. Také zde uvádím jednotlivé metody, které ve své práci používám. Nejvíce se budu zabývat optickým tokem, protože se pokouším právě o detekci vozidel dle optického toku. V další části této práce se pokusím o návrh aplikace pro detekci vozidel založené na určování optického toku. V další kapitole se pokusím popsat implementaci takto navržené aplikace. Poté se pokusím provézt testy na vytvořené aplikaci a zaměřit se především na její nedostatky zhodnocené v poslední části této práce.

2 Metody detekce vozidel

Ve své práci se budu zabývat detekcí vozidel v sekvenci snímků. V dnešní době existuje velmi mnoho různých přístupů k vyřešení tohoto problému, mnohé z těchto přístupů jsou zaměřeny na použití speciálního hardwaru, jako například laserů. Příkladem přístupu k detekci vozidel pomocí speciálního hardwaru je [1]. Já se však snažím o detekci vozidel v obraze, založeném na počítačovém vidění. Jedním z příkladů detekce vozidel, založeném na počítačovém vidění je [2]. V této práci je také použita metoda optického toku. Na rozdíl od mé práce je zde však optický tok použit jako druhořadý. Jak již bylo řečeno, ve své práci se zaměřuji na detekci vozidel dle optického toku. Výhodou tohoto přístupu je to, že detekuji pouze vozidla, která jsou něčím specifická. Touto specifikací je jejich pohyb.

Vzhledem k tomu, že často pracuji s histogramy, uvedu v této kapitole i bližší popis tohoto pojmu a práce s ním.

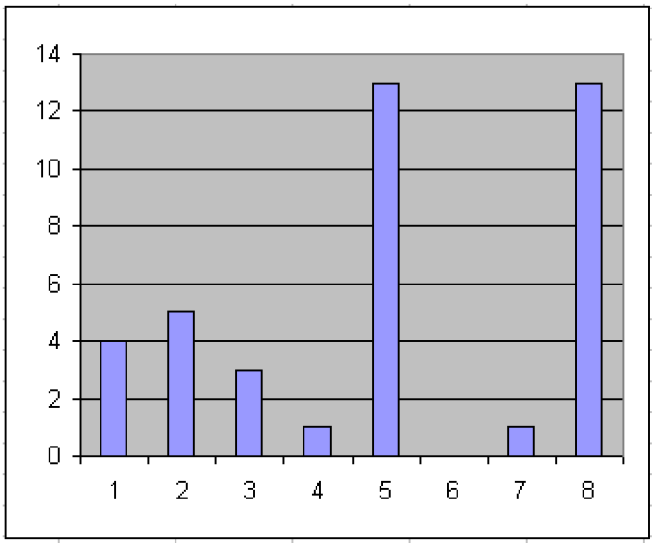
Pokud se snažíme nalézt. Objekty v obraze vždy musíme použít segmentaci obrazu, proto v této kapitole uvedu některé používané metody segmentace obrazu. Objekty ve videu lze rozdělit také dle toho, zdali se pohybují či nikoli. Pohybující se objekty nesou v rámci dvou snímků informaci o svém pohybu. Na tuto pohybovou informaci se zaměřují metody detekce pohybu.

Většina metod je zaměřena na detekci rozdílů mezi dvěma, nebo více snímky. První snímek označí jako pozadí a druhý jako popředí. Poté porovná snímky pozadí a popředí, pokud došlo ke změně je v obrazu nalezen pohyb. Problém nastává v případě, že mezi obrazy zobrazující statickou scénu dojde například ke změně jasu. Pro lidské vnímání není změna jasu problémem, protože zkoumá pouze významné změny obrazu. Počítačové vidění zkoumá každý bod obrazu a vyhodnotí změnu na obrazech, které vyhodnotí lidské vnímání za shodné. Přestože ve své práci používám detekci pohybu založenou na optickém toku, uvedu zde pro srovnání také jiné metody detekce pohybu.

2.1 Histogram

Ve statistice je histogram graf zobrazení četnosti daného jevu, zobrazeného jako sloupce. Histogram zobrazuje, které výskyty daného jevu spadají do jednotlivých kategorií. Jednotlivé kategorie jsou nepřesahující intervaly dané veličiny, nejčastěji stejné velikosti. Pokud použijeme intervaly o různé velikosti, stává se histogram méně pochopitelným. Tyto intervaly musí být přiléhající, aby nedošlo ke ztrátě dat a tudíž nesmyslnosti údajů zobrazených daným histogramem[3].

Histogramy jsou používány k zobrazení hustoty výskytu daného jevu. Při tvorbě histogramu je dobré převézt jej do normovaného tvaru. U normovaného tvaru histogramu odpovídá součet všech výskytů jedné. Příklad histogramu naleznete na obrázku 2.1.



Obrázek 2.1: Přiklad histogramu

2.2 Segmentace obrazu

Obsah této kapitoly jsem čerpal z [4]. Obecná definice segmentace říká, že je to proces dělení obrazu do částí, které korespondují s konkrétními objekty v obraze. Jinými slovy, každému obrazovému pixelu je přiřazen index segmentu vyjadřující určitý objekt v obraze. Segmentace je jeden z nejdůležitějších kroků analýzy obrazu. Informaci o rozdělení obrazu do jednotlivých segmentů využívají vyšší algoritmy zpracování obrazu. Snaží se porozumět obsahu obrazu. Konkrétním úkolem může být detekce přítomnosti příslušného objektu nebo nalezení a klasifikace objektů v obraze. Precizní segmentace je důležitá i pro 3D modelování objektů.

2.2.1 Metody vycházející z detekce hran

Tyto metody jsou orientovány na detekci významných hran v obraze. Hrany jsou detekovány na základě rozdílu hodnot okolních pixelů. Hrana popisuje rychlost změny a směr největšího růstu obrazové funkce. Hrana je pak definována velikostí a směrem. Základní metody detekce hran dělíme dle toho, jestli užívají první, nebo druhou derivaci.

2.2.2 Metody orientované na regiony v obraze

Tyto metody jsou principiálně stejné jako metody detekce hran. Jsme li schopní v obrázku identifikovat hrany, měly by tudíž tyto hrany ohraničovat regiony nalezené metodou orientovanou na detekci regionů. Toto však vždy neplatí, protože kontury regionů nemusí ohraničovat celý region,

nebo mohou být porušené. V opačném případě také není zaručeno, že regiony nalezené metodou segmentace orientované na regiony budou odpovídat hranám nalezeným hranovým detektorem.

2.2.3 Statistické metody

V tomto případě je základem segmentace statistická analýza obrazových dat, nejčastěji hodnot pixelů. Strukturní informace je obvykle zanedbávána.

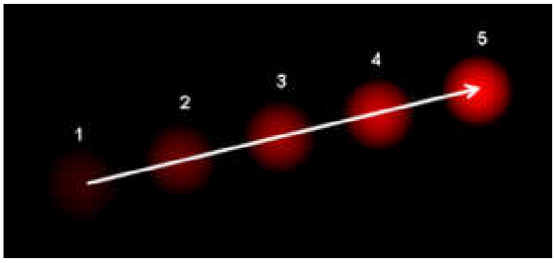
2.2.4 Znalostní metody

Znalost vlastností segmentovaných objektů (tvar, barva, struktura, apod.) mohou segmentaci značně ulehčit. Metody patřící do této kategorie využívají atlas předloh či modelů segmentovaných objektů (v případě medicínských dat to může být atlas lidských tkání). Atlas je generován automaticky ze souboru trénovacích dat, nebo jsou do něj informace vloženy ručně, na základě lidské zkušenosti. V průběhu segmentace algoritmus hledá transformaci známých objektů, šablon v atlasu, na objekty nalezené v obraze. Tento proces se obvykle nazývá atlas-warping a nejčastěji využívá lineární transformace.

2.3 Optický tok

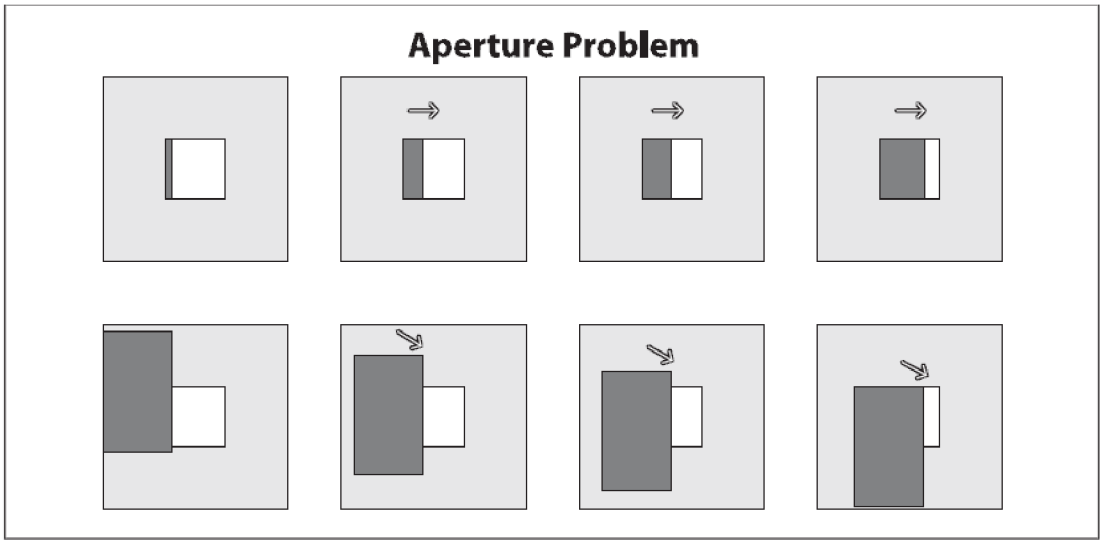
Optický tok je zjevný pohyb s kterým se shledáváme každý den při pohybu okolním světem. Předpokládejme, že sedíme ve vlaku, nebo automobilu a díváme se ven z okna. Zdá se, že stromy, budovy, krajnice atd. se pohybuje dozadu. Tomuto pohybu se říká optický tok. Tento pohyb nám také může říci, jak blízko se sledovaný objekt od nás nachází. Vzdálené objekty, jako například hory, mraky a měsíc se pohybují tak pomalu, že se jeví, že stojí na místě. Blízké objekty jako například budovy, nebo krajnice se pohybují dozadu v závislosti na vzdálenosti, kterou je vzdálen od pozorovatele. Objekty blíže k pozorovateli se pohybují vetší rychlosti, než objekty vzdálené [5].

V počítačové grafice je optický tok vektorové pole, které pro každý pixel daného obrazu určuje směr a rychlost pohybu oproti obrazu předchozímu. Vektor optického toku pohybujícího se objektu je na obrázku. Jak již bylo řečeno, objekty blíže k pozorovateli se pohybují větší rychlosti, než objekty vzdálené, tudíž i vektory optického toku vzdálených objektů musí být menší, než vektory optického toku objektů bližších [6].



Obrázek 2.2: Optický tok (převzato z [6])

Je důležité si zprvu uvědomit, že optický tok representuje pohyb v obraze, nikoli pohyb skutečné scény. Korespondence pohybu závisí na snímané scéně, a také na způsobu snímaní. Příkladem takovéhoto problému je problém apertury (angl. aperture problem). Snažíme li se detekovat pohyb scény z příliš malých obrazů dané scény, může se stát, že sledujeme pouze hranu, nikoli roh daného objektu. Z hrany lze složitě určit, s jakou intenzitou a jakým směrem se objekt pohybuje[7]. Příklad tohoto problému je na Obr. 2.3.



Obrázek 2.3: Problém apertury (převzato z [7])

Existuje několik přístupů k určování optického toku. Všechny tyto přístupy mají společné to, že vycházejí z předpokladu zachování intenzity. Mějme funkci intensity vyvíjející se v čase , kde reprezentuje prostorovou složku obrazu, t časovou dimenzi. Předpoklad zachování intenzity potom můžeme definovat vztahem (vzorec)

(2.1)

Kde $v = (u, v)T$ značí vektor pohybu v tomto případě v bodě p v čase t. To znamená, že hodnota obrazové funkce v pixelu v čase t je stejná, jako hodnota této funkce v čase $t + dt$. Avšak v bodě, jehož změna je závislá na velikosti optického toku. Tento výpočet lze zpracovat různými matematickými technikami, které člení metody pro výpočet optického toku do několika skupin, jak uvádí [11].

- Diferenční přístupy
- Vyhledávání oblastí
- Metody založené na energii
- Metody založené na fázi

2.3.1 Diferenční přístupy

Diferenční přístupy se objevily, jako první techniky k výpočtu optického toku. Tyto techniky používají parciální derivace prvního či vícero řádů. Získaná diferenciální rovnice je aproximací předpokladu zachování intenzity (vzorec). Z této aproximace vychází odhad chyby optického toku. Vzhledem k tomu, že se nám nepodaří nalézt optický tok splňující předpoklad zachování intenzity beze zbytku, zajímá nás tato zbytková chyba, kterou se snažíme minimalizovat.

Kvůli překonání problému apertury se zavádí další chybový člen, který má za úkol propagovat optický tok z míst, kde jej lze dobře určit do míst trpících nedostatkem gradientu. Další metody usměrňují propagaci optického toku podle výskytu hran reprezentující hranice objektů v reálné scéně. S přidáváním dalších omezujících předpokladů se snižuje problém apertury a snižuje vliv problému apertury. Problémem však je rostoucí složitost následného numerického řešení. Metody užívající diference vyšších řádu jsou z matematického hlediska přesnější, jsou však citlivější na existující chyby v obraze (nespojitosti, šum).

Horn-Skunkova metoda

Tato metoda byla objevena roku 1981 a popsána v [8]. Jedná se o první techniku používající předpoklad zachování intenzity. Horn-Schunkova metoda předpokládá hladkost toku celého obrazu. Snaží se tedy minimalizovat distorze optického toku a preferovat řešení, která jeví více hladkosti. Tok je formulován jako funkce globální energie, u které se snažíme o její minimalizaci. Tato funkce pro $2D + t$ dimenziální obraz takto:

$$f = \int((\nabla I \cdot \vec{V} + I_t)^2 + \alpha(|\nabla V_x|^2 + |\nabla V_y|^2))\mathrm{dxdy} \quad (2.2)$$

Kde $\nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}$ jsou derivace intenzity obrazu dle x a y složky, I_t je derivace dle času, $\vec{V}$ je vektor optického toku s komponentami V_x, V_y. Parametr α je váha kladená na požadavek homogenity optického toku. Větší hodnota této váhy vede k hladšímu toku. Tato funkce může být vyřešena vypočtením Euler-Lagrange equations odpovídajícím řešení rovnice výše. Z čehož vychází:

$$I_x\left(I_x\, V_x + I_y V_y + I_t\right) - \alpha \Delta V_x = 0 \qquad (2.3)$$
$$I_y\left(I_x\, V_x + I_y V_y + I_t\right) - \alpha \Delta V_y = 0$$

Kde Δ je Laplaecův operátor $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$.

Řešením těchto rovnic pomocí Gaus-Sidelovy metody pro komponenty toku V_x, V_y dostáváme iterační schéma:

$$V_x^{k+1} = \frac{\Delta V_x^k - \frac{1}{\alpha} I_x\left(I_y V_y^k + I_t\right)}{\frac{1}{\alpha} {I_x}^2} \qquad (2.4)$$
$$V_y^{k+1} = \frac{\Delta V_y^k - \frac{1}{\alpha} I_y\left(I_x V_x^k + I_t\right)}{\frac{1}{\alpha} {I_y}^2}$$

zde index $k+1$ znamená další iteraci, která má být vypočtena a k je poslední vypočtený výsledek. Jendou z cest k získání ΔV_i je:

$$\Delta V_i(p) = \frac{1}{4} \sum_{N\,(p)} V_i\left(N(p)\right) - V_i\,(p) \qquad (2.5)$$

kde $N(p)$je čtyřkolí pixelu p.

Výhodou Horn-Schunkova algoritmu je, že přináší husté pole vektorů optického toku, z důvodu vyplnění informace o toku homogenních objektů tokem s jejich ohraničení. Nevýhodou však je jeho veliká citlivost na šum v obraze[9].

Lucas-Kanadova metoda

Algoritmus Lucas-kanade byl původně navržen, jako algoritmus hustý (dense). Dnes, protože je snadno aplikovatelný i na sadu bodů ve vstupním obraze je používán i jako řídký (spase). Tento algoritmus se dá použít i jako spase, protože počítá pouze lokální informaci, která je derivována z malého okolí obklopujícího jednotlivé body zájmu. Nevýhodou použití malého okolí je, že velké pohyby mohou přesouvat body mimo toto okolí a tím znemožní jejich nalezení tímto algoritmem [7].

Základním předpokladem pro určení optického toku je, že místní tok (V_x, V_y) je konstantní v okolí velikosti $m \times m$, kde $m > 1$, se středem v bodě x, y a vnitřním číslováním $1 \ldots n, n = m^2$. Dostáváme tedy soustavu rovnic:

$$I_{x1} V_x + I_{y1} V_y = -I_{t1} \qquad (2.6)$$

$$I_{x2}V_x + I_{y2}V_y = -I_{t2}$$
$$\vdots$$
$$I_{xn}V_x + I_{xn}V_y = -I_{tn}$$

Je zde tedy více rovnic než neznámých a soustava je řešitelná. Odtud:

$$\begin{bmatrix} I_{x1} & I_{y1} \\ I_{x2} & I_{y2} \\ \vdots & \vdots \\ I_{xn} & I_{yn} \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} -t_1 \\ -t_2 \\ \vdots \\ -t_n \end{bmatrix} \tag{2.7}$$

K vyřešení této soustavy můžeme použít metodu nejmenších čtverců:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum I_{xi}^2 & \sum I_{xi}I_{yi} \\ \sum I_{xi}I_{yi} & \sum I_{yi}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_{xi}I_{ti} \\ -\sum I_{yi}I_{ti} \end{bmatrix} \tag{2.8}$$

Kde sumy provádíme od 1 do i.

Což znamená, že optický tok lze určit výpočtem derivací dle všech dimenzí [10].

2.3.2 Vyhledávání oblastí

Vyhledávání oblastí využívá korelačních funkcí, které dokáží určit míru podobnosti dvou různých obrazů. Na optický tok potom můžeme nahlížet, jako na transformační funkci, s jejichž pomocí lze určit obraz v čase t+dt pomocí transformace obrazu v čase t opět s využitím předpokladu zachovaní intenzity. Poté můžeme měřit přesnost odhadu optického toku pomocí korelační funkce mezi skutečným a transformovaným obrazem. Princip je tedy v nalezení optického toku s největší výše uvedenou korelační přesností z prostoru všech možných optických toků. Tyto techniky tudíž řeší v základní podobě výpočet optického toku hrubou silou. To způsobuje jejich univerzalitu a přesnost. Bohužel jsou kvůli nutnosti prověřování všech možných řešení výpočetně velice náročné. Většina pokročilejších metod se zaměřuje na omezení počtu řešení. [12].

2.3.3 Metody založené na energii

Tyto metody určují optický tok dle frekvenční domény obrazu. Metody tvoří sada filtrů, jejichž parametry jsou směr a rychlost posunu. Z energie těchto filtrů jsou poté určeny rychlosti pro jednotlivé pixely. Přiklad v [12].

2.3.4 Metody založené na fázi

Základním poznatkem těchto metod užívajících Fourierova obrazu vstupní sekvence je fakt, že posun v prostorové doméně obrazu se projeví posunem v doméně frekvenční. Rovnice pro výpočet rychlosti jsou pak analogické gradientním přístupům. Rozdíl je ve využití gradientu fáze, místo gradientu intenzity v obraze. Výhodou tohoto postupu je to, že gradient fáze je ve většině přístupů stabilnější, tudíž je vhodnější použít jej k výpočtu optického toku. Metodu užívajícu tuto techniku lze nalézt v [13]

2.4 Porovnání histogramů

Tato metoda je založena na porovnávání světelné charakteristiky aktuálního snímku se stejnou charakteristikou snímku jiného. Pokud se vyskytne ve scéně změna, projeví se v odlišnosti histogramu. Je li tato změna větší, než hodnota prahu je tato změna vyhodnocena, jako pohyb ve scéně. Výhodou této metody je to, že máme vždy vytvořen histogram s předcházejícího porovnávání a tudíž ušetříme režii pro jeho výpočet. Nevýhodou této metody je to, že udává pouze to, že ve scéně se vyskytuje pohyb. Neudává však to, kde se tento pohyb nachází. Tento problém se dá díky rychlosti tohoto algoritmu vyřešit tak, že snímek rozdělíme do menších oblastí vytvořit histogram pro jednotlivé oblasti a porovnáváme histogramy jednotlivých korespondujících oblastí. Pokud dojde ke změně v histogramu dané oblasti, můžeme přesně vyhodnotit nejen to, že k pohybu došlo, ale i to, kde k tomuto pohybu došlo. Počet a velikost oblastí volíme dle velikosti sledovaných objektů. Další výhodou tohoto přístupu je to, že můžeme přesně určit místa zájmu a v těchto místech vytvořit hustou síť menších oblastí. V místech, kde pohyb neočekáváme, nebo jej nechceme registrovat, neumístíme oblasti pro výpočet histogramů vůbec. Problémem u této metody by byla situace, kdy bychom snímali scénu kamerou, která se pohybuje. Při každém pohybu kamery by došlo ke změně histogramu tudíž vyhodnocení pohybu, které by bylo sice správné, ale nebylo by předmětem zájmu našeho sledování.

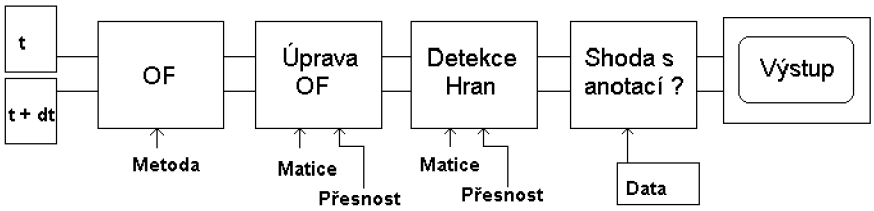
2.5 Sledování rozdílných bodů mezi snímky

Tato metoda pracuje tak, že porovná každý pixel pozadí s korespondujícím pixelem popředí. Odečtením korespondujících pixelů vytváří nový obraz reprezentující rozdíly těchto snímků. Z tohoto rozdílového snímku můžeme určit, ve kterých oblastech došlo ke změnám a které zůstaly nezměněny. Tato metoda je stejně, jako předchozí metoda náchylná na změnu světelných podmínek scény. Výpočetní náročnost této metody není příliš náročná, dá se však snadno optimalizovat použitím snímku ve stupních šedi namísto snímku barevného. Také se dá zredukovat počet porovnávaných

pixelů. Opět se zde střetáváme s problémem popisovaným u metody porovnávání histogramu a to situaci, kdy se pohybuje celá scéna z důvodu snímání pohybující se kamerou.

3 Návrh aplikace

Ve své práci se zaměřuji na detekci pohybujících se vozidel v sekvenci obrázků, založenou na určování optického toku. Implementaci realizuji v jazyce C++ pomocí knihovny OpenCV. Blokové schéma je na obrázku 3.1.



Obrázek 3.1: Blokové schéma aplikace

Vstupem aplikace budou dva snímky, jeden zobrazující scénu v daném čase t a druhý zobrazující scénu s časovým posuvem $t + d_t$. Další možností je použití vstup z videa uloženého na disku počítače, nebo z videokamery. Při použití jsou však tyto vstupy upraveny na sekvenci jednotlivých snímků, tudíž jejích použití vyžaduje pouze předřazení bloku rozdělujícího sekvenci snímku na jednotlivé snímky. Protože pracuji s anotovanými daty v podobě snímků, není nutné zde tento blok uvádět.

Prvním blokem je blok určující dle dané metody optický tok. Výstupem tohoto bloku jsou matice reprezentující velikost a směr vektorů optického toku.

Následujícím blokem je úprava optického toku. Výstupem sou upravené matice reprezentující velikost a směr vektorů optického toku. U tohoto bloku je možné nastavovat přesnost a velikost matice pro úpravu toku.

Dalším blokem je detektor hran, který opět pracuje s velikostí matice a přesností. Výstupem tohoto bloku je binární obraz.

Nepovinným blokem je blok hledající shodu s anotovanými daty.

Posledním blokem je blok výstupní, který nachází v binárním obrazu kontury a zobrazí uživateli výsledek detekce. Bližší popis jednotlivých bloků s jejich funkčností viz 4.

4 Implementace aplikace

Zde se pokusím popsat mnou naimplementovanou aplikaci sloužící pro detekci vozidel v obraze. Tato aplikace byla napsána v jazyce C++. Nebudu uvádět detailní popis tříd a jejích atributů, ale pouze obecnou definici jednotlivých funkčností.

4.1 OpenCV

OpenCV (Open Source Computer Vision) je platformě nezávislá knihovna zaměřená na počítačové vidění vyvinutá firmou Intel použitelná v jazycích C a C++. Nabízí mnoho funkcí pro práci s obrázky a videem. Nabízí také funkce pro výpočet optického toku, hledání kontur a práci s histogramy.

4.2 Optický tok

Základem detekce objektů v obrazech je segmentace. Jak již bylo řečeno segmentace je proces rozdělení obrazu na části odpovídající konkrétním objektům v obraze. Protože dle definice optického toku se objekty blíže pozorovateli pohybují větší rychlostí, nežli objekty vzdálené, rozhodl jsem se o segmentaci dle optického toku. Používaná knihovna OpenCV nabízí několik metod pro určování optického toku. Společným rysem těchto metod je vstup. Vstupem bývají vždy dva obrazy scény. Tyto obrazy se od sebe liší časovým posuvem. První obraz obsahuje scénu v čas t, druhý obraz v čase t+dt. Dle 2.3 je optický tok vektorové pole, které pro každý pixel daného obrazu určuje směr a rychlost pohybu oproti obrazu předchozímu. Výstupem těchto metod je tudíž právě toto vektorové pole, representováno buď koncovým bodem vektoru, nebo jednotlivými přírůstky souřadnic bodů matice.

Metody určující optický tok můžeme rozdělit nejen dle jejich přístupu k výpočtu optického toku (viz teorie). Můžeme je také rozdělit na husté (angl. dense) a řídké (angl. spase) jak již jejich názvy napovídají hlavním rozdílem je počet bodů obrázku vstupujících do výpočtu[7]. Řídké metody nepracují přímo se vstupními daty, ale pouze s vybranými body vstupu. Tyto body mohou být voleny náhodně, ale pro správnou detekci optického toku a omezení chyb by se mělo jednat o rohy, nebo hrany hledaných objektů, proto je výstup metody velice závislý na detektoru, který vybere které body do výpočtu zařadit a které nikoliv. Výhodou těchto metod je jejich větší přesnost. Nevýhodou je především výpočetní náročnost při použití většího množství vstupních bodů. Metoda také pracuje s chybovým členem, pokud je bod pozadí nalezen v popředí je jeho pohyb a posuv zaznamenán. Pokud ne, je zařazen mezi chyby a dále se s nim nepracuje. To způsobuje problém, že už tak nízké

množství bodů je ještě ochuzeno o chybové body. Také se nabízí otázka proč tuto metodu používat k segmentaci, když o vlastní segmentaci se musí starat metoda vybírající body pro vstup a v situaci, kdyby scéna obsahovala velké množství pohybujících se objektů, by došlo k potřebě značného množství vstupních bodů, což by vedlo k značné výpočetní a časové náročnost., proto tento přístup ve své práci nepoužívám.

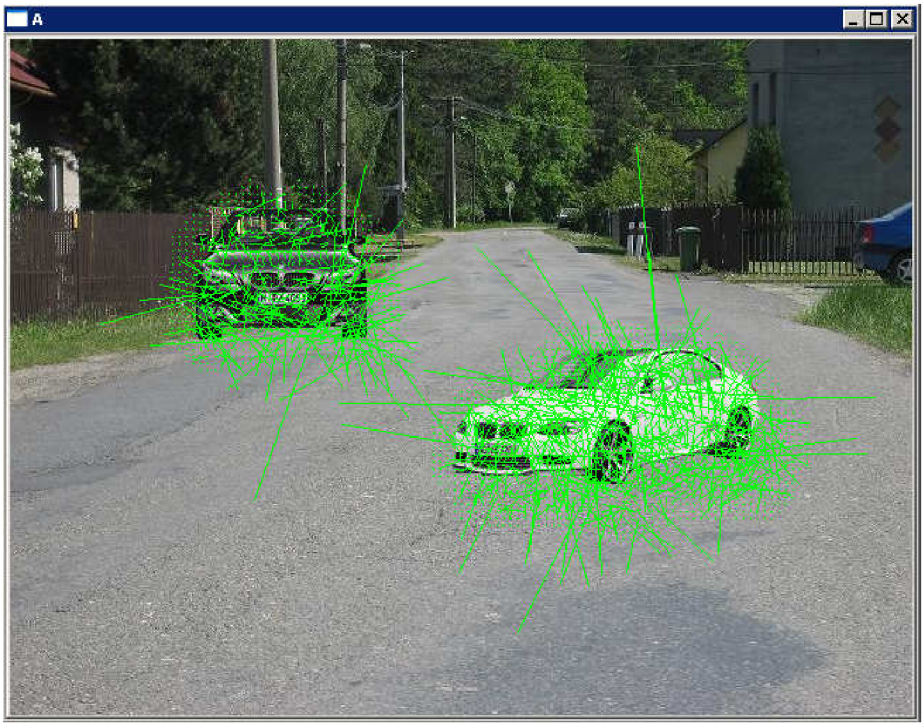
Metody husté pracují se všemi body vstupních obrazů. Výstupem jsou dvě matice odpovídající svými rozměry přesně rozměry vstupů. První matice obsahuje posuvy horizontální souřadnice daného bodu vstupu, druhá výstupní matice obsahuje vertikální posuvy vstupu. Metody tudíž pracují tímto způsobem. Pro všechny body prvního obrazu vstupu se snaží nalézt tyto body v obrazu výstupním. Největší nevýhodou těchto metod je jejich nízká přesnost.

V používané knihovně OpenCV jsou naimplementovány funkce pro výpočet optického toku dle Horn-Shunkova algoritmu a dle algoritmu Lucas-Kanade 2.3.1.

Funkce pro výpočet optického toku dle Horn-Skunkova algoritmu přijímá, jako argumenty obraz scény a obraz scény s časovým posunem. Výstupní obrazy pro uložení horizontální a vertikální rychlosti optického toku. Příznak, zdali se má použít posuvu nalezených v předchozím výpočtu metody. Tento příznak se používá, v případech kdy vstupní obrazy extrahujeme z videokamery, nebo videa. Při prvním běhu funkce jsou pro dané pixely nalezeny rychlosti pohybu v daném směru. V případě nastavení následujícího příznaku na 1 funkce nezačíná prohledávání přímo v okolí daného bodu, ale bere v potaz právě předchozí nalezené rychlosti dalším vstupním parametrem je *Lagrangův Násobitel* reprezentující podíl chyb v jednotlivých částech algoritmu, dalším argumentem jsou kritéria pro ukončení iteračních výpočtů. O těchto kritériích a *Lagrangově násobiteli* se lze blíže dozvědět v [7].

Funkce pro výpočet optického toku dle Lucas-Kanadova algoritmu přijímá také jako vstupní argumenty obraz scény a obraz scény s časovým posunem, dále velikost okolí ve kterém má být hledán optický tok.

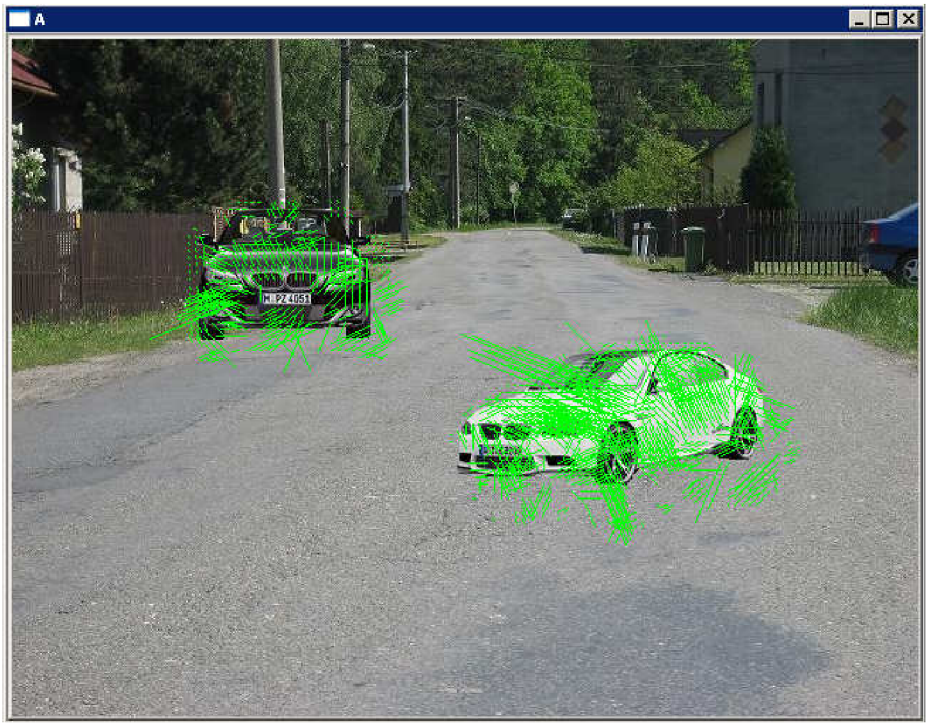
Kombinací těchto přístupů je metoda block matching, která pro blok vstupního obrazu, jehož velikost je definována programátorem hledá shodnou oblast v obraze s časovým posuvem. Pokud uspěje a oblast se oproti předchozímu obrazu pohnula, uloží do výstupních matic jednotlivé posuvy vstupního bloku. Výstupní matice jsou tudíž velikost bloku násobně menší, nežli vstupní obrazy, protože nerepresentují posuvy jednotlivých bodů obrazu, ale celého bloku. U této metody lze opět definovat velikost okolí časově posunutého obrazu, ve kterém se má hledat posunutý blok. Tak jako metoda Lucas-Kanade také umožňuje použití předchozí nalezené rychlosti pohybu pro cyklické volání této metody. Grafické znázornění vektorů optického toku je zobrazeno na obrázku 4.1.



Obrázek 4.1: Příklad zobrazení vektorů optického toku

4.3 Operátor upravující optický tok

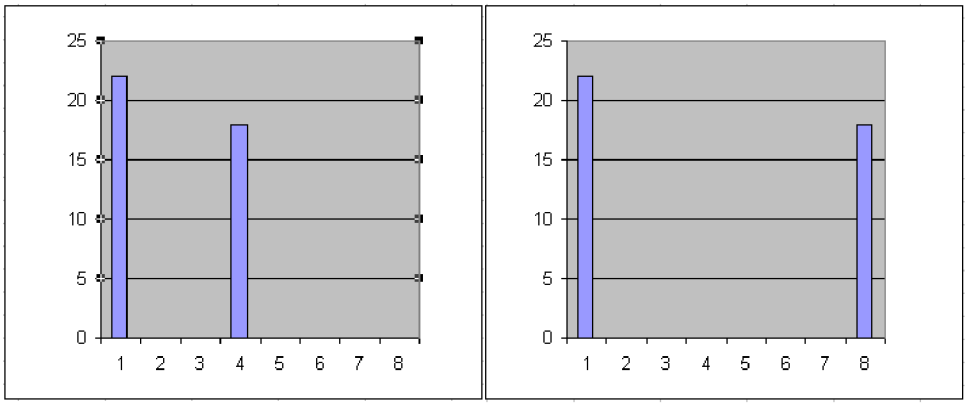
Jak již bylo řečeno výstup z metod pro určení optického toku je velice nepřesný. Proto pokud bych chtěl použít takto vytvořený výstup, jako vstup hranového detektoru byl by výsledek velice nepřesný (viz testy) je nutné tento výstup upravit. Tato nepřesnost spočívá především v nestejnorodosti směrů úhlů a velikostí vektorů, proto je nutné tyto velikosti a směry sjednotit. K sjednocení směrů a velikostí vektorů slouží následující algoritmus založený na použití histogramu. Jednotlivé matice přírůstků v horizontálním a vertikálním směru jsou převedeny na matice úhlů a velikostí vektorů v polárních souřadnicích. Tyto matice úhlů a velkostí vektorů jsou poté procházeny a pro každý prvek matice je v jeho okolí vytvořen histogram velikosti úhlu. Pokud takto vytvořený histogram obsahuje maximální hodnotu, jejíž velikost je vyšší, než požadovaná přesnost operátoru je tato hodnota nastavena i vstupnímu bodu. Neobsahuje, li histogram maximum, nebo je-li toto maximum nižší, než požadovaná přesnost je tento bod vynulován. Velikost vektoru se počítá obdobným způsobem. V případě, že maximum histogram úhlů okolí splňuje danou přesnost, se ze stejného okolí velikostí vektorů vytvoří aritmetický průměr velikostí, jejichž úhel odpovídá danému úhlu. Grafické znázornění upravených vektorů optického toku je zobrazeno na obrázku 4.2.



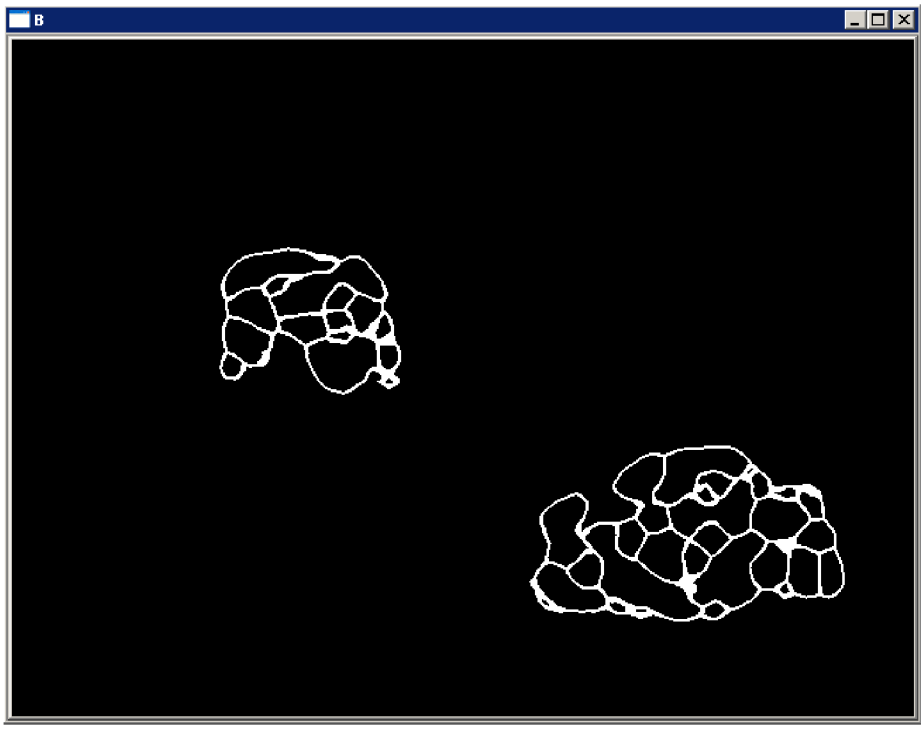
Obrázek 4.2: Příklad zobrazení vektorů optického toku po úpravě

4.4 Hranový detektor

Po úpravě optického toku je již možné použít hranový detektor. Hranový detektor je opět založen na histogramu úhlů v daném okolí. Tentokrát není však předmětem zájmu pouze nejvyšší hodnota. Pří prohledávání histogramu hledám také druhou nejvyšší hodnotu. Poté tyto dvě hodnoty porovnávám. Pokud nejvyšší hodnota splňuje danou přesnost, je porovnána s druhou nejvyšší hodnotou, pokud jsou tyto hodnoty podobné velikosti, je bod prohlášen za hranový. U rozhodnutí, zdali je bod označen za hranový, však také hraje významnou roli, kde se dané dvě nejvyšší hodnoty nacházejí. To vyplývá z faktu, že úhel nebývá hodnot od 0 do 360 stupňů, 360 stupňů se rovná úhlu 0 stupňů. Tudíž pokud se v histogramu vyskytují maxima odpovídající minimálnímu a maximálnímu úhlu nejedná se o hranu, ale pouze o zobrazení stejné hrany ve dvou různých intervalech. Příklady histogramů s detekovanou hranou a s falešnou detekcí hrany je na obrázku 4.3. Příklad výstupu z hranového detektoru je na obrázku 4.4.



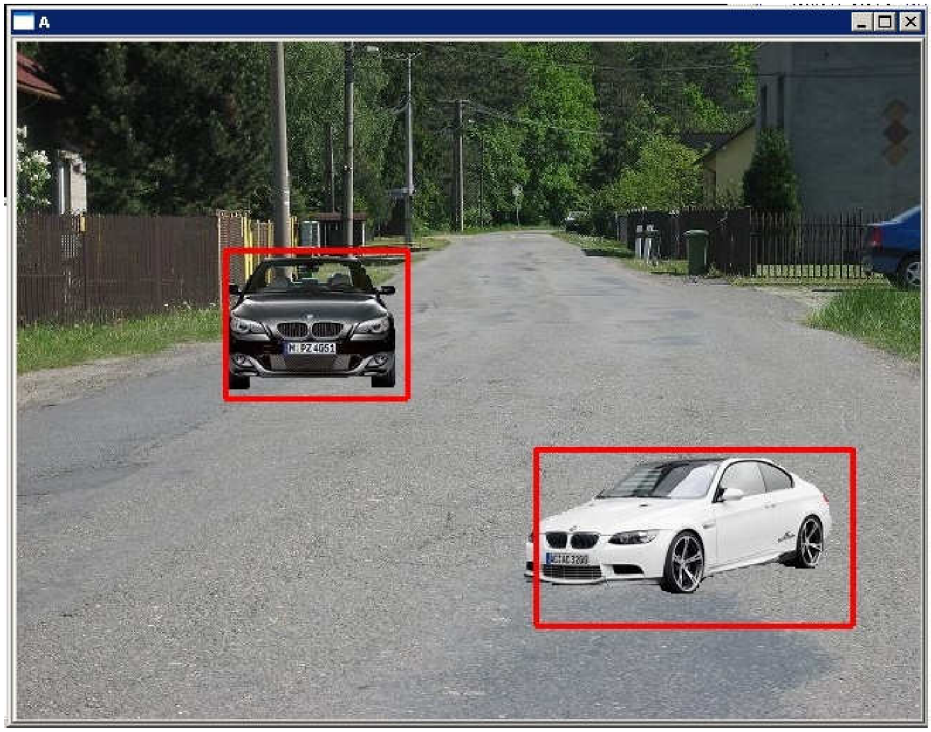
Obrázek 4.3: Správná detekce hrany (vlevo). Špatná detekce hrany (vpravo).



Obrázek 4.4: Binární obraz nalezených hran.

4.5 Zobrazení výsledků

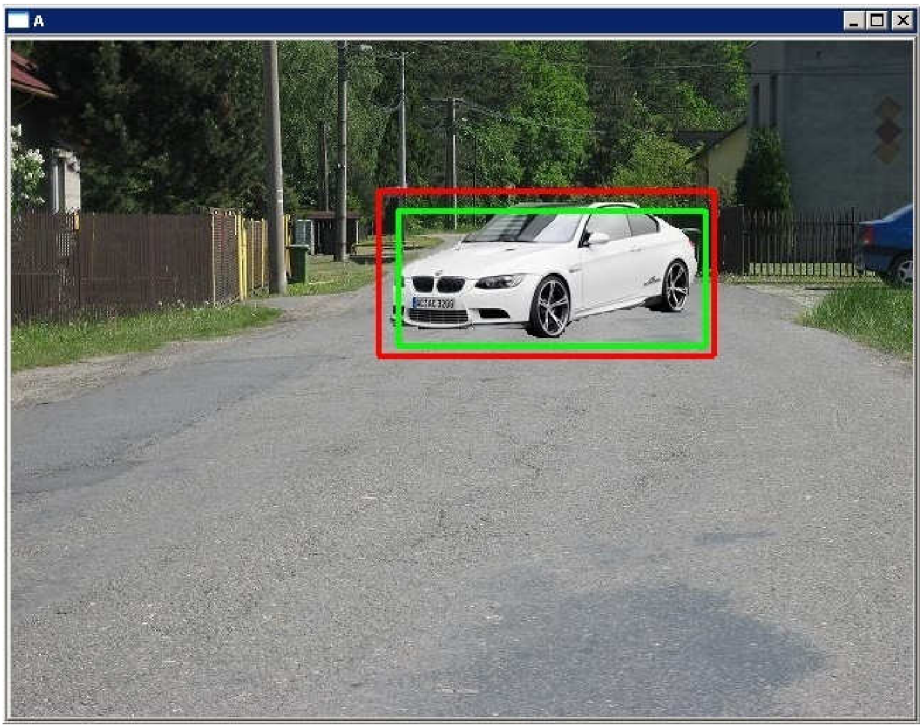
Dalším krokem je zobrazení nalezených hran uživateli. Na výstupu z detektoru hran jsou nalezeny kontury. Kontura je jednosměrný seznam bodů reprezentující křivky objektů v obraze. Každý prvek seznamu nese informaci o následujícím prvku seznamu. Pro každou nalezenou vnější konturu je nalezen obdélník, který přesně opisuje tuto konturu. Tento obdélník je poté vykreslen do vstupního obrazu a zobrazen uživateli.



Obrázek 4.5: Výstup aplikace s detekcí vozidel.

4.6 Zobrazení shody s anotovanými daty

Pro kontrolu shody nalezených objektů s anotovanými daty je zde tzv. anotovací režim. V tomto režimu je ze vstupních parametrů převzat také soubor s uloženými anotovanými daty. Z tohoto souboru je pro daný vstupní obraz převzata informace o pohybujícím se vozidel. Tato informace je uložená také jako obdélník přesně opisující obrys pohybujícího se vozidla. Tento obdélník je tedy také vykreslen uživateli a je vypočtena shoda mezi detekovaným objektem a objektem anotovaným.



Obrázek 4.5: Porovnání výstupu aplikace s anotovanými daty. Výstup aplikace (červená), anotovaná data (zelená).

5 Testování

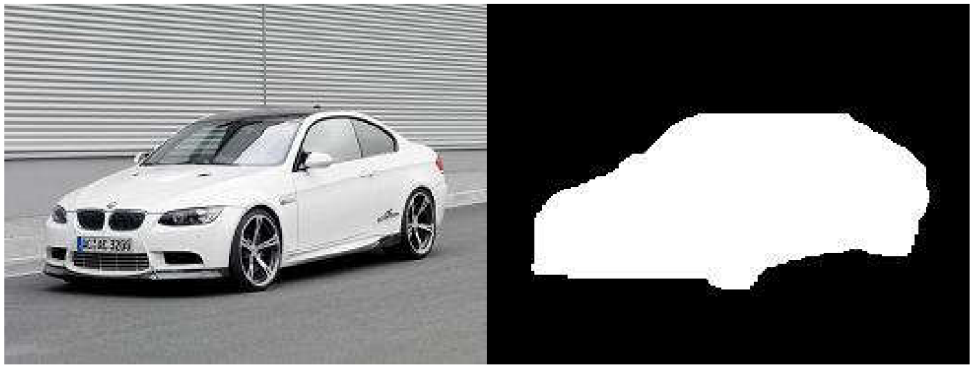
V této kapitole se pokusím zhodnotit úspěšnost detekce vozidel v obraze na anotovaných datech v závislosti na použité metodě, počtu vstupních objektů atd. Pokusím se zde ukázat nejen úspěšné detekce vozidel, ale i detekce neúspěšné, protože oprava neúspěšných detekcí patří spolu s klasifikací typu vozidla k možným směrům, kam se chci v budoucnu ubírat.

5.1 Anotovaná data

Aby byla možnost přesného určení pozice vozidla v obraze, je nutné mít k dispozici přesně anotovaná data. Z tohoto důvodu jsem vytvořil nástroj pro tvorbu těchto anotovaný dat.

Video je ve své podstatě sekvence mnoha po sobě jdoucích snímků. Všechny metody knihovny openCV, které používám, pracují při určení optického toku se dvěma snímky, prvním snímkem, který zobrazuje scénu v čase t, a druhým, který zobrazuje scénu t+dt. Pro určení optického toku se předpokládá, že ve scéně došlo ke změnám polohy některého z objektů – ve scéně došlo k pohybu. U anotovaných dat je nutností, aby byla známa přesná poloha pohybujícího se objektu v časech t i t+dt, aby bylo možné tato data posléze porovnat s výstupem z aplikace.

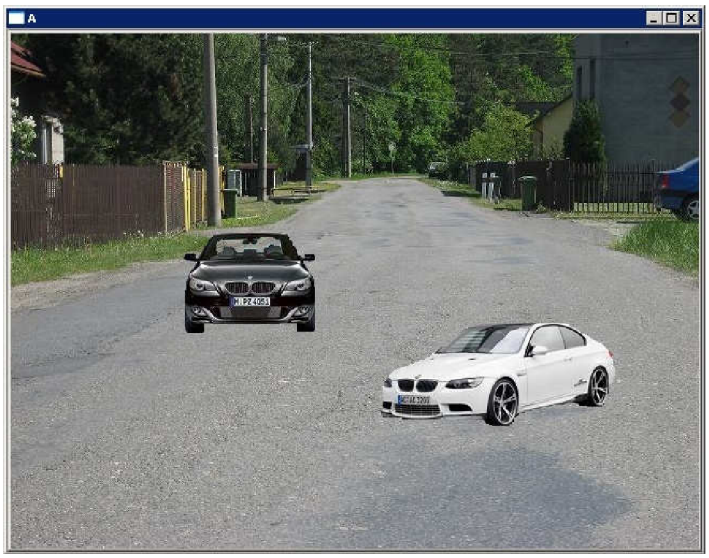
Nástroj pracuje se třemi typy obrázků. Prvním obrázkem je vždy obraz scény. Tento obraz bude sloužit pouze jako pozadí. Dalším obrazem je obraz vkládaného objektu – vozidla. Vzhledem k tomu, že obraz téměř nikdy neobsahuje pouze vozidlo, ale i jeho okolí. Mě však zajímá pohyb pouze vozidla a bylo by nežádoucí vkládat do scény i toto okolí, je třeba toto okolí zahodit. K tomuto účelu slouží maska vozidla. Maska je binární obraz, který obsahuje informace o tom, kde se přesně nachází vozidlo. Nástroj při umisťování vozidla do obrazu scény postupuje následujícím způsobem. Prochází obraz vozidla pixel po pixelu a pro a zároveň prochází korespondující pixely masky. Pokud pixel vozidla, odpovídá pixelu masky obsahující hodnotu logické jedničky (bílá barva). Je pixel zařazen do obrazu scény. Maska musí mít rozměry shodné s obrazem vkládaného vozidla. V masce je také nalezená kontura, pro kterou je určen obdélník který tuto masku ohraničuje. Souřadnice tohoto obdélníku se používají pro následné detailní porovnání s výstupem s aplikace. Příklad vozidla a jeho masky sloužící k zahození nepotřebných okrajů vozidla je na obrázku 5.1.



Obrázek 5.1: Vozidlo a jeho maska

Nástroj pracuje následujícím způsobem. Do obrazu scény vloží na přesné místo vymaskovaný obraz vozidla. Takto upravený obraz uloží do výstupního souboru a souřadnice kontury vozidla uloží do výstupního textového souboru. V dalším běhu umístí stejný obraz vozidla na pozici posunutou oproti původní pozici o přesný počet pixelů definovaný vstupními parametry. Takto vytvořený obraz opět uloží do výstupního souboru a jeho souřadnice do textového souboru.. V praxi je možné tímto způsobem vytvořit libovolně dlouhou sekvenci snímků s textovým souborem obsahujícím pozice vozidla v daném snímku.

Nástroj tímto způsobem umisťuje do obrazu maximálně dvě různá vozidla. Tento počet je pro testovací účely dostačující a v případě nutnosti není problém vytvořit masku a výřez dalšího vozidla. Příklad anotovaných dat je na obrázku 5.2.



Obrázek 5.2: Příklad anotovaných dat

Další vlastností nástroje je možnost použití „zoomu“. Vozidlo jedoucí směrem ke kameře mění v obraze scény nejen svou polohu, ale i velikost. Z tohoto důvodu je možnost použít při

generování dat funkci zoom. Tato funkce opět umístí vymaskovaný obraz vozidla do scény, nejen se zohledněním jednotlivých posuvů, ale tento vkládaný obraz také zvětší, nebo zmenší v závislosti použití funkce zoom.

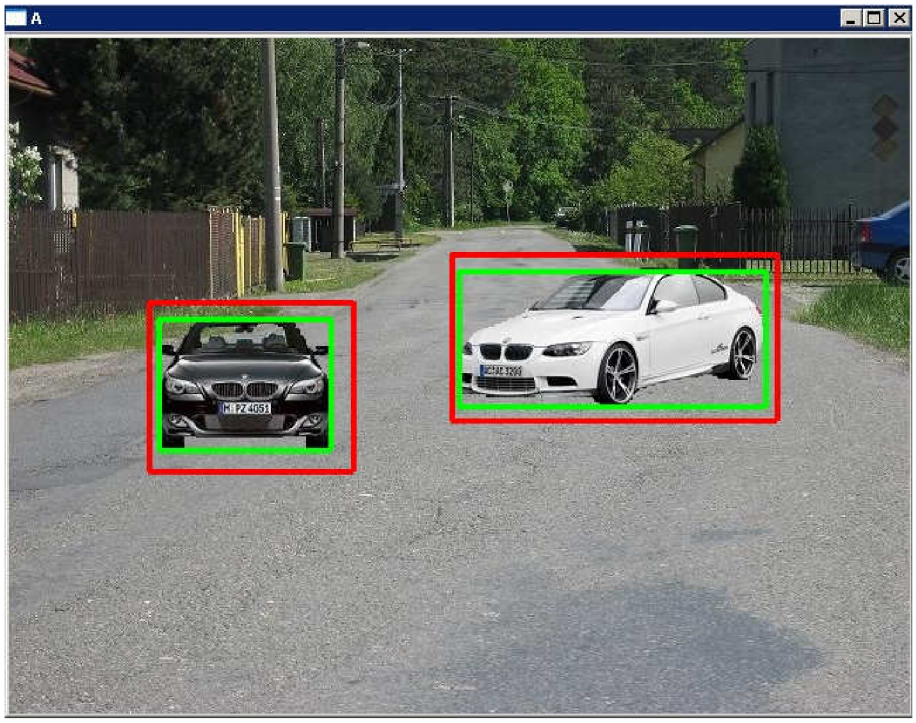
5.2 Detekce vozidla

Vzhledem k tomu, že aplikace detekuje jedno vozidlo velice spolehlivě. Ukázku této detekce Horn-Schunkovou metodu se stoprocentní shodou s anotovanými daty naleznete na obrázku 4.5.

5.3 Detekce dvou vozidel

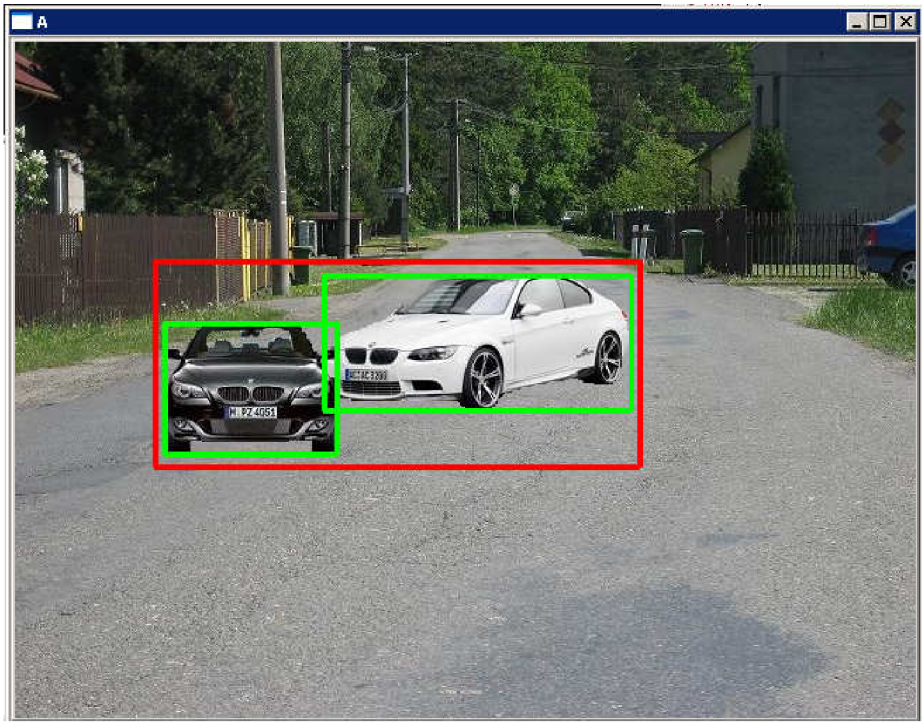
Zde demonstruji výsledky detekce vozidel Horn-Schunkovou metodou, protože výsledky metody Lucas-Kanade nebyly vyhovující. Chci také prezentovat rozdílné výsledky v závislosti na parametrech dané metody.

1. Kladná detekce: Aplikace je schopná detekovat za daných okolností také více vozidle zde je příklad kladné detekce dvou vozidel

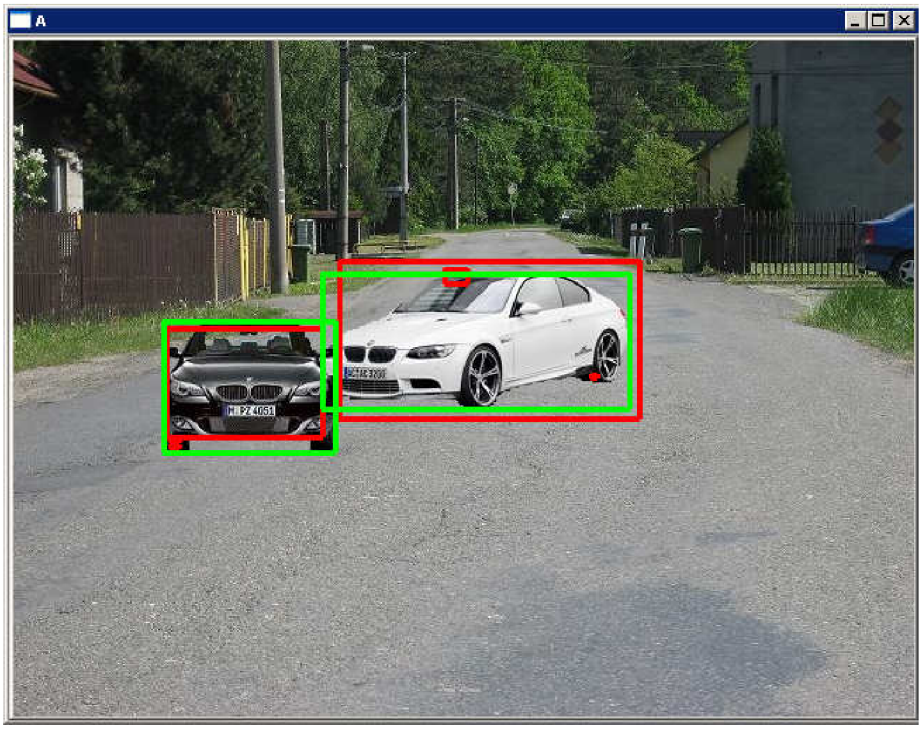


Obrázek 5.3: Kladná detekce dvou vozidel

2. Záporná detekce: U detekce dvou vozidel dochází k problému v případě, že jsou detekovaná vozidla příliš blízko sebe. V takovém případě je hranový detektor nedokáže od sebe rozeznat (viz. Obr. 5.4). Tento problém se dá U Horn-Schunkovy metody částečně řešit úpravou váhy α (viz 4.2), to však s sebou přináší možné riziko v nalezení falešných kontur. Výsledek po úpravě je na obrázku 5.5.



Obrázek 5.4: Záporná detekce vozidel blízko u sebe.



Obrázek 5.5: Kladná detekce vozidel blízko u sebe.

6 Závěr

Ve své práci jsem se zaměřil na detekci vozidel dle optického toku pomocí hustých metod jeho určování. Bohužel již v počátcích implementace jsem došel k závěru, že funkce pro husté určení optického toku jsou velice nepřesné a je nutné je upravit. Po úpravě výstupu již byla detekce možná avšak ani přesto nebylo možné od sebe jednoznačně oddělit objekty v případě, kdy se nacházeli velice blízko u sebe. V ostatních případech byla detekce spolehlivá a odpovídala anotovaným datům. Zároveň je možné díky mnou navrženému řešení vzájemně porovnávat funkce pro výpočet optického toku knihovny OpenCV.

6.1 Další vývoj a možná vylepšení

V prvé řadě bych se v dalším vývoji chtěl zaměřit na vytvoření kvalitnějšího operátoru na úpravu optického toku, protože na jeho správném určení závisí další části aplikace. Při detailním a přesném určení optického toku by již nebyl problém zaměřit se například na detekci vozidel, jejichž optický tok je vyšší, než definovaná konstanta, nebo se zaměřit na optický tok, jehož vektory mají pouze definovaný směr.

Další věcí, na které bych se chtěl v dalším vývoji zaměřit je snížení časové náročnosti, protože současná aplikace by se bohužel nedala použít v reálném čase.

Jednoznačným přínosem by také bylo přidání klasifikátoru určující například typ detekovaného vozidla.

Literatura

[1] **University of California Berkeley**: Trafic Surveilance, [online] 2009. [Citace: 1.Květen. 2009]
http://www.path.berkeley.edu/PATH/Publications/Media/FactSheet/TrafficSurveillance.pdf

[2] **Jaesik Choi:** Realtime On-Road Vehicle Detection with Optical Flows and Haar-like Feature Detector, [online]2006. [Citace: 4.Květen. 2009]
http://reason.cs.uiuc.edu/jaesik/projects/Vehicle_Detection/jaesik_cvpr.pdf

[3] **Wikipedia contributors**. Image histogram. Wikipedia, The Free Encyclopedia, [online] 2009. [Citace: 15. Květen 2009]
http://en.wikipedia.org/w/index.php?title=Image_histogram&oldid=289353543

[4] **Španěl, Michal a Beran, Vítězslav**: Obrazové segmentační techniky, [online] 2005. [Citace: 13. Květen 2009]
http://www.fit.vutbr.cz/~spanel/segmentace/

[5] **CentEye** : What is "Optic Flow"?, [online] 2008. [Citace: 5. Květen 2009]
http://www.centeye.com/pages/techres/opticflow.html

[6] **Wikipedia contributors**. Optical flow. Wikipedia, The Free Encyclopedia, [online] 2009 Apr 28, 09:58 UTC [Citace: 3. Březen 2009].
http://en.wikipedia.org/w/index.php?title=Optical_flow&oldid=286617987

[7] **Bradski, Gary a Kaehler, Adrian.** *Learning OpenCV: Computer Vision with the OpenCv Library.* Sebastopol : O'Reilly Media, Inc., 2008. 978-0-596-51613-0

[8] **Horn B.K.P and Schunck B.G**. Determining optical flow. AI 17,pages 185–204, 1981 [online] [Citace: 20. Březen 2009]
http://www.caam.rice.edu/~zhang/caam699/opt-flow/horn81.pdf

[9] **Wikipedia contributors**. Horn–Schunck method, Wikipedia, The Free Encyclopedia, [online] 2009 [Citace 23. Březen 2009]
http://en.wikipedia.org/w/index.php?title=Horn%E2%80%93Schunck_method&oldid=277571000

[10] **Wikipedia contributors**. Lucas–Kanade Optical Flow Method, Wikipedia, The Free Encyclopedia, [online] 2009 [Citace 17. Duben 2009]
http://en.wikipedia.org/w/index.php?title=Lucas%E2%80%93Kanade_Optical_Flow_Method&oldid=278936902

[11] **J.L. Barron, D.J. Fleet, S.S. Beauchemin, and T.A. Burkitt**. Performance of optical flow techniques. CVPR, 92:236–242

[12] **David J. Heeger**. Model for the extraction of image flow. J. of the Optical Society of America, 4(8):1455–1471, 1987

[13] **David J. Fleet and Allan D. Jepson**. Computation of komponent image velocity fromlocal phase information. International Journal of Computer Vision, 5(1):77–104, 1990