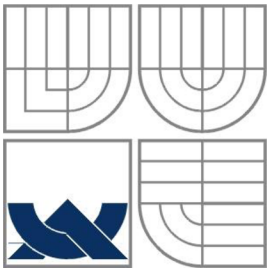
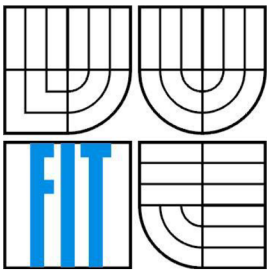


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SMART CAR: DETEKCE DOPRAVNÍCH ZNAČEK

SMART CAR: TRAFFIC SIGNS DETECTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

Martin Brzoza

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. Vítězslav Beran

BRNO 2009

Abstrakt

Tato bakalářská práce se zabývá známými postupy detekce dopravních značek ve video sekvenci. Z úvodu se věnuje jejich popsání a vytyčení hlavních výhod a nevýhod s ohledem na rychlost zpracování a přesnosti detekce. V dalších kapitolách je pak navržen kompletní systém pro detekci a klasifikaci dopravních značek. V závěru je tento systém otestován na vytvořené testovací sadě dat a zhodnoceny výsledky experimentů.

Práce je zaměřena na metodu Template Matching pro klasifikaci a model hodnocení barevného vzhledu CieCam97, pro hledání kandidátních oblastí.

Abstract

This bachelor's thesis is concerned about known methods of detection traffic signs in video sequence. In introduction are explained and located main benefits in the light of process speed and accuracy of detection. At the next chapters is designed system for traffic signs detection and classification. In the end is this system tested on prepared testing data and results are evaluated.

The work is focused on method Template Matching for classification and color appearance model CieCam97 for detecting candidate areas.

Klíčová slova

detekce dopravních značek, opencv, ciecam97, template matching, hledání kontur

Keywords

traffic signs detection, opencv, ciecam97, template matching, find contours

Citace

Brzoza Martin: SMART CAR: Detekce dopravních značek, bakalářská práce, Brno, FIT VUT v Brně, 2009

SMART CAR: Detekce dopravních značek

Prohlášení

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

……

Martin Brzoza

8. 4. 2009

Poděkování

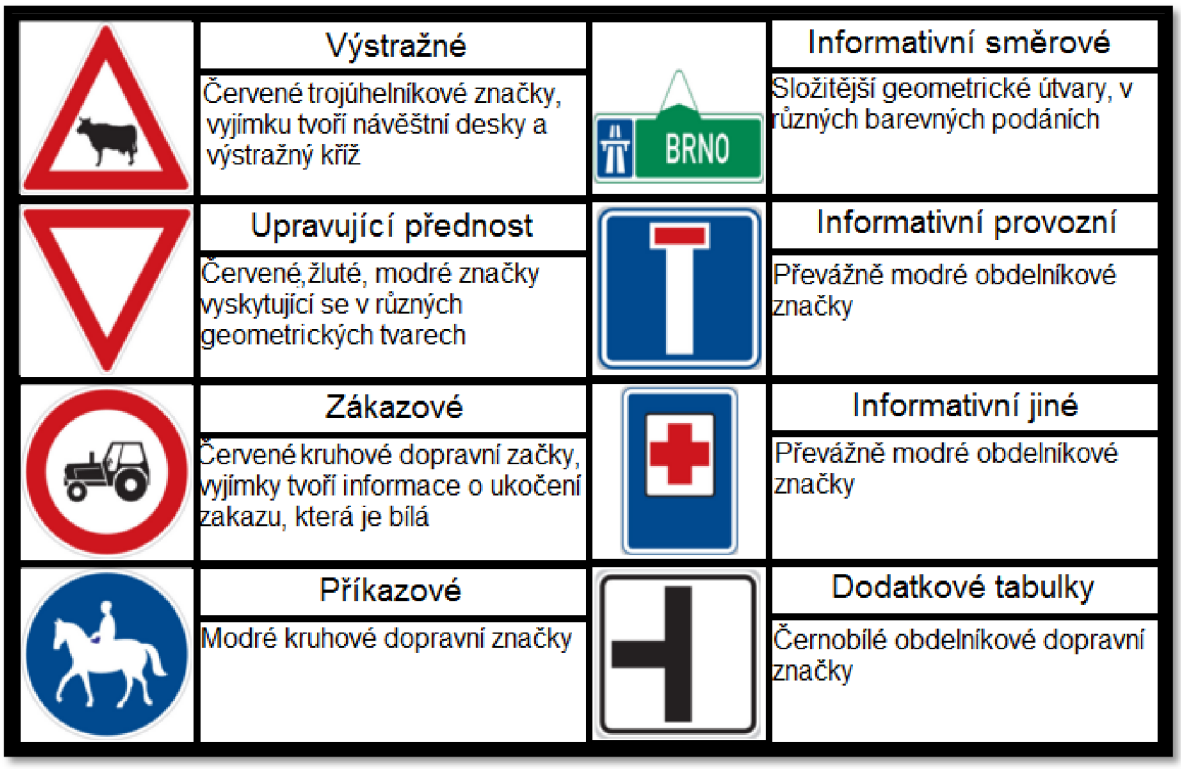
Chtěl bych velice poděkovat mému vedoucímu Ing. Vítězslavovi Beranovi za odbornou pomoc, cenné a užitečné informace při řešení této bakalářské práce.

Obsah

1 Úvod

Dopravní značky patří mezi základní prvky utvářející pravidla provozu na pozemních komunikacích. Upozorňují řidiče na případné nebezpečí, upravují přednost vozidel, informují o přítomnosti důležitých objektů. Pokud by naším cílem bylo vytvoření systému inteligentního vozidla, který by byl schopný řidiči podat důležité informace o provozu a umožnit mu tak soustředit se více na další důležité činnosti řízení, detekce dopravních značek by byla jedna z nejdůležitějších částí.

Dopravní značení v České republice a na Slovensku se začalo utvářet kolem roku 1935. Od té doby prodělalo velmi rozsáhlý vývoj. Přibylo několik nových kategorií barev i tvarů. Rozdělení dopravních značek do skupin, včetně specifického popisu si můžeme prohlédnout v obr. 1.1.



Obrázek 1.1: Přehled dopravního značení v České republice

2 Metody detekce dopravních značek

Mnoho známých metod detekce dopravních značek se zakládá na základních předpokladech o tvaru značky a barvy kontury, která ji ohraničuje. Tímto postupem jsou vybráni případní kandidáti, kteří by mohli být dopravní značkou. Pokud jsou v této části použity vysoké kritéria a metoda je dostatečně silná, podaří se nám najít s velkou pravděpodobností všechny značky v obraze.

Další částí je nalezenou oblast značky klasifikovat. Tato část nám řekne, již správné označení dopravní značky. Klasifikace se také používá pro redukci nalezené množiny potencionálních značek. V opačném případě, kdy nám detekce značky některou z oblastí nenašla, nám již klasifikace nepomůže.

2.1 Segmentace obrazu pomocí barev

Společným prvkem těchto metod je, že se snaží najít oblasti dopravních značek pomocí barevné informace. Většinou se hledá barva kontury, která značku ohraničuje, jako například červená v případě zákazových značek. Nevýhodou těchto metod bývá jejich závislost na kvalitě barvy značky, osvětlení scény a dalším podnebným podmínkám. Proto se často volí barevné modely podávající informace odpovídající více lidskému vnímání barvy.

2.1.1 Vytvoření binárního obrazu pomocí RGB

RGB je jeden z nejpoužívanějších a nejznámějších barevných modelů. Reprezentuje informaci o barvách většiny monitorů a projektorů. Používá aditivní způsob míchání tří barevných složek – červené (R), zelené (G) a modré (B). To znamená, že výsledná barva odpovídá kombinaci intenzit těchto tří barevných složek.

Tento barevný model je pro aplikaci detekce dopravních značek jen velmi obtížně použitelný, neboť zanedbává vliv osvětlení na celkové podávané informaci o barvě.

O použití tohoto barevného modelu se pokusil například autor [1]. Jedním z možných řešení vlivu osvětlení na celkovém podání barvy bylo vyřešeno použitím poměrů mezi jednotlivými barevnými složkami, jaké můžeme vidět ve vzorci (1), pro práh výstražných dopravních značek. Výsledkem této metody je pak binární obraz, kde bílé jsou body splňující daný práh a černé zůstanou ostatní. Jednou z významných výhod této metody je, že si nežádá žádné složité matematické výpočty pro převod a je tak velice nenáročná.

$$r/g > \alpha_w \,\&\, r/b > \beta_w \,\&\, g/b > \gamma_w \tag{1}$$

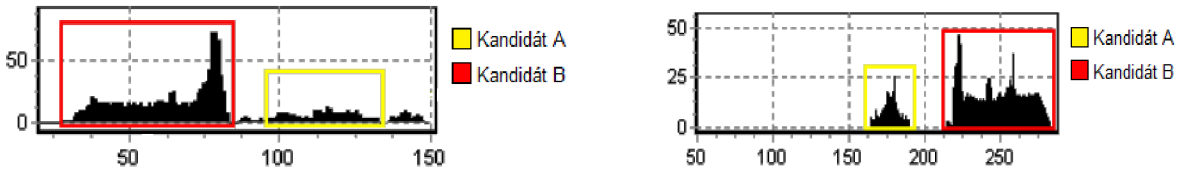
Proměnné r, g, b odpovídají hodnotám intenzit těchto tří barevných složek v daném bodě obrazu; α_w, β_w, γ_w jsou konstanty určující práh úrovní pro výstražné dopravní značky.

2.1.2 Redukce barev pomocí HSV

Jedním z barevných modelů, který se snaží popsat barvu mnohem věrněji, s ohledem na lidské vnímání barvy je HSV (Hue, Saturation, Value). Hue představuje samotný odstín barvy. Zapisuje se ve stupních, v rozsahu <0°,360°>. Saturation odpovídá nasycení barvy nebo také čistotě barvy. Udává se v procentech a určuje příměs šedé barvy do výsledného odstínu barvy. Poslední hodnotou je Value a jedná se o hodnotu jasu barvy.

Jeden z praktických využití modelu je možné vidět v [2]. Celý barevný prostor zkoumaného obrazu je v této práci redukován pomocí hodnot HSV do osmi základních barev- červené, žluté, zelené, azurové, modré, fialové, černé a bílé. V takto redukovaném obraze jsou pak případní kandidáti dopravní značky hledáni pomocí histogramu vertikální nebo horizontální projekce červené barvy (pokud tedy hledáme např. zákazové dopravní značky). Jedná se o histogram, který nám podává statistické informace o úrovních dané barevné složky pro určitý sloupec obrázku (vertikální) nebo řádek (horizontální). Vybrána je pak spojitá oblast s nejvyšší úrovní v těchto histogramech. Výhodou může být i to, že takový histogram můžeme porovnávat s histogramy projekce známých geometrických tvarů a získat tak informace o tvaru takovéto oblasti.

Obrázek 2.1: Vlevo originál, vpravo po redukci barev modelem HSV



Obrázek 2.2: Vlevo: histogram horizontální projekce červené barvy, vpravo: histogram vertikální projekce červené barvy

Obrázek 2.3: Vlevo: typická horizontální projekce pro trojúhelník, vpravo: vertikální

Vytvoření binárního obrazu pomocí modelu CieCam97

Model pro hodnocení barevného vzhledu přijatý v roce 1997 organizací CIE, se snahou o vytvoření určitého standardu na poli této problematiky. Tento model si klade za cíl podat věrohodné informace o vzhledu barvy za odlišných pozorovacích podmínek. Jelikož výčet všech vstupů a výstupů z modelu je celkem obsáhlý jejich přehled můžeme vidět v tabulce 2.1.

Tabulka 2.1: Vstupy modelu CieCam97

Označení	Popis
X, Y, Z	relativní tristimulus barevné složky
X_w, Y_w, Z_w	relativní tristimulus bílé barvy
La, Yb	jas adaptační oblasti, relativní jas pozadí
c, Nc, Fll, F	další parametry popisující okolí

Tabulka 2.2: Výstupy modelu CieCam97

Označení	Popis
J	světlost
h	odstín
C	sytost barvy
M	barevnost
S	nasycení barvy
Q	jas

Jak je vidět jedná se o velice škálovatelný model, který podává velké množství informací o zkoumané barvě. Abychom se nemuseli trápit měřením a výpočtem mnoha vstupních parametrů, bylo ustanoveno několik základních hodnot parametrů pro odlišné pozorovací podmínky, které můžeme vidět v tabulce 2.3. Organizace CIE pak také definovala standardní iluminant D65, který odpovídá dennímu světlu a ten definuje relativní tristimulus bílé barvy jako X=95.04, Y=100.00, Z=108.88.

Tabulka 2.3: Typické nastavení modelu CieCam97

Zkoumané okolí	F	c	Nc
Průměrné	1	0.69	1
Matné	0.9	0.59	0.95
Temné	0.8	0.525	0.8

Možné užití tohoto modelu pro detekci dopravních značek lze vidět například v [3]. Nejdříve jsou převedeny jednotlivé RGB barevné složky pomocí vztahu (2) na relativní XYZ tristimulus barevné složky a dále pak pro každý bod vypočteny hodnoty sytosti barvy (C) a odstínu (H). Z těchto hodnot je pak podle nastavených prahů vytvořen binární obraz. Autor se snaží využít plného potenciálu modelu a umožňuje podle odlišných pozorovacích podmínek nastavovat parametry modelu. Vybírá se mezi slunečným, deštivým a oblačným počasím. Pro každé nastavení se pak mění nastavení prahů hodnot sytosti a odstínu barvy a relativní tristimulus bílé barvy. Je uváděno až 94% úspěšnosti segmentace dopravní značky při slunečném počasí.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.2169 & 0.1068 & 0.048 \\ 0.1671 & 0.2068 & 0.0183 \\ 0.1319 & -0.0249 & 0.3209 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2}$$

Toto vše je bohužel vykoupeno náročností implementace a vysokou výpočetní náročností, díky častému používání mocnin a goniometrických funkcí. Model prošel již několika vylepšeními a nyní je k dispozici i verze CieCam02.

2.2 Segmentace obrazu pomocí hran

Hrana představuje v obraze místa s prudkou změnou úrovně jasu. Vstupem těchto metod bývá obrázek převedený do stupňů šedi. Problémem základních metod pro detekci hran v obraze bývá šum, proto se provádí práhování velikostí nalezené hrany, kdy zůstanou pouze ty výrazné a aplikování Gaussova filtru, před samotnou detekcí, pro redukci šumu.

Dopravní značky jsou tvořeny základními geometrickými tvary a ty je možné detekovat metodami uvedenými v kapitole 2.3, což je další krok po detekci hran v obraze. Nevýhodou těchto metod bývá detekce v terénu bohatém na geometrické obrazce (sídliště, městská centra), naopak významnou výhodou oproti metodám založených na barevné informaci bývá odolnost proti nekvalitnímu obarvení značky nebo rozdílným světelným podmínkám.

Další možností je podle nalezených hran, detekovanou oblast přímo klasifikovat. Možnou aplikaci můžeme vidět v [4], kde autor porovnává histogramy orientací nalezené oblasti a kandidátních značek.

Cílem této kapitoly je seznámení základních možností detekce hran v obraze a jejího možného uplatnění. Nakonec je pak uveden jeden z, v současnosti nejpoužívanějších detektorů – Cannyho hranový detektor.

Obrázek 2.4: Vlevo: zdrojový obraz, uprostřed: obraz ve stupních šedi, napravo: po použití Canny hranového detektoru

2.2.1 Hranové operátory

Používají se většinou k řešení první a druhé derivace jasové hodnoty obrazu. Mezi nejznámější hranové operátory řešící první derivaci patří Sobelův, Robertsův, pro druhou pak např. Laplaceův. První derivací obrazové funkce o dvou proměnných x a y získáváme gradient, čili informaci o směru prudké změny jasu v oblasti hrany a její velikost. Vzniká však velký problém se šumem, kdy je třeba volit správný práh, určující velikost hrany, která je ještě vybrána. Pokud zvolíme příliš nízký práh, propouštíme tím do výsledku mnoho šumu. Pokud naopak zvolíme příliš vysoký práh, můžeme ztratit některé podstatné hrany.

2.2.2 Cannyho hranový detektor

V současně době nejpoužívanější detektor. Podává velmi dobré výsledky. Na obraz je aplikován Gaussův filtr pro redukci šumu. Příliš široké hrany jsou ztenčeny hledáním lokálních maxim ve směru gradientu. Prahování velikostí gradientu hran je provedeno s hysterezí, což znamená, že v obrazu ponecháváme silné hrany, tenké pouze pokud navazuje na některou ze silnějších hran.

Možné využití pro detekci dopravních značek můžeme vidět v [5]. Autor se zde však omezuje pouze na kruhové dopravní značky. Elipsy jsou v obraze redukovaném pouze na hrany detekovány pomocí Houghovy transformace.

2.3 Hledání geometrických tvarů v obraze

Naším dalším krokem po segmentaci obrazu, ať již pomocí barev nebo hran může být hledání určitých souvislostí mezi nalezenými body. Můžeme hledat body, které leží na nějaké přímce nebo body, které utvářejí kružnici a klasifikovat další jednodušší geometrické obrazce. Možnou nevýhodou těchto metod může být narůstající náročnost metod, při složitějších útvarech.

2.3.1 Houghova transformace

Touto metodou lze hledat objekty, od nichž máme k dispozici analytický popis. Je vhodná pro hledání jednodušších typů objektů, jakou je například přímka nebo elipsa. Pokud například v obrazu hledáme přímku, výsledkem této metody nám bude rovnice přímky, na které leží nejvíce bodů z množiny bodů v prohledávaném prostoru. Základem je tedy rovnice (3). Hledané parametry jsou φ, r. Vybereme si libovolný existující bod v naší množině o souřadnicích x, y. Výčet všech možných řešení φ a r v daném prostoru nám vytvoří spojitou křivku. V místě, kde se tyto křivky jednotlivých bodů protnou, jsou parametry hledané přímky (na této přímce leží nejvíce bodů). S počtem neznámých parametrů v rovnici roste i výpočetní náročnost algoritmu. Nevyplatí se takto hledat tvary, které mají více než dvě neznámé. Využití pro hledání dopravních značek může být např. hledání elipsy dle rovnice (4).

$$x\cos\varphi + y\sin\varphi = r \tag{3}$$

$$(x-a)^2 + (x-b)^2 = r^2 \tag{4}$$

2.3.2 RANSAC

RANSAC(Random Sample Consensus) lze použít pro hledání geometrických tvarů, které lze popsat nějakým modelem nebo parametry. Náhodně vybíráme v obraze minimální počet bodů, ze kterých lze vytvořit parametry hledaného modelu. Můžeme si představit, že hledaný model v množině zkonstruujeme (například přímka) a hledáme body v prohledávaném prostoru, které s určitou tolerancí (prahem) ještě konstruovanému modelu náležejí. Pokud je množství dostačující skončíme a výsledkem nám jsou parametry hledaného modelu. Pokud ne pokračujeme další iterací algoritmu.

2.4 Metody rozpoznání dopravních značek

2.4.1 Template matching

Metoda vhodná pro hledání malých šablon v obraze. Šablona (template) je použita jako konvoluční maska, kterou aplikujeme na zdrojový obraz. V místě největšího výstupu se nachází hledaná šablona.

Nevýhodou této metody je docela velká výpočtová náročnost a závislost na velikosti a natočení šablony. Proto vznikají různé alternativy této metody.

2.4.2 Neuronové sítě

Dle [6] se jedná o základní výpočetní model používaný v umělé inteligenci. Pro naše účely se můžeme zaměřit na umělé neuronové sítě, jejichž vzorem je chování odpovídajících biologických struktur. Základním stavebním kamenem takovéto sítě jsou neurony, které jsou vzájemně propojeny. Podle jejich propojení pak rozlišujeme různé typy architektury sítě. Do jednoho neuronu může vstupovat signál z libovolného počtu neuronů. Každý takový signál může mít nastavenu určitou váhu, s kterou do neuronu vstupuje. Neuron může mít pouze jeden výstup, jehož hodnota je dána výsledkem specifické přenosové funkce.

Typickou vlastností je možnost učit se. Jde o nastavení vah vstupů a prahů přenosových funkcí, tak aby podávaly odpovídající výstupy. Rozlišujeme tyto základní přístupy:

- Učení s učitelem – je zde využito zpětné vazby. Síti je předložena sada trénovacích dat a máme základní nastavení vah vstupů a prahů. Podle výstupu sítě určíme chybu a provedeme korekci pro nové nastavení prahů a vah. Provedeme nový cyklus a vyhodnotíme novou chybu. Síť takto učíme, dokud nenalezneme takovou konfiguraci, abychom dosáhli námi stanovenou minimální chybu.
- Učení bez učitele – Výstup se nevyhodnocuje, protože není znám. Síť dostává na vstupu sadu vzorů a tyto vzory si třídí například podle typického zástupce.

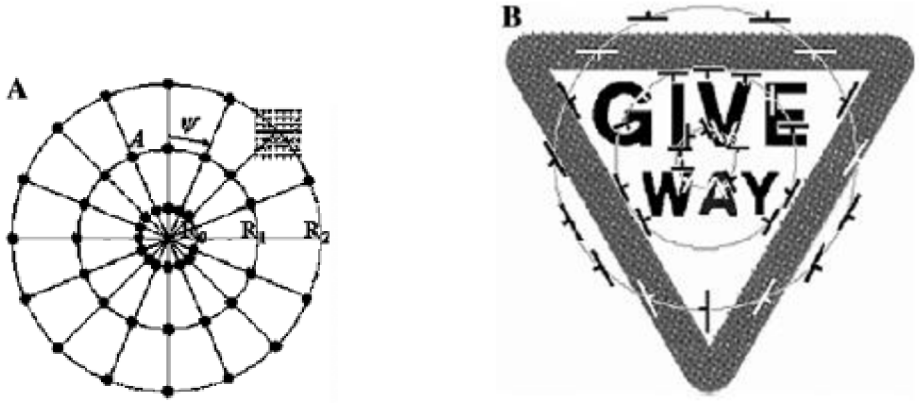
Použití vyhodnocování neuronových sítí pro rozpoznání dopravní značky můžeme vidět v [5]. Jako vstupní vektor je použito 63 uzlů, 3 určují průměrné hodnoty úrovně červené, modré, zelené barvy, dalších 30 vertikální histogram, zbytek horizontální. Hodnot je 30, protože velikost zkoumaného obrazu dopravní značky je normalizována na 30 řádků a 30 sloupců. Pro natrénování sítě byla vytvořena sada dopravních značek, kde pro každou značku bylo aplikováno několik typů zkreslení, jako například šum nebo rozmazání. Autor uvádí, že rychlost rozpoznávání takto natrénované neuronové sítě dosahuje průměrně 37.27 ms, pro zpracování jednoho snímku zkoumané scény.

2.4.3 FOST

Dle [1] je FOST(Foveal system for traffic signs) založen na metodě BMV(Behavioral model of visual perception and recognition) [7]. Tento systém se snaží imitovat některé z mechanismů lidského vnímání tvarů. Imituje změnu vnímání ostrosti směrem od centra oční sítnice až po její periferie a složitý proces zaměření na důležité fragmenty rozpoznávaného prvku. Systém je odolný vůči jednoduchému natočení rozpoznávaného prvku, posunutí i částečnému roztažení (změně velikosti).

Ve FOST je každý obrázek určený pro rozpoznávání analyzován pomocí speciálního senzoru, který můžeme vidět na obr. 2.5, důležité prvky tohoto systému pak můžeme shrnout do následujících bodů:

- Každé okolí jednoho ze 49 fixačních bodů senzoru je popsáno orientovanými segmenty, které se v něm nacházejí
- Fixační body senzoru se nacházejí na průsečících 16 přímek s třemi kružnicemi o rozdílných průměrech
- Orientace segmentu je pak dána rozdílem dvou orientovaných gaussových funkcí s prostorově posunutým středem a krokem 22.5°
- Prostorově variantní reprezentace rysů obrazu je emulována gaussovu konvolucí, s rozdílnou velikostí operátoru rostoucí s vzdáleností od středu senzoru



Obrázek 2.5: Vlevo senzor metody FOST, vpravo nalezené orientace hran okolí jednotlivých fixačních bodů

Souřadnice jednotlivých fixačních bodů senzoru můžeme vypočítat podle vzorců (5), (6).

$$x_i = X_0 + R_l \cos\psi_k \tag{5}$$

$$y_i = Y_0 + R_l \sin\psi_k \tag{6}$$

X_0,Y_0 je střed senzoru, R_l, odpovídá poloměru tří kružnic utvářející rozložení fixačních bodů (R_0=3,R_1=9,R_2=15), ψ_k = 22,5k, k = 0,1,..,15 je úhel rozestupu jednotlivých bodů.
Každý fixační bod A_i,i = 1, 2, …, 48 je popsán orientací dominantního úhlu α a jeho hustotou ρ.

$$\rho(A_i) = \max_{\varphi} \rho_{\varphi}(A_i) \tag{7}$$

$$\alpha(A_i) = \varphi \text{ kdy ž } \rho_{\varphi}(A_i) = \rho(A_i) \tag{8}$$

$$\rho_{\varphi}(A_i) = \rho_{\varphi}(x_i, y_i) = \frac{1}{S(x_i, y_i)} \sum_{m,n} Sg_{\varphi}(Or(m + x_i, n + y_i)) \tag{9}$$

$$Sg_p(x) = \begin{cases} 1 & \text{když x = p} \\ 0 & \text{jiné} \end{cases} \tag{10}$$

Or(x,y) je orientace detekované hrany v okolí se souřadnicemi (x, y). S(xi,y_i) je plocha oblasti zájmu a odpovídá 49 bodům; m, n = -3,…, 3; α = 0, 1, …, 15.
Z těchto dvojic je pak vytvořen vektor popisující značku (11).

$$\vec{F}(\vec{\alpha}, \vec{\rho}) = (\alpha(A_0) \cdots \alpha(A_{48}), \rho(A_0) \cdots \alpha(A_{48})) \tag{11}$$

Samotné rozpoznávání je pak prováděno dle vzorce (12). Máme v obraze kandidátní oblast, pro kterou máme vytvořen vektor (11) a máme vytvořenu databázi značek (vzorů), jejichž popis máme také redukován do těchto vektorů. Vzorec nám určí míru podobnosti mezi oblastí a vzorem. Většinou se prochází celá databáze značek a hledá největší podobnost, je však uváděno, že dobré výsledky podává i zvolený práh podobnosti na 25.

$$K^b = \sum_{i=0}^{48} \left[Sg\left(Or_i^b - Or_i^{rw}\right) \cdot \left(1 - \left|\rho_i^b - \rho_i^{rw}\right|\right) \right] \tag{12}$$

$$Sg(x) = \begin{cases} 1 & \text{když x = 0} \\ 0 & \text{jiné} \end{cases} \tag{13}$$

Or je orientace dominantní hrany v okolí fixačního bodu a ρ je její hustota. Horní index b označuje vzor značky z databáze, horní index rw označuje zkoumanou oblast.

Výhodou této metody je dle dostupných materiálů její rychlost a přesnost.

2.5 Možné problémy

Detekujeme prvky často podléhající venkovním vlivům, jako různá mechanická poškození, špatné pozorovací podmínky. Musíme se proto snažit nalézt způsob, jak se s nimi co nejlépe vypořádat. V této kapitole bude proveden souhrn a analýza nejčastějších problémů a uveden návrh jejich možných řešení.

Obrázek 2.6: Vlevo: příliš vybledlé barvy, uprostřed: nežádoucí vliv osvětlení, vpravo: rozmazání

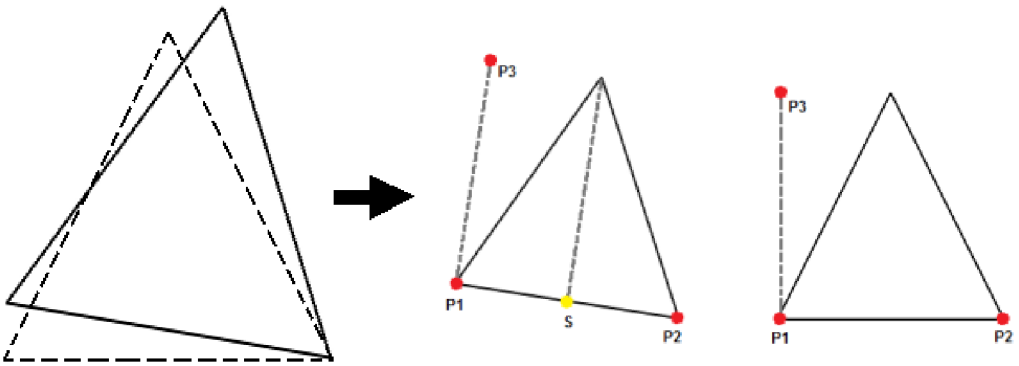
Vybledlé barvy

U hodně vybledlých barev, jaké můžeme vidět na obrázku 2.6, už selhávají metody založené na detekování podle barvy. Možným řešením by byla detekce pomocí hran. U méně vybledlých značek pomáhá snížení prahu pro hledanou barvu.

Rotace a perspektivní zkreslení

Značky mohou být snímány z různých úhlů, mohou být různě natočeny. Pokud pro klasifikaci značek používáme metodu, která není odolná proti takovýmto jevům, jako například Template Matching, musíme si nalezenou značku upravit. Pomohou nám základní afinní transformace, kdy víme, že afinní transformace je definována pomocí rovnoběžníku, který lze jednoznačně definovat pomocí tří bodů. Princip můžeme vidět na obrázku 2.7.

Toto nám však příliš nepomůže odstranit vliv perspektivního zkreslení, pro tento problém se používá např. hledání homografie, což už je složitější metoda odstraňující i perspektivní zkreslení. Pro její definování je potřeba minimálně čtyř bodů.



Obrázek 2.7: Úprava natočení značky

Rozmazání

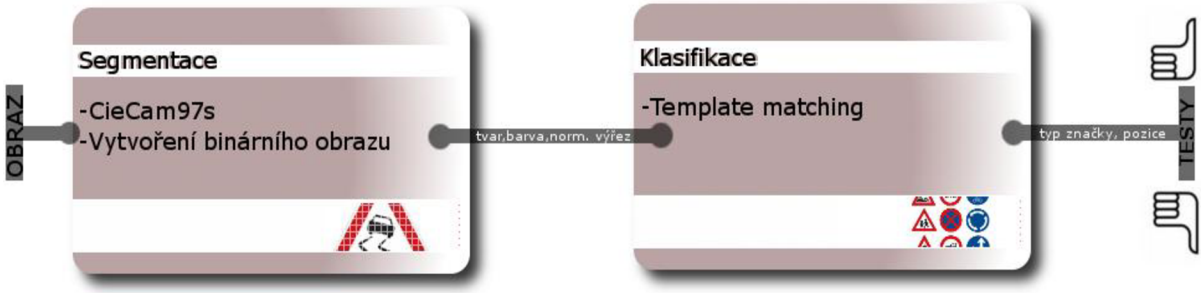
Nevýhodou detekce dopravních značek ve videu je možnost rozmazání obrazu, zvláště při vyšších rychlostech anebo použitím nekvalitní videokamery. Nalezení takové oblasti není obtížné. Problém

nastává při klasifikaci, kdy se nám značka jakoby zdvojí. Řešením by mohlo být umístění do databáze značek, vůči které klasifikujeme nalezené oblasti i značky v takto rozmazané podobě.

3 Návrh systému

Po fázi nastudování potřebných informací přichází část, kdy se pokusím navrhnout systém, který bude co nejlépe splňovat naše předpoklady. Těmi jsou robustnost, systém se musí umět vyrovnat s prudkými změnami, které se v reálném světě vyskytují. Rozšiřitelnost, systém bude složen z malých částí, které budou konat svou specifickou činnost a bude-li potřeba, lze tyto části lehce nahradit jinými. Rychlost, systém bude pracovat v reálném čase, proto je nutné volit mechanismy tak, aby nebyly příliš výpočetně náročné.

Návrh takového systému můžeme vidět na obr. 3.1.



Obrázek 3.1: Návrh systému pro detekci a klasifikaci dopravních značek

Vstupem takovéhoto systému je jeden snímek ze zpracovávaného videa. Tento snímek prochází segmentací. Segmentujeme pomocí barev, použitím modelu pro hodnocení barevného vzhledu CieCam97. Tento model splňuje naši dříve zmiňovanou robustnost. Model se snaží co nejpřesněji popsat zkoumané barvy a navíc je široce škálovatelný. Je možné volit vstupní parametry modelu podle podnebí. V současnosti existuje mnoho již ověřených parametrů modelu, jak pro slunečné počasí, tak pro zatažené. Práhovat budeme podle úrovně odstínu barvy (H) a sytosti barvy (C).

Výstupem segmentace je pro nás binární reprezentace obrazu, kde bílé oblasti reprezentují části obrazu, které vyhovují barevným prahům. Tato reprezentace je pro nás ještě nedostačující, proto budeme hledat v binárním obraze kontury. Takto dostaneme jasné hranice a obraz rozdělíme na přesné části, ve kterých by se měla nacházet značka. Pokud tyto části (kontury) správně aproximujeme polygony, dostaneme vypovídající informaci o tvaru.

Máme tedy informaci o tvaru, umístění a barvě značky, je tedy správný okamžik dostat informace v rozumné formě do další časti – klasifikace. Jelikož značka může být lehce natočená, provedeme na výřezu značky afinní transformace, jež byly popsány v předcházející kapitole. Výřez normalizujeme na standardní velikost a vložíme do seznamu oblastí potřebných k oklasifikování.

Klasifikace je část systému, která je odpovědna za pojmenování jednotlivých oblastí jejich skutečným jménem. K dispozici máme databázi značek rozdělenou podle tvaru a barvy. Pokud příslušná značka nevyhovuje žádné ze značek z databáze podle určených kritérií, je tato oblast

vyškrtnuta z oblastí, kde se nachází značka. Klasifikace značek je provedena metodou Template Matching. Jedná se o kompromis s ohledem na dostupný čas a jednoduchost implementace. Metoda je relativně časově náročná. Pokud nám ale pomůže použitý barevný model správně redukovat prohledávané oblasti a databázi značek inteligentně rozdělíme podle barvy a tvaru, je dle mne možné dosáhnout celkem dobré úrovně.

4 Implementace a testy

Celý systém byl implementován v jazyce C++ s využitím knihovny OpenCV a otestován pod systémem Windows. Tato kapitola se věnuje popisu realizování jednotlivých částí dle návrhu a částečně předkládá i možnosti další práce s celým systémem. Na závěr jsou popsány sady testů, kterými prošel a zobrazeny výsledky.

4.1 OpenCV

Je jedna z nejvýznamnějších knihoven na poli počítačového vidění. Jedná se o celek čítající přes 500 optimalizovaných algoritmů pro zpracování obrazu, vyvinutý v začátcích firmou Intel. Současně nabízí základní rutiny pro vytváření uživatelského rozhraní a přehrávání videa. Tuto oblast obstarává část knihovny nazvaná HighGui.

V aplikaci využívám verzi 1.1, jak pro samotné zpracování, tak vytvoření jednoduchého uživatelského rozhraní.

4.2 Segmentace obrazu

4.2.1 Popis

Samotná segmentace obrazu na části je provedena pomocí modelu pro hodnocení barevného vzhledu CieCam97, byla použita revidovaná implementace uvedená v [8]. Hlavní výpočetní kód modelu je ukrytý ve třídě `CieCam`. Model provádí velkou sadu výpočtů a velké množství z nich lze provést jen jednou, protože závisí na vstupních parametrech modelu. Proto byla vytvořena privátní metoda `precomputeVals()`, která je volána při nastavování vstupních parametrů modelu metodou `setConfig()` a přepočítává všechny tyto jednorázové výpočty. Práce s třídou je intuitivní. Zdrojový obraz je nastaven pomocí metody `setInput()` a klient této třídy pak volá metodu `get()` která vrací výstup výpočtu hodnot pro konkrétní bod v obraze.

Tuto třídu pro svou práci využívá třída `CCBinaryMap`, která vytváří binární reprezentaci obrazu podle hodnot výstupu modelu. Bílé budou oblasti, které vyhovují prahům hodnot a černé ostatní oblasti. Hlavní metody jsou `setInput()`, která nastaví používanou instanci `CieCam` a metoda `make()`, jejímž výstupem je binární obraz.

Máme tedy reprezentaci bodů, ve kterých by se mohla nacházet značka, nyní tyto oblasti potřebujeme ohraničit, nějak je popsat a rozdělit. K tomu slouží třída `Countours`. Mezi její hlavní

veřejné metody patří `setInput()`, která nastaví zpracovávaný obraz binární mapy a `make()`, která rozdělí vstupní obraz podle kontur z binární mapy na jednotlivé části a pokusí se určit jejich tvar. Metoda `make()` využívá přednostně algoritmů z knihovny OpenCv. Nejdříve se v binárním obraze naleznou kontury pomocí metody `cvFindContours()`. Všechny jsou testovány na velikost, nevyhovující (příliš malé) jsou vyřazeny ze zpracování. Vyhovující kontury jsou dále aproximovány na polygony pomocí metody `cvApproxPoly()`. Pokud počet hran takto aproximovaného polygonu souhlasí s některým ze známých tvarů (trojúhelník, obdélník,…), tak je provedena pomocí afinních transformací rotace a změna velikosti. Natočení značky je určeno pomocí dvou bodů základny tvaru, podrobněji je možno dočíst v kapitole 2.5.

Aby byla v následující části vylepšena klasifikace dopravní značky, je okolí kontury vyplněno šedou barvou. Tím je odstraněn vliv pozadí značky na výslednou klasifikaci.

Obrázek 4.1: Postup detekce dopravních značek

Určování prahu barev dopravní značky dle modelu CieCam97

V systému bylo zatím implementováno a odzkoušeno hledání červených dopravních značek, přesto zde uvedu i možnosti určení barevného prahu pro ostatní barvy.

Samotný práh je určen na dvou hodnotách a to kompozici odstínu - H_c pro jednotlivé barvy, který je přepočítán z hodnoty odstínu – h, pomocí vztahů uvedených v [8] a hodnoty sytosti barvy - C. Přehled algoritmu pro převod a jejich použitý práh lze vidět v tabulce 4.1.

Tabulka 4.1: Barvy dopravních značek v modelu CieCam97

Barva	Převod na Hc barvy	Práh Hc	Práh C

	`if (H>300) Hc = H-300;`
	`else if (H<100) Hc = 100-H;`
	`else Hc= 0;`
		$H_c > 40$	C > 10

	`if (H<=100) Hc = H;`
	`else if(H<200) Hc = 200-H;`
	`else Hc=0;`
		$H_c > 40$	C > 10

	`If (H>300) Hc = 400-H;`
	`Else if (H>200) Hc=C54-200;`
	`Else Hc=0;`
		$H_c > 40$	C > 10

Hledání kontur v knihovně OpenCV

I když máme vytvořen binární obraz reprezentující body, které vyhovují našim prahům, ještě nemáme informace o spojitých oblastech a jejich hranicích. O toto se nám postará funkce `cvFindContours()`.

Kontura je seznam bodů, který popisuje v jednom směru křivku v obraze. Existuje několik způsobů, jak tuto křivku popsat. V OpenCV jsou kontury popsány sekvencemi bodů, kdy každý prvek v sekvenci nese informaci o dalším bodu ležícím na křivce. Způsob, který funkce používá k nalezení těchto křivek je odvozen od metody, kterou představil Suzuki. OpenCV uchovává nalezené kontury ve specifické stromové struktuře. Více se lze o tomto dočíst v [9].

Nyní se podrobněji podívejme na samotnou funkci:

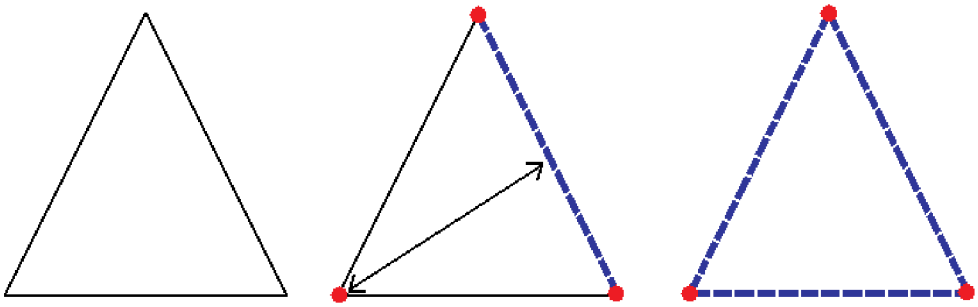
```
int cvFindCountours(
    IplImage* img,
    CvMemStorage* storage,
    CvSeq** firstContour,
    int headerSize = sizeof(CvContour),
   CvContourRetrievalMode mode = CV_RETR_LIST,
   CvChainApproxMethod method = CV_CHAIN_APPROX_SIMPLE
);
```

První argument označuje obrázek, ve kterém hledáme kontury (nejčastěji se jedná o binární obraz nebo obraz hran). Druhý argument je úložiště, kde bude moct funkce vytvářet své dynamické proměnné. Třetí parametr – `firstContour` je ukazatel na paměťové místo první nalezené místo kontury. OpenCV nám ukazatel na tuto konturu sama naplní. Velikost hlavičky neboli argument `headerSize` je závislá od způsobu uložení bodů kontury a může být např. `sizeof(CvContour)` nebo `sizeof(CvChain)`. Předposlední argument definuje způsob uložení kontur a poslední definuje způsob reprezentace bodů definující křivku kontury. Pro naše

potřeby je užitečné použít `CV_RETR_EXTERNAL` pro způsob uložení kontur, což způsobí, že z více kontur na jednom místě bude vybrána pouze ta nejokrajovější. Funkce vrací počet nalezených kontur.

Mezi další užitečné funkce pracující s výsledkem nalezených kontur jsou `cvBoundingRect()`, která vrátí obdélník ohraničující konturu, `cvDrawContours()`, která kontury vykreslí a `cvAproxPoly()`. Poslední jmenovaná je obzvláště pro nás velice užitečná, protože může pomoci v analýze tvaru nalezené kontury. Vstupem metody je jedna kontura, kterou chceme aproximovat a míra preciznosti výsledné aproximace.

Princip algoritmu aproximace kontury ideálního trojúhelníku dopravní značky lze vidět na obrázku 4.2. Nejprve jsou vybrány dva nejvzdálenější body a je vytvořena první přímka aproximované kontury. Dalším krokem je nalezení nejvzdálenějšího bodu od této přímky a jeho přidáním do aproximovaného polygonu. Celý proces pokračuje takto dál přidáváním nejvzdálenějších bodů, dokud tato vzdálenost není menší než hodnota nastavení preciznosti aproximace.



Obrázek 4.2: Princip aproximace kontury polygony

Experimentováním jsem došel k závěru, že pro účely rozhodnutí jestli se jedná o obdélník, čtverec nebo trojúhelník je vhodné volit větší rozptyl preciznosti nacházení aproximací s ohledem na velikost kontury. Metoda je pak méně náchylná k porušeným konturám způsobené nedokonalostí vytvořeného binárního obrazu.

4.2.2 Řešené problémy

Jedním z hlavních problémů této části je velmi vysoká výpočtová náročnost modelu CieCam97, která znemožňovala detekci v reálném čase. Rozhodl jsem se proto hodnoty modelu vypočítat do trojrozměrného pole, jehož indexy odpovídají hodnotám R, G, B. Tyto vypočítané hodnoty jsou uloženy do binárního souboru, který se načítá při dalším spuštění programu. I toto řešení však není úplně správné, neboť takovýto soubor vypočítaných hodnot má velikost přibližně 120 MB a tato velikost se nám také promítne do obsazení operační paměti. Mnohem lepší by bylo řešení, kdy bychom měli vyhledávací tabulku obsahující pouze prvky kombinací R, G, B vyhovující našim prahům. Toto by však vyžadovalo implementaci převráceného modelu CieCam97, která je velmi náročná a vyžadovala by vymezení delšího času na implementaci i přesto však je urychlení značné.

4.3 Klasifikace dopravní značky

4.3.1 Popis

Aby bylo podle čeho klasifikovat oblasti nalezené v předchozí části je potřeba mít připravenu databázi dopravních značek. O tuto úlohu se postará třída `SignDatabase`. Mezi její hlavní veřejné metody patří metoda `load()`, která přijímá jako parametr soubor s definicí dopravních značek. Značky jsou načteny a normalizovány na standardní velikost 50x50 bodů. Dále jsou značky rozděleny do kategorií podle tvaru a barvy. Další důležitou metodou je `next()`, která vrací další značku z databáze o zadaných parametrech a `reset ()`, která naopak nastaví iterátory značek na první pozici.

Za samotnou klasifikaci nalezených oblastí odpovídá třída `SignClassificator` a její hlavní veřejná metoda `classify()`. Pro řešení je použita, jak již bylo zmíněno v návrhu, metoda Template Matching. Tady znovu přichází na řadu OpenCv a funkce `cvMatchTemplate()`. Funkce nabízí několik typů implementací. Po vyzkoušení jsem si vybral metodu nalezení schody korelací, která má ve funkci označení `CV_TM_CORR_NORMED`, podávala totiž nejlepší výsledky. Metoda porovnává šablonu s obrázkem podle vzorce (14). T je hledaná šablona, I je zkoumaný obrázek. V místě kde se šablona bude nejvíce shodovat s oblastí v obrázku, bude výstup funkce největší.

$$R(x,y) = \sum_{x',y'} \left[T(x',y') \cdot I(x+x', y+y') \right]^2 \tag{14}$$

Výsledek metody je navíc normovaný, což je užitečné neboť umožňuje redukovat vliv rozdílu osvětlení mezi šablonou a obrázkem. Normalizační koeficient je vypočten dle vztahu (15). Výsledek metody je pak normalizován vztahem (16).

$$Z(x,y) = \sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x', y+x')^2} \tag{15}$$

$$R_{norm}(x,y) = \frac{R(x,y)}{Z(x,y)} \tag{16}$$

4.3.2 Řešené problémy

Jedním z hlavních problémů, které jsem řešil, bylo vůbec použití metody Template Matching. Nejdříve jsem v databázi použil předkreslené značky a hledal největší podobnost. Jelikož se však v předkreslených značkách neodráží vliv osvětlení na celkovém podání značky, klasifikace nepodávala dobré výsledky. Proto jsem se místo předkreslených značek rozhodl použít v databázi značky nafocené přímo v terénu, některé i vícekrát za různých podmínek. Tady pak vznikal problém, že metoda dávala často vysokou míru podobnosti i na nesprávné oblasti, proto jsem se rozhodl nastavit vysoký práh hodnoty podobnosti vracející Template Matching. Pokud není nalezena podobnost přesahující tento práh, o oblasti je rozhodnuto, že není značkou. Další problémem je, že s velikostí databáze narůstá i výpočetní náročnost celé klasifikace, protože metoda není nijak optimalizována a porovnává vzor a kandidát bod po dobu.

Obrázek 4.3: Ukázka šablon použitých v databázi dopravních značek

4.4 Testy

Jelikož cílem bylo vytvořit systém rozpoznávání dopravních značek ve videu. Byly vytvořeny dva ukazatele kvality. První, jehož výsledky jsou uvedeny v tabulce 4.3, mají za cíl ukázat relativní výkonnostní schopnosti systému pro práci v reálném čase a ukazují průměrné časy potřebné pro detekci značek v obrazu a jejich klasifikaci. Druhá část testů, uvedená v tabulce 4.2, zahrnuje kvalitativní hodnocení systému, jeho úspěšnost v detekci a klasifikaci dopravní značky. Testovaných značek není sice moc, reprezentativní by byl o něco větší vzorek, ale snažil jsem se volit různé typy značek s různou pozorovací kvalitou.

Pozn. testováno, pro představu, bylo na konfiguraci s dvoujádrovým procesorem frekvence 1.6 MHz. Výsledky s přepočítáním znamenají, že byl použit předpočítaný model CieCam97.

Tabulka 4.2: Výsledky testování kvality

Celkový počet značek	29
Počet správně detekovaných	27
Počet špatně detekovaných	1
Počet správně klasifikovaných	25
Počet špatně klasifikovaných	2

Tabulka 4.3: Výsledky rychlosti zpracování

	S přepočítáním	Bez přepočítání
Detekce	5.755 ms	51.5 ms
Klasifikace	17.482 ms	17.619 ms

Pro snadnější testování byla výsledná aplikace vybavena i testovacím režimem, kdy na vstupu přijímá soubor anotací značek. Každý takovýto soubor je tvořen následujícími řádky:

```
Soubor[.png|.jpg], x_souradnice, y_souradnice, popis_znacky
```

Např.

```
test1.png, 70, 89, Deti
```

Můžeme uvést i více řádků se stejným názvem souboru. Bude pak bráno, že v tomto obrázku je více značek. Po spuštění a proběhnutí testů pak program vypíše informace podobné těm, které můžeme vidět v předchozí tabulce, včetně názvů obrázků ve kterých došlo k chybě detekce nebo klasifikace.

Obrázek 4.4: Výstupy detekce pro vybrané vzorky, uprostřed chybné výsledky klasifikace

Na obrázku 4.4 můžeme vidět několik výstupů z aplikace pro detekci značek. Uprostřed lze vidět špatné výsledky klasifikace. Důvod chyby obrázku vlevo, kde byla správně detekovaná oblast, ale špatně klasifikována jako výstražná značka Děti, je zřejmý. Chybu zakrytím části značky, by bylo možné vykompenzovat rozšířením databáze o tyto případy. Otázkou je, jestli je to nutné, neboť se jedná o ojedinělé případy. Chyba v klasifikaci napravo je způsobena nedokonalostí vytvořeného binárního obrazu, v kontuře tvaru dopravní značky tak vznikly přerušené části. Díky tomu nebylo možné určit pozadí značky a následně ho odstranit, protože vnitřek kontury nebyl uzavřený. Do porovnávání se vzory se tak započetla i barevná informace o pozadí značky. Řešením v těchto případech může být použití dilatace na binární obraz, která menší přerušení slepí. Ta nám však oblasti jednotlivých značek také rozšíří a může se stát, že pokud se nachází více značek těsně pod sebou, spojí se touto operací dohromady a vytvoří jednu oblast.

Obrázek 4.5: Ukázka špatně odstraněného pozadí a špatně vytvořené binární mapy

5 Závěr

Jak je vidět z výsledků testů, úspěšnost je docela veliká. Je to dáno víceméně tím, že zkoumaný vzorek není příliš velký a také tím, že v databázi značek byly použity značky, které byly nafoceny přímo z reálného prostředí a testovány pouze případy značek, které se nachází v této databázi. V současnosti databáze obsahuje velkou část červených trojúhelníkových značek a nějakou část kruhových. Rozšíření by nemělo dělat další problémy, stačí další značky nafotit a dodat je do databáze podle dokumentace na přiloženém CD. Je důležité, aby byla pro použitou metodu Template Matching vytvořena větší databáze nafocených dopravních značek, protože metoda porovnává podobnost k šabloně. Je dobré mít pro jeden typ značky i více vzorů, nafocených za různých podmínek. Bohužel čím větší databáze, tím větší časová náročnost na porovnávání. Možným urychlením by mohlo být zvolení určité hranice podobnosti, který by přerušil algoritmus hledání nejlepšího výsledku podobnosti.

Velice jsem byl zvědavý, jaké výsledky bude podávat model pro hodnocení barevného vzhledu CieCam97, použitý v tomto systému. Bohužel jsem neměl možnost detekce např. v deštivých podmínkách a vyzkoušet tak různé typy nastavení parametrů modelu. I přesto model podává dobré výsledky, problémy nastávaly v případech, kdy se v obraze objevila například sytě červená střecha, ale tady tyto oblasti odstranila následná klasifikace.

Celý systém je nyní ve fázi, kdy kromě toho, že je schopen předvést základní detekci dopravních značek v obraze, je také schopen ukázat jak by pak mohla vypadat taková detekce v reálném nasazení. Programový kód jsem se snažil strukturovat a psát tak, aby bylo možné jeho snadné rozšíření a díky objektovému členění na jednotlivé části – detekce, klasifikace i snadnou změnu použitých metod. Kód je také kvalitně komentovaný a obsahuje komentáře, které směrují systém k dalšímu vývoji a rozšiřování.

Závěrem se ještě uchýlím k zhodnocení systému a zamyšlení nad jeho ideální podobou. Myslím si, že ideální systém, který nám pohotově nalezne a rozpozná značku v obraze zatím navrhnout nelze. Vždycky zůstanou aspoň částečné problémy s různými pozorovacími podmínkami, poškozeními značky, zakrytími a různými světelnými podmínkami. Velmi kvalitní schopnost detekce modelu CieCam97 přebíjí veliká výpočetní náročnost, proto bych v této fázi teď volil raději kompromis v podobě modelu HSV. Rozpoznávání tvarů pomocí aproximace kontur na polygony je docela nepřesné a hodně záleží na kvalitě binární mapy, ve které byla vyhledávána kontura. Lepší by nejspíš bylo použít klasifikaci tvarů pomoci FOST. Použití tohoto modelu by se tato část programu skloubila zároveň s klasifikací značky a podle toho co se lze dočíst, je tato metoda o dost rychlejší, než použitý Template Matching, jehož nevýhodou je i potřeba velké databáze porovnávaných dopravních značek. Možnou další alternativou pro klasifikaci značky by mohli být i neuronové sítě.

Literatura

[1] **Schneier, Michael.** Road sign detection. *IEEE Computer Society International Conference on Computer Vision and Pattern Recognition.* [Online] 2005. [Citace: 1. Květen 2009.] http://www.isd.mel.nist.gov/documents/shneier/Road_Sign_Detection.pdf.

[2] **Chieh Hsien, Jung, Sheng Liou, YI and Yuan Chen, Shu.** Road Sign Detection and Recognition Using Hidden Markov Model. *Asian Journal of Health and Information Sciences.* [Online] 2006. [Cited: Duben 2, 2009.] http://www.asia.edu.tw/ajhis/no1/07-his06018.pdf.

[3] **Xiaohong, Gao, a další.** Colour Vision Model-Based Approach for Segmentation of Traffic Signs. *EURASIP Journal on Image and Video Processing.* 2008. [Citace: 1. Duben 2009.]

[4] **Alefs, Bram, a další.** Road Sign Detection from Edge Orientation Histograms. *Intelligent Vehicles Symposium.* 2007. [Citace: 1. Květen 2009.]

[5] **Lorsakul, Auranuch a Suthakorn, Jackrit.** Traffic Sign Recognition Using NeuralNetwork on OpenCV: Toward Intelligent Vehicle/Driver Assistance System. *Center for Biomedical and Robotics Technology.* [Online] [Citace: 4. Květen 2009.] http://www.bartlab.org/Dr.%20Jackrit%27s%20Papers/ney/1.TRAFFIC_SIGN_Lorsakul_ISR.pdf.

[6] Neuronová síť. *Wikipedie: Otevřená encyklopedie.* [Online] 7. Únor 2009. [Citace: 5. Květen 2009.] http://cs.wikipedia.org/w/index.php?title=Neuronov%C3%A1_s%C3%AD%C5%A5&oldid=3595710.

[7] **Rybak, I.A., a další.** A model of attention-guided visual perception and recognition. [Online] 1997. [Citace: 5. Duben 2009.] http://www.jdl.ac.cn/project/faceId/paperreading/Paper/ffang_20070713_Vision%20Research1998_AttensionGuidedPerception.pdf.

[8] **Fairchild, Mark D.** A Revision of CIECAM97s for Practical Applications. *Munsell Color Science Laboratory.* [Online] [Citace: 10. Duben 2009.] http://www.cis.rit.edu/fairchild/PDFs/PAP10.pdf.

[9] **Gao, X.W., a další.** Recognition of traffic signs based on their colour and shape features extracted using human vision models. *Journal of Visual Communication and Image Representation.* [Online] 2005. [Citace: 3. Duben 2009.] http://nisms.krinc.ru/papers/jvci2006.pdf.

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

[11] Template matching. *Wikipedia, The Free Encyclopedia.* [Online] 27. Duben 2009. [Citace: 5. Květen 2009.] http://en.wikipedia.org/w/index.php?title=Template_matching&oldid=286401156.

[12] RANSAC. *Wikipedia, The Free Encyclopedia.* [Online] 15. Květen 2009. [Citace: 16. Květen 2009.] http://en.wikipedia.org/w/index.php?title=RANSAC&oldid=290171220.

[13] HSV. *Wikipedie: Otevřená encyklopedie.* [Online] 23. Duben 2009. [Citace: 3. 5 2009.] http://cs.wikipedia.org/w/index.php?title=HSV&oldid=3880324.

[14] Hough transform. *Wikipedia, The Free Encyclopedia.* [Online] 27. Duben 2009. [Citace: 1. Květen 2009.] http://en.wikipedia.org/w/index.php?title=Hough_transform&oldid=286511231.

[15] Edge detection. *Wikipedia, The Free Encyclopedia.* [Online] 28. Duben 2009. [Citace: 2. Květen 2009.] http://en.wikipedia.org/w/index.php?title=Edge_detection&oldid=286568477.

[16] Cannyho hranový detektor. *Wikipedie: Otevřená encyklopedie.* [Online] 18. 1 2008. [Citace: 20. 4 2009.] http://cs.wikipedia.org/w/index.php?title=Cannyho_hranov%C3%BD_detektor&oldid=2166951.