

Detekce a zpracování obrazových dat dopravních značek pro rozpoznávání umělými neuronovými sítěmi

Detection and Image Data Processing of Road Signs for Recognition by Means of Artificial Neural Networks

Bc. Josef Hrubý

Diplomová práce
2014



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2013/2014

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Josef Hrubý**
Osobní číslo: **A12419**
Studijní program: **N3902 Inženýrská informatika**
Studijní obor: **Počítačové a komunikační systémy**
Forma studia: **prezenční**

Téma práce: **Detekce a zpracování obrazových dat dopravních značek pro rozpoznávání umělými neuronovými sítěmi**

Téma anglicky: **The Detection and Processing of Road Sign Image Recognition Data by Means of Artificial Neural Networks**

Zásady pro vypracování:

1. Seznamte se s oblastí zpracování obrazu a rozpoznávání vzorů.
2. Seznamte se s oblastí neuronových sítí a především přípravou dat pro trénovací a testovací množiny.
3. Připravte metodiku zpracování obrazových dat s ohledem na tvorbu trénovacích a testovacích množin pro neuronové sítě.
4. Připravte vhodná data s dopravními značkami pro trénovací a testovací množinu.
5. Exportujte data do vhodných formátů pro další zpracování neuronovou sítí.

Rozsah diplomové práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. ZELINKA, Ivan. Umělá inteligence I. Volume 1. Zlín: Vutium, Brno, 1998, 126 p. ISBN 80-214-1163-5.
2. RAHMAN, Atiqur. Computer vision and action recognition: a guide for image processing and computer vision community for action understanding. Amsterdam: Atlantis Press, c2011, xxi, 211 s. ISBN 9789491216206.
3. ŠNOREK M., JIŘINA M.: Neuronové sítě a neuropočítače, ČVUT, 1996, ISBN 80-01-01455-X.
4. BÍLA J.: Umělá inteligence a neuronové sítě v aplikacích, ČVUT, 1996, ISBN 80-01-01275-1.
5. BOSE N.K., LIANG P.: Neural Network Fundamentals with Graphs, Algorithms, and Applications, McGraw-Hill Series in Electrical and Computer Engineering, 1996, ISBN 0-07-006618-3.
6. FREEMAN J. A.: Simulating Neural Networks with Mathematica, Adison-Wesley Publishing Company, 1994, ISBN 0-201-56629-X.
7. NOVÁK M., FABER J., KUFUDAKI O.: Neuronové sítě a informační systémy živých organismů, Grada, 1993, ISBN 80-58424-95-9.
8. The Mathematica Book, manuál softwaru Mathematica.

Vedoucí diplomové práce:

doc. Ing. Zuzana Komínková Oplatková, Ph.D.
Ústav informatiky a umělé inteligence


Datum zadání diplomové práce:

7. února 2014

Termín odevzdání diplomové práce:

27. května 2014

Ve Zlíně dne 7. února 2014


prof. Ing. Vladimír Vašek, CSc.
děkan




prof. Ing. Karel Vlček, CSc.
ředitel ústavu

ABSTRAKT

Práce se zabývá návrhem algoritmu pro detekci a rozpoznávání dopravních značek pomocí umělé neuronové sítě metodami zpracování obrazu v softwaru Wolfram Mathematica. Popsány jsou postupy při vývoji algoritmu a tvorba učících či testovacích dat pro umělou neuronovou síť. Tato práce je součástí týmové spolupráce, kdy výsledkem je aplikace naprogramovaná v jazyku C#. Ta umí rozpoznávat dopravní značky z videozáznamů i statických snímků.

Klíčová slova: Dopravní značka, zpracování obrazu, Wolfram Mathematica, umělá neuronová síť

ABSTRACT

The work deals an algorithm for the detection and recognition of traffic signs using artificial neuronové network image processing methods in the software Wolfram Mathematica. Procedures are given for the development and creation of learning algorithms or test data for the artificial neural network. This work is part of teamwork, the result is an application programmed in C #. It can recognize traffic signs from video clips and images.

Keywords: Traffic sign, image processing, artificial neuron network

Děkuji za cenné rady a připomínky vedoucí diplomové práce
doc. Ing. Zuzaně Komínkové Oplatkové, Ph.D.

Motto:

„Smyslem vzdělání nemůže být, že každý umí vysvětlit Einsteinovu teorii relativity, ale nikdo neumí spravit kapající kohoutek.“ N. Blum

Prohlašuji, že

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

Prohlašuji,

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

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	11
1 DOPRAVNÍ ZNAČKY	12
1.1 ROZDĚLENÍ.....	12
1.1.1 Rozdělení dle významu	12
1.1.2 Rozdělení dle provedení.....	13
1.2 TECHNICKÉ PŘEDPOKLADY A UMÍSTĚNÍ	14
1.2.1 Umístění vzhledem k vozovce	14
1.2.2 Umístění vzhledem k dopravním značkám	15
1.3 DOPRAVNÍ ZNAČKY V ZAHRANIČÍ	16
1.3.1 Evropský systém	16
1.3.2 Americký systém.....	17
1.4 SYSTÉMY NA ROZPOZNÁVÁNÍ DOPRAVNÍCH ZNAČEK	17
2 ZPRACOVÁNÍ OBRAZU	19
2.1 BAREVNÉ MODELY	19
2.1.1 RGB.....	19
2.1.2 CMY(K)	20
2.1.3 HSV (HSB)	21
2.1.4 Lab.....	22
2.2 ZPRACOVÁNÍ BINÁRNÍHO OBRAZU A JEHO EXTRAKCE	23
2.2.1 Segmentace	23
2.2.2 Prahování a histogramy	24
2.2.3 Gaussův filtr	25
2.2.4 Hledání hran	26
2.2.5 Cannyho hranový detektor	27
2.2.6 Morfologické operace dilatace a eroze.....	27
2.2.7 Měření objektů – kruhovost, pravoúhlost a podlouhlost.....	28
3 NEURONOVÉ SÍTĚ	30
3.1 ZÁKLADNÍ POPIS NEURONOVÉ SÍTĚ.....	30
3.1.1 Dopředné neuronové sítě.....	30
3.1.2 Metoda Levenberg-Marquardt	32
4 POUŽITÁ TECHNIKA	34
4.1 DIGITÁLNÍ ZRCADLOVKA NIKON D3000	34
4.2 MOBILNÍ TELEFON SAMSUNG GALAXY S2	35
5 WOLFRAM MATHEMATICA	38
5.1 ZÁKLADNÍ POJMY	39
5.1.1 Notebookový systém	39
5.1.2 Programovací jazyk.....	39
5.1.3 Nápověda – Wolfram mathematica 9 DOCUMENTATION CENTER	40
5.1.4 Práce s grafikou	40
5.2 IMAGE PROCESSING – ZPRACOVÁNÍ OBRAZU	40
5.2.1 Vybrané příkazy Image Proceingu	41

5.3	IMPORT A EXPORT	42
5.3.1	Import	42
5.3.2	Export	43
II	PRAKTICKÁ ČÁST	44
6	PŘÍPRAVNÉ PRÁCE A TVORBA DATABÁZE DOPRAVNÍCH ZNAČEK	45
6.1	DATABÁZE DOPRAVNÍCH ZNAČEK	45
6.2	ÚPRAVA FOTOGRAFIÍ A POUŽITÝ SOFTWARE	47
7	DETEKČNÍ ALGORITMY	48
7.1	HLEDÁNÍ KOMPROMISŮ MEZI VÝVOJOVÝMI PROSTŘEDÍMI	48
7.1.1	Barevný model	48
7.1.2	Morfologické operace	49
7.2	KOMPLEXNÍ VYHLEDÁVACÍ ALGORITMUS	49
7.2.1	Barevný model LAB	51
7.2.2	Barevné modely HSB a RGB	56
7.3	OSTATNÍ ALGORITMY	59
7.3.1	Algoritmus dle RGB barevného prostoru	61
7.3.2	Algoritmus dle LAB a HSB barevného prostoru	61
8	PŘÍPRAVA DAT PRO NEURONOVOU SÍŤ	63
8.1	PRVNÍ PŘÍPRAVA DAT	63
8.2	DRUHÁ PŘÍPRAVA DAT – DĚLENÍ DLE KATEGORIÍ	66
8.3	TŘETÍ PŘÍPRAVA DAT – OŘEZY PIKTOGRAMŮ 34X34 PIXELŮ	67
8.3.1	Příkazové značky	68
8.3.2	Zákazové značky	69
8.3.3	Výstražné značky	71
8.4	ČTVRTÁ PŘÍPRAVA DAT – FINÁLNÍ UČÍCÍ MNOŽINY	72
8.5	TESTOVACÍ A VALIDAČNÍ VEKTORY	75
9	POPIS VÝSLEDNÉHO PROGRAMU	77
	ZÁVĚR	80
	ZÁVĚR V ANGLIČTINĚ	82
	SEZNAM POUŽITÉ LITERATURY	84
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	87
	SEZNAM OBRÁZKŮ	89
	SEZNAM TABULEK	91
	SEZNAM PŘÍLOH	92

ÚVOD

Cílem této diplomové práce je účast na skupinovém vývoji softwaru pro rozpoznávání dopravních značek z videosekvencí a statických snímků. Moje část se zabývala převážně návrhem detekčního algoritmu pomocí metod zpracování obrazu. Dále zahrnovala tvorbu učicích, testovacích a validačních vektorů pro umělou neuronovou síť a jejich následný export. Všechna moje účast na projektu probíhala převážně v programovacím prostředí Wolfram Mathematica. Hlavní aplikace byla vyvíjena v jazyku C# spolupracovníkem Bc. Janem Dospivou [31] a slučovala moji diplomovou práci s prací kolegy Bc. Tomáše Vinklera [30], který se zabýval umělými neuronovými sítěmi. Společně jsme nejen konzultovali a diskutovali jednotlivé části, ale podíleli se i na tvorbě ukázkových scénářů.

Do oblasti metod zpracování obrazu se řadí velké množství úkonů a úprav obrázků či pořizování statických nebo dynamických snímků. Nejdůležitější částí v této oblasti je správný výběr barevného modelu, v mém případě LAB a znalost jeho předností ve vývojovém prostředí. Mezi základní metody se řadí segmentace, která obraz rozdělí dle společných vlastností a dokáže oddělit pozadí od hledaného objektu. Pomocí prahování či detekce hran v obraze se pak provádí další operace, které mají za následek nalezení kandidáta, v případě detekčního softwaru dopravní značku. Tento kandidát poté projde přes filtrační podmínky, tzv. příznaky, které mají za úkol ještě více nalezený objekt otestovat.

Tvoření učicích, testovacích a validačních vektorů obsahuje sestavení množin dopravních značek, rozdělených do kategorií dle domluvy se spolupracovníkem [30], které detekuje navržený algoritmus jak v softwaru Wolfram Mathematica, tak v hlavní výsledné aplikaci. Dále se ze sestavených snímků získají obrazová data, která se vkládají po úpravách do umělé neuronové sítě, jejímž výstupem jsou rovnice, tzv. vztahy z jednotlivých výstupních neuronů v závislosti na vstupních údajích. Tyto rovnice se aplikují do hlavního výsledného programu, který rozpozná příslušnou dopravní značku.

Diplomová práce je rozdělena na 2 části – teoretickou a praktickou. První, teoretická, se zabývá popisem dopravních značek, metodami zpracování obrazu a morfologickými operacemi a možnostmi těchto metod v softwaru Wolfram Mathematica. V druhé, praktické, se práce zabývá návrhem vhodného algoritmu pro rozpoznávání dopravních značek, přípravou učicích, testovacích a validačních vektorů a popisem hlavní

výsledné aplikace. V závěru je provedeno hodnocení výsledků. Výstupními soubory je několik notebooků softwaru Mathematica se sestavenými algoritmy a pak učící a testovací množiny pro umělou neuronovou síť.

I. TEORETICKÁ ČÁST

1 DOPRAVNÍ ZNAČKY

Hlavním úkolem každé dopravní značky je označování dopravní situace, která nastane. Pro použití je rozhodující jejich význam, který je pevně stanoven a definován zákonem č 361/2000 Sb. o provozu na pozemních komunikacích. Zásady pro dopravní značení jsou platné pro všechny druhy pozemních komunikací, to je pro silnice, dálnice, místní či účelové komunikace.[1]

1.1 Rozdělení

Nejčastějšími dopravními značkami nejen na českých komunikacích jsou značky svislé, které jsou zobrazeny na tabulích či panelech. Jiným druhem jsou značky vyobrazené na komunikacích, označované jako vodorovné.



Obr. 1: Ukázka dopravního značení v ČR[2]



1.1.1 Rozdělení dle významu

Dle významu se dělí dopravní značky na kategorie:

- Výstražné (skupina A) – značky, které upozorňují účastníka provozu na nebezpečí či na místa, kde musí dbát zvýšené opatrnosti.[1]

- Zákazové (skupina B) – udávají (spolu s dodatkovými tabulemi) zákazy či omezení účastníkovi provozu.[1]
- Příkazové (skupina C) – říkají účastníkovi provozu příkazy, dle kterých se má řídit na pozemní komunikaci. Doplnění ke značkám je možné jen na dodatkových tabulích.[2]
- Upravující přednost (skupina P) – značky stanovující přednosti v provozu.[3]
- Informativní (základní označení I) – účastníkovi provozu podávají informace, slouží k orientaci nebo ukládají určité povinnosti. Dodatkové informace se udávají buď přímo na značce, není-li ohrožena její čitelnost, nebo na dodatkové tabuli
 - Provozní (skupina IP)
 - Směrové (skupina IS)
 - Jiné (skupina IJ) [4]

Tabulka 1: Parametry kategorií značek

Skupina	A	B	C	P	I
Počet	40	40	32	12	133
Tvar	Trojúhelník	Kruh	Kruh	Různý	Různý
Barva	Červená	Červená	Modrá	Různá	Různá
Ukázka					

Dle tabulky (Tabulka 1) lze pozorovat, že nejpočetnější kategorií jsou informativní značky, protože jejich významnost není tak vysoká. Naopak nejvýznamnějšími jsou značky upravující přednost, výstražné či zákazové, které jsou dané jasnými barvami a tvary. Platí zde pravidlo, že nejvýznamnější značky jsou vždy červené.[5]

1.1.2 Rozdělení dle provedení

Dle provedení se dělí dopravní značky na skupiny:

- Stálé – umístění je stanoveno pevně vedle vozovky nebo nad vozovkou, na sloupku nebo konstrukci.[6]

- Přenosné – nejdůležitějším parametrem je nadřazenost nad svislými dopravními značkami s tím, že jsou umístěny buď přímo na vozovce (výjimečně) nebo vedle vozovky na místech stálých dopravních značek na červeno bílém reflexním sloupku.[7]
- Proměnné – jejich výhodou je okamžitá reakce na dopravní situaci s tím, že jsou umístěny na panelech vedle nebo nad vozovkou. Pokud je zobrazována příkazová značka, musí mít barevné spektrum stejné, jako značka svislá. Značky upravující přednost nesmí být provedeny jako proměnné. Zobrazení je spojitě či nespojitě:
 - Spojitě – zobrazení je totožné s příslušnou svislou značkou.
 - Nespojitě – značka je sestavena z jednotlivých svítících bodových elementů s tmavým pozadím.[8]

1.2 Technické předpoklady a umístění

Význam dopravních značek uvádí vyhláška č. 30/2001 Sb.

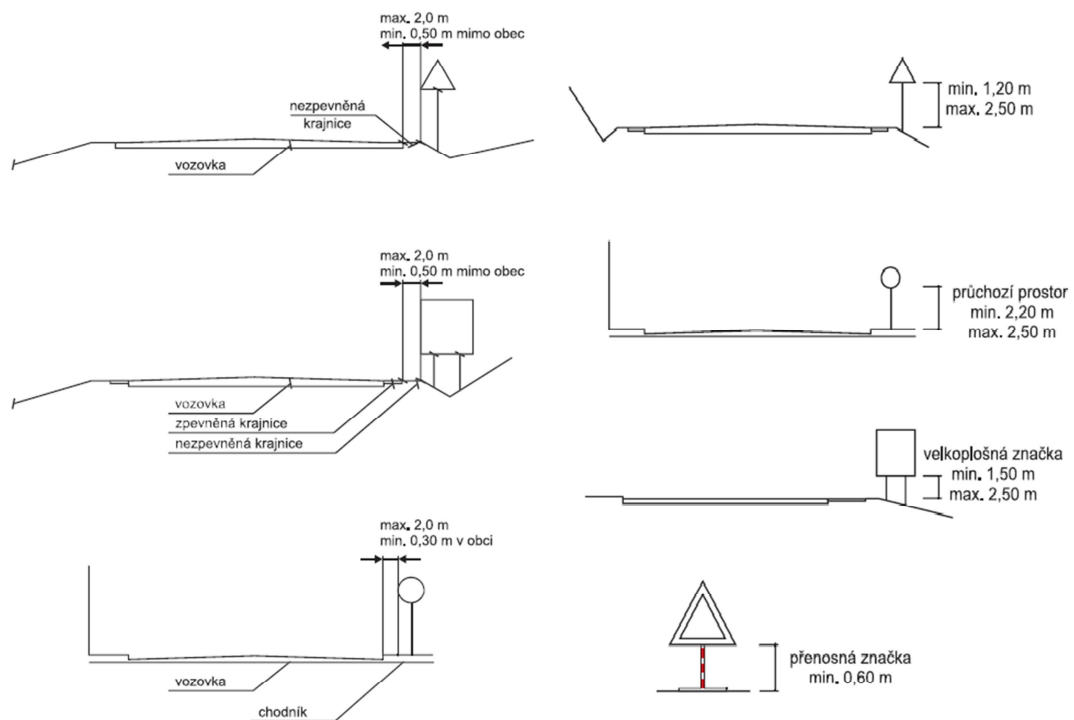
Tvary symbolů mají danou jasnou předlohu a nesmí se měnit s výjimkou vyobrazených číslic nebo znaků, které mohou být obráceny (směrové šipky).[9,3]

Barevnost značek v ČR je omezena na několik základních barev. Nejzákladnější barvou je bílá, ke které se nejčastěji přidává červené ohraničení s černým znakem pro významnost. Dle potřeby a významu se přidává barva modrá, občas zelená.

Velkou výhodou některých dopravních značek je jejich reflexní zvýraznění, které upravuje norma ČSN EN 12899-1. Takové značky upozorňují zejména na významné dopravní situace (křižovatky, změna organizace dopravy a jiné). Jejich význam je zvýrazněn při osvětlení vnějším světelným zdrojem. Reflexní zvýraznění může být i žlutou reflexní barvou, která kopíruje tvar dopravní značky.[10,3]

1.2.1 Umístění vzhledem k vozovce

Svislé dopravní značky jsou umístěny nad úroveň pozemní komunikace ve vzdálenosti dle obrázku (Obr. 2).



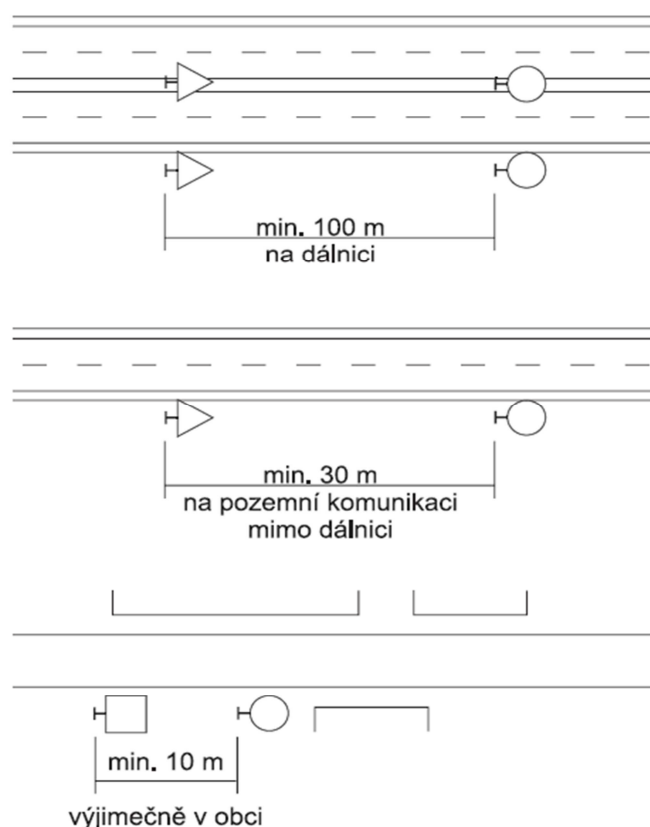
Obr. 2: Umístění svislých dopravních značek vzhledem k vozovce[3]

Značky se umísťují z pravidla kolmo ke směru jízdy na pravou stranu. Výjimkou jsou dálnice a silnice pro motorová vozidla, kdy se dopravní značka umístí na obě strany kvůli většímu počtu jízdních pruhů.

Reflexní značky se umísťují tak, aby neoslňovaly řidiče. Viditelnost reflexní vrstvy však musí být viditelná mimo obec na 100 metrů a v obci na 50 metrů.[3]

1.2.2 Umístění vzhledem k dopravním značkám

Vzdálenost dopravních značek od jiných dopravních značek se liší od třídy komunikace dle obrázku (Obr. 3). Na dálnicích a silnicích pro motorová vozidla se vzdálenosti mezi značkami liší v rámci stovek metrů. Mimo tyto komunikace pak minimálně ve vzdálenosti 30 metrů. Kvůli častým změnám dopravní situace ve městech je vzdálenost určena na minimálně 10 metrů (výjimečně) pro přehlednost.[1,3]



*Obr. 3: Umístění svislých dopravních značek
vzhledem k jiným dopravním značkám[3]*

1.3 Dopravní značky v zahraničí

V zahraničí se dopravní značky liší vždy jednotlivými vyhláškami země. Ve většině zemí však platí 2 systémy, a to evropský či americký případně jejich kombinace. Znamky v systémech jsou vesměs podobné či udávají podobná pravidla. Rozdíl může být například v barevnosti.

1.3.1 Evropský systém

Téměř ve všech státech Evropské Unie i ostatních státech je standardizován tvar i znaky na dopravních značkách (Tabulka 1). Oproti jiným zemím světa je tak na těchto značkách minimum psaného textu, naopak důležitost vyjadřují znaky. Barevnost se liší od vyhlášek jednotlivých zemí. Například v Polsku a Švédsku je místo bílého podkladu značky podklad žlutý. V Evropě je jedinou výjimkou Irská republika, kde platí systém americký.[4,5]

Tabulka 2: Rozdíly ve značkách v zemích EU[3]

Německo	Polsko	Španělsko	Francie	Itálie	Chorvatsko
					
					

Dalšími zeměmi mimo Evropu s tímto systémem značení jsou například Singapur nebo Indie.

1.3.2 Americký systém

Dopravní značky amerického systému jsou orientovány spíše více na nápisy různých významů než na znaky. Ze standardních znaků jsou využívány hlavně značka STOP či dej přednost v jízdě s doplňujícím nápisem. Základem ostatních značek je žlutý kosočtverec se znakem či písemné vyjádření v čtvercové tabulce. Mezi země, kde je tento systém využíván, patří například Austrálie či Brazílie.[5]

1.4 Systémy na rozpoznávání dopravních značek

Již existující systémy rozpoznávající dopravní značky (TSR) [8] využívají automobilky po celém světě. Používány jsou většinou v kombinaci s ostatními asistenty pro řidiče, jako například systémem pro rozpoznání jízdního pruhu nebo GPS. Úkolem je tak maximálně zabezpečit řidiče nejen před porušením dopravních předpisů, ale i upozornit ho na omezení, příkazy, křižovatky či zákazy, případně mu poskytnout informační servis. [7]

Všechny již vyvíjené systémy pracují pouze se svislými dopravními značkami podél komunikace. Spoustu z nich dokáží rozeznat i dodatkové tabulky a pomocí zvukového signálu na ně upozornit. Některé systémy jsou propojeny s tempomatem, takže po akceptování od řidiče dokáží automaticky snížit rychlost dle dopravního značení. Automobilky se ovšem především zabývají výzkumem systémů, které dávají řidiči informace jen o značkách omezujících rychlost, případně o značkách zakazujících předjíždění.[7]

Základem každého asistenčního systému je vestavěná kamera (Obr. 4), která je nejčastěji umístěná ve zpětném vnitřním zrcátku. Ta je využita pro asistenty, kteří kontrolují provoz a dění před vozem. Poté se signál z kamery rozebírá, zpracovává a je vyhodnocován nejčastěji v řídicí jednotce, kdy je snímaná značka porovnána s předlohou. Naposledy následuje zobrazení předlohy buď na HUD displeji (displej, který promítá informace přímo do zorného pole řidiče) nebo do řídicí části přístrojové desky (tachometr). Některé systémy využívají zabudované GPS navigace, která má údaje o rychlostech na pozemních komunikacích vložené na určených GPS souřadnicích. Řidič je tak na dané souřadnici upozorněn na změnu rychlosti (zejména v obcích).



Obr. 4: Kamery asistenčních systémů – vlevo Škoda, vpravo Opel[10]

U světových automobilek, které využívají asistenční systémy se zabudovaným TSR, lze od roku 2008 sledovat značný pokrok. Zejména v evropských zemích se tento asistent stal velmi oblíbeným. Kromě koncernu Volkswagen (automobilky Škoda [10], Audi a Volkswagen) využívají TSR i automobily značek Ford, Opel, BMW a švédský Saab či Volvo.[8] Z nezávislých výrobců vyvíjí tyto systémy zejména společnosti Hella Aglaia TSR [9] nebo společnost Siemens.

2 ZPRACOVÁNÍ OBRAZU

Vědeckotechnická disciplína se zabývá zpracováním obrazu, především jejich úpravou, tvorbou a získáváním informací z obrazových dat. Základem je mít snímky reálného světa v digitální podobě. Takový obraz se dá získat například pomocí digitálního fotoaparátu, digitální videokamery či dalším snímáním obrazu, jako například skenerem.

2.1 Barevné modely

Většina barevných modelů, které se využívají v moderní počítačové technice pro vyjádření barev, vznikly na principu míchání známých barev z reálného života či malířství. Jiné zase vycházejí z vnímání barev pomocí lidského oka. Barevný model umožní vytvořit matematicky určité množství barev. Toto množství je však většinou menší, než množství viditelných barev, které lze rozlišit lidským okem.[11]

Naopak barevný prostor představuje rozsah barev v rámci viditelného spektra. Je tak považován za variantu nějakého barevného modelu.

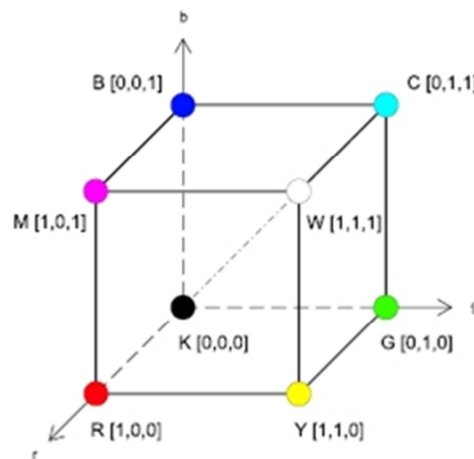
Mezi nejznámější a nejpoužívanější barevné modely patří aditivní RGB, subtraktivní CMY nebo CMYK, HSV (HSB), HLS či CIE 1976 L*a*b*.[11]

2.1.1 RGB

Základním a nejznámějším představitelem barevných modelů je bezesporu RGB. Vzniká pomocí aditivního míchání barev, tedy jednotlivé složky barev se sčítají a vytváří světlo větší intenzity. Složkami jsou 3 barevná světla, a to R (Red – červená), G (Green – zelená) a B (Blue – modrá), které se zobrazují v určitých intenzitách. Pokud jsou umístěny dostatečně blízko, tak lidské oko bude vnímat jen jejich výslednou barvu. Barevný prostor RGB tak vnímá barvy takovým způsobem, jakým barvy vnímá i lidské oko.[11,12]

Dle aditivního principu míchání barev vznikají další barevné odstíny dle obrázku (Obr 5):

- **Bílá** – maximální intenzita všech základních barev v modelu
- **Černá** – minimální (nulová) intenzita všech základních barev v modelu
- **Stupně šedi** – skládání základních barev se stejnou barvou, u které se postupně snižuje její intenzita
- **Ostatní barvy** – skládání základních barev s různou intenzitou



Obr. 5: Reprezentace RGB barevného modelu[12]

Intenzita se uvádí v celočíselném rozsahu 0 až 255, kde hodnota 0 (0%) je nulové zastoupení barvy a hodnota 255 (100%) je maximální zastoupení barvy. Celkový počet barevných odstínů je 256^3 tedy 16 777 216.[11,12]

Odvozené prostory od RGB:

1. **sRGB** – Definován firmami Microsoft a HP jako standard pro periférie počítačů. Používá se však i jako paleta barev v HTML jazyce.[11]
2. **Adobe RGB** – Využívá mírně odlišné barvy a jeho rozsah je o něco větší než u sRGB. Využití je především u digitálních fotoaparátů. Většina monitorů však nedokáže zobrazit všechny barvy, které jsou v tomto modelu reprezentovány.[11]
3. **RGBa** – nejedná se o odvozený barevný prostor, ale o doplnění (na osmi bitech) k 3 základním barvám. Původní model je rozšířen o další informaci, která se týká průhlednosti, tedy tzv. alfa kanál.[11]

2.1.2 CMY(K)

Barevný model CMY respektive CMYK je reprezentantem subtraktivního míchání barev, to znamená, že s každou přidanou barvou se ubírá část světla. Základem jsou 3 barevné pigmenty a to C (Cyan – tyrkysový), M (Magenta – purpurový) a Y (Yellow – žlutý). Často je přidáván ještě pigment čtvrtý a to K (black – černý).[11]

Protože je model využíván především v tiskařské oblasti, její princip je tak založený na barvě čistého listu papíru. Ten pohlcuje určitou část viditelného spektra barev. Bílá

barva se tak odrazí, černá naopak pohltí. Model vychází tedy z principu, kdy se při tisku nanáší inkousty, jejichž úkolem je snížit schopnost papíru odrazit dopadající paprsky. Barvy jsou tak vytvářeny odečtem základních barev od bílé. [11,12]

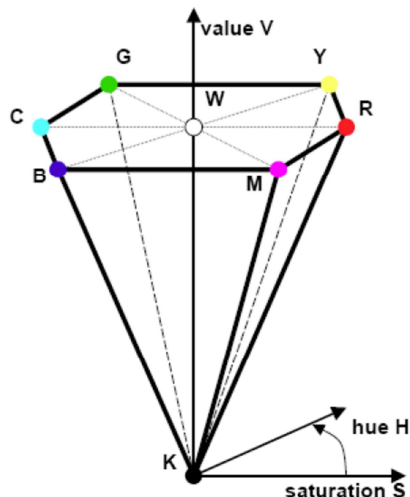
2.1.3 HSV (HSB)

Oproti RGB či CMY je tento barevný model zaměřený více na uživatele, než na zobrazovací zařízení. Základem jsou tak 3 veličiny - odstín, sytost a jas. Nové barvy vznikají tak, jak je přirozené pro člověka, tedy přidáním bílé nebo černé barvy. Tímto vznikají další nádechy a odstíny.[11]

- **H (Hue) – barevný tón** – základní barva, jejíž hodnota se vybírá v hodnotách 0° až 360° , dle kruhu barev v duze. Stupně jsou pak v PC přepočítány na hodnoty 0 až 255.[11]
- **S (Saturation) – sytost** – poměr čisté barvy a bílé barvy v rozsahu 0 až 1.[11]
- **V (Value), B (Brightness) – jasová hodnota** – poměr čisté barvy a černé barvy v rozsahu 0 až 1.[11]

HSV lze v prostoru popsat jako šestiboký jehlan postavený na svém vrcholu (Obr. 6).

Na osách mu leží souřadnice hodnot H, S a V a svůj střed má v nule.[11]



Obr. 6: Repräsentace HSV barevného modelu [12]

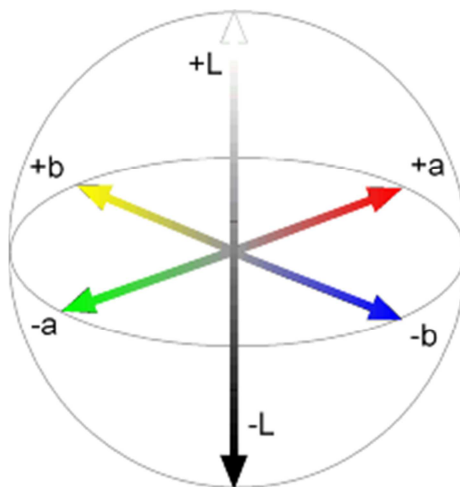
2.1.4 Lab

CIE 1976 L^*a^*b , což je oficiální název tohoto barevného modelu, byl původně vědecký pokus a je navržen tak, aby byl zcela nezávislý na zobrazovacím zařízení.

„Barevný prostor Lab zachycuje všechny barvy z viditelné části spektra. [11, str. 247]“

Specifickou vlastností tohoto modelu je oddělení jasové složky od barevné. Barvy lze tak upravovat beze změny světlosti. Je složen ze tří kanálů:

- **L (Lightness) – jasový kanál** – světlost bodu v hodnotách 0 až 100. Hodnota minimální, tedy 0 představuje černou barvou a naopak maximální hodnota 100 je reprezentována barvou bílou.[11]
- **A – barevný kanál a** – plynulý přechod mezi červenou (a) a zelenou (-a) barvou. Nabývá hodnot -128 až 127.[11]
- **B – barevný kanál b** – plynulý přechod mezi žlutou (b) a modrou barvou (-b). Hodnoty jsou stejné jako u kanálu a.[11]



Obr. 7: Reprezentace Lab barevného modelu[12]

Kladné hodnoty představují teplé barvy, studené barvy jsou reprezentovány hodnotami zápornými. Barva je nejsytější, pokud se její číslo pohybuje co nejvýše v absolutních hodnotách. Nulové hodnoty vyjadřují stupně šedi od bílé po černou.

Využití tohoto modelu je především v počítačové grafice a metodách zpracování obrazu. Často se využívá jako přechod mezi jinými barevnými modely v grafických

editorech. Pomocí převodních algoritmů se barvy přepočítávají a do souborů pak ukládají v barevném modelu RGB.[11]

2.2 Zpracování binárního obrazu a jeho extrakce

Pojem binární obraz lze chápat jako obraz, který se skládá ze dvou hodnot - jedniček a nul. Všechny pixely tak nabývají pouze dvou čísel, z čehož lze usoudit, že obraz bude černobílý. V takovýchto obrazech se zpravidla nejlépe hledají jednotlivé segmenty a části, protože hodnoty pixelů nesou jasnou informaci.

Digitální zpracování obrazu má za svůj hlavní cíl především nalezení částí v obrazu, které nás zajímají, tzv. extrakce obrazu. Existuje několik metod, pomocí kterých lze z jakéhokoliv obrazu rozpoznat požadovaný objekt, ať je jakékoliv barvy nebo tvaru. Téměř vždy se však převádí do binárního stavu.

2.2.1 Segmentace

Segmentace je hlavním procesem, kterého lze využít pro nalezení objektu zájmu v obraze. Objekty jsou separovány od nezajímavého pozadí především pomocí hledání hran [13] jednotlivých objektů případně pomocí detekce hran celých oblastí, kterými jsou objekty reprezentovány. Výsledkem jsou pak soubory, které obsahují nepřekrývající se oblasti. Segmentace se dělí na:

- Kompletní segmentace - objekty vstupního obrazu se jednoznačně shodují s nepřekrývajícími se oblastmi.[13]
- Částečná segmentace – segmenty přímo nemusí souhlasit s objekty obrazu. Obraz je rozdělen do samostatných částí, které jsou k určitým vlastnostem, jako je barva, jas, odrazivost nebo textura homogenní po zpracování ve zvolených rysech.[13]

Hlavním cílem tohoto procesu je získání částečné segmentace, kterou je možné zpřesnit při použití operací vyšších úrovní. Přínosem je pak redukce objemu dat výrazným způsobem. Je třeba brát zřetel na nejednoznačnost dat, které mohou být doprovázené informačním šumem.

Metody segmentace lze rozdělit do tří skupin dle dominantních vlastností:

1. Metody využívají znalosti celého obrazu nebo jeho části. Hlavním reprezentantem je pak histogram vlastností.

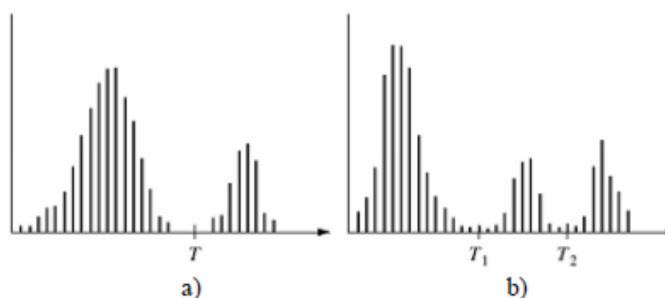
2. Postupy, které vycházejí z určování hranic mezi oblastmi obrazu nebo částmi obrazu.
3. Postupy, které přímo vytvářejí oblasti nebo jejich zájmy.[13,14]

Každá z těchto metod má odlišné výsledky, které jsou ovlivněny charakteristikami, jako jsou například jas nebo textura. Výčet těchto charakteristik se využívá při hledání hranic nebo při tvorbě jednotlivých oblastí. Výsledky lze tak kombinovat a vytvářet nové jednotlivé popisné struktury.

2.2.2 Prahování a histogramy

Princip nejstarší a nejjednodušší metody zpracování obrazu, prahování, je založený na tom, že objekty a pozadí mají jinou úroveň intenzity jasu. Pokud se tedy vyseparují pixely s určitými intenzitami jasu, lze oddělit pozadí od objektů zájmu. Nejlépe se prahovací oblasti sledují pomocí histogramů.[14]

Z jasových histogramů, tedy speciálních grafů, které zobrazují rozložení jasů v obraze, se dle rozložení hodnot můžeme seznámit s místy tmavých a světlých pixelů. Nejideálnější histogram má rovnoměrné rozložení kopce. Takový histogram však v reálném světě není možné nikdy získat. Informace o barvách dává histogram jednotlivých barev RGB kanálu (jednající se o RGB barevný model).[15]



Obr. 8: Ukázka jasového histogramu[14]

Jednoduché prahování, jehož histogram lze vidět na obrázku (Obr. 8 část a) je přiřazen k obrázku, který se skládá ze světlých bodů tvořících objekt a tmavých bodů, tvořících pozadí. Do dvou dominantních modulů se tak v takovémto případě seskupí pixely objektu a pozadí. Pomocí prahovací úrovně T lze objekty vyseparovat z pozadí. Bod, o souřadnicích (x,y) , pro který platí, že $f(x,y) > T$ je pak bodem objektu. Pokud je to naopak, bod je reprezentantem pozadí.[13,14]

Na obrázku (Obr. 8 část b)) je zobrazen histogram pro víceúrovňové prahování. Lze tak z něj vyčíst, že v obraze jsou 3 dominantní moduly (například 2 světlé objekty a 1 tmavé pozadí). Bod o souřadnicích (x,y) pak patří jednomu objektu, pokud platí, že $T1 < f(x,y) \leq T2$, druhému objektu, pokud $f(x,y) > T2$ a naposledy pozadí, pokud platí, že $f(x,y) \leq T1$. [13,14]

Obecně lze však prahovaný obrázek, označovaný jako $g(x,y)$ popsat

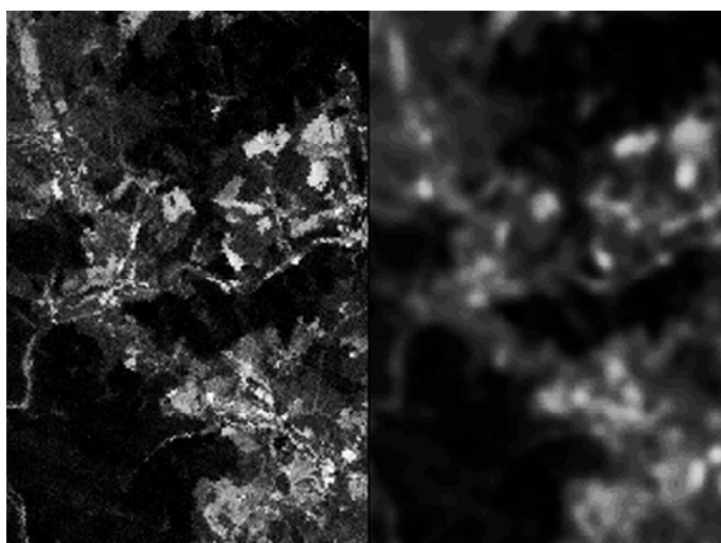
$$g(x,y) = \begin{cases} 1 & f(x,y) > T \\ 0 & f(x,y) \leq T \end{cases} \quad (1)$$

kde číslo 1 odpovídá pixelům objektu a číslo 0 pixelům pozadí. [13,14]

2.2.3 Gaussův filtr

Každý pořízený snímek obsahuje nějaký druh šumu, který tvoří nežádoucí prvky v obraze. Takový šum je zapotřebí odstranit pomocí vhodného filtru, například Gaussovým. Konvoluční maska filtru se vytváří tak, že zvyšují váhy bodu ve středu masky, případně jeho čtyř-okolí. Za čtyř-bod se považují takové body, které mají s bodem ve středu jednu souřadnici společnou a druhou lišící se o jednotku. Velikost masky tedy nemůže být nekonečná.

Využití Gaussova filtru je převážně k vyhlazování obrazu, kdy dochází k jeho rozmazání, respektive rozostření (Obr. 9). [18]



Obr. 9: Ukázka použití Gaussova filtru [18]

2.2.4 Hledání hran

Pojem hrana se v počítačové technice a metodách zpracování obrazu vysvětluje jako náhlá jasová změna pixelů v obraze. Jako příklad pro reprezentaci hran lze považovat perokresbu, kdy linie odpovídají právě náhlým jasovým změnám. Hrana je dána vlastnostmi obrazového elementu a jeho okolím. Nástrojem pro hledání hran a studium změn jasových složek jsou parciální derivace. Gradient, vektorová veličina ∇ (určuje směr největšího růstu funkce) a strmost, která udává změnu obrazové funkce $f(x,y)$. Pixely, které mají velký modul gradientu se nazývají hrany.[13,14,16,17]

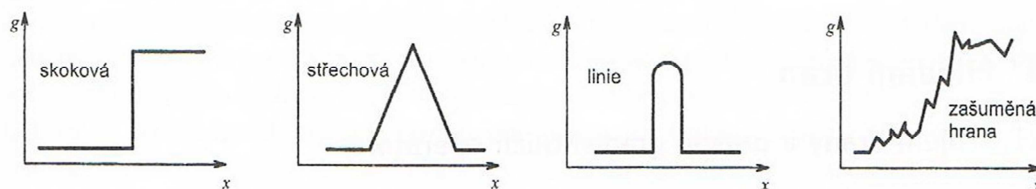
Velikost gradientu $|\nabla f(x,y)|$ a směr gradientu ψ jsou dány jako vztahy:

$$|\nabla f(x,y)| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}, \quad (2)$$

$$\psi = \arg\left(\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}\right), \quad (3)$$

kde $\arg(x,y)$ je úhel v radiánech mezi osou x a radiusvektorem k bodu (x,y).[13]

Směr hrany se většinou definuje jako kolmý na směr gradientu ψ z důvodu spojování hranových pixelů. Pokud jsou takto orientovány, tak se mohou hrany přirozeně spojit do hranic. Třídění je pak dle jednorozměrného jasového profilu ve směru gradientu v pixelu.



Obr. 10: Jasové profily nejběžnějších hran[13]

První 3 jasové profily na obrázku (Obr. 10) - skoková a střechová hrana a tenká linie jsou idealizované. Zašuměnou hranu lze najít v jakémkoliv reálném obrázku.[13]

Velkou souvislost s hranami a gradientními operátory má ostření obrazu. To má za úkol upravit obraz do takové podoby, aby v něm byly pozorovatelné strmější hrany. Ostření lze také popsat ve frekvenčním spektru, a to zdůrazněním vysokých frekvencí. Je dokázáno, že čím vyšší frekvence, tím vyšší je amplituda její derivace. Využití ostření je zejména pro zvýraznění kontrastu při zobrazení jak na obrazovce, tak při tisku.

2.2.5 Cannyho hranový detektor

Klíčovým problémem při hledání hran je správná volba měřítka, to znamená okolí použitelné pro výpočet. Správná velikost závisí na velikosti objektů v obraze. Nutnost k volbě měřítka je, aby byl obraz interpretovatelný. Úkolem Cannyho detektoru je hledat nejlepší možné rozlišení.[16] Základní myšlenka vychází z toho, že filtrem se hledá skoková změna v obrázku. Za splnění jistých požadavků je popsán tento filtr jako úloha variačního počtu.

Pro skokové hrany je pak Cannyho detektor optimální vzhledem ke třem kritériím:

- **Detekční kritérium** – významné hrany nesmí být přehlédnuty a na jednu hranu nesmí být vícenásobné odezvy
- **Lokalizační kritérium** – rozdíl mezi skutečnou a nalezenou polohou musí být minimální
- **Požadavek jedné odezvy** – detektor nesmí reagovat na jednu hranu v obraze vícenásobně. Zde je velká svázanost s prvním kritériem, které tuto podmínku z části splňuje s tím rozdílem, že toto kritérium je zaměřeno na zašuměné a nehladké hrany.[13]

2.2.6 Morfologické operace dilatace a eroze

Jak dilatace, tak eroze patří do kategorie binární matematické morfologie. Již z názvu kategorie lze poznat, že se tato metoda využívá zejména u binárních, neboli černobílých obrázků.

Dilatace, značící se znakem \oplus [13,14], pomocí vektorového součtu pixelů skládá body z dvou množin do jedné. Tento jev, zobrazený na obrázku Obrázek 10, se projevuje zejména na okolí dilatovaného objektu. Jeho pozadí či náhlé změny barvy či jasu splynou s objektem. Objekty se rozrostou o jeden sloupec pixelů, respektive zvětší se. Přesná definice zní

$$X \oplus B = \{p \in \varepsilon^2, p = x + b, x \in X, b \in B\} \quad (4)$$

„Dilatace $X \oplus B$ je bodovou množinou všech možných vektorových součtů pro dvojice pixelů. Vždy pro jeden z množiny X a jeden z množiny B .“[13, str. 213]

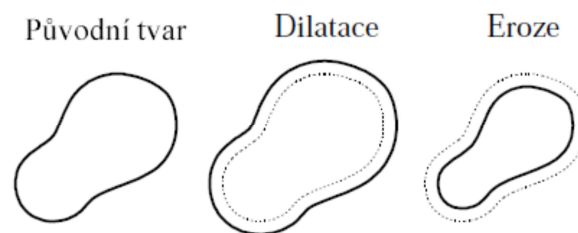
Obecné využití dilatace je především pro zaplnění malých děr, úzkých zálivů a pro složitější operace. Pokud se však takto využívá, je třeba často zachovat původní rozměr, proto se kombinuje s erozí.

Na rozdíl od dilatace nereprezentuje eroze žádnou vektorovou operaci a je k dilataci duální. Značí se \ominus [13,14] a je vyjádřena pomocí předpisu

$$X \ominus B = \{p \in \varepsilon^2, p + b \in X, \forall b \in B\} \quad (5)$$

„Pro každý bod obrazu p se ověřuje, zda pro všechna možná $p+b$ leží výsledek v X . Pokud ano, zapíše se v reprezentativním bodě do výsledného obrázku 1 a v opačném případě 0.“[13, str. 215]

Eroze se používá pro zjednodušení obrázků. Složitější objekty se rozdělí na jednodušší a objekty tloušťky 1 se ztratí. Dojde tak ke zmenšení, které je patrné na obrázku Obrázek 11. Tímto způsobem se dají lehce získat obrysy objektů, které jsou zobrazeny na obrázku. Není už těžkým úkolem odečíst erodovaný obrázek od původního.[13,14]



Obr. 11: Reprezentace a ukázka dilatace a eroze[14]

2.2.7 Měření objektů – kruhovost, pravoúhlost a podlouhlost

Provedení morfologických operací, popsanych v kapitole 2.2.5 ještě nezajišťuje, že všechny objekty jsou vhodnými kandidáty na dopravní značku. Pro konečné rozpoznání se provádí tzv. měření objektů, při kterém se počítají vlastnosti jako je kruhovost, pravoúhlost nebo podlouhlost. Pro tyto výpočty je zapotřebí získat z jednotlivých objektů v obraze číselné hodnoty, neboli příznaky, které vytváří vektor příznaků. Takový vektor pak obsahuje veškeré potřebné a důležité informace. Výpočty se tedy neprovádí pro celý binární obraz, ale pro jednotlivé objekty.[29]

Důvodem k měření objektů je především tvorba síta, složeného z podmínek, které musí splňovat testovaný objekt. Pokud jsou podmínky, složené z vlastností uvedených výše, splněny, kandidát na dopravní značku byl nalezen nebo naopak.

Nezákladnější vlastností je kruhovost. Ta se počítá jako poměr délky hranice objektu P a jeho plochy A dle vztahu (6)

$$C = \frac{P^2}{A}. \quad (6)$$

Výsledek kruhovosti se v různých objektech liší. Například pro čistý kruh vychází 4π , což je však číslo určené z teoretické délky hranice. Pro čtverec je výsledná hodnota většinou 16 a objekt má nepravidelný tvar. Výsledek je pak vyšší. Výsledek kruhovosti může být ovlivněn výpočetním postupem, kdy se může lišit zjištěná délka hranice od té teoretické. Tyto 2 hodnoty mohou být k sobě přiblíženy průměrováním, vyhlazením či prokládáním hranice objektu.[29]

Kruhovitost se nejčastěji pro tvorbu filtračního síta dále kombinuje s vlastnostmi pravoúhlost a podlouhlost. Principem těchto dvou vlastností je postupná rotace hranice vyšetřovaného objektu v rozsahu od $0-90^\circ$ po krocích o hodnotě 5° . Jakmile je rotace dokončena, kolem hranice objektu se opíše rovnoběžník, jehož vnitřní úhly jsou pravé, tzv. pravoúhelník. U tohoto pravoúhelníku musí platit, že jeho strany jsou rovnoběžné se stranami obrazu. Z těchto útvarů se vybírá ten, který má nejmenší plochu.[29]

Pravoúhlost R a podlouhlost S se získá dle vztahů

$$R = \frac{A_0}{A_R}, \quad S = \frac{a}{b}, \quad (7)$$

kde jako proměnná A_0 je označována plocha objektu, A_R plocha nejmenšího pravoúhelníku se stranami a a b . [29]

3 NEURONOVÉ SÍTĚ

Využití neuronových sítí v oblasti rozpoznávání dopravních značek je nesdílňnou součástí každého systému, který nevyužívá GPS s pevně zabudovanými pozicemi značek. Při použití neuronové sítě dochází k porovnávání sejmutého snímku, respektive ohraničené značky, s předlohou dle norem. Tato předloha je pak řidiči patřičným způsobem prezentována.

Historie sítí spadá do období první poloviny 20. století, kdy americký vědec W. S. McCulloch publikoval první práci na téma neuronů. Dnešní popis neuronů, jak je známý, publikoval ve 40. letech student W. Pitts a v roce 1958 na základě tohoto výzkumu vytvořil F. Rosenblatt první funkční perceptronovou síť. V 80. letech pak vznikaly další neuronové sítě, jako Hopfieldova síť, Kohonenova síť a ART síť.[19,20]

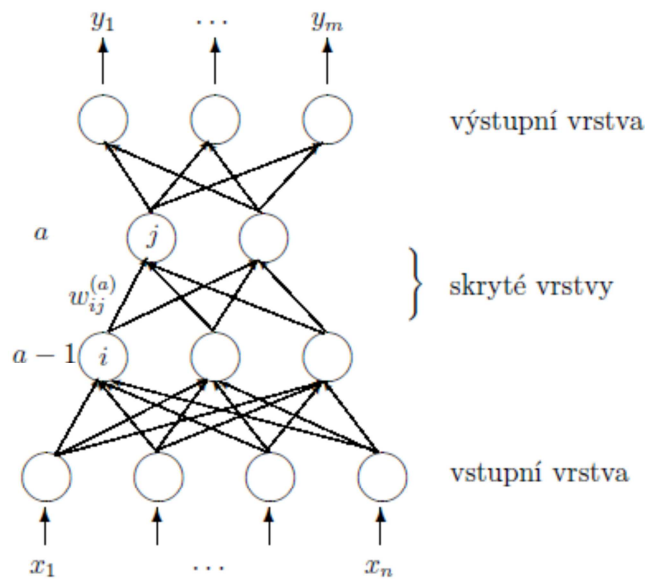
3.1 Základní popis neuronové sítě

Nejlépe se problematika neuronové sítě vysvětluje na rozdílu klasického počítače vs. neuronová síť. Při použití klasického počítače je nutnost psaní programu pro řešení daného problému. Má-li programátor na začátku tvorby dostatek informací, je schopný pomocí cyklů a podmínek zahrnout do programu všechny okolnosti, které mohou nastat. Co však může nastat, je situace, kdy je program postaven před problematiku, která je známá, ale zároveň dost odlišná. Takové problémy jsou ignorovány nebo je programátor či obsluha programu upozorněna, že se vyskytl takovýto případ. V takovém případě je nutný opět zásah programátora, který program upraví.[19]

V případě neuronové sítě není potřeba programátora, který by programoval či nějak program upravoval. Neuronová síť je v případě správného naučení schopna reagovat i na novou informaci a s velkou pravděpodobností zařadí informaci na správné místo či do správné třídy. Pokud je nových informací mnoho, neuronovou síť je třeba doučit, případně změnit její konfiguraci (počet neuronů, počet iterací a jiné).[19]

3.1.1 Dopředné neuronové sítě

Dopřednou neuronovou síť lze brát jako orientovaný graf, v němž každý uzel je neuron a orientovaná hrana je spoj mezi nimi. Každá hrana má svou váhu w , tzv. vážené spoje. Vstupem a výstupem do neuronové sítě, respektive výstupem z ní, je vždy neprázdná množina.[19]



Obr. 12: Dopředná neuronová síť[19]

Orientovaný graf na obrázku (Obr. 12) představuje dopřednou neuronovou síť, skládající se ze čtyř vzájemně disjunktčních podmnožin tzv. vrstev, indexovaných jako písmeno a . Počet těchto vrstev se označuje L . Počet neuronů ve vrstvách může být pokaždé jiný. Hrany vedou z vrstvy $a-1$ do a . [19]

První vrstva, kdy $a=1$ je vždy vstupní a obsahuje vstupní neurony se stejnou přenosovou funkcí. Prahová hodnota je nulová. Úkolem neuronů v této vrstvě je pak propagace vstupu pro celou neuronovou síť (x_1 až x_n). Naopak výstupní vrstva, tedy poslední, kdy $a=L$, nemá žádné další následníky a z výstupů se čte výstup celé sítě (y_1 až y_m). Ostatní vrstvy, splňující podmínku $1 < a < L$ jsou skryté vrstvy se skrytými neurony. Oproti vstupům mají vždy výstupní a skryté vrstvy jako přechodovou funkci logickou sigmoidu. [19]

Počet neuronů ve skrytých vrstvách se určuje exponenciálně, naopak počet všech vah a prahových hodnot se určí tzv. stupněm volnosti, pro který platí, že by neměl být větší, než celkový počet vstupních hodnot N . Tento počet se pak vypočítá jako velikost trénovací množiny $|X| * n$, kde n je dimenze vstupních vektorů. Trénovací množiny X je třeba k učení neuronové sítě a skládá se z uspořádaných dvojic $\langle x, d \rangle$, kde x je vzor a d požadovaný výstup sítě. Další určující skutečností je stanovení chyby, s kterou se liší skutečný a požadovaný výstup. Celková chyba se stanoví dle kritériální funkce

$$E = \frac{1}{2} \sum_k \sum_j (y_{j,k} - d_{j,k})^2 \quad (8)$$

kde:

k uspořádané dvojice v trénovací množině,

$y_{j,k}$ skutečný výstup j -tého neuronu ve výstupní vrstvě po předložení k -tého trénovacího vzorku x_k ,

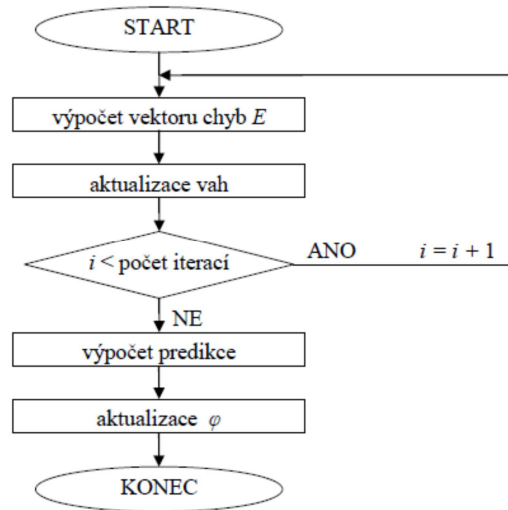
$d_{j,k}$ požadovaný výstup j -tého neuronu ve výstupní vrstvě po předložení k -tého trénovacího vzorku x_k .

Kriteriální chyba se musí co nejvíce minimalizovat, což je hlavním cílem učení. Toho se dosahuje pomocí algoritmu zpětného šíření, pomocí kterého se dá po každém předložení vzorku x určit, jak se mají modifikovat váhy a prahové hodnoty neuronů. Algoritmus pak pracuje tak, že po zjištění kriteriální chyby na výstupu každé vrstvy propaguje tuto chybu od výstupní vrstvy směrem k vstupní a postupně upravuje váhy.[20]

Trénování probíhá v epochách. V každé z epoch jsou neuronové síti předloženy trénovací množiny a celý proces se pak opakuje v dalších epochách. Po projití všech epoch platí poměr, že čím menší je kriteriální chyba, tím lépe je síť naučená, pokud nedojde k přeučení sítě. V takovém případě se provádí test naučenosti sítě pomocí ověřovací množiny, která se skládá ze vzorů, které se při učení nepoužily.[19,20]

3.1.2 Metoda Levenberg-Marquardt

U této iterační metody je princip založený na hledání globálního minima chyby výstupů a výstupů z trénovací množiny. Jedná se tak o učící algoritmus při použití dopředné neuronové sítě.[21]



Obr. 13: Algoritmus učení metodou

Levenberg-Marquardt[21]

Algoritmus učení, zobrazený na obrázku (Obr. 13) funguje tak, že se vypočítá vektor chyb E a aktualizují si váhy. Poté dochází k rozhodování, kdy nesmí být splněna podmínka, že krok výpočtu i musí být menší než počet iterací. Pokud je podmínka splněna, i se zvýší o číslo 1 a pokračuje se k prvnímu kroku. Je-li podmínka vyhodnocena záporně, dojde k výpočtu predikce (předpokladu) a aktualizuje se vektor známých funkcí.[21]

4 POUŽITÁ TECHNIKA

Pro pořízení snímků, respektive fotografií značek, bylo využito dvou zařízení. Prvním byla digitální zrcadlovka Nikon D3000 s objektivem 18-105mm. Druhým zařízením byl mobilní telefon Samsung Galaxy S2, u kterého byla využita především jeho optika pro pořízení videozáznamů a fotografií a také GPS modul pro sběr GPS dat.

4.1 Digitální zrcadlovka Nikon D3000

Digitální zrcadlovka, nebo-li DSLR, Nikon D3000 (Obr. 14) je fotoaparát formátu DX. Velikost jeho obrazového snímače je 24 x 16 mm. CCD snímač má rozlišení 10,2MPix o poměru stran 3:2.[22] Vychází ze svého předchůdce Nikonu D40 a další generací je Nikon D3100.[23]



Obr. 14: Nikon D3000 s objektivem 18-105mm[23]

Svojí plastovou konstrukcí je poměrně malý lehký a levný. Jelikož se řadí mezi poloprofesionální zrcadlovky, tak na první pohled nepotěší absence stavového LCD displeje. O informacích nastavení se stará hlavní 3“ display, který díky zabudovanému senzoru má schopnost obracet vypsané informace dle polohy fotoaparátu.[23]

Ovládání fotoaparátu probíhá pomocí několika tlačítek, kterých není velké množství a řada funkcí je tak nastavována pomocí přehledného anglického menu.

Fotografovaná scéna je zobrazována pouze v hledáčku, který zabere zhruba 95 fotografované scény. Nikon je v této problematice napřed, protože využívá systém matice jako průhledového displeje s možností zobrazení pomocné mřížky bez nutnosti výměny matice aktivací v menu. [23] V hledáčku se nachází i základní informace například o citlivosti ISO, o stavu baterie či stavu místa na paměťové kartě SD.[23]

Hlavní součástí je snímací čip, tedy senzor typu CCD formátu APS-C, který je schopný zpracovávat fotografie s obrazovým rozlišením 3872x2592 bodů. Snímky lze ukládat na paměťovou SD nebo SDHC kartu ve formátech JPEG či RAW (U Nikonu je tento formát označován jako NEF). Barevné prostory jsou RGB a sRGB. Citlivost se pak pohybuje v rozmezí ISO 100 až ISO 3200.[22,23]

Nikon D3000 nabízí spoustu režimů pro fotografování. Speciální funkcí je GUIDE MENU, které naučí zacházet s fotoaparátem jakéhokoliv amatérského fotografa. K manuálním režimům, kdy si uživatel nastavuje hodnotu clony nebo čas závěrky sám (režimy M, A, S a P) [22] lze využívat i automatické režimy pro různé scény, jako noční fotografie, krajina a jiné. Plně automatický režim je označován jako AUTO.

Pro výměnu objektivů slouží bajonet Nikon F s AF kontakty. Nevýhodou D3000 je absence ostřicího motoru přímo v těle fotoaparátu. Pokud tedy chce uživatel využívat automatického ostření, musí na to mít objektiv s označením VR. K fotoaparátu je nejčastěji dodáván objektiv AF-S 18-55mm f/3.5-5.6 G VR v základním setu.[22] Pro dosažení nejlepšího poměru kvalita/cena je však nejvhodnějším objektivem AF-S NIKKOR 18-105mm f/3.5-5.6 G VR, který byl využíván i pro moji diplomovou práci.

Aktuální průměrná cena se nyní pohybuje kolem 11 500 Kč. Minimální je zhruba 8900 Kč.[23]

4.2 Mobilní telefon Samsung Galaxy S2

Galaxy S2, který na trh přišel jako vlajková loď společnosti Samsung v roce 2011 a který vychází z modelu S, patří dodnes mezi mobilní telefony s dobrou hardwarovou i softwarovou výbavou. Jeho rozměry jsou 125x66x8,5 mm a hmotnost 116 gramů.[24]



Obr. 15: Mobilní telefon Samsung Galaxy S2[24]

Mezi jeho základní funkce patří vedle telefonování a zaslání SMS či MMS velice dobře propracovaný systém Android 4.1.2 Jelly Bean, který nabízí téměř vše, na co si uživatel vzpomene. Od aplikací na fotografování, natáčení videozáznamů, po kvalitní zábavu v podobě her či kancelářské aplikace.[24]

Procesor tohoto mobilního telefonu pracuje na frekvenci 1,2 Ghz a jedná se o dvoujádrový procesor Cortex A9. O zobrazování informací se stará kapacitní Super AMOLED Plus displej s velikostí 4,27“ a rozlišením 480x800 pixelů. Propojení, neboli konektivita, může být nejen s PC pomocí kabelů USB nebo HML, ale i bezdrátově, a to přes BlueTooth či připojení k internetu přes Wi-Fi. Paměťové médium je Flash Paměť o velikosti 16 GB s možností doplnění o microSD či microSDHC kartu. Systémová paměť, tedy paměť RAM, dosahuje hodnoty 1024 MB. Za zmínku stojí další hardwarové komponenty, jako jsou GPS modul, akcelerometr, gyroskop či digitální kompas.[24]

Společnost Samsung se u tohoto modelu poměrně dost zaměřila na kvalitu fotoaparátu. Ten pořizuje fotografie v rozlišení 3264x2446 pixelů (8MPx) nebo širokoúhle 3264 x 1968 bodů. V standardně dodávané aplikaci od společnosti Samsung lze nastavovat základní vlastnosti, jako je citlivost ISO (maximálně 800), bod ostření po celé dotykové obrazovce, různé scénické režimy či režimy zaostřování jako je detekce obličeje, makro nebo automatický režim.[24]

O natáčení videa v rozlišení Full HD, tedy 1920x1080 pixelů, se stará stejná optika (a aplikace) jako u fotografování. Videosekvence je pořizována rychlostí 30 snímků za sekundu s bitrate kolem 18 Mb/s. Výsledné video je ukládáno do formátu MP4.[24]

5 WOLFRAM MATHEMATICA

„Over 25 Years of Experience – Millions of Users.“[28]

Software Mathematica od společnosti Wolfram představuje jeden z nejznámějších systémů pro numerickou a symbolickou matematiku a vizualizaci dat. Díky tomuto softwaru se dá matematika aplikovat nejen pro řešení malých projektů libovolného rozsahu nebo rutinních výpočtů, ale i na velkosystémová řešení.[27,28]

První zmínky se datují do roku 1987, kdy vznikla společnost Wolfram Research Inc. První vydání pak vyšlo rok nato, tedy v roce 1988. Již od verze 1.0 byl brán jako nepřehlédnutelná součást softwarového odvětví. Během několika prvních měsíců se Mathematica roznesla mezi desítky tisíc uživatelů.[25]

Svojí charakteristikou je velice podobná několika programovacím jazykům, kterými jsou například Java, SQL, XML nebo Jazyk C. Od každého z těchto jazyků přebírá některé vlastnosti a spojuje je do symbolického jazyka vhodného pro aplikování matematických metod. Symbolický jazyk je pojmem, který umožňuje zobrazit grafy, representovat data či psát programy.

Z počátku bylo využití softwaru Mathematica převážně v oborech, jako je fyzika, engineering a matematika. V dnešní době se však software rozšířil do dalších odvětví, jakými jsou například přírodní, biologické či sociální vědy. Velké uplatnění má též pro studijní účely.[25,26]

Software lze využít pro plnění úkolů, jakými jsou:

- Zpracování komplexních symbolických výpočtů, které obsahují velké množství členů
- Zavádění, analýza a vizualizace dat
- Řešení rovnic, diferenciálních rovnic a minimalizace problémů numericky nebo symbolicky
- Výzkum chování numerických modelů a simulací, od jednoduchých regulačních systémů až po srážku galaxií, finančních nebo biologických systémů. Chemických reakcí či vlivů na prostředí
- Výroba profesionálně kvalitních, interaktivních technických dokumentů pro elektronickou distribuci či tisk

- Ilustrace matematických nebo vědeckých konceptů pro studenty
- Sazba technických informací
- Provádění odborných prezentací a seminářů
- Usnadnění rychlého rozvoje engineeringových společností a finančních institucí [25, str. 10]

5.1 Základní pojmy

Tím nejzákladnějším pojmem v prostředí Wolfram Mathematica může být interpretace $f[x]$. Tímto výrazem může být reprezentována nejen matematická funkce, ale i jakákoliv grafika, zvuk, program nebo celý Mathematica dokument či notebook.

5.1.1 Notebookový systém

Jako notebook se označuje interakční nezávislý dokument, který kombinuje textovou formu, respektive textový procesor se zřetelně definovanou soustavou buněk, které jsou seřazeny svisle v rozbalovacím okně. Celé toto okno vypadá, jako by bylo rozdělené na odstavce. Výhodou notebookového systému je komunikace mezi nimi, protože po zadefinování proměnné v jednom notebooku je tato proměnná funkční i v notebooku jiném.[25]

Pomocí buněk lze od sebe oddělovat jednotlivé příkazy programovacího jazyka, lze v nich provádět celé naprogramované části, či zobrazovat vstupy, výstupy, grafiku nebo nadpisy a různé texty.

Celý notebook s příponou .nb je možné převést do jiného formátu jakým je například HTML či TeX a je možné s ním jakkoliv jinak pracovat, nebo ho odeslat přes e-mail, uložit na server nebo webové stránky. [25]

5.1.2 Programovací jazyk

„Mathematica poskytuje silné programové vývojové prostředí.“ [25, str. 8]

Čitelnost celého programového kódu dělá především přehlednost a poměrná jednoduchost programovacího jazyka. Přejít z jiného jazyka je tak mnohem jednodušší a efektivnější. Velkou výhodou je, že programátor nemusí předdefinovat proměnnou ani její typ, nemusí dimenzovat matice či pole nebo překládat program. Příkazy pro provedení

jakékoliv operace jsou definovány buď vhodnou zkratkou nebo celým anglickým názvem či spojením. Všechny příkazy jsou obsaženy v přehledné interaktivní nápovědě. [25,26]

5.1.3 Nápověda – Wolfram mathematica 9 DOCUMENTATION CENTER

Nápověda v softwaru Mathematica je tvořena jako samostatný indexovaný notebook s pokročilými vyhledávacími schopnostmi. Obsahem takového notebooku je popis veškerých nastavení hledaného příkazu, alternativy, vzorové vypracované příklady použití či komplexnější využití v tomto prostředí. Celá nápověda je zpracována i na oficiálních stránkách společnosti Wolfram.[26]

Rozdílem od ostatních nápověd v jiných programovacích rozhraních je možnost jakékoliv změny parametrů příkazu přímo v notebooku nápovědy. Lze tak měnit výsledky i vyhodnocovat výsledky s vlastním zadáním. Otevře-li si tak programátor nápovědu pro daný příkaz může libovolně měnit jeho parametry, aniž by v budoucnu zůstaly změny v nápovědě uloženy. V případě, že dojde k nepožadovaným úpravám, je možno celou stránku jednoduše obnovit znovu vyhledáním hledaného příkazu.

5.1.4 Práce s grafikou

Wolfram Mathematica obsahuje množství grafických vzorů pro vizualizaci matematických výsledků. Výsledky se dají zobrazit pomocí grafů a to nejen v 2D ale v 3D prostoru.[26] Množství druhů grafů je též obsáhlé (vrstevnicové grafy, grafy hustoty, ekonomické či statistické grafy a jiné).

K pohodlné práci s grafikou patří bezesporu i provádění animací, zpracování videozáznamů (velice paměťově náročné na počítač) či různá práce s obrázky či fotografiemi, tzv. image processing.

5.2 Image Processing – zpracování obrazu

Image processingem ve Wolfram Mathematice se rozumí zejména problematika zpracování obrazu, a to tvorba, úprava, manipulace a celkově práce s obrazovými daty. Úzce spjata je s Image Processingem i problematika importu či exportu dat, která je popsána v kapitole 3.3.

Obecně lze tedy zjistit z obrázku téměř všechny informace, které jsou uloženy v jeho útrokách či exif souboru. Základní parametry, jako jsou informace o barvách, rozměry,

použité barevné modely nebo typ či formát obrázku jsou k zjištění po napsání správné formulace příkazu a uvedení názvu importovaného obrazu.

Wolfram Mathematica pracuje se spoustou grafických formátů. [26] Nejpoužívanějším je formát JPEG, který díky své oblíbenosti a celkové dostupnosti je vhodný i pro problematiku rozpoznávání dopravních značek. Protože však není možné importovat více souborů najednou, využívaným grafickým formátem je víceobrazový TIFF.

5.2.1 Vybrané příkazy Image Processingu

Software nabízí několik desítek příkazů pro zpracovávání obrazu. [26] Níže jsou vybrány příkazy využívané při práci na algoritmech pro hledání kontur či barevných dopravních značek.

- Tvorba obrázků
 - Tvorba: Image, Rasterize
 - Zachycení informací: ImageCapture, \$ImageDevices, \$DefaultImagingDevice
 - Image Processing v reálném čase: Dynamic, Manipulate
- Reprezentace obrázků
 - Vlastnosti: ImageType, ImageData, ImageColorSpace, ImageChannels, ImageDimensions, ImageAspectRatio, ImageValue, PixelValue, ImageValuePositions, PixelValuePositions,
 - Nastavení: Options, ColorSpace, MetaInformation, ImageSize, ImageResolution
 - Barvy a úrovně: ImageHistogram, FindTreshold, DominantColors, Threshold, FindThreshold, Binarize, Histogram, HistogramTransform, ColorConvert, ColorSpace, ImageColorSpace, ColorSeparate, RemoveAlphaChannel, ImageApply, ImageScan, Colorize, ColorNegate
- Základní práce s obrázky

- Geometrické operace: ImageCrop, ImageTrim, ImageTake, ImageResize, ImageRotate, ImageReflect
- Úpravy: ImageAdjust, ImageClip, Sharpen, Blur, Lighter, Darker
- Porovnání: ImageDifference
- Morfologické zpracování obrazu
 - Základní operace: Dilation, Erosion, Opening, Closing
 - Morfologické analýzy: MorphologicalComponents
 - Analýza komponent: ComponentMeasurements, SelectComponents, DeleteSmallComponents, DeleteBorderComponents,
- Detekce hran
 - Detekce zajímavých bodů: ImageCorners
 - Detekce kontur: EdgeDetection, CrossingDetect, ContourDetect, FindFaces, TextRecognize
- Filtry: Wolfram Mathematica obsahuje celkem 36 filtrů, já jsem využil pouze jeden a to GaussianFilter.
- Ostatní funkce použitelné pro image processing: Manipulate, ListAnimate, TabView, SlideView.

5.3 Import a export

Data, připravená metodami zpracování obrazu jsou dále připravována tak, aby je bylo možné využívat v jiných programovacích prostředích či jazycích. K tomuto úkolu slouží příkaz pro export. Naopak import má za svůj cíl nahrát do prostředí data vhodná pro práci.[26]

5.3.1 Import

V případě rozpoznávání dopravních značek se importují do softwaru Mathematica především obrázky ve formátu JPEG či TIFF případně videozáznamy ve formátu AVI.

`Import["soubor"]` – importuje data ze souboru

`Import["soubor", elementy]` – importuje data ze souboru s vybranými elementy.

`Import["http://url", ...]` – importuje data z URL adresy.

`Import["ftp://url", ...]` – importuje data z FTP serveru.[26]

5.3.2 Export

Exportování probíhá převážně v textové formě. Důvodem je příprava pro zpracování obrazových dat neuronovou sítí. Nejvhodnějším je formát TXT.

`Export["soubor.přípona", funkce]` – exportuje data do souboru s podporovanou příponou. Pole *funkce* označuje proměnnou nebo jakoukoliv funkci, která má být exportována.

`Export["soubor", funkce, "format"]` – exportuje data dle stejného způsobu jako v předchozím případě jen, s jiným zápisem.

`Export["soubor.přípona", funkce, element]` – exportuje data do souboru s podporovanou příponou, kdy pole *element* nastaví další dílčí vlastnosti.[26]

II. PRAKTICKÁ ČÁST

6 PŘÍPRAVNÉ PRÁCE A TVORBA DATABÁZE DOPRAVNÍCH ZNAČEK

Cílem této diplomové práce byl vývoj softwaru pro rozpoznávání dopravních značek ve skupinové spolupráci, kdy se tato část zabývala přípravou vstupních dat pro umělé neuronové sítě a vývojem, respektive výběrem vhodného algoritmu pro detekci objektů z fotografií či videozáznamů. Z mé práce vycházeli spolupracovníci, kteří se zabývali neuronovými sítěmi [30] a kompletací všech společných částí v jazyce C# [31].

Praktická část se tak zabývá nejen programováním v oblasti zpracování obrazu, tzv. image processingu, ale i hledáním vhodných kandidátů v terénu a celkově přípravou dat pro výstupní program. Programovací část byla sepsána v prostředí programu Wolfram Mathematica a většina databáze byla získána fotografovacím strojem Nikon D3000 či mobilním telefonem Samsung Galaxy S2.

Všechny 4 algoritmy jsou závislé na rozpoznávání dle barvy značky, tedy je možné nimi rozpoznávat značky červené (výstražné, zákazové, upravující přednosti), modré (informativní, příkazové) a žluté (hlavní). Značky zelené jsou záměrně ignorovány, protože jejich výskyt je na českých či evropských silnicích minimální (především jde jen o značky označující dálniční síť či městský okruh). Po výběru z těchto barev se provádí různé odfiltrování nepotřebných částí především pomocí detekce hran v obraze či jiných pomocných funkcí. Nakonec se provede zobrazení kandidáta na dopravní značku.

Zobrazenou značku je možné přidat do dat, která se připravují pro učení neuronové sítě. Ta je zpracována rovněž v prostředí Wolfram Mathematica a jejím výstupem je soustava rovnic. Rovnice se zapisují do programovacího rozhraní C#. V něm je naprogramován celý kompletní výstupní program, který zahrnuje nejvhodnější algoritmus pro rozpoznávání z obrazu a videozáznamu, srovnávání nalezených kandidátů s předlohou a zobrazování GPS dat na mapě. Uživatel tak má k dispozici po dodání natočeného scénáře kompletní servis.

6.1 Databáze dopravních značek

Základem mé databáze jsou předlohy pro všechny druhy dopravních značek s vyobrazenou značkou a bílým pozadím. Ostatní fotky byly nalezeny v terénu či na internetu. Zdrojem pro fotografování dopravních značek jako kandidátů pro rozpoznávání mi bylo především město Zlín. Důvodem výběru je zejména místo, studia a pak také

zlínská architektura, která je díky svému zbarvení do červena či ruda vhodná pro rozeznávání kontrastu mezi značkou a pozadím. Postupně tak vznikaly situace, kdy bylo třeba odfiltrovat červenou budovu s červenou značkou. Časová náročnost této části byla cca 36 hodin a bylo získáno celkem 620 fotografií, z toho přibližně polovina snímků z internetu.

Během fotografování jsem dospěl ke zjištění, že zhruba 45% dopravních značek ve městech je nějakým způsobem deformovaná. Ať už se jedná o deformaci vyblednutých barev, polepení nálepkami, koroze, různě zohýbaných či skrytých za zelení (Obr. 16). V takovém případě je velký problém rozeznat barvu či tvar značky (výjimkou jsou polepené značky, které se s největší pravděpodobností nerozeznají až u neuronové sítě). Dalším nepříznivým vlivem je počasí. Jak sluneční paprsky, které svítí přímo proti objektivu, tak mlha či hustý déšť ovlivnili velkou mírou snímání dopravních značek jak z fotografií, tak z videozáznamu. Nejvhodnějším počasím bylo polojasno či zimní období za předpokladu, že značky nebyly pokryty sněhovou vrstvou.



Obr. 16: Deformace dopravních značek

Jako grafický formát pro fotografie a internetové obrázky jsem volil formát s příponou JPG. Tento nejpoužívanější a nejoblíbenější formát i přes svou nevýhodu komprese nabízí dobrý poměr datové velikosti snímku ke kvalitě. Některé další formáty totiž obsahují i informaci o alfa kanálu (GIF nebo PNG), a ta by byla pro učící data nepříznivá. Podobně byl také nevhodný formát BMP, jehož informace díky nekomprimovaným datům měly příliš vysokou velikost. Dalším vhodným grafickým formátem byl TIFF, který je specifický svou vlastností vícenásobného souboru. Je tedy

schopný nést více JPG obrázků, což jsem využíval při importu několika kandidátů najednou a učících či testovacích množin do softwaru Mathematica.

6.2 Úprava fotografií a použitý software

Některé fotografie a obrázky bylo nutné patřičným způsobem upravit. Většinou jsem zanechával původní barevnost dle nastavení fotoaparátu, ovšem pro rychlejší zpracování jsem změnil rozlišení fotografie dle rozumného uvážení. Všechny fotografie měly rozměry maximálně do tisíce pixelů na delší straně.

U některých fotografií bylo třeba kontrolovat i hodnoty v histogramu, což jsem prováděl přímo na fotoaparátu. Přesvětlené nebo podsvětlené fotografie by mohly nepříznivě ovlivnit snímek a celkové vyhodnocení kandidáta.

Pro úpravy rozlišení a mírně doladění fotografií jsem využíval softwaru ACDSee 9.0, který byl nainstalován jako trial verze. Tento program je svou jednoduchostí velice lehce ovladatelný a dostatečně postačil k úpravám. Jeho velkou výhodou byla možnost hromadného přejmenování a zmenšení snímků na požadovanou velikost. Dalším využívaným softwarem byl volně šiřitelný xnView. Ten posloužil především pro tvorbu vícenásobných souborů TIFF. Je třeba dodat, že oba problémy jsou velmi nestabilní na systému Microsoft Windows 8. Práci jsem tak musel provádět na nižší verzi, Microsoft Windows 7.

Protože se využívaly k testování i videozáznamy z kamery telefonu Samsung Galaxy S2, bylo třeba převést formát MP4 do formátu vhodného pro import do softwaru Mathematica. K tomu jsem používal software Adobe Premiere CS6, opět v měsíční trial verzi s prodloužením. Zde jsem nejen převáděl formáty pro import videozáznamů, ale i vytvářel obrazové sekvence, které jsem rovněž importoval a testoval. Tyto videozáznamy byly pořízeny pouze jako testovací v první fázi vývoje a zkoušení možností. Nejedná se o plnohodnotné scénáře, které byly využity ve finální verzi softwaru pro rozpoznávání. Rovněž zkoušeným softwarem pro převod video formátů byl Xilisoft Video Converter a spolupracující využíval Sony Vegas.

7 DETEKČNÍ ALGORITMY

Část s detekčními algoritmy se zabývá popisem a rozbohem různých postupů, které byly vybrány jako vhodné pro vyhledávání dopravních značek. Zaměřil jsem se na takový algoritmus, který je možné aplikovat či napodobit v prostředí C#.

Další vývoj jiných algoritmů byl zaměřen na porovnání funkcí zpracování obrazu v oblasti rozpoznávání tvarů či barev v digitálním obraze. Vyzkoušeno bylo několik barevných modelů, několik morfologických operací či několik možností binarizace obrazu.

Pro demonstrační popis algoritmů jsem použil snímky na dvojobrázku (Obr. 17), které byly importovány do notebooku Rozpoznávací algoritmy a uloženy do proměnné `im`.



Obr. 17: Demonstrační snímky

7.1 Hledání kompromisů mezi vývojovými prostředími

Z počátku bylo třeba ověřit a porovnat, jaké odlišnosti představují mezi sebou obě vývojová prostředí. Na návrh spolupracujícího kolegy jsme se vzájemně dohodli na nejvhodnějším postupu, který bude mít podobné výsledky jak ve Wolfram Mathematica, tak v C#, ve kterém byl zpracováván výsledný program na rozpoznávání dopravních značek.

7.1.1 Barevný model

Výběr barevného modelu byl zvolen dle jeho vlastností a možností v prostředí C#. Byly postupně vyzkoušeny modely RGB, LAB a HSB. Po výsledcích analýzy jsme se se spolupracovníkem [31] dohodli na hlavním zaměření se na barevný prostor LAB a jeho použití v hlavním programu. Důvodem k tomu byla možnost nastavování barevných

kanálů A a B u tohoto modelu, kdy se daly jednoduše vyseparovat v C# potřebné barevné objekty z obrázku po nastavení příslušných číselných hodnot.

Barevný model RGB byl vhodný z počátku na zkoušení a testování metod zpracování obrazu. Ve Wolfram Mathematica bylo u něj možné nastavit barvu, která se dala separovat od pozadí. Avšak pro další využití a hlavně použití v C# nebyl tento model vhodný. Podobné výsledky měl i model HSB, kdy práce s jedním kanálem v obou programovacích prostředích a hlavně nemožnost nastavit barvy v kanálech nebyla dostatečná pro další podrobné zaměření na něj.

7.1.2 Morfologické operace

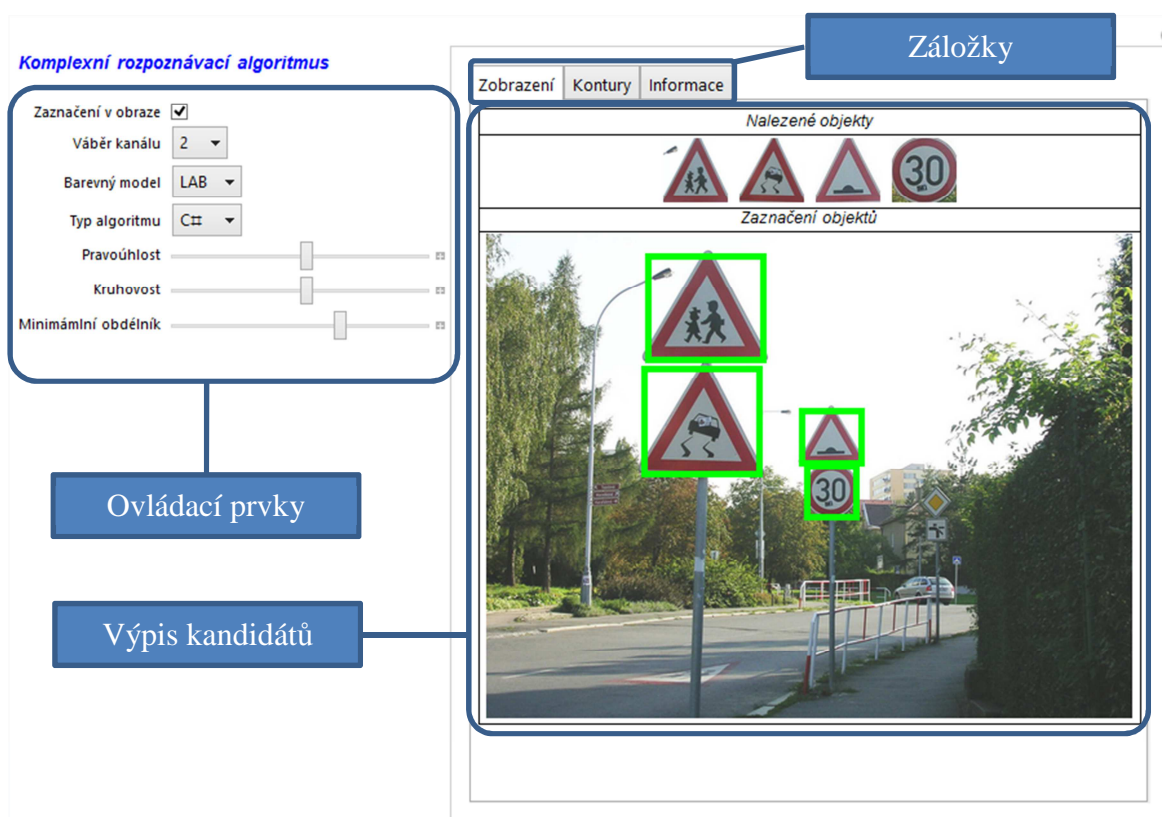
Možností morfologických operací je v Mathematice dlouhý výčet. Nejvýznamnější, respektive vyzkoušené pro vývoj, jsou sepsány v kapitole 5.2.1. Podobně jako v předchozím případě při výběru barevného modelu, muselo dojít zejména ke shodám ve funkcích obou vývojových prostředí v oblasti morfologických operací. Nicméně zkoušeno bylo i více postupů, které nebyly již dále aplikovány do C#, ale měly pouze prověřit teoretickou funkčnost některých metod zpracování obrazu.

Se spolupracovníkem [31] jsme dohodli postup vývoje, aby bylo možné naleznout co nejpodobnější výsledky v obou prostředcích. Mnoho stejně nastavených hodnot však nezaručilo tento požadavek. Například hodnoty nastavování kruhovosti či pravoúhlosti byly zcela odlišné a jsou popsány v kapitole 7.2.1.

7.2 Komplexní vyhledávací algoritmus

Název tohoto algoritmu byl zvolený díky jeho využitelnosti. Je možné si vybrat barevný model (LAB, HSB a RGB) i číslo kanálu (1. 2. a 3. kanál), ze kterého se mají snímat kandidáti na dopravní značky. Tímto jsem si mohl ověřit v jedné funkci možnost vybraných vlastností zpracování obrazu. Dalšími funkcemi je zaznamenání nalezeného kandidáta v pořízeném, respektive importovaném snímku pomocí zatrhávacího tlačítka. Velkou předností je výběr typu algoritmu, který se má provést. Možnosti zde jsou buď algoritmus navržený pro C# nebo Wolfram Mathematica. Posledními (nejdůležitějšími) ovládacími prvky je nastavování hodnot pravoúhlosti, kruhovosti a minimálního obdélníku (příznaků) pomocí posuvníků, dle kterých je možné separovat nalezené objekty. Optimální hodnoty jsou vypsány v kapitole 7.2.1 u těchto příkazů tvořící příznaky). Všechny testovací obrázky se nachází ve složce „Import - dopravní značky“ a jejich nastavení

prahových hodnot příznaků lze vidět v příloze PI. Na obrázku níže (Obr. 18) se nachází úvodní obrazovka s popisem jednotlivých funkcí.



Obr. 18: Ukázka a popis hlavní obrazovky algoritmu

Na výběr je ze tří záložek (režimů zobrazení) a to „Zobrazení“, „Kontury“ a „Informace“. První režim, „Zobrazení“, nabízí pohled na nalezené kandidáty v části „Nalezené objekty“ a dále pohled na importovaný snímek (část „Zaznačení objektů“), ve kterém jsou kandidáti zaznačeni zeleným ohraničením. Druhý režim „Kontury“ nabízí pohled na všechny tři kanály zvoleného barevného modelu (popsáno v kapitolách 7.2.1 až 7.2.3) a poslední, třetí, záložka slouží k zobrazení informací o souboru, kde se nachází tabulka s výpisem na Obr. 19.

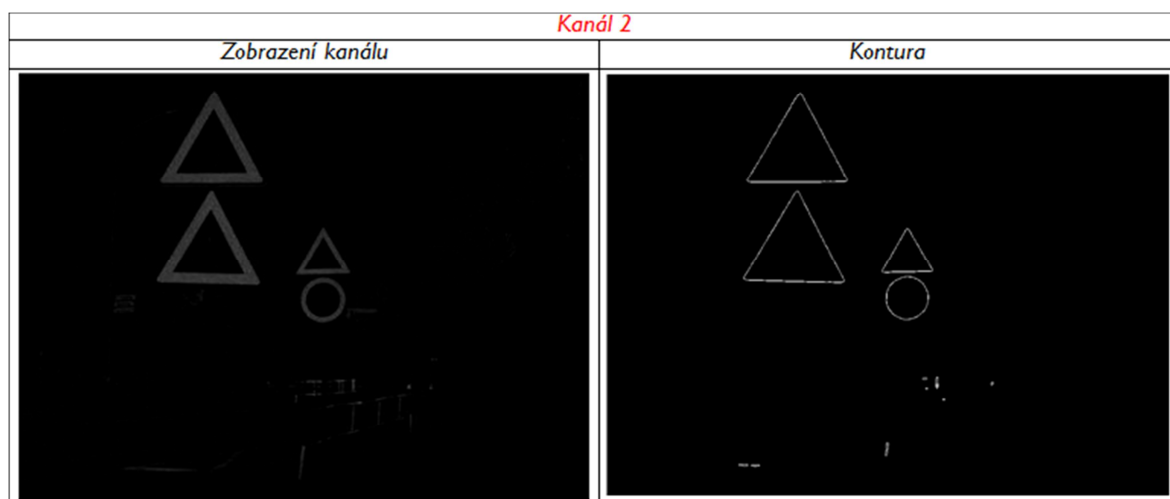
Informace	
Šířka snímku	1000
Výška snímku	750
Počet kanálů	3
Barevný model - vstup	RGB
Barevný model - zpracování	LAB

Obr. 19: Ukázka výpisu informací k importovanému snímku

Celý algoritmus je zapsaný do příkazu `Module`, který zajistí neovlivňování jednotlivých proměnných mezi sebou, a tak lze algoritmus spustit v jakémkoliv notebooku, pokud byl uložený do kernelu.

7.2.1 Barevný model LAB

Výběr barevného modelu pro snímání kandidátů na dopravní značky je hlavním předpokladem pro správný výsledek vyhledávání. Barevný model LAB, jehož hlavní přednosti jsou popsány v kapitole 2.1.4, pracuje především se dvěma kanály A a B. Pro univerzálnost mezi všemi barevnými modely jsem kanály u všech modelů očísloval vzestupně od 1 do 3. Ve druhém kanálu byly hledány značky červené, ve třetí žluté (teoreticky modré).

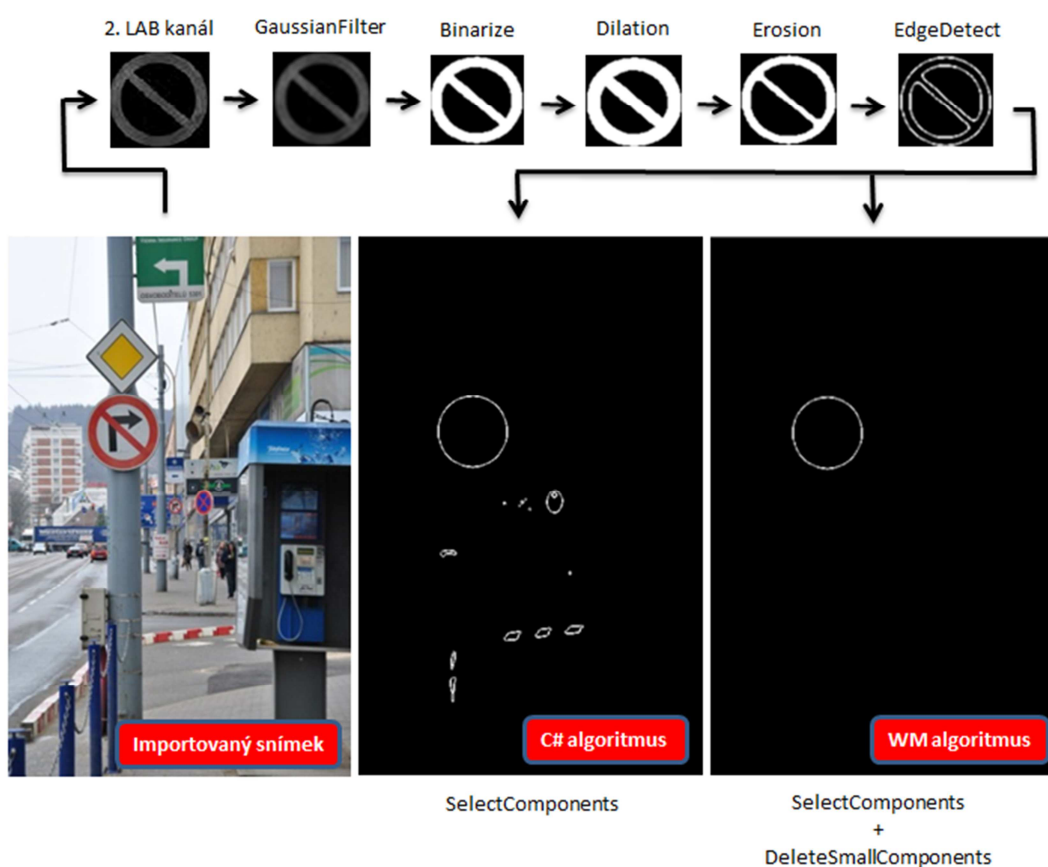


Obr. 20: Kanál 2 barevného modelu LAB při použití C# algoritmu

Na obrázku (Obr. 20) lze pozorovat rozdíly mezi obyčejným zobrazením kanálu v levé části obrázku a zobrazením kontury v pravé části při použití algoritmu navrženého pro C#. Zobrazení kanálu obsahuje bílé útvary, které reprezentují červenou barvu v obraze. Pozadí se tak od kontury oddělí a Cannyho detektor může najít hrany. K doladění a minimalizaci rušivých vlivů pak pomohou další metody zpracování obrazu. Snímek s konturou je posléze uložen do proměnné `konturyVnejsi1`. Zapsání celého příkazu je:

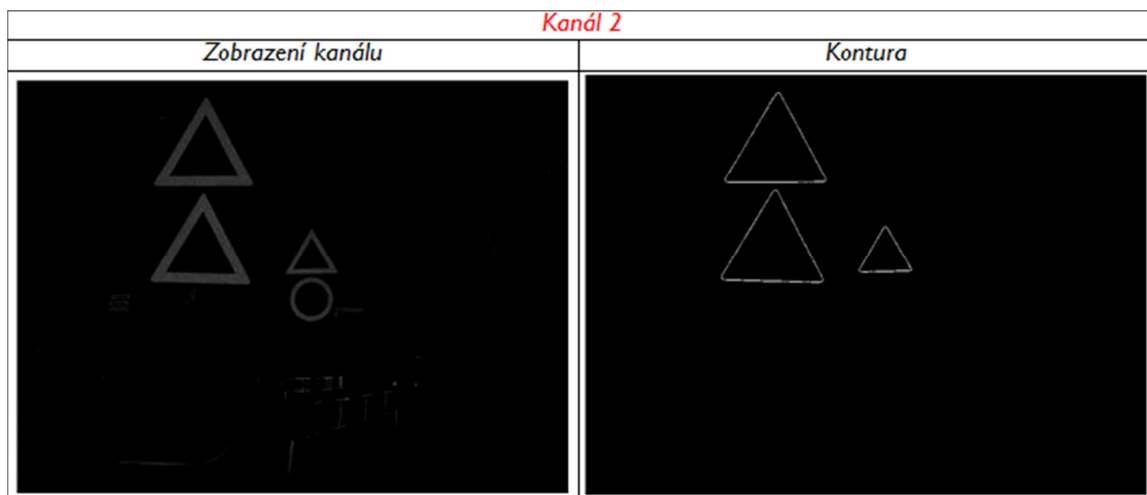
```
hranyVnejsi1=
Table[SelectComponents[EdgeDetect[Erosion[Dilation[Binarize[
GaussianFilter[CielabData[[i]],{{5,5},{1,1}}]],1],2],1,
Method->"Canny"],"EnclosingComponentCount",#==0&],{i,1,3}];.
```

Z proměnné `CielabData` se postupně vybíraly kanály 1 až 3 a zapisovaly se do proměnné `hranyVnejsi1` pomocí cyklu `Table`. Před zapsáním na tyto snímky bylo postupně aplikováno několik příkazů. Provedla se úprava Gaussovým filtrem – `GaussianFilter` a dále binarizace – `Binarize`, dilatace – `Dilation` a eroze – `Erosion`. První zmíněná úprava měla za výsledek rozmazání obrazu a odstranění šumu, druhá převedla snímek do binární podoby (do monochromatických barev). Třetím a čtvrtým krokem bylo využití morfologických operací dilatace a eroze, které byly popsány v kapitole 2.2.5. Elementem nastaveným pro dilataci bylo číslo 1 a číslo 2 bylo zvoleno pro erozi. Tyto hodnoty jsem zjistil experimentováním. Po provedení převodu do podoby hran příkazem `EdgeDetect` s argumentem `Method->"Canny"` bylo nutné vyčistit obraz od hran, které byly pro hledání kandidátů nevhodné. Základní takovou očištěnou byla separace vnějších hran příkazem `SelectComponents` s argumentem `EnclosingComponentCount`, kdy doplňujícím parametrem zde byla experimentální hodnota. Celý zmíněný postup (Obr. 21 – C# algoritmus) byl proveden jen v případě, pokud byl ve výběrovém menu zvolený algoritmus navržený pro C#.



Obr. 21: Postup morfologických operací obou navržených algoritmů

Druhý algoritmus (Obr. 21 – WM algoritmus), tedy navržený pro možnosti softwaru Wolfram Mathematica se liší v několika příkazech a provádí se až po potvrzení podmínky `If`, kdy hodnota rozhodnutí v proměnné `wolfram` je „WM“.

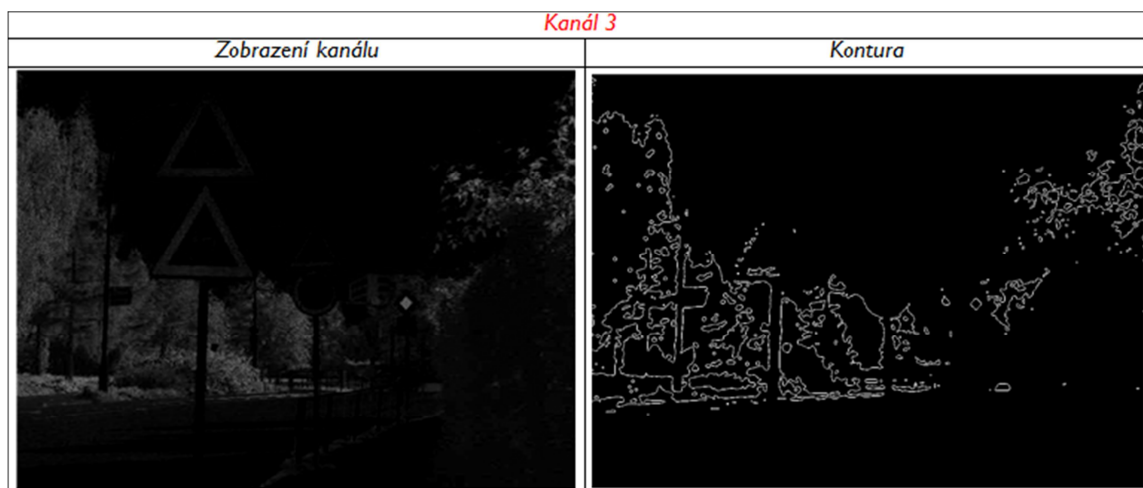


Obr. 22: Kanál 2 barevného modelu LAB při použití wolfram algoritmu

Po porovnání dvou obrázků Obr. 20 a Obr. 22 lze zpozorovat změny v pravé části, tedy v části se zobrazenou konturou. Malé částečky v pozadí (včetně jedné dopravní značky) byly odstraněny. Tento výsledek by se dal považovat za nevhodný, protože jeden z kandidátů tak nebude nalezený. Ovšem na jiných snímcích měl tento postup lepší vyhledávací schopnosti, zejména díky odstranění nepotřebných kontur a celkovému vyčištění obrazu. Takováto kontura se nalezne po provedení proměnné `hranyVnejsi1` a následujícího postupu:

```
hranyVnejsi1=
Table[DeleteBorderComponents[DeleteSmallComponents[
SelectComponents[EdgeDetect[Erosion[Dilation[Binarize[
GaussianFilter[CielabData[[i]],{{5,5},{1,1}}]],1],2],1,
Method->"Canny"],"EnclosingComponentCount",#==0&]],,
{i,1,3}];
```

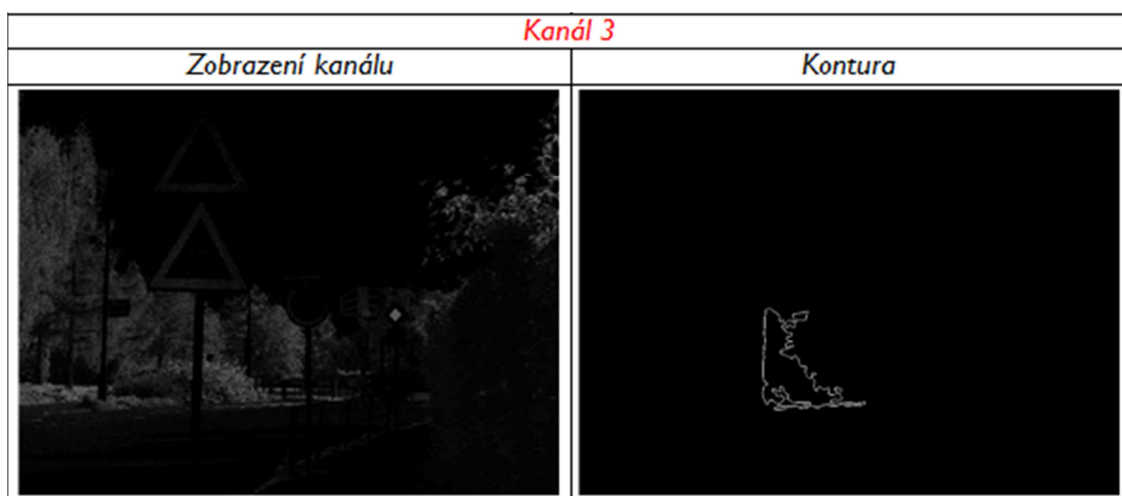
Podobnost wolfram algoritmu a C# algoritmu je zřejmá i na zápisu. Na konci provádění však přibily ještě 2 metody zpracování obrazu, specifické jen pro Wolfram Mathematica. Nebylo je možné aplikovat do jiných prostředí. Jedná se o příkazy `DeleteSmallComponents`, který odstraní z obrazu malé zanedbatelné či jinak ignorovatelné částečky, a `DeleteBorderComponents`, který odstraní nepotřebné části podél okrajů snímku.



Obr. 23: Kanál 3 barevného modelu LAB při použití C# algoritmu

Kanál 3 (Obr. 23 a Obr. 24) se vyznačuje separací žluté a modré (teoreticky) barvy. Jasně bílé místo je u importovaného snímku pouze u značky hlavní silnice. Problematické však mohou být ostatní neodstraněné objekty z obrazu. Výsledkem byla soustava 16 nalezených objektů, z nichž jeden, 9., je značka hlavní silnice. Počet nalezených objektů lze měnit u všech kanálů laděním pomocí posuvníků pro nastavení hodnot pravoúhlosti, kruhovosti a minimálního obdélníku.

Při použití wolfram algoritmu došlo k nalezení jediného objektu v obraze. Již dle kontury (Obr. 24) však lze konstatovat, že obrazec není dopravní značka. Výsledek je tak při použití této kombinace nastavení negativní a kandidát na dopravní značku nebyl nalezen. Příkazy `DeleteSmallComponents` a `DeleteBorderComponents` zde odstranily téměř vše, a v případě velkého počtu malých objektů v kontuře došlo i k odstranění potřebných částí.



Obr. 24: Kanál 3 barevného modelu LAB při použití wolfram algoritmu

Dalším posunem v algoritmu byl průchod přes několik podmínek, tzv. příznaků, přes které musel projít každý nalezený objekt v kontuře. Parametry podmínek, tedy hodnoty kruhovosti a pravouhlosti, lze měnit v daném rozmezí, které bylo zjištěno experimentováním. Rozmezí se pohybuje od čísla 3 do čísla 20 a dochází tak ke změnám v počtu nalezených objektů v obraze. Na některých snímcích je tak možné ručně eliminovat rušivé vlivy. Optimální hodnoty jsou kolem čísla 12.

Jedním z nejdůležitějších příkazů byl při tvorbě příznaků příkaz `ComponentMeasurements` s danou vlastností. Pro kruhovost jsem využil "Circularity" a pro pravouhlost "Rectangularity". Výsledkem bylo číslo ve formátu s desetinnou čárkou. Zde lze pozorovat rozdíl oproti teoretickým znalostem, popsanych v kapitole 2.2.6 a rozdílu mezi metodami jazyku C# , kde se hodnoty pohybovaly dle teoretických předpokladů. Pro dosažení výsledků v desítkových číslech jsem musel použít zaokrouhlování příkazem `Round` s elementem `0.001` a vynásobením výsledné hodnoty číslem 100. Tyto hodnoty se mi posléze mnohem lépe hledaly a experimentování s nimi bylo uživatelsky příjemnější. Celou popsanou činnost jsem provedl pomocí soustavy příkazů:

```
u=ComponentMeasurements[hranyVnejsi1[[pop]], "Rectangularity"]
v=ComponentMeasurements[hranyVnejsi1[[pop]], "Circularity"];
zaokPrav=Table[Round[u[[i,2]],0.001]*100,{i,1,Length[u]}];
zaokKruh=Table[Round[v[[i,2]],0.001]*100,{i,1,Length[u]}];
podm=Table[If[zaokPrav[[i]]<=p&&zaokKruh[[i]]<=k,1,0],
{i,1,Length[zaokPrav]}];
```

Vektor, vycházející z podmínky `podm` obsahoval čísla 1 a 0, dle kterých jsem rozpoznal objekty, které by mohly být vhodnými pro označení jako kandidáta na dopravní značku. Jejich rozměry, respektive obdélníky kolem těchto objektů, jsem zjistil pomocí stejného příkazu jako v minulém případě, tentokrát s vlastností "BoundingBox":

```
rozmary=
Table[ComponentMeasurements[hranyVnejsi1[[pop]], "BoundingBox"
][[Flatten[Position[podm,1]][[i]]]],{i,1,Length[Flatten[
Position[podm,1]]]}];
```

Dle získaných rozměrů bylo možné zjistit šířky a výšky obdélníků, díky čemuž jsem mohl sestavit poslední filtrační podmínku a tou byl výpočet nejmenšího obdélníku kolem kontury. Provedl se výpočet rozdílu mezi pozicemi pixelů a ukládal je do proměnných

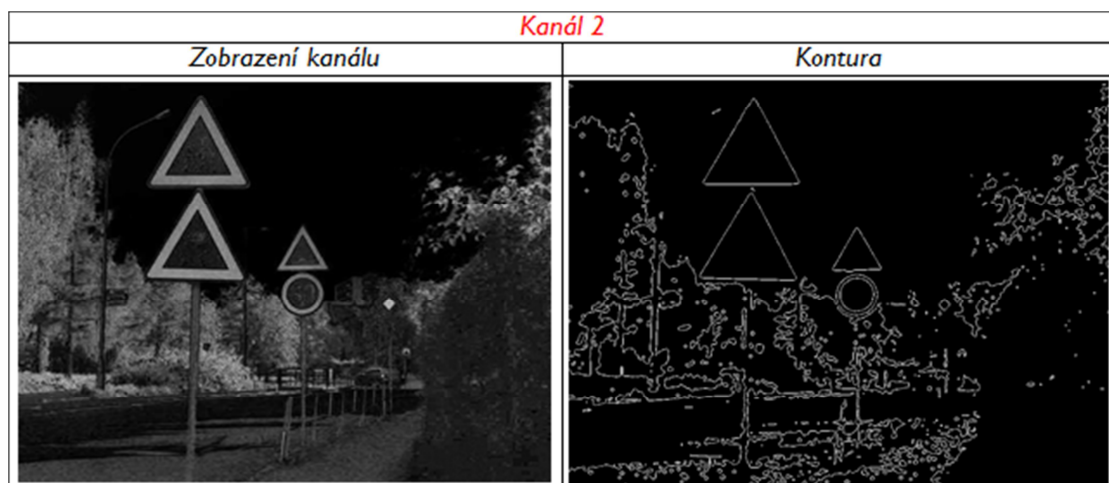
šířka a výška. Poté se hodnoty převedly do kladných čísel a tato hodnota mi řekla, jaký je rozdíl mezi nimi. Dle toho jsem mohl vyseparovat objekty obdélníkových tvarů, které měly velký rozdíl mezi šířkou a výškou. Zde jsem vycházel z předpokladu, že kandidát na dopravní značku je vždy ohraničený čtvercem či obdélníkem s malým rozdílem mezi těmito čísly. Hranice rozdílu je opět možné měnit v ovládacím prvku pomocí posuvníku (proměnná `o`) a jeho hodnoty se mění od 2 do 30, kdy pro ideální kruh je hodnota nulová, pro trojúhelník 10.

Protože výsledkem podmínky pro minimální obdélník byly opět hodnoty 1 a 0, posledním krokem ke zjištění kandidátů byl převod čísel 1 na rozměry výsledných objektů dle příkazu `AppendTo`, který přidával do vektoru `poziceNova` rozměry, které pak byly vykreslovány a zpracovány ve výsledném zobrazení zelenou barvou. Stejně tak se pomocí těchto rozměrů ořezávali kandidáti a zobrazovali jako jednotlivé omezené objekty v záložce „Nalezené objekty“.

7.2.2 Barevné modely HSB a RGB

Testování a vývoj v barevných modelech HSB a RGB byl proveden jen pro ověření, který z nich je vhodnější jako konkurence pro model LAB. Ani jeden z modelů nebyl aplikován v programovacím prostředí C#. Model RGB se tam používá pouze do doby, než dojde k převedení na prostor LAB. Postup všech příkazů byl naprosto stejný jako u prostoru LAB v kapitole 7.2.1, jen se měnil argument příkazu `ColorConvert` na potřebný barevný model.

Kanály, které mě zajímaly, byly u HSB a RGB rozdílné. U prvně zmíněného se nejlepší výsledky dostavovaly u 2. kanálu, naopak u RGB u všech tří. Model HSB má ve druhém kanálu uloženy hodnoty nejsytějších barev (Saturation). Není tedy třeba pro kandidáty znát barvu značky. Naopak u RGB je třeba znát jak barvu, tak i kanál, ve kterém se nachází (podobně jako u modelu LAB). Červené značky jsem hledal v prvním kanálu, žluté v prvním a druhém, a modré jen ve třetím.



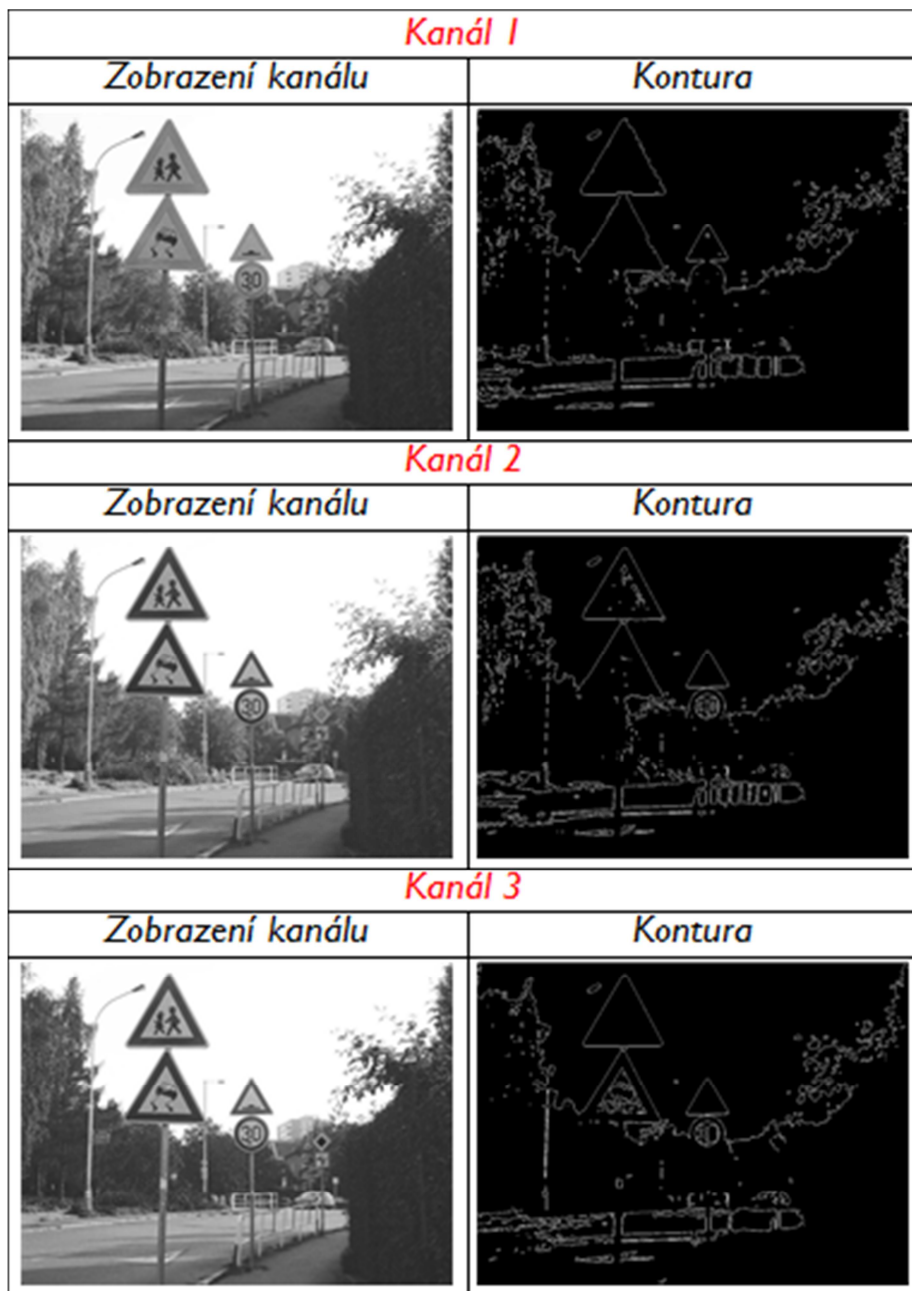
Obr. 25: Kanál 2 barevného modelu HSB při použití C# algoritmu

Na Obr. 25 a Obr. 26 lze opět pozorovat rozdíl mezi režimem algoritmu pro C# a Wolfram Mathematicu pro barevný model HSB ve druhém kanálu. Oproti modelu LAB je při zobrazení kanálu nalezeno více bílé barvy, tedy zajímavých míst pro detektor hran. Proto kontura obsahuje vysoké množství malých objektů (Obr. 23 – pravá část). Naopak při použití Wolfram Mathematica algoritmu se díky příkazům pro mazání malých objektů smažou veškeré důležité části včetně kandidátů.



Obr. 26: Kanál 2 barevného modelu HSB při použití wolfram algoritmu

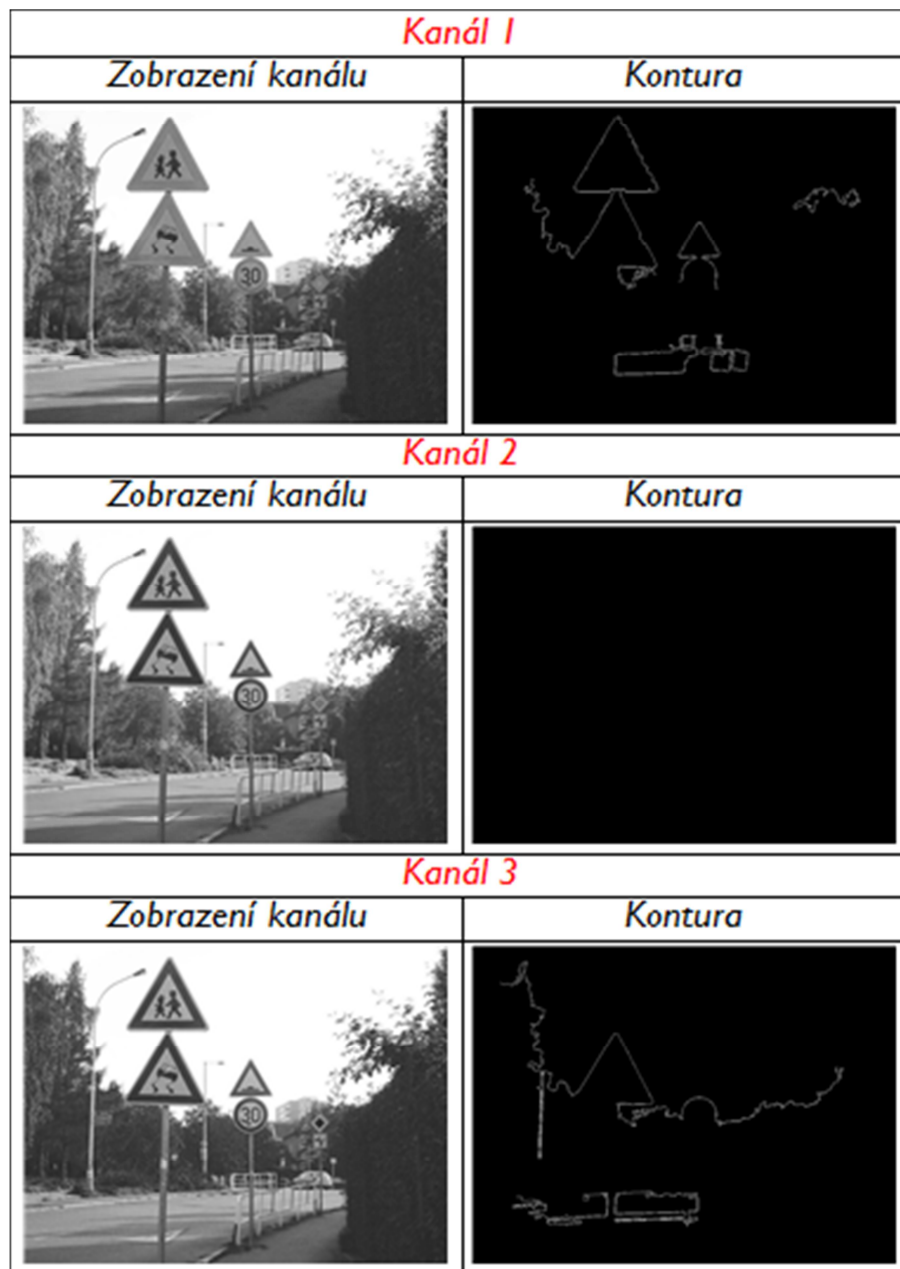
U některých snímků dopravních značek mi barevný model HSB podával dostatečné a kvalitní výsledky, ovšem kvůli poměrně vysokému počtu rušivých vlivů a objektů se v mnoha případech musela provádět ruční separace a musel jsem si sám vybírat snímky, které jsem dle svého uvážení za kandidáta označil.



Obr. 27: Kanály barevného modelu RGB při použití C# algoritmu

U modelu RGB je výsledek poměrně problematický, protože je třeba číst ze všech tří kanálů, které jsou zobrazeny na Obr. 27. V tomto obrázku jsou pozorovatelné velké rušivé vlivy v pozadí ve všech kanálech. Počet nalezených objektů byl velice vysoký a zastavil se na čísle 12 pro červený kanál R i pro modrý kanál B. Žluté značky se velmi špatně při použití této kombinace nastavení identifikují ve všech třech kanálech.

Velká čistka objektů v kontuře nastala při použití Wolfram Mathematica algoritmu (Obr. 28). Ve druhém kanálu došlo k úplnému smazání všech objektů. Počet nalezených kandidátů se tak snížil cca o 84%.



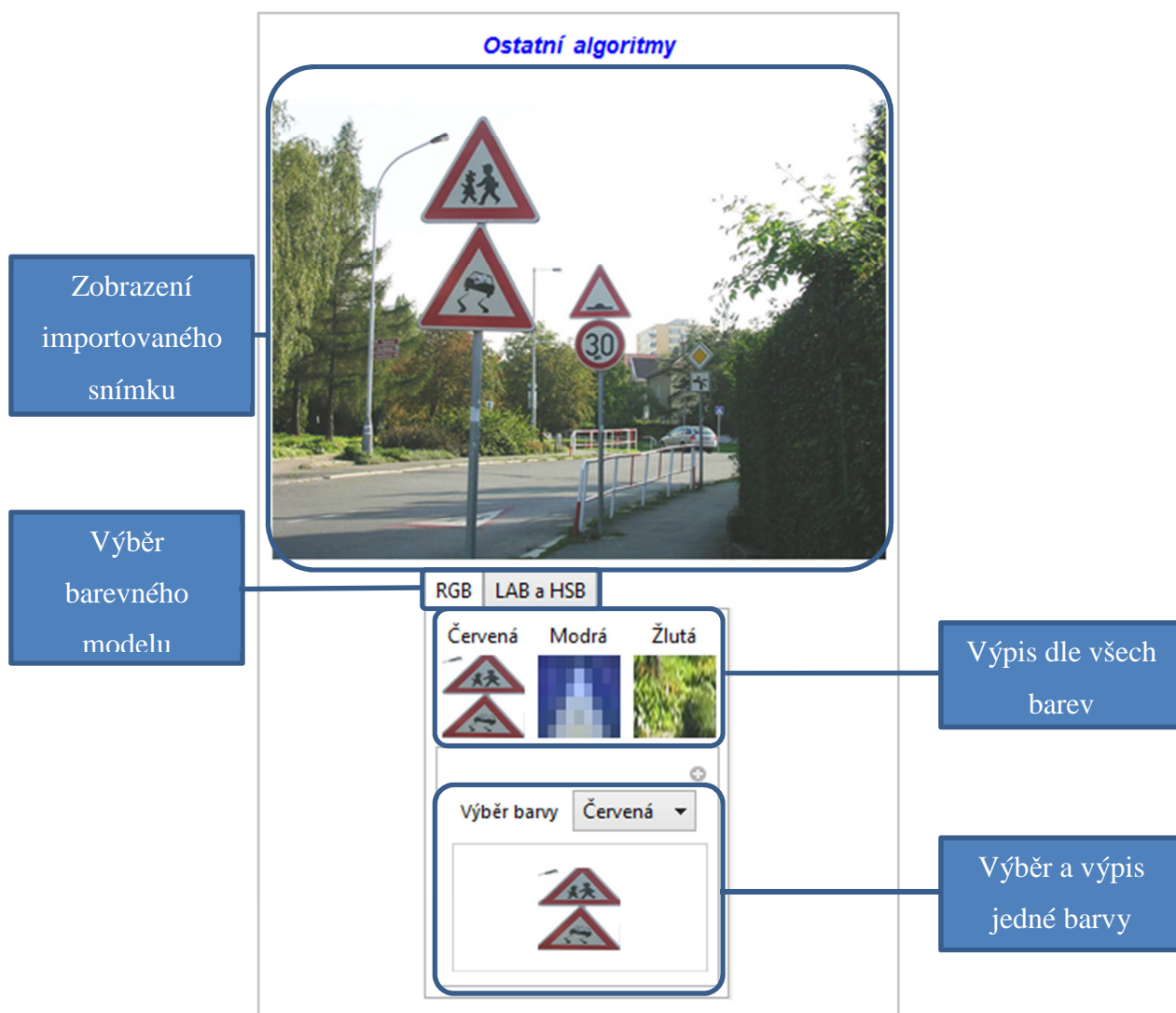
Obr. 28: Kanály barevného modelu RGB při použití wolfram algoritmu

Využití komplexního vyhledávacího algoritmu bylo především nalezení té správné cesty k vývoji a výzkumu hlavního programu, který vytvářel spolupracující. Pomocí omezené a přesně nastavené verze jsem ho využil i pro tvorbu učících, testovacích a validačních vektorů v kapitole 8.

7.3 Ostatní algoritmy

Tyto algoritmy mi sloužily především jako výchozí pozice pro vývoj komplexního vyhledávacího algoritmu. Byly to první nalezené postupy, které podávaly požadované, respektive potřebné výsledky při hledání vhodných kandidátů na dopravní značky. Oproti

komplexnímu vyhledávacímu algoritmu se u těchto nevyskytuje detektor hran. Nelze tak rozeznávat tvary a hledání probíhá pouze na základě barev. Vzhled je patrný na obrázku Obr. 29. Manipulate obsahuje minimum ovládacích prvků. Lze měnit zobrazení dle barevných modelů na RGB a společné LAB a HSB. Důvodem je použití jiného pracovního principu u prvně zmíněného prostoru.



Obr. 29: Vzhled zobrazení ostatních algoritmů

Výpis obsahu je uspořádán u RGB modelu jako zobrazení dle barvy značky buď najednou, nebo je možno pomocí výběrového menu vybrat z barev červené, modré a žluté. Po přepnutí na záložku LAB a HSB lze zpozorovat výběr barevného modelu (LAB nebo HSB) a výpis buď zobrazení kanálů, nebo informace o pořízeném, respektive importovaném snímku. Nad celým výběrovým menu lze spatřit importovaný snímek.

7.3.1 Algoritmus dle RGB barevného prostoru

V podstatě je tento algoritmus velice jednoduchý svým principem. Díky použitým příkazům v oblasti digitálního zpracování obrazu v softwaru Wolfram Mathematica ovšem není jednoduchá jeho integrace do jiného vývojového prostředí. Proto byl tento postup mým prvním, který mi podával první kandidáty a sloužil spíše jako vnoření se do problematiky zpracování obrazu.

Oproti výslednému Komplexnímu vyhledávacímu algoritmu je postavený na naprosto jiných základech. Využívá pro detekci hlavní barvy dopravní značky. Binarizace se provádí příkazem `ChanVeseBinarize` s parametrem určujícím barvu v anglickém překladu. Separují se tak objekty s barvou červenou, modrou nebo žlutou a ukládají do proměnné:

```
konturaOkraj=  
DeleteBorderComponents[ChanVeseBinarize[znackaVstup,  
"TargetColor"->barva]];
```

Po uložení do této proměnné bylo zapotřebí testovat, jestli snímek obsahuje nějakou vykreslenou konturu (objekt) či nikoliv, aby mohly být zjištěny pozice ořezu. Sestavil jsem tak podmínku, ve které se zjistí počet černých pixelů. Pokud je počet černých pixelů rovný počtu všech pixelů, vypíše se chybová hláška "Nenalezen žádný objekt". V opačném případě se provedlo mazání malých objektů pomocí příkazu `DeleteSmallComponents`. Tímto byla získána výsledná kontura, jejíž data obsahovaly pouze čísla 1 a 0 (díky binarizaci).

Vykreslení kandidáta proběhlo pomocí zjištění pozice krajních jedniček nalezeného objektu. Tyto hodnoty se zapsaly na určené pozice příkazu `ImageTake` a na konec mohlo být provedené `ImageResize` pro změnu rozměrů kandidáta. Vypsání na výstupu jsou 2 podobné algoritmy, u nichž jeden vypisuje zmíněné barvy automaticky a u druhého je nutný výběr.

7.3.2 Algoritmus dle LAB a HSB barevného prostoru

Principem u tohoto algoritmu je separace objektů zájmu dle barevných předloh pomocí vykreslování barevné kontury. Každá malá část v obraze dostala přidělenou specifickou barvu, která byla pomocí morfologických operací dilatace a eroze upravována, stejně jako u Komplexního vyhledávacího algoritmu. Nakonec se provedla opačná

binarizace, protože při obyčejné měly pixely mého zájmu barvu černou. Vše se ukládalo do proměnné:

```
konturaOkraj =
```

```
ColorNegate[Binarize[Dilation[Image[Colorize[ImageForestingCo  
mponents[Cie]]],1]]];
```

Vykreslování a testování, zdali byla kontura prázdná, probíhala stejně, jako u algoritmu dle RGB barev, popsaného v kapitole 7.3.1. Protože jsou vypisovány všechny 3 kanály barevných modelů, celý postup musel být uzavřený do příkazu `AppendTo`, a ten zapisoval do proměnné výsledek nalezených kandidátů ve všech kanálech.

8 PŘÍPRAVA DAT PRO NEURONOVOU SÍŤ

Mnou navržené algoritmy pro zpracování obrazu a separaci dopravních značek slouží nejen k pomocnému vývoji výsledného algoritmu v prostředí C#, ale i k získání dat, vhodných pro učení neuronové sítě. Za tato data se dají označit hodnoty všech pixelů v pořízeném snímku. Obsah testovacích a učících vektorů byl vždy jiný a stejně i jeho délky. Každá skupina se skládala ze snímku dle normy (bílé pozadí) a snímků z reálného světa.

Pro učení neuronové sítě byly vytvářeny 3 varianty barevnosti kandidátů - barevné, černobílé a šedé snímky o rozměrech 50x50 pixelů. Tyto vlastnosti se dále kombinovaly s ořezem nalezených kandidátů, kdy jsem získával detailní vyobrazení znaku ve značce. Dalším zkoušením a vyhodnocováním výsledků učení sítě jsme se s kolegou, který se zabýval problematikou neuronové sítě [30] dohodli na postupech a kombinacích, které zahrnovaly experimentování nad rozměry výřezů detailních piktogramů vyjadřujících význam značky.

Učících množin vznikalo v průběhu tvorby této diplomové práce spousty. Důvodem k poměrně vysokému počtu bylo především zkoušení různých kombinací mezi vektory, přidávání nových značek pro učení či testování a převody mezi barevnostmi. Každý nový či upravený vektor vznikl po vzájemné diskuzi s kolegou.

Testovací data byla vždy tvořena tak, aby se ověřila funkčnost jednotlivých kategorií a správné přiřazení značek do nich. Obsahem nikdy nebyly skupiny, ale jednotliví kandidáti získaní z reálného prostředí. Tímto vektorem jsme s kolegou ověřovali, jestli je neuronová síť správně naučená.

Celá tato část diplomové práce byla rozdělena do 4 příprav, kdy každá příprava přinesla nové výsledky, ze kterých se vycházelo v přípravách dalších.

8.1 První příprava dat

Na začátku tvorby testovacích vektorů vznikaly vektory, které obsahovaly snímky pro testování nastavení neuronové sítě. Proto obsahem byl poměrně malý počet snímků. Všechny skupiny v učícím vektoru jsem popsal do tabulky (Tabulka 3) s tím, že jednotlivé fáze příprav jsem čísloval dle zkoušeného učení.

Tabulka 3: Kroky prvního stádia příprav dat

Fáze přípravy	Počet skupin DZ	Počet DZ celkem	Barevnost
1	6	18	SŠ, ČB
2	7	21	SŠ, ČB
3	8	24	RGB

První tvorba přípravných dat se skládala z několika testovacích vektorů o různé barevnosti a počtu dopravních značek. V první fázi bylo nachystáno 6 skupin, kdy každá obsahovala 3 dopravní značky stejného typu (dle tvaru), se stejným znakem ve stupních šedi a černobílé barvě. Obsahem této fáze příprav byly značky: „Jiná nebezpečí“, „Zákaz vjezdu v obou směrech“, „Stop, dej přednost v jízdě“, „Hlavní silnice“, „Dej přednost v jízdě“ a „Kruhový objezd“. Zvýšení počtu značek v druhé fázi mělo za následek rozšíření této množiny o kandidáta na značku „Zákaz zastavení“. Přípravný vektor vypadal následovně:

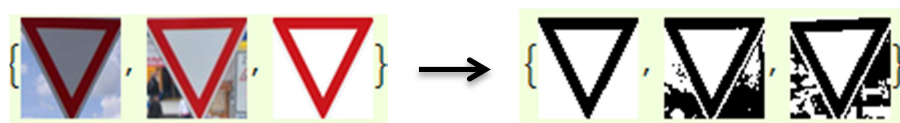
```
znackyPriprava=
{znackyHlavni, znackyPrednost, znackySTOP, znackyPrikazove,
znackyVystrazne, znackyZakazove, znackyZakazZastaveni,
znackyZakazStani} ;
```

Skupiny snímků dopravních značek, zapsané v proměnné `znackyPriprava`, se dále mohly upravovat dle potřeb na černobílé snímky (Obr. 30) nebo snímky ve stupních šedi (Obr. 31). V případě monochromatických barev se využila klasická binarizace:

```
velikosti=
Table[Length[znackyPriprava[[i]]], {i, 1, Length[znackyPriprava]}];
znackyUceniCB=
Table[Table[ImageData[Binarize[znackyPriprava[[j, i]]]],
{i, 1, velikosti[[j]]}], {j, 1, Length[znackyPriprava]}].
```

Základem jsou dva cykly `Table`, kterých se docílilo pohybem výběru v proměnné `znackyPriprava`, kdy iterační proměnná `i` je omezena číslem 1 jako začínající hodnotou a číslem, které udává velikost skupiny se snímky. Tato velikost, respektive délka vnořeného vektoru, je výsledkem proměnné `velikosti`. Například pro třetí fázi přípravy dat obsahuje celkem osmkrát číslo 3. Postup byl chronologicky takový, že po načtení obrázku z vektoru se provedla binarizace (Obr. 30) `Binarize` a převod obrazových

hodnot do dat pomocí ImageData. Naposledy se mohl provádět export do textového souboru TXT.



Obr. 30: Převod barevného vektoru na monochromatický

Převod snímků do stupňů šedi (Obr. 31) se provedl velmi podobným postupem, jako u monochromatických snímků. Proměnná `znackyUceniDataStupneSedi` se liší pouze v části `sBinarize`, které bylo nahrazeno příkazem `ColorConvert` s argumentem "GrayScale".

```
znackyUceniDataStupneSedi=
Table[Table[ImageData[ColorConvert[znackyPriprava[[j,i]],
"Grayscale"]],{i,1,velikosti[[j]]}],{j,1,
Length[znackyPriprava]}]
```



Obr. 31: Převod barevného vektoru na stupně šedi

Při převodech kandidátů do černobílých snímků často docházelo k různým deformacím obrazu. První taková deformace se projevila už při druhé fázi této přípravy. Kombinace barev červené a modré u značek „Zákaz stání“ a „Zákaz zastavení“ způsobila promíchání těchto barev a při převodu na hodnoty 1 a 0 se kandidát vykreslil do černé barvy (Obr. 32). Neuronová síť by tak neměla šanci tyto značky rozeznat. Způsobeno je to vysokou jasovou hodnotou v kanálech pro červenou a modrou barvu. Tuto deformaci jsem vyhodnotil jako nevhodnou. Spolupracující, zabývající se problematikou neuronových sítí [30] tento problém vyřešil v jazyce C#, avšak ani po průzkumu možností příkazů ve Wolfram Mathematica jsem nenašel způsob, jak tuto situaci nasimulovat v tomto programovacím prostředí. Situace je po zhodnocení podobná jako s příkazem `ColorSeparate`, kdy není možné nastavovat hodnoty pro barvy v kanálu.



Obr. 32: Deformace binarizací
obrazu

8.2 Druhá příprava dat – dělení dle kategorií

Několik fází příprav, označených jako druhá příprava dat, se skládala z rozdělování dopravních značek do skupin dle kategorií a významu s vyšším počtem dopravních značek. Učící data byla poskládána především z kandidátů dle normy, doplněná o snímky z reálného světa. Počet kategorií se změnil pouze jednou a je zapsán v Tabulce 4.

Tabulka 4: Kroky druhého stádia příprav

Fáze přípravy	počet skupin DZ	Počet DZ celkem	Počet DZ ve skupinách						
1	7	156	6	6	6	6	42	44	46
2	6	153	6	6	6	0	42	44	49
3	6	107	1	1	1	0	34	34	36
4	6	18	1	1	1	0	5	5	5
5	6	21	2	2	2	0	5	5	5
6	4	147	0	6	6	0	0	44	49

První 3 skupiny (4 v první fázi příprav) se skládaly ze samostatných dopravních značek - „Hlavní silnice“, „Dej přednost v jízdě“ a „Stůj, dej přednost v jízdě“ (u první fáze příprav „Přednost protijedoucích vozidel“). V dalších skupinách je patrný velký přírůstek prvků v případě prvních třech fází. Důvodem bylo rozšíření skupin o všechny kandidáty z dané kategorie, kde kategorie byly rozděleny dle významů značek na „Příkazové“ - modré, „Výstražné“ - trojúhelníkové a „Zákazové“ – kruhové.

Všechny zmíněné skupiny jsem dle potřeby zapsal do proměnné znackyPriprava. Barevnost odstínu při těchto fázích nebyla řešena. Dále se pracovalo především se značkami v RGB barvách. Pozitivním a použitelným výsledkem byla poslední fáze příprav. Došlo k odstranění skupin značek „Hlavní silnice“ a „Příkazové značky“, protože po zdokonalení algoritmu v prostředí C# a jeho filtračním podmínkám mohl tento vektor (Obr. 33) pracovat souběžně s tímto algoritmem, a tak pomáhal při základním rozdělování kandidátů dle kategorií.

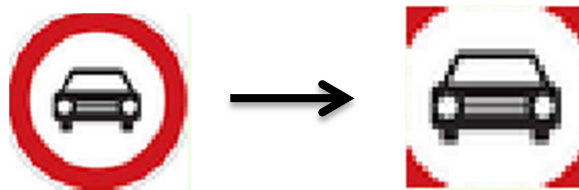


Obr. 33: Použitý vektor do výsledného programu

K první a druhé přípravě byl vypracovaný notebook s názvem 1. a 2. příprava, jehož obsahem jsou nejen postupy tvorby, ale i všechny vytvořené vektory připravené k jednoduchému zpracování a exportu jeho dat.

8.3 Třetí příprava dat – ořezy piktogramů 34x34 pixelů

Doposud největšího posunu a vývoje při učení neuronové sítě se dosáhlo při třetí přípravě dat. Byly přichystány učící a testovací vektory přímo dle druhu dopravních značek, respektive jejich barev, kterými se odlišily skupiny příkazových, zákazových a výstražných značek. Oproti předchozím přípravám zde byly exportované vektory velmi rozdílné, protože data se skládala ze snímků s menším počtem pixelů, kdy se provedly ořezy kandidátů na rozměry 34x34 obrazových bodů pomocí příkazu ImageCrop. 8 krajních pixelů z každé strany se odstranilo a ve snímku zůstal detailní pohled na piktogram dopravní značky.



Obr. 34: Separace piktogramů u kruhové DZ

```
znackyUceniTypyOrez=
Table[Table[ImageCrop[znacky[[j,i]],34],{i,1,velikosti[[j]]}]
,{j,1,Length[znackyZakazove]}];
```

Argumentem proměnné `ImageCrop` bylo číslo, které nastavilo ořez snímku ze všech stran na daný poměr stran, v tomto případě 34. Klasickým postupem přes 2 cykly `Table` jsem ořezané vektory ukládal do proměnné `znackyUceniTypyOrez` a dále (většinou manuálně) skládal nové skupiny.

8.3.1 Příkazové značky

Nejméně obsáhlými učícími vektory byly při tomto stádiu příprav skupiny příkazových značek. Po prozkoumání všech kandidátů jsem usoudil, že kromě značek, které ruší příkazovou značku, je většina kandidátů v převládající modré barvě. Aplikoval jsem binarizaci a porovnával výsledky této operace (Obr. 35).








Obr. 35: Porovnání binárních snímků

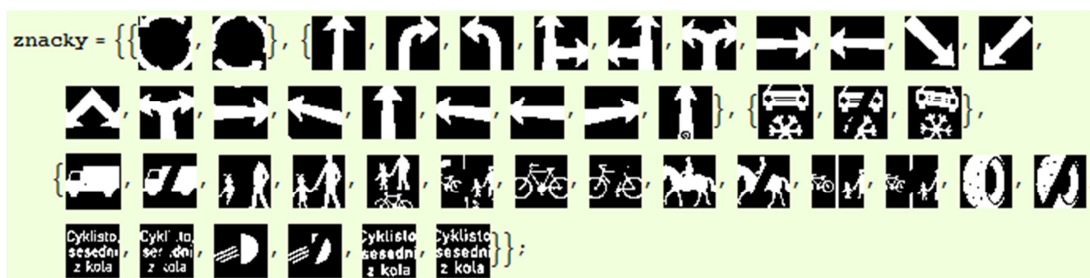
Piktogramy v levé části Obr 35. jsou reprezentanty dopravních značek s modrým pozadím. V pravé části lze pozorovat deformaci obrazu při zrušení platnosti této značky pomocí červeného pásu. Červená barva se při binarizaci opět míchala s modrou a nelze oddělit tyto 2 barvy od sebe. I přes deformaci tohoto druhu jsem usoudil (a testování

výsledků neuronové sítě mě v o tomto utvrdilo), že rozdíl mezi dvěma piktogramy je poměrně výrazný.

Tabulka 5: Kroky třetího stádia příprav dat

Fáze přípravy	Počet skupin DZ	Počet DZ celkem	Počet DZ ve skupinách					
								Ost.
1	6	38	2	15	2	3	2	14
2	6	37	2	12	2	3	2	16
3	6	40	2	15	2	3	2	16
4	4	44	2	19	0	3	0	20

Dopravní značky, zobrazující kategorie v Tabulce 4, jsou pouze informativní a reprezentují danou skupinu. K výrazné změně v počtu dopravních značek nedocházelo. V konečné fázi na základě výsledků učení neuronové sítě jsem zrušil 2 kategorie a sloučil je s poslední skupinou. Čtvrtá fáze (Obr. 36) proto byla vybrána jako použitelná pro aplikování do hlavního programu.



Obr. 36: Použitý vektor do výsledného programu

8.3.2 Zákazové značky

Změna rozměrů snímků s kandidáty (34x34 pixelů) se u dopravních zákazových značek projevila negativně. V detailu piktogramu (Obr. 32 pravá část) lze pozorovat červené pixely v rozích ořezaného snímku. I přes tento nedostatek jsem dále připravoval data (Tabulka 6, Tabulka 7 a Tabulka 8) a rozděloval si skupiny do kratších vektorů.

Tabulka 6: 1. dvě fáze třetího stádia příprav dat pro zákazové značky

Fáze přípravy	Počet skupin DZ	Počet DZ celkem	Počet dopravních značek ve skupinách												Ost.
															
1	12	43	2	7	4	3	2	4	5	2	6	4	2	2	0
2	13	57	2	7	4	3	2	4	5	2	6	4	2	2	14

Nejdelším vektorem, dle Tabulky 6, byla druhá fáze přípravy. Obsahovala všechny typy zákazových dopravních značek. Vzniklo 13 skupin, rozdělených dle druhu piktogramu tak, aby případný pozitivní výsledek učení neuronové sítě mohl značky

přirázovat do skupin dle správnosti a logiky. Například v 7. skupině se nacházely značky omezující rychlost, v 10. značky „Zákaz stání“ a „Zákaz zastavení“ a v poslední ostatní jiné zákazové značky.

Tabulka 7: 3.a 4. fáze třetího stádia příprav dat pro zákazové značky

Fáze přípravy	Počet skupin DZ	Počet DZ celkem	Počet dopravních značek ve skupinách								
											Ost.
3	9	56	4	11	8	5	2	6	4	2	14
4	9	59	4	13	8	5	2	6	8	2	11








Po výsledcích učení musely být vektory výrazně zkráceny a některé skupiny sloučeny (Tabulka 7). „Zákaz vjezdu všech vyznačených vozidel“ byl sloučen s „Zákazem vjezdu vyobrazených vozidel“ a dopravní značky upozorňující na limitní omezení se též seskupily. „Zákaz vjezdu v obou směrech“ jsem sloučil se skupinou první, kdy vznikla velká skupina „Zákazů vjezdu“. Skupiny v exportovaných vektorech byly ještě doplněny o další kandidáty z reálného prostředí.

Tabulka 8: 5.-7 fáze třetího stádia příprav dat pro zákazové značky

Číslo přípravy	Počet skupin DZ	Počet DZ celkem	Počet DZ ve skupinách	
				Ost.
5	9		12	47
6	8		17	52
7	8		13	56

Kvůli velké výraznosti a odlišnosti dopravních značek „Zákaz stání“ a „Zákaz zastavení“ vznikly speciální vektory, které měly za úkol oddělit tyto specifické dopravní značky do jedné skupiny a ostatní zákazové značky do skupiny jiných zákazů (Tabulka 8). Hierarchie rozpoznávání by pak teoreticky vypadala tak, že po rozdělení kandidátů dle barvy a rozeznání zákazové značky by se provedlo další dělení. Počet značek ve skupinách byl náhodný. Výsledek učení byl pozitivní a teoreticky použitelný při snímcích v odstínech šedi. Po dlouhé konzultaci jsme se však se spolupracovníky dohodli, že tohle dělení nepoužijeme [30].

Tabulka 9: Ostatní fáze třetího stádia příprav dat pro zákazové značky

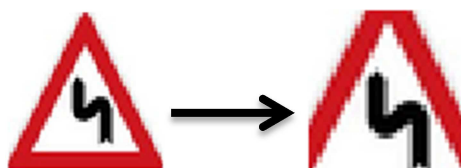
Fáze přípravy	Počet skupin DZ	Počet DZ celkem	Počet dopravních značek ve skupinách							
										Ost.
8	8	0	4	11	8	5	2	6	2	13
9	7	0	4	14	13	0	2	6	2	7

Osmá a devátá fáze (Tabulka 9) se už skládala z finálních šesti kategorií a byla by použitelná za předpokladu, že by se použilo dělení z Tabulky 8. Celkově jsem dělením zákazových značek získal poznatek, že u těchto kandidátů je rušivým vlivem okraj značky a ani ořezem na rozměry 34x34 jsem se tohoto problému úplně nezbavil.

Pro urychlení procesu učení jsem se podílel na trénování a nastavování neuronové sítě u zákazových dopravních značek. Proces tak mohl probíhat nezávisle na sobě na dvou PC.

8.3.3 Výstražné značky

Výstražné dopravní značení mělo při odstranění krajních pixelů na rozměr 34x34 obrazových bodů (Obr. 37) pohledově ještě negativnější výsledky, než v předchozí kapitole. Piktogram tolik nevynikal, jako u zákazových značek a velkým rušivým vlivem bylo jak pozadí za značkou, tak samotný trojúhelníkový okraj



Obr. 37: Separace piktogramů u trojúhelníkové DZ

Zjištění nepěkného odstranění krajních pixelů mě vedlo k tomu, že jsem se dále těmito značkami nezabýval a po skupinové konzultaci jsme došli k závěrům, že bude třeba provádět ořezy piktogramů dle kategorií pokaždé s jinými hodnotami. Tabulka 10 a Tabulka 11 obsahuje vektory dohromady tří fází příprav výstražných značek, a však ani jedna fáze nebyla nijak využita ve finálním programu.

Tabulka 10: 1. a 2. fáze třetí přípravy dat pro příkazové značky

Fáze přípravy	Počet skupin DZ	Počet DZ celkem	Počet DZ ve skupinách																	Ost.			
1	19	50	2	2	2	2	2	2	2	1	2	2	2	2	2	4	7	4	4	4	2	2	0
2	20	57	2	2	2	2	2	2	2	1	2	2	2	2	2	4	7	4	4	4	4	2	7

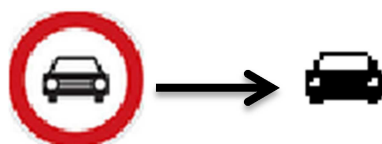
Tabulka 11: 3. fáze třetí přípravy dat pro příkazové značky

Fáze přípravy	Počet skupin DZ	Počet DZ celkem	Počet DZ ve skupinách								Ost.	
3	9	0	2	2	2	4	7	4	4	4	2	26

Notebook, ve kterém jsou zapsány všechny učící vektory, jsem pojmenoval jako 3. příprava dat.

8.4 Čtvrtá příprava dat – finální učící množiny

Podoba finálních vektorů se opět odvíjela od zkušeností z předchozích příprav. Po průzkumu ořezů u všech kategorií dopravních značek jsem dospěl k závěru, že bude nutná separace větší, než 34x34 pixelů. Značka hlavní silnice se z původních rozměrů 50x50 pixelů škálovala na rozměr 20x20 obrazových bodů. Zákazové dopravní značky byly nejprve škálovány na 36x36 pixelů a poté ořezány na 18x18 (zákazové) a výstražné na experimentální hodnotu 15x15 pixelů z původního rozměru. Všechny kategorie se musely nejprve převést dle postupu demonstrovaného na Obr. 40. Nicméně na obrázku níže (Obr. 39) je ukázaná úplná separace piktogramu u kruhové dopravní značky. Po srovnání s obrázkem (Obr. 32) v kapitole 8.3 lze jasně vidět odstranění rušivých rohových pixelů.



Obr. 38: Úplná separace piktogramů
u kruhové DZ

Ve vektoru příkazových značek (Obr. 36) se další dělení týkalo jen kandidátů s piktogramem příkazaného směru. Snímky se vyřezávaly do obdélníků kolem směrové šipky a rozměr byl zvolený experimentálně na 23x28 pixelů. Počet odebraných obrazových bodů všech skupin je uvedený v Tabulce 12.

Tabulka 12: Rozměry a ořezy snímků ve 4. přípravě

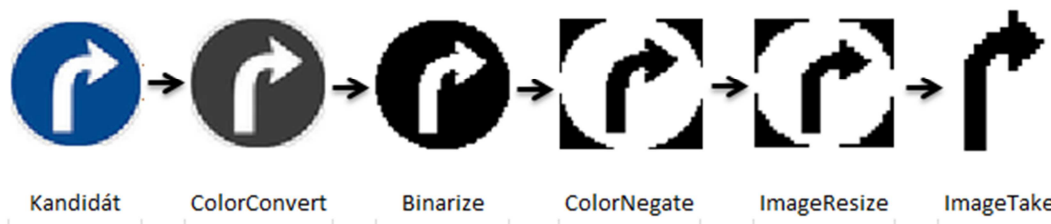
Kategorie značek	Původní rozměr	Škálovaný rozměr	Ořezaný rozměr	Odstranění pixelů			
				Z hora	Z dole	Z leva	Z prava
Zákazové	50x50	36x36	18x18	9	9	9	9
Výstražné	50x50	-	15x15	19	6	14	13
Příkazaný směr jízdy	50x50	36x36	23x28	5	6	8	6
Hlavní silnice	50x50	20x20	-	9	9	9	9

znackyOrez=

```
Table[Table[ImageTake[ImageResize[ColorNegate[Binarize[
ColorConvert[sipky[[j,i]], "Grayscale"]]]], {36,36},
```

```
Resampling->"Bicubic"], {5, 32}, {8, 30}], {i, 1, velikosti[[j]]}],
{j, 1, Length[sipky]]}]
```

Proměnná `znackyOrez` obsahuje vektor skupin snímků omezených na dané rozměry dle určeného postupu, kdy se kandidáti (konkrétně šipky příkázaného směru) upravovali dle metod zpracování obrazu (Obr. 39) pomocí příkazů `ColorConvert`, `Binarize`, `ColorNegate`, `ImageResize` a `ImageTake`.

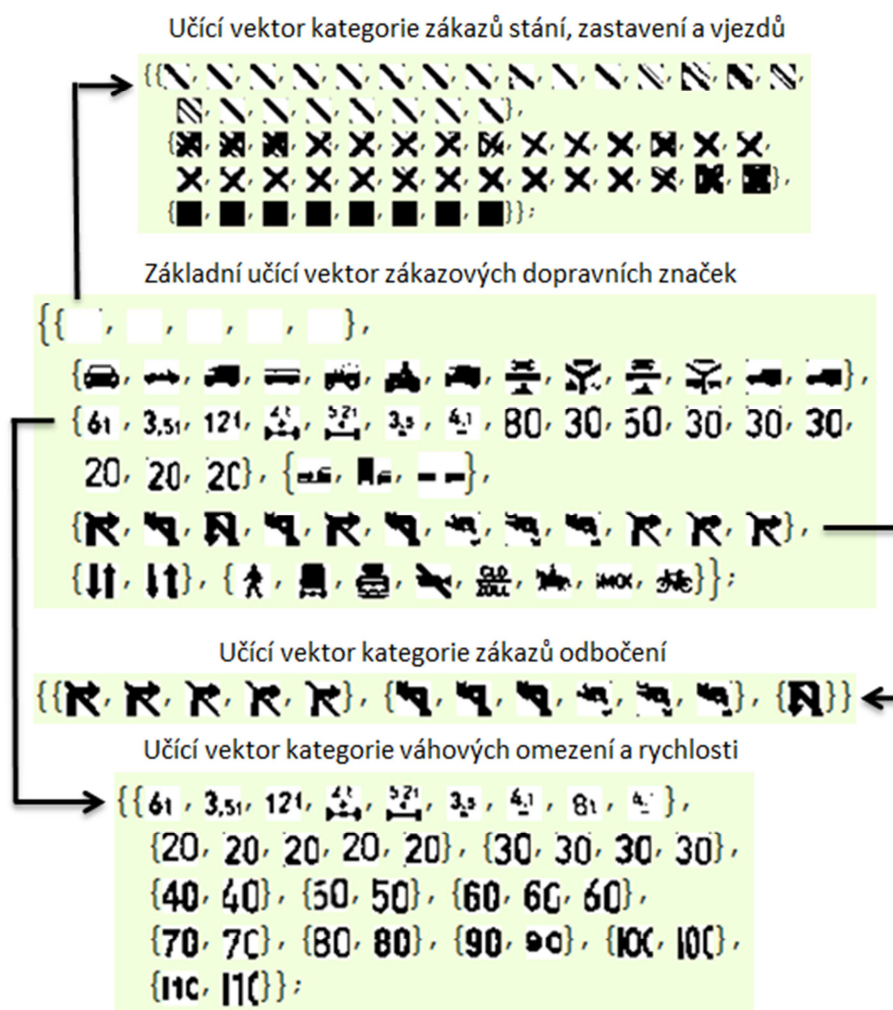


Obr. 39: Postup aplikování metod zpracování obrazu

Kandidátem ve výše zmíněném obrázku (Obr. 39) byla příkazová dopravní značka o rozměrech 50x50 pixelů. Prvně zmíněná instrukce `ColorConvert` byla pro prostředí Wolfram Mathematica zbytečná, nýbrž pro C# se prováděl nepostradatelný převod do stupňů šedi. Tento převod je podmínkou pro metodu `TreshHoldAdaptive`, kterou se více zabýval kolega [30]. Další krok `Binarize` provedl klasickou binarizaci na obrazová data obsahující 1 a 0 a `ColorNegate` význam těchto čísel obrátil, takže vytvořil negativní snímek. `ImageResize` škáloval kandidáta na rozměry 36x36 s bikubickou interpolací, čímž více zmenšil počet obrazových hodnot. Finální úpravou bylo odstranění nepotřebných bílých a černých pixelů pomocí `ImageTake` na rozměry dle Tabulky 12.



Obr. 40: Použitý vektor do výsledného programu



Obr. 41: Použitý vektor do výsledného programu - hierarchie

Způsobem, zmíněným v předchozím odstavci, jsem postupoval i u zákazových a příkazových dopravních značek (Obr. 40 a Obr. 41). Mohl se však vypustit krok s ColorNegate, protože piktogram u těchto typů byl vždy černý (až na minimální výjimky). 1. skupina se dělila na učící vektor značek „Zákaz stání“, „Zákaz zastavení“ a „Zákaz vjezdu“. Dělení 3. skupiny bylo voleno dle čísla v piktogramu, respektive číselného rozměrového či váhového omezení. Posledním dělením u zákazových dopravních značek byly zákazy odbočení na „Zákaz odbočení vlevo“, „Zákaz odbočení vpravo“ a „Zákaz otáčení“. K této přípravě dat vznikl notebook pojmenovaný jako 4. příprava.

U dopravních značek „Hlavní silnice“, se jako učící vektor použily snímky, které vyseparoval hlavní program v jazyku C#. Nijakým způsobem nedošlo z mé strany k žádným úpravám, pouze k exportu jeho dat. Jeho skladbu lze pozorovat na Obr. 42.



Obr. 42: Použitý vektor do výsledného programu

8.5 Testovací a validační vektory

Testovacím vektorem (Obr. 43) se nazývá množina všech kandidátů, která slouží k otestování správného naučení neuronové sítě v prostředí Wolfram Mathematica. Rozměrově se tak neskládá ze skupin, ale ze samostatných kandidátů. Podobně validační vektor (Obr. 44) je množina kandidátů, která má stejné vlastnosti jako testovací, ale slouží k pomocnému přiučení neuronové sítě. Neměla by tak obsahovat náhodné množství dopravních značek, ale přesný počet dle skupin v učícím vektoru, kdy každý kandidát je reprezentantem právě této skupiny.

Postup tvorby byl v obou případech stejný. Skládání probíhalo buď ručním tříděním, nebo pomocí skupiny obrázků, uložených do formátu TIFF a importovaných do WM. V podstatě se všechny další vektory odvíjely od prvního navrženého, do kterého jsem doplňoval značky, které byly třeba pro testování.

Struktura notebooku „Testovací a validační vektory“, který byl vytvořený pro ukládání těchto množin, se odvíjela podle stádií příprav v předchozích kapitolách. Ukázka postupu při tvorbě a převodu testovacího vektoru, respektive dělení dle barevnosti, je zobrazeno na obrázku Obr. 43. Zde jsem využil náhodného výběru testovacího vektoru, který je shodný pro některé z učení z 3. a 4. přípravy.



Obr. 43: Vývoj testovacího vektoru pro zákazové značky pro 3. a 4. Přípravu dat

Při postupu hromadného aplikování metod zpracování obrazu na testovací vektor, se využívaly již v této diplomové práci popsané příkazy ImageCrop, Binarize, ColorConvert, ImageResize či ColorNegate dle požadovaných vlastností přípravných dat.

Validačních vektorů vzniklo minimální množství, protože jsme se tímto způsobem při učení neuronové sítě zabývali pouze okrajově. Celkově jsem tak přichystal 3 validační vektory, z nichž barevný se nachází na obrázku níže (Obr. 44).



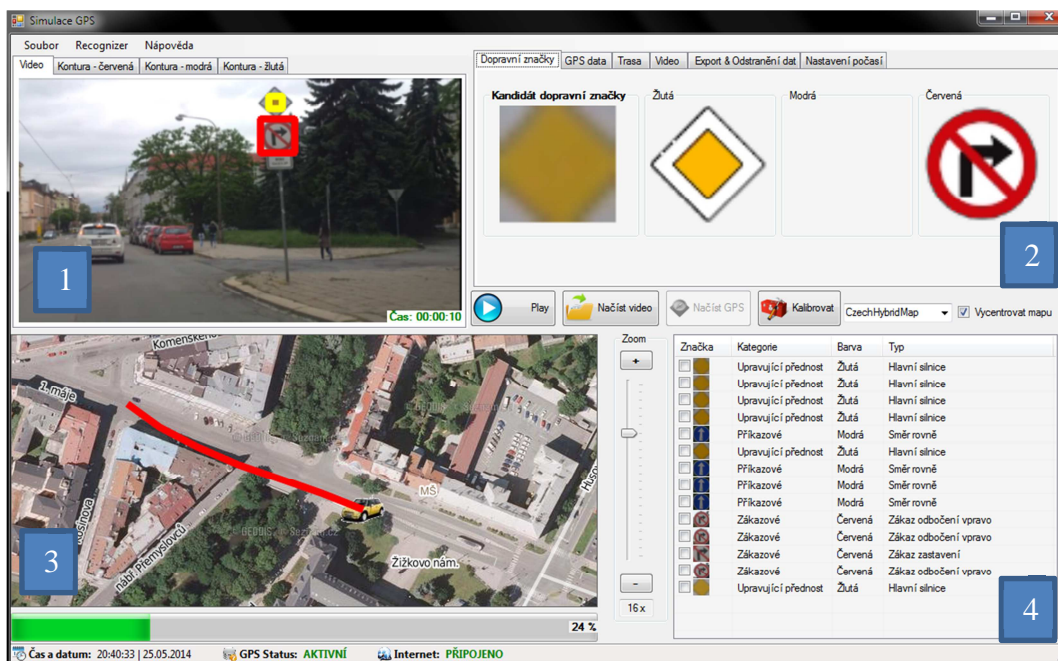
Obr. 44: Ukázka barevného validačního vektoru

9 POPIS VÝSLEDNÉHO PROGRAMU

Spojením této diplomové práce s kolegou, zabývajícím se neuronovými sítěmi [30] byl hlavní program [31], který rozpoznával vybrané dopravní značení z fotografií i videozáznamů. Programovaný byl jazykem C# přes vývojové prostředí Visual Studio 2010 od společnosti Microsoft s použitím knihoven pro zpracování obrazu Emgu CV a gmaps.NET.

Můj podíl práce na tomto softwaru spočíval v dodání návrhu algoritmu, který rozpoznával ze statického obrazu kandidáty na dopravní značky a připravě učících či testovacích dat. Dále jsem se podílel na samotném učení neuronových sítí a při natáčení videosekvencí, které sloužily buď jako testovací pracovní snímky pro získání kandidátů na dopravní značky, nebo jako plnohodnotné scénáře. Při poslední zmíněné aktivitě jsem zajistil technické zázemí.

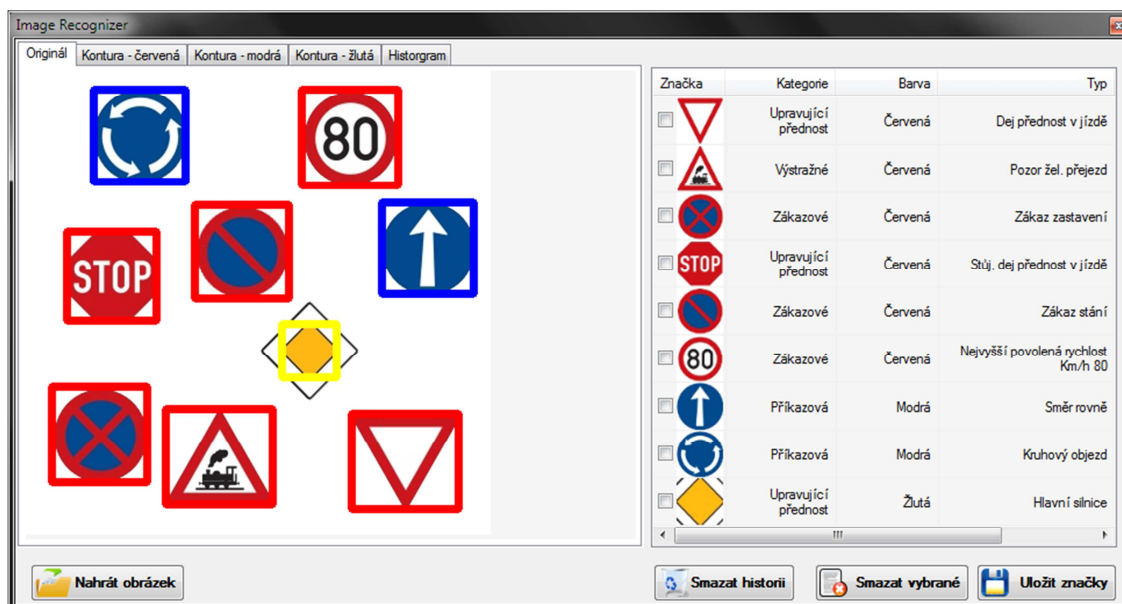
Scénáře se natáčely na náhodných místech ve městech Zlín, Olomouc a Zábřeh na Moravě. Pořízeny byly snímacím zařízením mobilního telefonu Samsung Galaxy S2. Spolu s nimi byla získána i GPS data, která se dala využít pro zaznačení natočeného scénáře do mapy. Městské ulice jsme volili dle předpokládaného počtu dopravních značek. Nastavení kamery bylo přenecháno automatickému mobilnímu telefonu a umístění tohoto zařízení v automobilu se zvolilo na vnitřní zpětné zrcátko, proto scénáře nejsou plně centralizované s vozovkou.



Obr. 45: Hlavní okno DVH recognizeru

Hlavní výsledná aplikace, pojmenovaná DVH Recognizer, zobrazuje několik informací za jeden okamžik. Uživatel má na výběr rozeznávat snímky z videozáznamů (hlavní priorita) a z obrázků, tzv. Image Recognizer (Obr. 46). V obou prostředích jsou k dispozici stejné rozeznávací algoritmy. Okno hlavního programu pro rozeznávání z videosekvencí je skládá ze 4 částí (Obr. 45, popisky 1 až 4):

- 1 – Část, zobrazující videozáznam, ve kterém jsou zvýrazněni nalezení kandidáti na dopravní značku dle barvy.
- 2 – Volba režimů zobrazení. Uživatel si z vybraných záložek vybírá různá nastavení programu. Hlavním oknem je zobrazení nalezených kandidátů a jemu přiřazené předlohy z databáze. Dále je možnost nastavení GPS dat, výpis informací o trase a videozáznamu. Nalezené snímky je z historie možné exportovat jako obrázky. Poslední záložkou je nastavení přednastavených prahových hodnot dle zvoleného scénáře.
- 3 – GPS data jsou zapisována do map a interaktivně zobrazována. Mapa se pomocí tlačítka centralizuje na pozici v datech.
- 4 – Historie nalezených značek ve vzestupném pořadí dle nalezení a jejich přesné přiřazení do skupin.



Obr. 46: Image Recognizer

Image Recognizer (Obr. 46) snímá ze statických snímků kandidáty a přiřazuje jim nalezené dopravní značky z databáze. V levé části se nachází image box se snímkem,

případně lze k originálním snímkům zobrazit kontury nebo histogram pomocí záložek. Pravá část obsahuje historii nalezených dopravních značek podobně jako u videozáznamů.

Snímání kandidátů ze statických snímků lze spustit ve výběrovém menu „Recognizer“, ve kterém se nachází i další záložka „Live Capture!“, což je příprava pro rozeznávání reálného prostředí živě kamerou v automobilu.

Hierarchie učení a jednotlivé fáze lze pozorovat v příloze PII. Přesné postupy při přehrávání scénářů a celkové ovládání popsal kolega v diplomové práci [31].

ZÁVĚR

Hlavním cílem mé práce byl vývoj detekčního algoritmu a příprava učících či testovacích vektorů v prostředí Wolfram Mathematica pro umělou neuronovou síť ve skupinové spolupráci. S kolegy, Bc. Tomášem Vinklerem, který se zabýval neuronovou sítí [30], a Bc. Janem Dospivou skládajícím dohromady všechny části v jazyku C# [31], jsme společně dospěli k výsledné aplikaci, která detekuje kandidáty na dopravní značky ve videozáznamech či statických obrazech a přiřazuje jim vhodné předlohy z databáze.

Pomocí odborných materiálů byla zhotovena teoretická část práce týkající se především metodami zpracování obrazu. Byly v ní vysvětleny základní pojmy jako segmentace, prahování, detekce hran nebo příznaky. Další popis jsem věnoval použitým zařízením, umělým neuronovým sítím, programovacímu rozhraní Wolfram Mathematica a základnímu popisu dopravní infrastruktury především v ČR. Praktická část se zabývala tvorbou databáze fotografií a obrázků dopravních značek v reálném prostředí, návrhem algoritmu pro rozpoznávání dopravního značení pomocí metod zpracování obrazu a přípravou učících a testovacích vektorů pro umělou neuronovou síť.

Tvorba databáze probíhala v terénu velkých měst, kde se nacházela spousta kandidátů (dopravních značek) všech druhů. Mohl jsem tak určit, že nejčastější značky v městských ulicích jsou „Zákaz stání“, „Zákaz zastavení“ a „Hlavní silnice“. Z pořízených snímků byla zhotovena databáze, ze kterých jsem tvořil data pro neuronovou síť.

Vývoj detekčního algoritmu se odvíjel od realizovatelnosti v jazyce C#. Po diskuzi jsem dospěl k postupu, který je aplikovatelný jak ve Wolfram Mathematice, tak v jazyce C# s podobnými výsledky. Rozdílnost mezi algoritmy byl v nastavitelnosti prahovacích hodnot u barevného modelu LAB, který se stal základem pro detekci v obou prostředích. Pomocí několika metod, jako je separace kanálů, binarizace snímku či Cannyho detektor hran jsem dospěl k nalezení objektů, které dále procházely filtračními podmínkami (příznaky) kruhovosti, pravoúhlosti a minimálního obdélníku. Kandidát na dopravní značku byl následně vykreslený do zobrazovacího prostředí ve velikosti 50x50 pixelů. Ve vyhledaných kandidátech se algoritmy rozdílných prostředí lišily v několika řadách či sloupcích pixelů

Celkově byly testovány barevné modely LAB, HSB a RGB v několika rozdílných nastaveních s různými výsledky. Neoptimálnějším se stal model LAB s tím, že rozeznával jen červené dopravní značky. Všechny barvy dokázal rozeznat jen model HSB, ovšem

nebylo možné ho aplikovat do prostředí C#. RGB model byl využitý jen na začátku vývoje. Vyhledávací schopnost závisela na prahových hodnotách příznaků – kruhovosti, pravouhlosti a minimálního obdélníku. U testování tohoto algoritmu se projevila velká náročnost metodik zpracování obrazu na paměť počítače. Na PC sestavě s procesorem Intel i5 a operační paměti RAM o velikosti 8 Gb měl algoritmus při vysokém rozlišení snímku s kandidátem vysokou časovou i výpočetní náročnost.

Navržený detekční algoritmus figuroval nejen jako základ pro hlavní aplikaci, ale i pro tvorbu učících a testovacích vektorů. Tyto vektory se skládaly z kandidátů na dopravní značky rozdělených dle požadovaných kritérií, na kterých jsme se ve skupině domluvili. Z těchto vektorů se získávala data. Ta pak sloužila jako vstup pro umělou neuronovou síť. Postup vývoje nových vektorů se odvíjel od předešlých zkušeností, vznikly tak 4 přípravy dat. První byla považována za seznamovací a testovací, z druhé se pak do aplikace využila hierarchie rozdělení do kategorií dle červených značek. Třetí příprava se skládala z kandidátů škálovaných a ořezaných na rozměr 34x34 pixelů. Z těchto dat bylo využito základní rozdělení příkazových značek. Poslední, čtvrtá příprava, doplnila kompletní hierarchii všech dopravních značek v různých rozměrech.

Pro výslednou aplikaci bylo zhotoveno 9 video nahrávek s dopravními značkami z reálného prostředí. Délka všech scénářů je 460 minut. Počet kandidátů je celkem 71, z čehož 21 značek se vůbec nedetekovalo vyhledávacím algoritmem. Důvodem může být nesplnění podmínek příznaků nebo poškození kontury objektu. Velký vliv mělo i počasí a slunečné podmínky. Celkově nejméně úspěšnými značkami byly výstražné, naopak příkazové značky podávaly úspěšnost nejlepší.

Cíle diplomové práce byly splněny v plném měřítku s většími či menšími problémy. Mohl jsem se tak soustředit i na pomoc při učení neuronové sítě a do určité míry i práci na hlavním programu či natáčení scénářů. Velkou výhodou mi byla interaktivní nápověda softwaru Wolfram Mathematica, naopak nevýhodou nestabilita tohoto vývojového prostředí. Svými možnostmi se z tohoto softwaru stal dokonalý nástroj pro zpracování obrazu, nýbrž díky nestabilitě (plyne ze zaplnění operační paměti), nemožnosti práce s videozáznamy a celkově pomalé reakci ho po zkušenostech doporučuji jen jako simulaci pro jiná vývojová prostředí.

ZÁVĚR V ANGLIČTINĚ

The main aim of my thesis was the development of detection algorithms and preparation of learning or test vectors in Wolfram Mathematica for artificial neural network in group collaboration. With colleagues, Bc. Tomas Vinkler, which dealt with neural networks[30], and Bc. Jan Dospiva composed together all the parts in C #[31], we collectively came to the final application that detects candidates for traffic signs in your video or static Pictures and assigns them to the appropriate template from the database.

Using specialized materials was made theoretical part of the work relating primarily to image processing methods. It had explained the basic concepts such as segmentation, thresholding, edge detection or symptoms. Additional description I gave of used equipment, artificial neural networks, Wolfram Mathematica programming interface and basic description of transport infrastructure, especially in the Czech Republic. The practical part deals with the creation of a database of photos and images of road signs in real environment, design of algorithms for recognition of traffic signs using the methods of image processing and preparation of learning and test vectors for artificial neural network.

Creating the database took place in the field of large cities, where there was plenty of candidates (road signs) of all kinds . I was able to determine that the leading brands in the city streets are "No parking ", " no stopping " and "Highway ". The captured images were made database from which I generated data for the neural network .

Development of detection algorithm was based on the viability of the C # language. After the discussion, I came to a procedure which is applicable to the Wolfram Mathematica and the C # language with similar results. The divergence between the algorithms was adjustability thresholding values in LAB color model, which became the basis for detection in both environments. Using several methods, such as channel separation, the binarization image or Canny edge detector, I came to find objects that pass through the filter further conditions (symptoms) roundness, squareness and minimum rectangle. A candidate for the traffic sign was subsequently rendered to the display environment in the size of 50x50 pixels.

Overall color models were tested LAB, HSB and RGB in several different settings, with different results. The most optimal model with LAB that could see a red traffic signs. All colors could make out only HSB model, but it was not possible to apply it to C#. RGB model was utilized only at the beginning of development. The search capability was

dependent on the thresholds of symptoms - roundness, squareness and minimum rectangle. For testing of this algorithm showed great complexity of the method of image processing on the computer's memory. The PC configured with an Intel i5 processor and RAM, 8 GB algorithm was at a high resolution image with candidate high temporal and computational complexity.

The proposed detection algorithm figured not only as a basis for the application, but also for creating learning and test vectors. These vectors are composed of candidates for traffic signs, divided according to the required criteria on which we agreed in the group. Of these vectors obtain the data. She then served as input for the neural network. The procedure of developing new vectors was based on the previous experience , thus creating four training data. The first was regarded as reconnaissance and testing, the latter was then used in the application hierarchy divided into categories according to the red marks . The third preparation consisted of candidates scaled and cropped to the size of 34x34 pixels. These data were used to the basic division command brands. The fourth preparation, adding a complete hierarchy of road signs in various sizes.

The resulting application was made 9 video recordings of traffic signs from real environment. The length of all scenarios is 460 minutes. The number of candidates is a total of 71, of which 21 marks are not detected at all search algorithm. The reason may be the failure to meet the terms of symptoms or damage to the contours of the object. I had a great influence weather and sunny conditions. Overall, the least successful brands have been warning the contrary, the command marks the best administered success.

The objectives of the thesis were met in full scale with more or less problems. It allowed me to concentrate on assistance in learning neural networks and to some extent also working on a major program or shooting scenarios. The great advantage of the interactive help me Wolfram Mathematica software, while the disadvantage of instability development environment. Their capabilities of this software has become the perfect tool for image processing, but due to instability (resulting from the full system memory), inability to work with video and generally slow response after the experience I recommend it only as a simulation for different development environments.

SEZNAM POUŽITÉ LITERATURY

- [11] Revize TP 65 Zásady pro dopravní značení na pozemních komunikacích. 2013.
- [2] Dopravní značky u Tupolevovy. 2012, Dostupné z:
<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4621285>
- [3] Zásady pro dopravní značení na pozemních komunikacích. 2002.
- [4] An overview of traffic sign detection methods. In: An overview of traffic sign detection methods [online]. 2013 [cit. 2014-05-23]. Dostupné z: http://www.fer.unizg.hr/_download/repository/BrkicQualifyingExam.pdf
- [5] European road signs and traffic signals. Idea Merge [online]. 2013 [cit. 2014-05-23]. Dostupné z: <http://www.ideamerge.com/motoeuropa/roadsigns/>
- [6] Dopravní předpisy v USA. CESTOVÁNÍPOUSA.CZ [online]. 2013 [cit. 2014-05-23]. Dostupné z: <http://www.cestovani-po-usa.cz/pages-cz/dopravni-predpisy-usa-amerika.html>
- [7] EICHNER, Marcin L. a Toby P. BRECKON. Integrated speed limit detection and recognition from real-time video. 2008 IEEE Intelligent Vehicles Symposium [online]. IEEE, 2008, s. 626-631 [cit. 2014-05-23]. DOI: 10.1109/IVS.2008.4621285. Dostupné z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4621285>
- [8] Traffic Sign Detection. Mobileye [online]. 2014 [cit. 2014-05-23]. Dostupné z: <http://www.mobileye.com/technology/applications/traffic-sign-detection/>
- [9] Hella Aglia - Front Camera. Hella Aglia [online]. 2010 [cit. 2014-05-23]. Dostupné z: <http://www.aglaia-gmbh.com/aglaia-com-en/579.html>
- [10] ŠKODA Octavia - Traffic Sign Recognition. Škoda [online]. 2013 [cit. 2014-05-23]. Dostupné z: <http://www.skoda.co.uk/models/HotspotDetail?HotspotName=New%20Octavia%20Estate%20-%20Traffic%20Sign%20Recognition&WebID=3e1f7ded-131f-435f-8be0-87500e18c873&Page=technology>
- [11] DANNHOFEROVÁ, Jana. Velká kniha barev: kompletní průvodce pro grafiky, fotografy a designéry. 1. vyd. Brno: Computer Press, 2012, 352 s. ISBN 978-80-251-3785-7.

- [12] Základní pojmy o grafice a rastrová grafika [online]. Brno, 2013 [cit. 2014-05-23]. Dostupné z:http://www.ped.muni.cz/wtech/03_studium/apg/APG_03.pdf. Studijní opory. MU.
- [13] HLAVÁČ, Václav a Miloš SEDLÁČEK. Zpracování signálů a obrazů. Vyd. 2. Praha: ČVUT, 2005, 255 s. ISBN 80-010-3110-1.
- [14] SOJKA, Eduard. Digitální zpracování a analýza obrazů [online]. 1. vyd. Ostrava: VŠB - Technická univerzita, 2000, 133 s. [cit. 2014-05-23]. ISBN 80-707-8746-5.
- [15] Jak číst v histogramu. Blog milujeme fotografii [online]. 2012 [cit. 2014-05-23]. Dostupné z: <http://www.milujemefotografii.cz/jak-cist-v-histogramu>
- [16] JOHANSSON, Björn. Road Sign Recognition from a Moving Vehicle. 2012.
- [17] HÁJOVSKÝ, Radovan, Radka PUSTKOVÁ a František KUTÁLEK. Zpracování obrazu v měřicí a řídicí technice [online]. Vyd. 1. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, 2012 [cit. 2014-05-23]. 1 DVD-ROM. ISBN 978-80-248-2596-0.
- [18] Jak číst v histogramu. Blog milujeme fotografii [online]. 2012 [cit. 2014-05-23]. Dostupné z: <http://www.milujemefotografii.cz/jak-cist-v-histogramu>
- [19] ZELINKA, Ivan. Umělá inteligence, aneb, Úvod do neuronových sítí, evolučních algoritmů--. Vyd. 2. Ve Zlíně: Univerzita Tomáše Bati, 2005. 127 s. Učební texty vysokých škol (Univerzita Tomáše Bati ve Zlíně. Technologická fakulta). ISBN 80-731-8277-7. Skripta. UTB Zlín.
- [20] MRÁZEK, MICHAL. ADAPTIVNÍ OPTIMÁLNÍ REGULÁTORY S PRINCIPY UMĚLÉ INTELIGENCE V PROSTŘEDÍ MATLAB [online]. Brno, 2008 [cit. 2014-05-23]. Dostupné z: <https://dspace.vutbr.cz/xmlui/handle/11012/12842>. Diplomová. VUT Brno.
- [21] BÍLA J.: Umělá inteligence a neuronové sítě v aplikacích, ČVUT, 1996, ISBN 80-01-01275-1.
- [22] Nikon D3000 – Uživatelská příručka. 2010.
- [23] Test digitální zrcadlovky Nikon D3000. In: KUPSA, Michal. Fotoaparát.cz [online]. 2010 [cit. 2014-05-23]. Dostupné z:<http://www.fotoaparát.cz/article/10874/1>

- [24] Samsung Galaxy S II: prostě ten nejlepší. Mobilmania.cz [online]. 2011 [cit. 2014-05-23]. Dostupné z: <http://www.mobilmania.cz/clanky/samsung-galaxy-s-ii-proste-ten-nejlepsi-recenze/uvod-konstrukce-design-vnejsi-pohled/sc-3-a-1316358-ch-1051748/default.aspx#articleStart>
- [25] CHRAMCOV, Bronislav. Základy práce v prostředí Mathematica. Vyd. 2. Ve Zlíně: Univerzita Tomáše Bati, 2006, 122 s. ISBN 80-731-8510-5.
- [26] Wolfram Mathematica Documentation Center. Wolfram Mathematica Documentation Center [online]. 2014 [cit. 2014-05-23]. Dostupné z: <http://reference.wolfram.com/mathematica/guide/Mathematica.html>
- [27] Wolfram Mathematica 8. Wolfram Mathematica 8 [online]. 2012 [cit. 2014-05-24]. Dostupné z: http://www.mathematica.cz/produkty.php?p_mathematica
- [28] Wolfram Mathematica 9. Wolfram Mathematica 9 [online]. 2014 [cit. 2014-05-24]. Dostupné z: <http://www.wolfram.com/mathematica/>
- [29] DETEKCE OBJEKTU VE VIDEOSEKVENCÍCH [online]. Brno, 2010 [cit. 2014-05-25]. Dostupné z: https://dspace.vutbr.cz/xmlui/bitstream/handle/11012/6156/LMST_SEBELA_ID23634.pdf?sequence=1. Diplomová. VUT Brno.
- [30] VINKLER, Tomáš. Rozpoznávání dopravních značek pomocí umělých neuronových sítí. Zlín, 2014. Diplomová. UTB Zlín. Obhajoba Červen 2014.
- [31] DOSPIVA, Jan. Systém využívající rozpoznávání dopravních značek pomocí umělých neuronových sítí pro hlášení ve značení. Zlín, 2014. Diplomová. UTB Zlín. Obhajoba Červen 2014.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AVI	Audio Video Interleave
CCD	Charge-coupled device
CMY(K)	Cyan Magenta Yellow (BlacK)
ČSN	České státní normy
DSLR	Digital Single-Lens Reflex
GIF	Graphics Interchange Format
GPS	Global Positioning Systém
HD	High-Definition
HLS	Hue Lightness Saturation
HSB	Hue Saturation Balance
HSV	Hue Sarution Value
HTML	HyperText Markup Language
HUD	Heads-up Display
ISO	Citlivost
JPG	Joint Photographic Experts Group
LCD	Liquid Crystal Display
MMS	Multimedia Messaging Service
MHL	Mobile High-Definition Link
MP4	MPEG-4 Part 14
PNG	Portable Network Graphics
RAM	Random Access Memory
RGB	Red Green Blue
SD	Secure Digital
SDHC	Secure Digital High Capacity

SMS	Smart Messaging Service
TIFF	Tagged Image File Format
USB	Universal Serial Bus
WM	Wolfram Mathematica

SEZNAM OBRÁZKŮ

Obr. 1: Ukázka dopravního značení v ČR[2]	12
Obr. 2: Umístění svislých dopravních značek vzhledem k vozovce[3].....	15
Obr.3: Umístění svislých dopravních značek vzhledem k jiným dopravním značkám[3]	16
Obr. 4: Kamery asistenčních systémů – vlevo Škoda, vpravo Opel[10]	18
Obr. 5: Reprezentace RGB barevného modelu[12]	20
Obr. 6: Reprezentace HSV barevného modelu [12]	21
Obr. 7: Reprezentace Lab barevného modelu[12]	22
Obr. 8: Ukázka jasového histogramu[14]	24
Obr. 9: Ukázka použití Gaussova filtru[18].....	25
Obr. 10: Jasové profily nejběžnějších hran[13]	26
Obr. 11: Reprezentace a ukázka dilatace a eroze[14]	28
Obr. 12: Dopředná neuronová síť[19]	31
Obr. 13: Algoritmus učení metodou Levenberg-Marquardt[21]	33
Obr. 14: Nikon D3000 s objektivem 18-105mm[23]	34
Obr. 15: Mobilní telefon Samsung Galaxy S2[24]	36
Obr. 16: Deformace dopravních značek	46
Obr. 17: Demonstrační snímky	48
Obr. 18: Ukázka a popis hlavní obrazovky algoritmu	50
Obr. 19: Ukázka výpisu informací k importovanému snímku.....	50
Obr. 20: Kanál 2 barevného modelu LAB při použití C# algoritmu	51
Obr. 21: Postup morfologických operací obou navržených algoritmů.....	52
Obr. 22: Kanál 2 barevného modelu LAB při použití wolfram algoritmu	53
Obr. 23: Kanál 3 barevného modelu LAB při použití C# algoritmu	54
Obr. 24: Kanál 3 barevného modelu LAB při použití wolfram algoritmu	54
Obr. 25: Kanál 2 barevného modelu HSB při použití C# algoritmu	57
Obr. 26: Kanál 2 barevného modelu HSB při použití wolfram algoritmu	57
Obr. 27: Kanály barevného modelu RGB při použití C# algoritmu.....	58
Obr. 28: Kanály barevného modelu RGB při použití wolfram algoritmu	59
Obr. 29: Vzhled zobrazení ostatních algoritmů	60
Obr. 30: Převod barevného vektoru na monochromatický	65
Obr. 31: Převod barevného vektoru na stupně šedi	65

Obr. 32: Deformace binarizací.....	66
Obr. 33: Použitý vektor do výsledného programu	67
Obr. 34: Separace piktogramů u kruhové DZ.....	68
Obr. 35: Porovnání binárních.....	68
Obr. 36: Použitý vektor do výsledného programu	69
Obr. 37: Separace piktogramů u	71
Obr. 38: Úplná separace piktogramů u kruhové DZ.....	72
Obr. 39: Postup aplikování metod zpracování obrazu	73
Obr. 40: Použitý vektor do výsledného programu	73
Obr. 41: Použitý vektor do výsledného programu - hierarchie.....	74
Obr. 42: Použitý vektor do výsledného programu	75
Obr. 43: Vývoj testovacího vektoru pro zákazové značky pro 3. a 4. Přípravu	76
Obr. 44: Ukázka barevného validačního vektoru	76
Obr. 45: Hlavní okno DVH recognizeru.....	77
Obr. 46: Image Recognizer	78

SEZNAM TABULEK

Tabulka 1: Parametry kategorií značek	13
Tabulka 2: Rozdíly ve značkách v zemích EU[3]	17
Tabulka 3: Kroky prvního stádia příprav dat	64
Tabulka 4: Kroky druhého stádia příprav	66
Tabulka 5: Kroky třetího stádia příprav dat	69
Tabulka 6: 1. dvě fáze třetího stádia příprav dat pro zákazové značky	69
Tabulka 7: 3.a 4. fáze třetího stádia příprav dat pro zákazové značky	70
Tabulka 8: 5.-7 fáze třetího stádia příprav dat pro zákazové značky	70
Tabulka 9: Ostatní fáze třetího stádia příprav dat pro zákazové značky	70
Tabulka 10: 1. a 2. fáze třetí přípravy dat pro příkazové značky	71
Tabulka 11: 3. fáze třetí přípravy dat pro příkazové značky	71
Tabulka 12: Rozměry a ořezy snímků ve 4. přípravě	72

SEZNAM PŘÍLOH

PI: Přesné nastavení hodnot pro ukázky dopravních značek

PII: Fáze učení neuronové sítě

PIII: CD s ukázkovými příklady

PŘÍLOHA P I: PŘESNÉ NASTAVENÍ HODNOT PRO UKÁZKY DOPRAVNÍCH ZNAČEK

<p>Zaznačení v obraze <input checked="" type="checkbox"/></p> <p>Váběr kanálu 2</p> <p>Barevný model LAB</p> <p>Typ algoritmu C_#</p> <p>Pravouhlost 12</p> <p>Kruhovost 12</p> <p>Minimální obdélník 20</p> <p>Ukázkový snímek 1</p>	<p>Zaznačení v obraze <input checked="" type="checkbox"/></p> <p>Váběr kanálu 3</p> <p>Barevný model LAB</p> <p>Typ algoritmu C_#</p> <p>Pravouhlost 14</p> <p>Kruhovost 14</p> <p>Minimální obdélník 22</p> <p>Ukázkový snímek 2</p>
<p>Zaznačení v obraze <input checked="" type="checkbox"/></p> <p>Váběr kanálu 3</p> <p>Barevný model LAB</p> <p>Typ algoritmu C_#</p> <p>Pravouhlost 15</p> <p>Kruhovost 15</p> <p>Minimální obdélník 42</p> <p>Ukázkový snímek 3</p>	<p>Zaznačení v obraze <input checked="" type="checkbox"/></p> <p>Váběr kanálu 2</p> <p>Barevný model HSB</p> <p>Typ algoritmu C_#</p> <p>Pravouhlost 13</p> <p>Kruhovost 12</p> <p>Minimální obdélník 27</p> <p>Ukázkový snímek 4</p>
<p>Zaznačení v obraze <input checked="" type="checkbox"/></p> <p>Váběr kanálu 2</p> <p>Barevný model HSB</p> <p>Typ algoritmu C_#</p> <p>Pravouhlost 13</p> <p>Kruhovost 12</p> <p>Minimální obdélník 27</p> <p>Ukázkový snímek 5</p>	<p>Zaznačení v obraze <input checked="" type="checkbox"/></p> <p>Váběr kanálu 2</p> <p>Barevný model LAB</p> <p>Typ algoritmu C_#</p> <p>Pravouhlost 18</p> <p>Kruhovost 18</p> <p>Minimální obdélník 27</p> <p>Ukázkový snímek 6</p>

Zaznačení v obraze

Váběr kanálu 2

Barevný model HSB

Typ algoritmu C++

Pravoúhlost 13

Kruhovost 13

Minimální obdélník 3

Ukázkový snímek 7

Zaznačení v obraze

Váběr kanálu 2

Barevný model LAB

Typ algoritmu C++

Pravoúhlost 12

Kruhovost 12

Minimální obdélník 20

Ukázkový snímek 8

Zaznačení v obraze

Váběr kanálu 2

Barevný model LAB

Typ algoritmu C++

Pravoúhlost 12

Kruhovost 12

Minimální obdélník 25

Ukázkový snímek 9

Zaznačení v obraze

Váběr kanálu 2

Barevný model LAB

Typ algoritmu C++

Pravoúhlost 12

Kruhovost 12

Minimální obdélník 24

Ukázkový snímek 10

PŘÍLOHA PII: FÁZE UČENÍ NEURONOVÉ SÍTĚ

