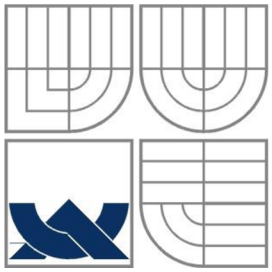
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

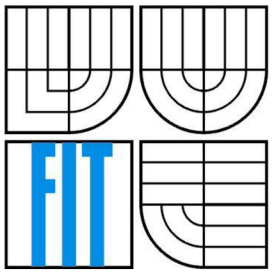
Fakulta informačních technologií
Faculty of Information Technology

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

Brno, 2016 Bc. Petr Huták



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

ROZPOZNÁVÁNÍ TOPOLOGICKÝCH INFORMACÍ Z PLÁNU KŘIŽOVATKY

TOPOLOGY RECOGNITION FROM CROSSROAD PLAN

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. Petr Huták

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. RNDr. Jitka Kreslíková, CSc.

BRNO 2016

Zadání diplomové práce/18244/2015/xhutak00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2015/2016

Zadání diplomové práce

Řešitel: **Huták Petr, Bc.**
Obor: Počítačová grafika a multimédia
Téma: **Rozpoznávání topologických informací z plánu křižovatky**
Topology Recognition from Crossroad Plan
Kategorie: Zpracování obrazu

Pokyny:

1. Nastudujte metody zpracování obrazu umožňující detekci obrazových segmentů v obraze a získávání jejich příznaků. Seznamte se s postupy získávání sémantických znalostí ze segmentovaného obrazu.
2. Na základě provedené analýzy a po konzultaci s konzultantem firmy Siemens, s.r.o., vyberte vhodné metody pro získávání topologických informací z plánu křižovatky v rastrovém obrazu.
3. Pořiďte anotovanou sadu dat.
4. Zvolte vhodné vývojové prostředí a po dohodě s vedoucí implementujte prototyp navržené aplikace.
5. Proveďte experimenty a vyhodnocení systému na anotované sadě dat vybrané po dohodě s vedoucí.
6. Zhodnoťte dosažené výsledky a diskutujte možnosti dalšího rozvoje vytvořeného produktu.

Literatura:

- Bradski, G., Kaehler, A.: *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, 2008, ISBN-13: 978-0596516130.
- Sonka, M., Hlaváč, V., Boyle, R.: Image Processing, Analysis, and Machine Vision, CL-Engineering, 2007, ISBN-13: 978-0495082521.

Při obhajobě semestrální části projektu je požadováno:

- Bez požadavků.

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

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

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

Vedoucí: **Kreslíková Jitka, doc. RNDr., CSc.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 25. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Tato diplomová práce se zabývá průzkumem, návrhem a tvorbou postupů pro rozpoznávání topologických informací z plánu křižovatky. Vysvětluje metody používané v oblasti zpracování obrazu za účelem segmentace obrazu, rozpoznávání objektů v obraze, popisuje existující přístupy zpracování map reprezentovaných rastrovými obrazy a cílové prostředí, do kterého bude praktická část práce integrována. Práce je zaměřena především na porovnání různých přístupů získávání příznaků z rastrových map křižovatek a určení jejich sémantického významu. Praktická část je realizovaná v jazyku C# s využitím knihovny OpenCV.

Abstract

This master‘s thesis describes research, design and development of system for topology recognition from crossroad plan. It explains the methods used for image processing, image segmentation, object recognition. It describes approaches in processing of maps represented by raster images and target software, in which the final product of practical part of project will be integrated. Thesis is focused mainly on comparison of different approaches in feature extraction from raster maps and determination their semantic meaning. Practical part of project is implemented in C# language with OpenCV library.

Klíčová slova

Topologie křižovatek, C#, OpenCV, EmguCV, AForge.NET, Houghova transformace, Morfologické operátory, Vektorizace map, Segmentace obrazu, MSER, detekce textu

Keywords

Topology of crossroad plan, C#, OpenCV, EmguCV, AForge.NET, Hough transform, Mathematical morphology, Map vectorization, Image segmentation, MSER, Text detection

Citace

Huták Petr: Rozpoznávání topologických informací z plánu křižovatky, diplomová práce, Brno, FIT VUT v Brně, 2016

Rozpoznávání topologických informací z plánu křižovatky

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Doc. RNDr. Jitky Kreslíkové, CSc., další informace mi poskytl Ing. Zdeněk Jurka a Mgr. Martin Procháska. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……

Bc. Petr Huták

24. května 2016

Poděkování

Rád bych poděkoval vedoucí mé diplomové práce Doc. RNDr. Jitce Kreslíkové, CSc., konzultantům Ing. Zdeňku Jurkovi a Mgr. Martinu Prochárkovi a společnosti Siemens za odborné vedení, rady a jejich čas, které mi při tvorbě práce poskytli.

Obsah

1 Úvod

Diplomová práce na téma Rozpoznání topologických informací z plánu křižovatky se zabývá analýzou rastrových plánů dopravních uzlů/křižovatek za účelem získání informací důležitých při návrhu plánů řízení dopravy na řadičích ovládajících světelnou signalizaci těchto křižovatek. Získané informace jsou využity coby vstupní informace pro software umožňující vytváření právě těchto plánů. Základní ideou je tedy usnadnění práce dopravním inženýrům, kdy namísto manuálního vyplňování údajů přicházíme s úplnou nebo alespoň částečnou automatizací a s ní spojenou časovou úsporou celého procesu. Práce byla zadána firmou Siemens, konkrétně centrálním úsekem Corporate Technology Development Center. Ten v České republice čítá dvě pracoviště, nacházející se v Praze na Stodůlkách a v Brně na Olomoucké ulici, skládá se z pěti oddělení, kde pracuje přes 100 zaměstnanců. Úkolem tohoto úseku je především vývoj hardwaru, softwaru a firmwaru pro interní divize společnosti v zahraničí (převážně v Německu). Ze zde vyvíjených produktů je možné jmenovat i software Sitraffic smartCore či Sitraffic Office, do kterých je plánováno možné budoucí začlenění výsledků praktické části práce.

Prvním úkolem v rámci práce na diplomovém projektu bylo nastudovat vhodné metody zpracování obrazu umožňující detekci obrazových segmentů v obraze a získávání jejich příznaků a dále pak seznámení se s postupy získávání sémantických znalostí ze segmentovaného obrazu. Následující fáze se týkala využití nastudovaných metod při návrhu systému pro získávání topologických informací v plánu křižovatky z rastrového obrazu, na což navazovala implementace jednotlivých metod. Za účelem testování byla pořízena anotovaná sada dat odpovídající výsledkům, které jsou požadovány od výsledného systému. Implementace byla provedena v jazyku C# s využitím knihoven OpenCV a AForge.NET. Po dokončení této fáze a otestování navrhnutých postupů na testovací sadě dat následuje vyhodnocení dosažených výsledků a popis dalšího možného vývoje, či možné integrace algoritmů do stávajících produktů firmy Siemens.

Celá práce je rozčleněna do 8 kapitol. Druhá kapitola (2) se ve stručnosti zabývá teoretickými podklady zpracování obrazu později využitých v následujících částech práce. Následující kapitola (3) je věnována podobným systémům, které slouží pro vektorizaci rastrových map a detekci vodorovného dopravního značení. Kapitola (4) dává náhled na software Sitraffic smartCore a SitrafficOffice společnosti Siemens, které jsou budoucími cílovými produkty, a do kterých může být v budoucnosti výsledek praktické části integrován. Pátá kapitola (5) je zaměřena na analýzu problému rozpoznání jednotlivých požadovaných informací z rastrového obrázku obsahujícího plán křižovatky, šestá (6) pak na návrh, implementaci a experimentování s navrženými metodami. V kapitole (7) je popsáno shrnutí dosažených výsledků a závěr lze nalézt v poslední kapitole (8), která pak dále popisuje nástin dalšího možného vývoje.

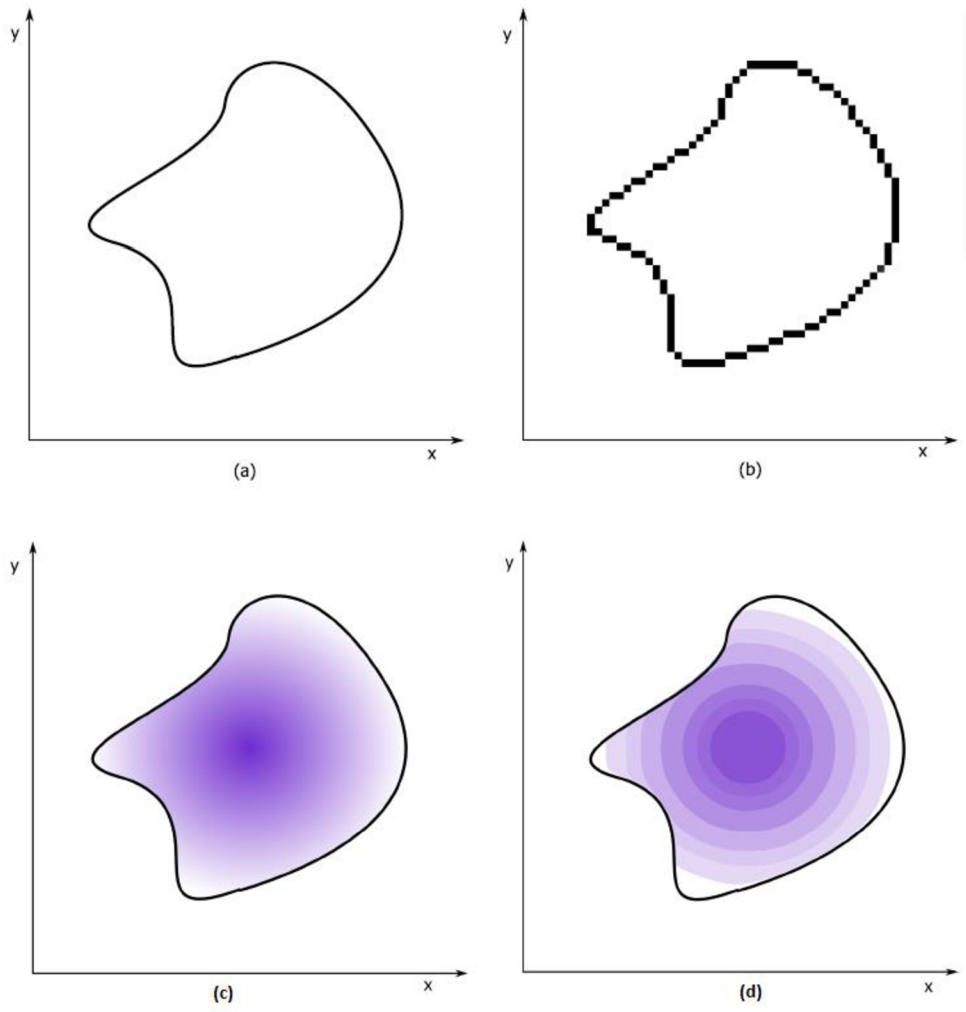
2 Metody zpracování obrazu

Tato kapitola se zabývá metodami, které byly použity při návrhu, rozboru a implementaci praktické části práce. Nalezneme zde metody pro předzpracování obrazu, segmentaci obrazu, detekci hran a detekci objektů, které jsou pro výsledek práce důležité.

2.1 Rastrový obraz

Rastrový (bitmapový) obraz je jako celek popsán pomocí bodů reprezentujících jednotlivé pixely. Tyto body jsou uloženy ve struktuře, která odpovídá dvourozměrné mřížce. Každý bod má svoji specifickou polohu a nese informaci o barvě, která je dána použitým barevným modelem. Kvalita zobrazení rastrového obrazu je dána především rozlišením (rozměry mřížky) a barevnou hloubkou.[1]

Reprezentace obrazu v počítačích je omezena diskrétním prostorem. Z toho plyne, že pokud máme na vstupu spojitý obraz a chceme ho převést do rastrové reprezentace, je nutné použít pro převod vhodnou transformaci. K tomu využíváme vzorkování a kvantizace.[11]



Obr. 1 Příklady vzorkování (nahoře) a kvantizace (dole) [11]

- **Monochromatický obraz –** obraz, kde informace pixelu o barvě nabývá pouze dvou úrovní, např. černé a bílé. Velmi často je označován také jako binární obraz.
- **Šedotónový obraz –** pixely obrazu obsahují informace o barvě reprezentované jednou barevnou složkou nabývající až 256 různých hodnot. To odpovídá 8 bitové barevné hloubce, a tedy pro každý obrazový bod je potřeba 1 Bytu paměti.
- **Barevný obraz –** v případě barevné reprezentace obrazu obsahuje každý pixel tři informace o barvě, kterou nese. Tyto informace odpovídají jednotlivým barevným složkám (nejčastěji červené, zelené a modré – model RGB), kdy výsledný obraz získáme prostým aditivním složením.

Obr. 2 Zleva monochromatický (a), Šedotónový (b), barevný (c) obraz [11]

2.2 Morfologické operátory

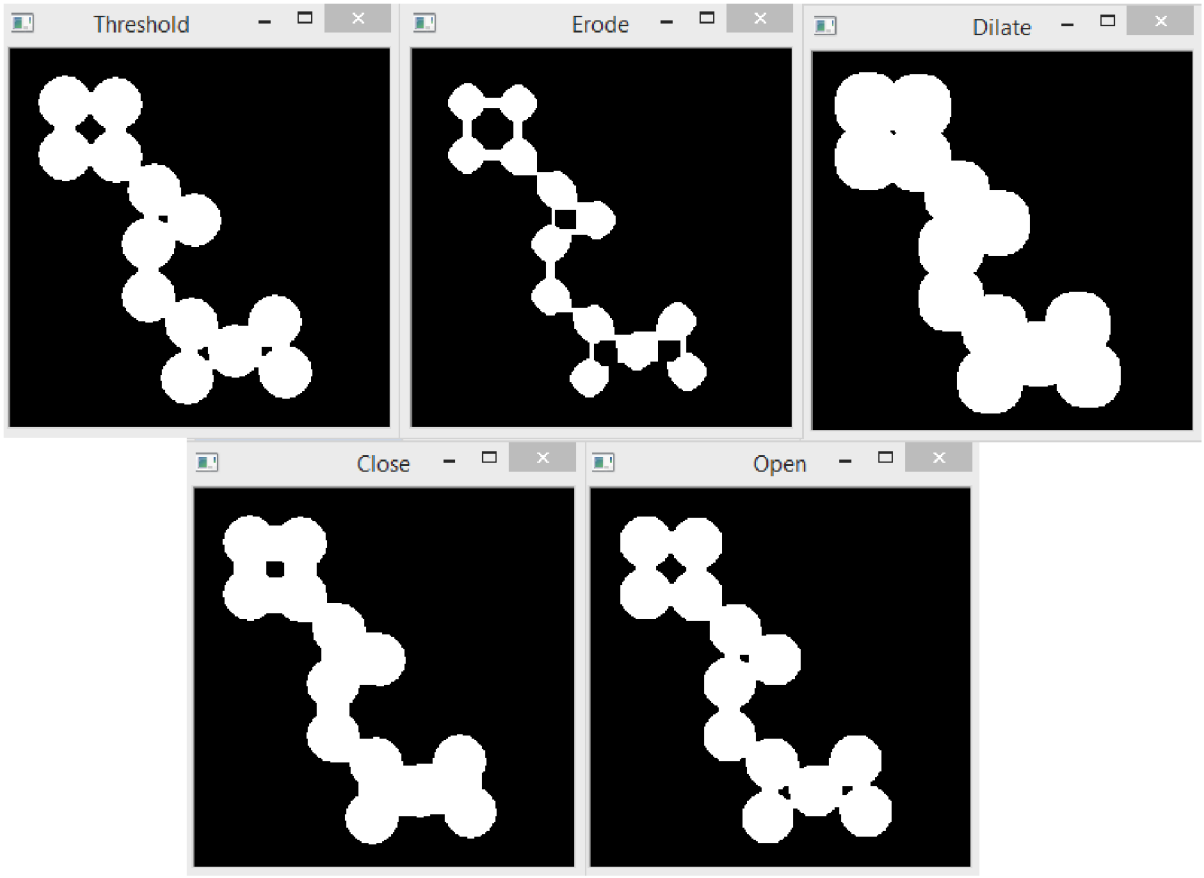
Matematická morfologie je založena na vlastnostech bodových množin. Její nejčastější využití nalezneme u binárních či šedotónových obrazů (je možné ji zobecnit i na obrazy barevné) a to u operací určených pro předzpracování obrazu (odstranění šumu, zjednodušení tvarů), zvýraznění struktury elementů (kostra, ztenčení, zesílení, obálka, izolace/spojení objektů), extrakce obrazových komponent, nalezení děr v obraze, nalezení obrazových přechodů, či pro popis elementů daných jinými číselnými charakteristikami (např. popis tvaru, plochy, atd.). [3] [4]

Jak už bylo řečeno, morfologické operace jsou založeny na teorii množin a jako takové jsou realizovány pomocí relace vstupního obrazu s jinou menší množinou, kterou nazýváme strukturní element. Výsledná transformace odpovídá odezvě systematického pohybu strukturního elementu po vstupním obrazu a to podle zvoleného typu operace. Základním typem jsou eroze a dilatace. Následuje jejich popis společně s popisem dalších, složitějších morfologických operací.

- **Dilatace –** Dilatace je operace, jejímž výsledkem je zvýraznění objektů s vysokým jasem.
- **Eroze –** Jedná se o opačnou operaci k operaci dilatace. Zmenšuje velké objekty, malé pak úplně odstraňuje. To, zda je objekt malý či velký, a tedy či bude odstraněn, je dáno velikostí použitého

strukturního elementu. Operace je vhodná pro odstranění šumu a zpřesnění/zvýraznění detekovaných hran.

- **Otevření** – jde o kombinaci operací eroze a dilatace. Nejdříve je použita operace eroze, po které následuje dilatace. Otevření se používá za účelem oddělení oblastí v obraze při zachování jejich struktury, to se hodí např. při potřebě spočítat oblasti v binárním obraze.
- **Uzavření** – opačná operace k otevření je operace uzavření. Nejdříve je použita operace dilatace a následuje operace eroze. Uzavření se využívá při potřebě spojit blízko položené objekty. Jak otevření, tak uzavření jsou výsledkem velmi podobné samotným dílčím operacím dilatace a eroze. Obdržený výsledek je však přesnější, operace jsou méně destruktivní.
- **Morfologický gradient** – tato operace je výsledkem rozdílu operace dilatace a eroze daného obrazu. Po její aplikaci je možné detekovat/izolovat/zvýraznit hrany existujících objektů v obrazu.
- **Top Hat -** operace Top Hat je výsledkem odečtení otevření od zdrojového obrazu. Tímto způsobem je možné detekovat oblasti s vyšším jasem než je jejich okolí – toho lze využít např. při detekci prasklinek či kapek ve zdrojovém obrazu.
- **Black Hat** – operace Black Hat je výsledkem odečtení zdrojového obrazu od uzavření. Takto lze naopak zvýraznit oblasti s nižším jasem než je jejich okolí.



Obr. 3 Příklady morfologických operátorů – zleva nahoře vstupní obraz, eroze, dilatace, zleva dole uzavření, otevření

2.3 Segmentace obrazu

Segmentace obrazu je jedním ze stěžejních úkolů automatického zpracování obrazu. Ve své podstatě se jedná o rozčlenění vstupního obrazu na celky, které by měly představovat každý jeden celý objekt obrazu. Z principu samotné segmentace je téměř nemožné obecně dosáhnout dokonalé a stoprocentní segmentace, vždy záleží na konkrétním případu, volbě vhodného algoritmu a zejména apriorních znalostech o vstupních datech – musíme vědět, co konkrétně nás v obrazu zajímá. Samotná segmentace je proces, který je složen z několika postupných kroků. Nejdříve je většinou nutné vstupní obraz vhodným způsobem předzpracovat – odstranit šum, či malé nepodstatné objekty, které nás nezajímají. Následuje samotná segmentace. Po této fázi již máme segmentovaný obraz. Výsledek však málokdy odpovídá požadovanému cíli, je tedy nutné provést dodatečné úpravy – většinou spojujeme příliš malé segmenty do jednoho celku nebo naopak některé celky potřebujeme dodatečně oddělit.

V následujících odstavcích jsou popsány metody segmentace obrazu pomocí prahovacích funkcí a hledání oblastí v obraze (Watershed algoritmus). [5]

2.3.1 Prahování

Prahování lze označit za jednu z nejstarších a nejjednodušších segmentačních metod, i přesto však jde o velice použitelný a oblíbený způsob, jak přistupovat k segmentaci v rámci jednoduchých zadání. Lze jej použít jak samostatně, tak jako část pokročilejších metod. Jeho výhodou je snadná implementace i velmi malá časová náročnost výpočtů. Základní myšlenka spočívá v předpokladu, že objekty a pozadí mají rozdílné úrovně intenzity. Na základě definovaného prahu pak dělíme pixely na ty, které mají menší hodnotu než práh a ty, které mají hodnotu větší. Po jediném průchodu algoritmu tak získáváme objekty rozdělené do disjunktních částí s popisem co je objekt a co pozadí. Samotná segmentace se pak dokončí popisem jednotlivých částí, čímž získáme popis každého z objektů. Jediným průchodem algoritmu tak získáváme výsledek segmentace [1].

Vybrané způsoby prahování:

- **Globální prahování –** základní verze prahování, vyjádřena vztahem:

$$\begin{matrix} f(i,j) = 1 & \text{je-li } g(i,j) \geq T \\ 0 & \text{jinak} \end{matrix} \tag{2.1}$$

- **Procentní prahování –** hodnota prahu není explicitně určena jako úroveň šedi, ale jako procentní zastoupení bodů, které jsou rovny danému vhodnému prahu.
- **Poloprahování** – od základní verze prahování se liší tím, že pixely nabývající vyšších hodnot, než je hodnota prahu, si ponechají svoji hodnotu, zatímco pixely nabývající hodnot nižších jsou vynulovány.
- **Adaptivní prahování –** při tomto druhu prahování je obraz rozdělen do několika částí, nejčastěji čtverců nebo obdélníků a prahování je provedeno pro každou takovou část zvlášť. Výhodou je lepší chování prahování v případě různorodého osvětlení, nevýhodou pak skokové a neurčité chování na hranicích jednotlivých prahovaných částí.

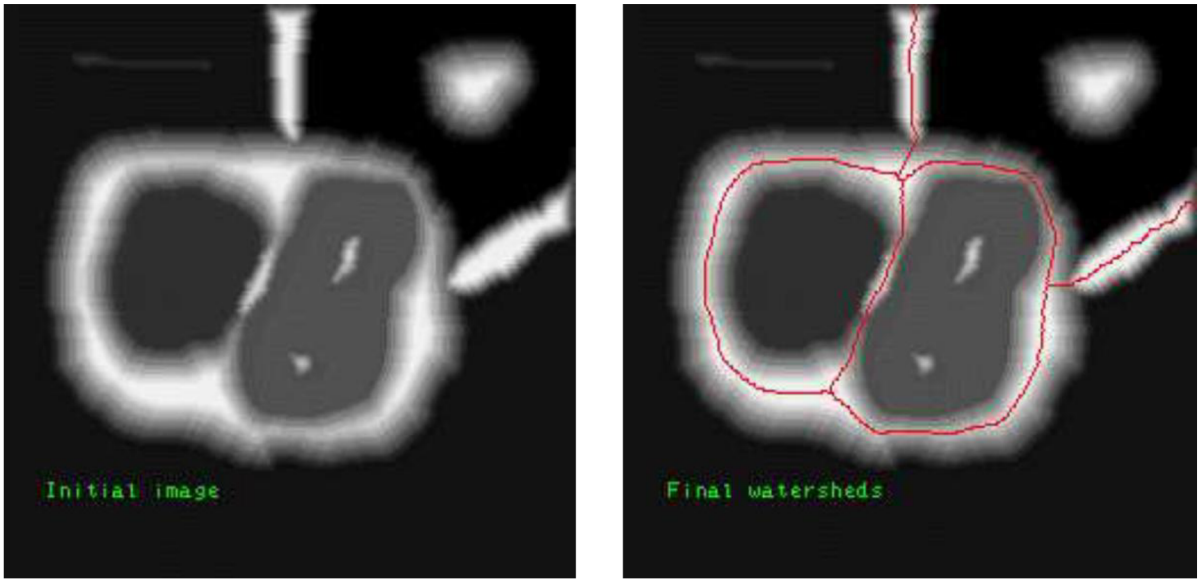
- **Vícestupňové prahování** – označované za multitreshold rozděluje obraz na více než dvě množiny.

$$\begin{aligned} f(i,j) &= 1 \quad \text{je-li } g(i,j) \geq T_1 \& g(i,j) < T_2 \\ f(i,j) &= 2 \quad \text{je-li } g(i,j) \geq T_2 \& g(i,j) < T_3 \\ f(i,j) &= 3 \quad \text{je-li } g(i,j) \geq T_3 \& g(i,j) < T_4 \\ &\vdots \\ f(i,j) &= n \quad \text{je-li } g(i,j) \geq T_n \end{aligned} \tag{2.2}$$

- **Hysterezní prahování** – double treshold, úprava základní verze prahování přidáním druhého prahu. Objektem je označena oblast která splňuje podmínky obou prahů, např. pixely nabývají hodnot vyšších než je první práh a zároveň nižších než je práh druhý.

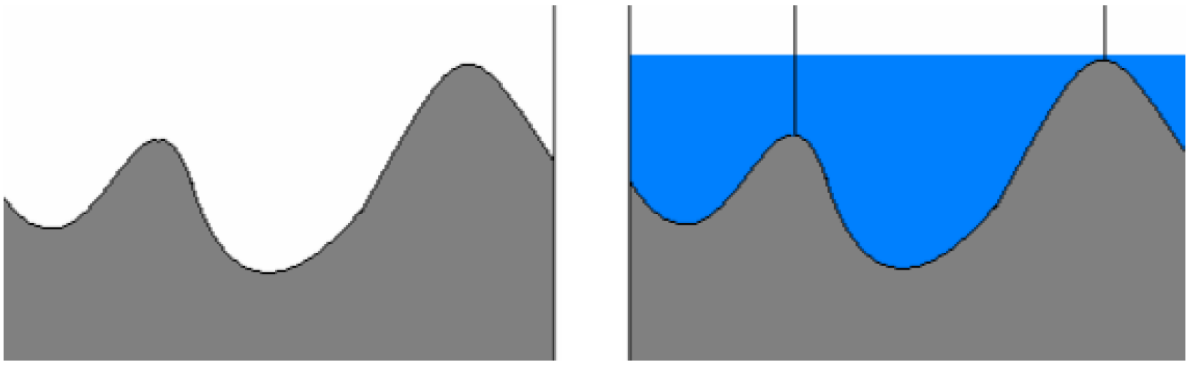
2.3.2 Watershed algoritmus

Algoritmus *Watershed* transformace (*watershed* = rozvodí, povodí či vodní předěl) [5] patří mezi region-based segmentační přístupy. Tato metoda segmentace je založena na myšlence pocházející z geografie. Obraz je chápán jako topografická mapa, který je postupně zaplavována vodou.



Obr. 4 Příklad nalezení hranice mezi objekty pomocí watershed algoritmu [5]

V místech, kde by se voda ze dvou různých rozvodí mohla slít v jedno, jsou vytvořeny hráze (*dams*). Zaplavování pokračuje tak dlouho, až dosáhneme nejvyššího bodu terénu (globálního maxima obrazu). Výsledkem je obraz rozdělený do regionů, jednotlivé regiony jsou odděleny hrázemi. Vzniklé hráze jsou nazývány *watershed lines*, nebo jednodušeji *watersheds*. Každé rozvodí je označeno unikátním indexem, hráze jsou pak označeny speciálním indexem odlišným od ostatních.



Obr. 5 Ukázka zaplavování a tvorby hrází algoritmu watershed [6]

Algoritmus je velmi citlivý na lokální minima, z toho pak plyne i citlivost na šum. Velice často se stává, že výstupem je přesegmentovaný obraz. Toto chování lze potlačit předzpracováním obrazu - konkrétně odstraněním šumu a případných nežádoucích detailů, např. aplikací mediánového filtru. Nebo následným slučováním blízkých regionů.[6]

2.4 Houghova transformace

Oblast zpracování obrazu se velice často zabývá problémem nalézt základní tvary typu přímky, kružnice či elipsy. Kromě hranových detektorů, které jsou často díky chybějícím informacím a nedokonalostem v obrazu ne příliš vhodné, existuje metoda Houghovy transformace.[9]

Princip metody je poměrně jednoduchý, postupným hlasováním v rámci určitých tříd tvarů nacházíme nedokonalé instance hledaných objektů. Toto hlasování probíhá v prostoru parametrů, které získáme coby lokální maxima z tzv. akumulačního prostoru, který je algoritmicky zkonstruován během Houghovy transformace.

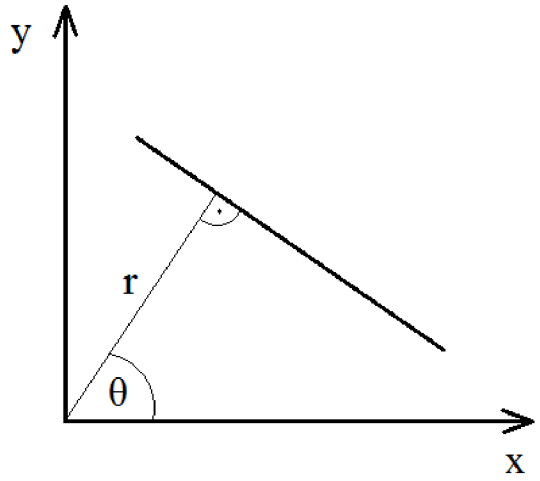
Klasická podoba Houghovy transformace byla navržena pro detekci přímek, později však byla rozšířena a zobecněna pro detekci jakýchkoliv (i složitějších) tvarů – tedy i kružnic, či elips. Tato rozšířená verze byla představena pány Richard Duda a Petr Hart v roce 1972 jako "generalized Hough transform".[2]

2.4.1 Houghova transformace pro detekci přímek

Základní myšlenkou Houghovy transformace je převod z kartézského souřadného systému do parametrického prostoru, v kterém jsou přímky definovány. Uvažujme případ, kdy máme přímku v obraze. Ta je dána následující rovnicí:

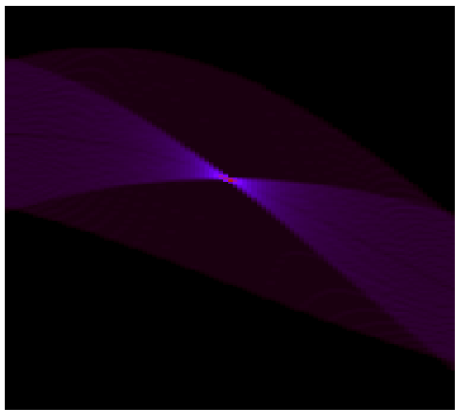
$$x \cdot \cos\theta + y \cdot \sin\theta = r \qquad (2.3)$$

kde úhel určuje odchylku od normály vedené směrem od středu souřadného systému k přímce a osou x, r pak udává vzdálenost mezi středem souřadného systému a přímkou, viz obrázek. [1] [7]



Obr. 6 Zobrazení parametrů přímky – úhlu a vzdálenosti od středu souřadného systému [9]

Pokud pak dosadíme souřadnice některého z bodů tvořící přímku do zmíněné rovnice a vykreslíme všechny možné hodnoty úhlu a vzdálenosti, vzniká v akumulačním prostoru spojitá křivka. Vykreslením všech bodů ležících na přímce do akumulačního prostoru lze vypozorovat, že se protínají v jednom jediném bodě, který představuje hledané parametry dané přímky.



Obr. 7 Akumulační prostor Houghovy transformace [7]

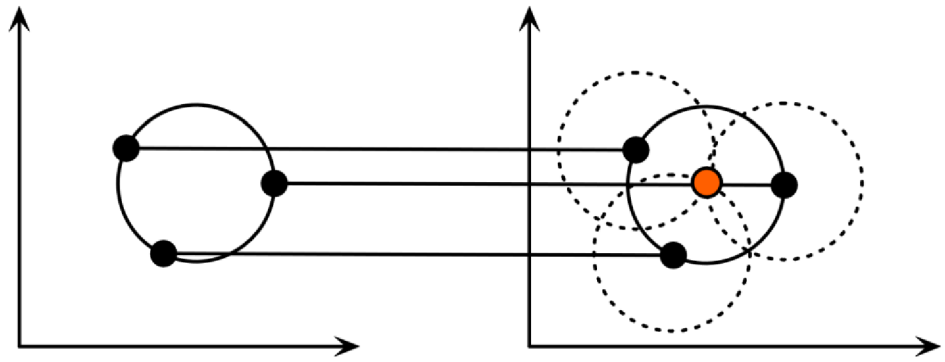
2.4.2 Houghova transformace pro detekci kružnic

Kružnice jsou častým geometrickým útvarem, který nás zajímá v aplikacích zaměřených na počítačové vidění a právě Houghova transformace může být použita k určení parametrů kružnice. Kružnici danou poloměrem a středem lze vyjádřit pomocí parametrických rovnic následovně:

$$\begin{aligned} x &= a + R\cos(\theta) \\ y &= b + R\sin(\theta) \end{aligned} \tag{2.4}$$

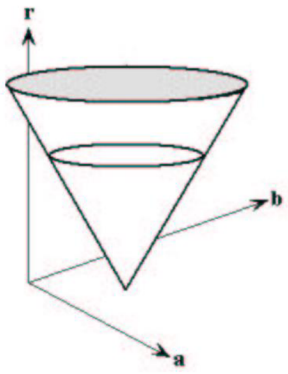
Pokud se pak v daném úhlu a v dané vzdálenosti nachází nějaké body dané souřadnicemi x, y, tvoří obvod kružnice. V okamžiku kdy obraz obsahuje více bodů splňující tuto podmínku, algoritmus pomocí nich dokáže určit parametry hledané kružnice. Fakt, že prohledávaný prostor je trojrozměrný (hledáme tři parametry), pak dělá Houghovu transformaci pro detekci kružnic v porovnání s detekcí přímek mnohem více výpočetně i časově náročnou.

Za předpokladu že známe poloměr hledaných kružnic, lze prohledávaný prostor omezit na dvourozměrný a cílem nám zůstává nalézt pouze souřadnice středu. Uvažujme, že každý bod v geometrickém prostoru je promítnut coby střed kružnice s daným poloměrem do prostoru parametrického. Potom střed hledané kružnice je v místě průsečíku takto promítnutých kružnic. [8]



Obr. 8 Promítnutí bodů ležících na kružnici z prostoru geometrického do parametrického [8]

V případě, že poloměr hledaných kružnic neznáme, je prohledávaný prostor trojrozměrný a parametrický prostor je promítnut na kužel, kdy body obvodu kružnice tvoří jeho povrch. Hledané parametry kružnice pak odpovídají místu v akumulačním prostoru, kde se protíná nejvíce kuželů. Následující nákres zobrazuje generování povrchu kuželu v parametrickém prostoru. Kužel je tvořen kružnicemi o různých poloměrech na každém stupni.



Obr. 9 Trojrozměrný parametrický prostor tvořený kuželem [8]

2.5 Detekce a klasifikace objektů v obraze

Tato kapitola ze začátku představuje základní pojmy, se kterými se při tvorbě detektoru objektů v obraze můžeme setkat. Dále popisuje několik vybraných metod používaných při detekci objektů v obraze.

Obecně můžeme metody detekování objektů v obraze rozdělit na metody detekující podle textury vstupního obrazu a na metody detekující podle tvaru detekovaného vzoru. Mezi metody detekující podle textury řadíme např. detektory detekující podle barvy hledaného objektu nebo detektory využívající přístupu LBP[17]. Metody detekující podle tvaru hledaného objektu využívají detektory detekující na základě natrénovaného klasifikátoru, přičemž metodou pro trénování klasifikátoru může být např. metoda AdaBoost[19].

V této kapitole je uveden pouze zlomek metod používaných pro detekci a klasifikaci objektů v obraze, pokrývá tak především metody, které byly v rámci experimentální a implementační fáze práce uvažovány nebo použity.

2.5.1 Klasifikátor

Rozdělení vstupních dat do n různých tříd má na starosti algoritmus, který se nazývá klasifikátor [20]. Klasifikátory se podle hodnoty n dělí na:

- Binární – $n=2$
- Vícehodnotové – $n > 2$

Binární klasifikátory obvykle vrací hodnoty -1 a +1. Zatímco první hodnota označuje třídu, která popisuje data neobsahující hledaný vzor, druhá hodnota označuje třídu dat, která hledaný vzor obsahují. Data, která hledaný vzor obsahují, se nejčastěji označují jako pozitivní vstupní data. Naopak data, která hledaný vzor neobsahují, se označují pojmem negativní vstupní data.

Vícehodnotové klasifikátory vrací n hodnot v předem definovaném rozsahu. Z toho plyne, že každé třídě tak odpovídá jiná hodnota a to podle toho, jakým způsobem odpovídá míře pravděpodobnosti, že vstupní data z této třídy obsahují hledaný vzor. Počáteční hodnota definovaného rozsahu odpovídá nulové pravděpodobnosti, poslední hodnota pak pravděpodobnosti stoprocentní – vstupní data tedy v této třídě obsahují hledaný vzor. Jinak řečeno čím větší hodnotu klasifikátor pro vstupní data vrátí, tím vyšší je pravděpodobnost, že se hledaný vzor vyskytuje v těchto datech.

Klasifikátory obvykle nevyužívají jako vstup původní reprezentaci vstupních dat. Nejčastěji se data převedou na z nich vyextrahované příznaky, např. Haarovy příznaky či LBP. Takto extrahovaných příznaků může být obecně m. Klasifikátor pak musí vyhodnotit m vstupů, resp. Vstupní vektor o velikosti m. Pro tento vstupní vektor vrátí jeden výstup. Volba příznaků, extrahovaných z původních dat, je velmi důležitá. Ovlivňuje totiž výslednou funkci klasifikátoru, tudíž je této volbě důležité věnovat při návrhu klasifikátoru patřičnou pozornost.

2.5.2 Návrh a učení klasifikátoru

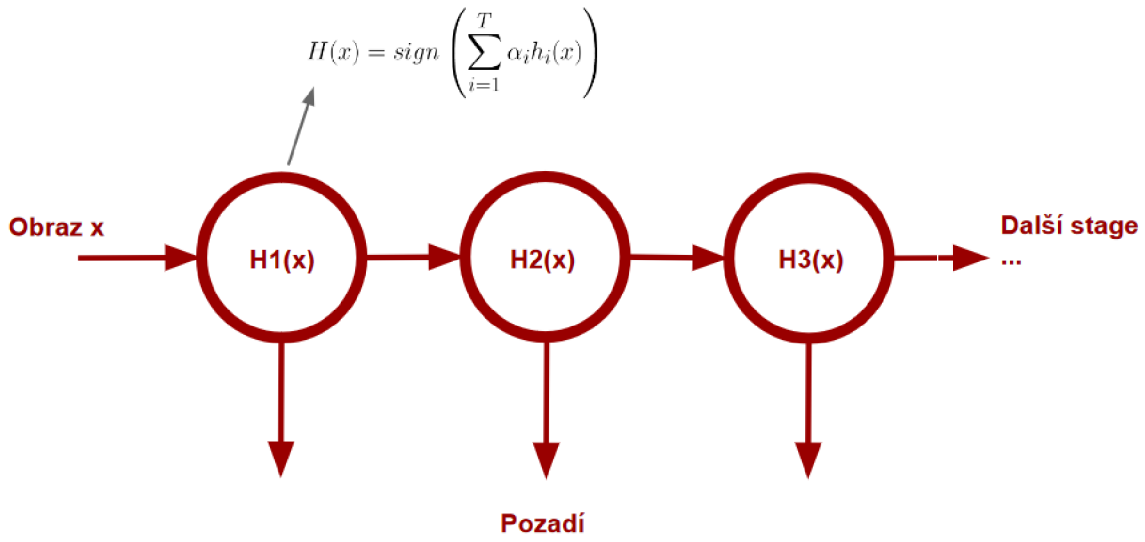
Aby klasifikátor mohl správně klasifikovat obecná vstupní data, je ho nejdříve zapotřebí naučit, resp. natrénovat, co a jak má klasifikovat [20]. Klasifikátor se trénuje na množině trénovacích dat. Její kvalita je důležitá pro kvalitu klasifikování a tím pádem i celého detektoru. Pokud vybíráme vhodná data, tak platí, že čím rozsáhlejší trénovací množina dat je, tím kvalitněji je klasifikátor natrénovaný. S tím je však spojená i vyšší časová náročnost. Pro učení klasifikátorů se používají dva obecné přístupy:

- **Učení bez učitele** – využívá vstupních dat, u kterých není předem známo, zda jde o data pozitivní či negativní
- **Učení s učitelem** – u vstupních dat předem víme, zda obsahují hledaný vzor

Varianta učení s učitelem se v praxi používá častěji. Navíc se kromě trénovací sady dat využívá i sady dat testovacích, na které je pak možné funkcionalitu klasifikátoru otestovat a určit jeho chybovost.

2.5.3 Kaskáda klasifikátorů

Kaskádové zapojení klasifikátorů je technika vycházející ze skutečnosti, že v hledaných datech se vyskytuje často mnohem méně oblastí obsahujících hledaný vzor než oblastí, jenž hledaný vzor neobsahují. Jde o techniku, kdy rozdělíme silný klasifikátor na několik dílčích klasifikátorů, z nichž každý je tvořen kombinací slabších klasifikátorů. Nově vzniklé klasifikátory jsou navzájem propojeny a tvoří tak kaskádu [16].



Obr. 10 Kaskáda klasifikátorů [16]

Obecně se při tvorbě kaskády postupuje tak, že první klasifikátor v kaskádě má jen několik slabých klasifikátorů a čím dále pokračujeme v kaskádě, tím je počet slabých klasifikátorů větší. Struktura kaskády se pak využívá jak při trénování výsledného klasifikátoru tak i při jeho použití pro detekci. První fáze kaskády je trénována pomocí celé sady trénovacích vzorků. Následně je natrénovaný

klasifikátor aplikován na trénovací vzorky a ty, které nezamítne, se použijí při trénování další fáze kaskády. Takto se pokračuje v trénování celé kaskády, kdy přitom dochází k zpřesňování natrénovaných slabých klasifikátorů.

Výhodou kaskády klasifikátorů je především rychlost, s jakou pracují při detekci. První fáze sice obsahuje pouze slabé klasifikátory, ale je schopná poměrně rychle rozhodnout o přítomnosti hledaného vzoru ve vstupních datech. Pokud pak chceme výsledek zpřesnit, pošleme data do další fáze, která je sice pomalejší než první, ale je přesnější a pracuje s méně vstupními daty díky tomu, že první fáze již některé vyřadila. Počet takto zapojených klasifikátorů může být obecně n, přičemž míra tolerance k negativním vstupním datům se s narůstajícím stupněm kaskády snižuje.

2.6 Detekce textu v obraze – metoda MSER

Metoda maximálně stabilních extrémních regionů (Maximally Stable External Regions - MSER) [21] spadá do oblasti blob detektorů. Blob detektory je obecně možné rozdělit na dvě skupiny:

- Metody založené na první derivaci – dochází k vzájemnému porovnání intenzity bodů a určujícím faktorem je rozdíl.
- Metody založené na lokálních extrémech intenzit – hledá se lokální maximum a minimum (v podstatě matematická druhá derivace)

Původně tyto algoritmy vznikly pro potřeby geografie, avšak později se usoudilo, že i obraz je možné chápat jako reliéf krajiny, čehož se nyní využívá pro potřeby v počítačovém vidění.

Metoda MSER byla představena v roce 2002 Jiřím Matasem jako rychlá metoda řešící korespondenci dvou obrazů pořízených z rozdílného úhlu. Metodou se detekují oblasti významných bodů nebo regionů v obraze, které jsou tmavší nebo světlejší než jejich okolí. Tyto extrémní regiony jsou definovány intenzitou v regionu a na jejich vnější hranici a jsou tedy tvořeny plochami pixelů z prahovací funkce s malou změnou napříč různými prahy – počet prahů, kdy je region stále stabilní, je pak nazýván rozpětím regionu.

Vlastní algoritmus pro výpočet MSER regionů je následující: zkoumaný obraz převedeme na obraz ve stupních šedi nebo provádíme výpočet pro každý barevný kanál zvlášť. Všechny body obrazu seřadíme vzestupně podle intenzity, body následně procházíme a vkládáme do nového obrazu od nejvyšší intenzity v sestupném nebo vzestupném pořadí. Body, které splňují podmínky prahování, tvoří extrémní regiony. Případné spojení regionů předurčuje výskyt menšího regionu, který je ve vzniklém obsažen. Složitost tohoto algoritmu je $O\ (n \log \log n)$. Výstup detektoru je reprezentován polohou minimálních místních intenzit a hodnotou prahu.

U mnoha obrazů je metoda stabilní ve velkém rozsahu prahových hodnot v určitých regionech. Tyto regiony jsou pak zkoumanými oblastmi a mohou splňovat:

- Stabilita extrémních regionů, které jsou neměnné pro více prahových hodnot
- Kovariance na přilehlost
- Invariance na afinní transformaci obrazu intenzit
- Detekce v maximálním měřítku
- Množina všech extrémních regionů může být vyčíslena složitostí $O\ (n \log \log n)$

Obr. 12 Ukázka výstupu MSER detektoru. Nalevo jsou zobrazeny nalezené regiony. Napravo je detail elips nalezených MSER regionů [21]

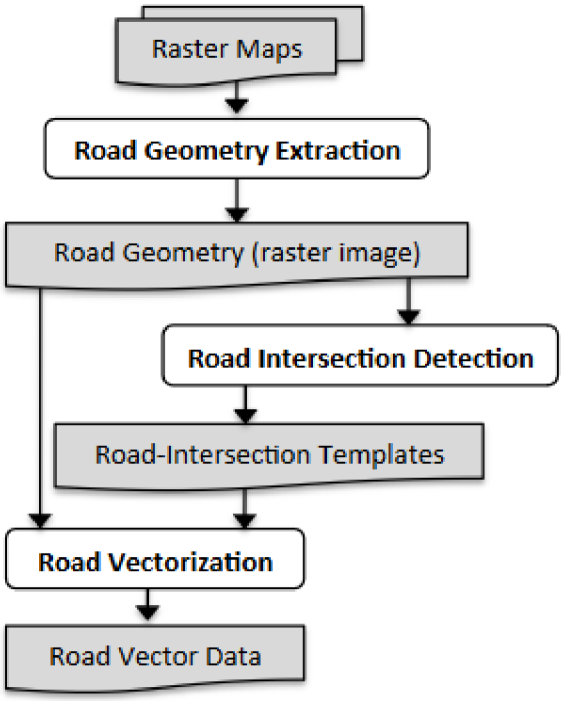
3 Stávající situace, podobné systémy

V rámci počátečního průzkumu byla snaha nalézt již existující podobná řešení, která by alespoň částečně svým obsahem odpovídala zadání. Již při zadávání práce bylo zmíněno, že podobné řešení v rámci takto úzké problematiky, jako je dopravní inženýrství zabývající se řízením křižovatek, nemusí existovat a práce tedy bude zejména o prozkoumání možností jak k danému problému efektivně přistoupit.

Většina stávajících prací zabývajících se zpracováním rastrových map je spojena s digitalizací a vektorizací mapových podkladů a to buď za účelem čisté archivace nebo spojení se systémy GIS (Geographic Information System) [10]. Další zkoumanou oblastí byly systémy zabývající se detekcí a rozpoznáním vodorovného dopravního značení. Následující text se tedy bude zabývat především těmito přístupy.

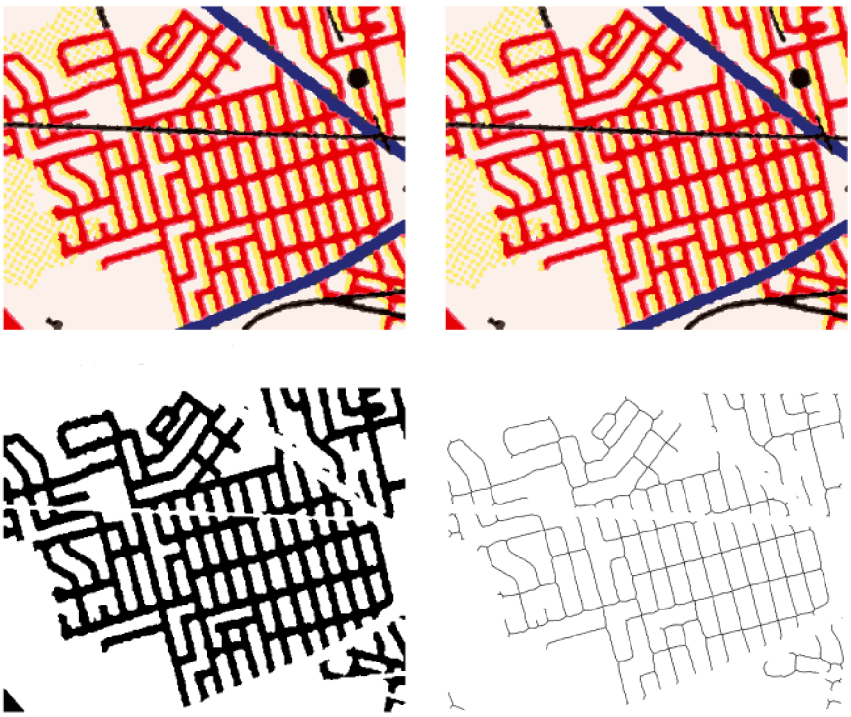
3.1 Vektorizace map

Vektorizace map v kontextu GIS se zabývá tvorbou vektorové reprezentace vybraných prvků vyskytujících se na mapách, příkladem může jít o komunikace, vodní toky, geografické jednotky jako jsou města a obce, či jiné geologické a dokumentační body. Tyto prvky jsou převedeny do vektorové reprezentace v podobě bodů, linií nebo polygonů (ploch), přičemž pro každý prvek je nutné zvolit vhodný typ, například vodní toky a komunikace lze reprezentovat pomocí linií, dokumentační tělesa zase užitím bodů, atd. Hlavním účelem vektorizace je možnost lepší analýzy poskytnutých dat, než je tomu v případě map reprezentovaných rastrem. Lze pak data vyhodnocovat, vytvářet statistiky, filtrovat a to jak podle jejich polohy tak samotných atributů jednotlivých objektů. I samotné automatizované vyhledávání, či navigace v elektronické mapě jsou dány díky převedení map do vektorové reprezentace, logického propojení jednotlivých prvků a jejich strojovému zpracování.



Obr. 13 Schéma vektorizace map [12]

V rámci této kapitoly a jako inspirace pro samotnou praktickou část byly prostudovány texty autorů Ojaswa Sharma - A methodology for Raster to Vector Conversion of Colour Scanned Maps [11] a Yao-Yi Chiang, Craig A. Knoblock - General Approach for Extracting Road Vector Data from Raster Maps [12]. Tyto texty pojednávají o postupech a metodách pro vektorizaci map (na obr. 13 je možné shlédnout schéma vektorizace map z [12]). A i když jsou především zaměřeny na rozpoznávání objektů na základě jejich barevné informace, poskytují přehled zajímavých technik, hlavně pak co se týká předzpracování obrazu. Jsou zde popsány morfologické operátory, detektory hran, Houghovy transformace, segmentace obrazu, skeletonizace obrazu (obr. 14) i rozšíření klasických algoritmů a jejich užití v praxi.



Obr. 14 Ukázka skeletonizace barevné mapy [11]

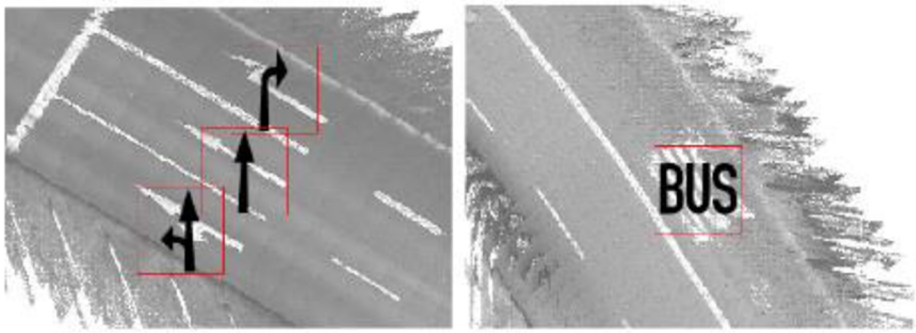
3.2 Detekce vodorovného dopravního značení

V rámci průzkumu aplikací pro detekci dopravního značení stojí za povšimnutí především práce, které se zabývají detekcí vodorovného dopravního značení. Tyto projekty většinou vznikají za účelem podpory řízení automobilů, kdy je takto získané informace možné dále použít za účelem navigace, autonomního řízení, či zvýšení bezpečnosti během přepravy.

Algoritmy používané v těchto aplikacích jsou obvykle postaveny na skutečnosti, že zpracovávají snímky pořízené před vozidlem pod určitým úhlem nebo přímo pod vozidlem (tedy pod kolmým úhlem). Snímky takto získaného vodorovného dopravního značení jsou pak velmi podobné snímkům dopravního značení, které je možné získat z plánků křižovatek. Z nastudovaných materiálů je možné zmínit práce[22][23], které se zabývají detekcí jízdních pruhů a směrových šipek. Je z nich možno

čerpat jak ohledně algoritmů obstarávajících předzpracování snímků, tak algoritmů rozpoznávající vodorovné dopravní značení.

Obr. 15 Analýza vodorovného dopravního značení pro autonomní vozidlo [22]



Obr. 16 Systém lokalizující vodorovné značení pomocí OCR [23]

4 Aplikace dopravního inženýrství

Tato kapitola je zaměřena na software pro řízení a plánování světelných křižovatek, který je vyvíjen společností Siemens. Jsou zde popsány produkty Sitraffic smartCore a Sitraffic Office, u kterých je předpokládáno budoucí využití výsledků této práce.

4.1 Sitraffic smartCore

Sitraffic smartCore je software vyvinutý společností Siemens za účelem návrhu a tvorby signálních plánů křižovatek. Signální plán je program řízení dopravy v křižovatce, zjednodušeně jde tedy o posloupnosti stavů, ve kterých se řízení křižovatky (světelné signály) může vyskytovat. Samotný Sitraffic smartCore podporuje principy jak pevného tak dynamického řízení dopravy a je využíván zejména v dopravních kancelářích.

Samotný produkt je původně navrhnut především pro návrh méně komplikovaných řízení křižovatek – v menších městech a jejich okolí. Za tímto účelem je navrhnuto i jeho uživatelské rozhraní, stručné, přehledné a co možná nejjednodušší. Celý koncept je uvažován jako průvodce nastavením signálního plánu, kdy postupně krok za krokem uživatel prochází jednotlivé položky a sestavuje tak výsledný plán. Na následujících řádcích jsou za účelem seznámení se se softwarem, do kterého bude výsledek práce zaintegrován, popsány jednotlivé dílčí kroky nastavení, které produkt poskytuje.

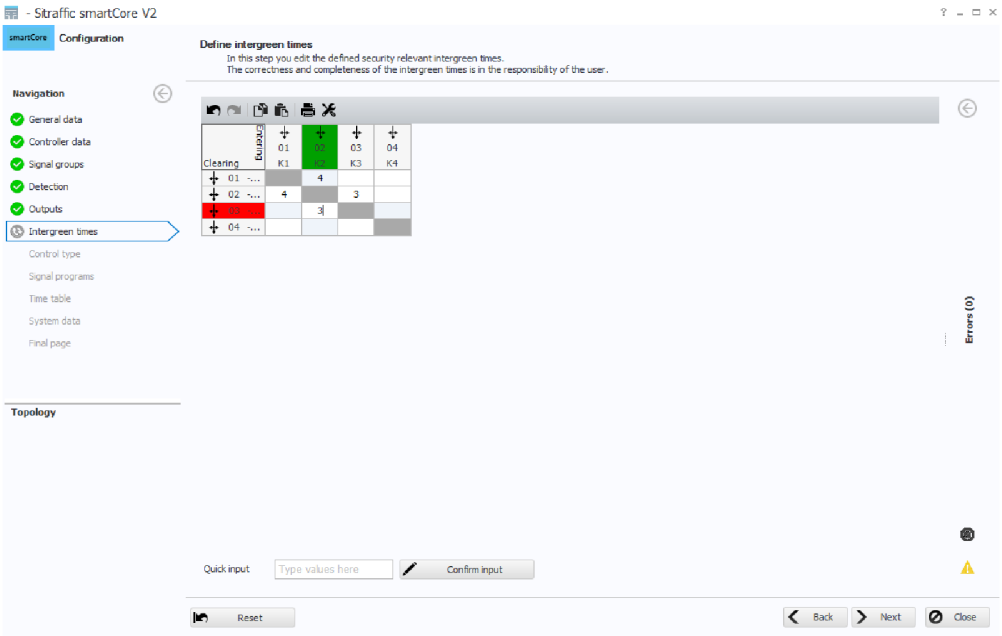
4.1.1 Popis funkcionality

Po spuštění lze vybrat, zda chceme otevřít existující konfiguraci, stáhnout konfiguraci z řadiče nebo vytvořit konfiguraci novou. Při volbě konfigurace nové, je uživatel postupně proveden jednotlivými záložkami, které slouží k jejímu nastavení.

- **Obecné informace (General data) –** Jde o úvodní záložku pro vyplnění jak popisných, tak třeba také informací geografických. Uvádíme jméno křižovatky, město, či bližší popis konkrétní lokality.
- **Kontrolér (Controller data) –** Nastavení dat specifických pro daný kontrolér – napájecí napětí kontroléru a signálních světel, dostupnost záložního napájení, typ komunikace s řídícím střediskem.
- **Signální skupiny (Signal groups) –** Jednotlivá ramena křižovatky jsou popsána právě zde. Pro křižovatku zde uvádíme její tvar, jednotlivá ramena a dopravní proudy. Pro každé rameno pak popis jízdních pruhů, tedy jejich směr. Dopravní proudy pak lze rozřadit podle toho, pro koho jsou určeny – auta, chodci, apod.

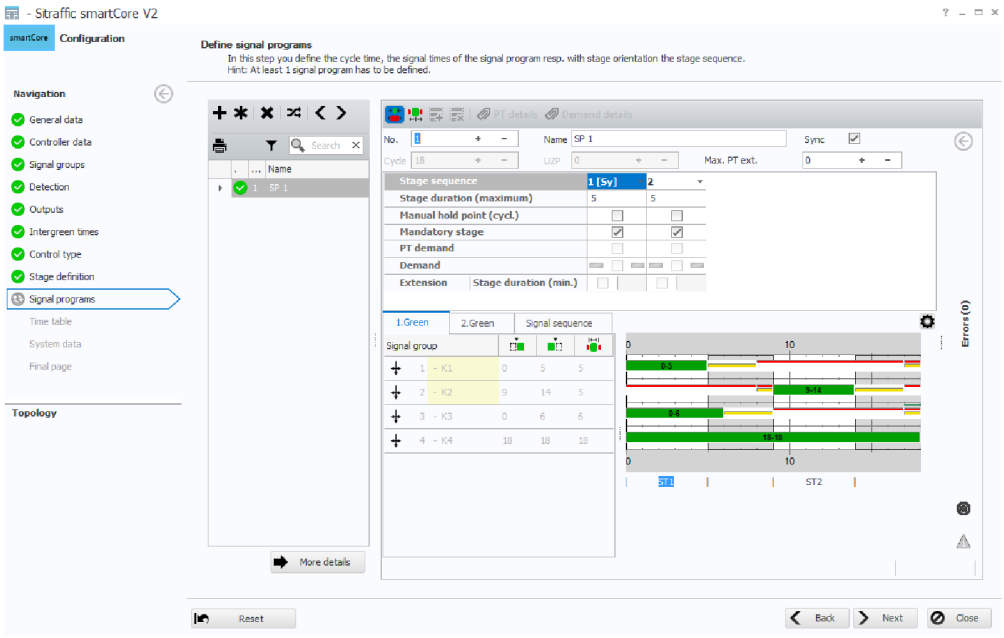
Obr. 17 Signální skupiny v Sitraffic smartCore

- **Detektory (Detection) –** Záložka je zde k nastavení vstupů řadiče, sloužících ke sběru a vyhodnocení informací pro dynamické řízení křižovatky. Jde o vstupy např. z indukčních smyček, tlačítek pro chodce, radary, apod.
- **Výstupy (Outputs)** – Výstupy slouží co by zpětná komunikace z řadiče jak v místě křižovatky, tak např. pro centrální středisko řízení dopravy. Je zde tedy umožněno nastavit typy těchto výstupních informací.
- **Matice mezičasů (Intergreen times)** – Mezičas je bezpečná doba, která je potřeba při přepínání aktivity z jednoho proudu do druhého (např. doba pro bezpečné opuštění křižovatky). Matice následně ukazuje všechny možné kombinace a umožňuje nastavení jednotlivých mezičasů.



Obr. 108 Matice mezičasů v Sitraffic smartCore

- **Typ ovládání (Control type)** – Nastavení uvádějící, zda se pro řízení křižovatky použije signální plán využívající fázová schémata, zda je použita ochrana pro vyskytující se chodce a zda si přejeme automaticky vygenerovat signální plán.
- **Fázové shcéma (Stagge definition)** – Fázové schéma rozděluje dopravní proudy do několika segmentů. Tyto segmenty jsou vzájemně bez konfliktů, to znamená, že v každém segmentu jsou proudy, které se v okamžiku, kdy jsou aktivní, neprotínají. Nastavení softwaru umožňuje fázová schémata jak automaticky vygenerovat, tak manuálně upravovat dle představy uživatele, přičemž v případě sloučení kolizních proudů do jednoho schématu je uživatel upozorněn chybovými hlášeními.
- **Signální plán (Signal programs)** – Tato záložka zobrazuje jednotlivá fázová schémata, umožňuje modifikaci délky jednotlivých zelených signálů, či přechodových stavů mezi signály. Je zde také možná další úprava fázových schémat.
- **Přepínací časy (Time table)** – Vzhledem k tomu, že řízení křižovatek vyžaduje rozdílné chování pro jednotlivé části dne, či týdne, je možné zde nastavit různé signální plány např. pro dopravní špičku, noc, víkendový provoz, apod.



Obr. 19 Signální plán konfigurace v Sitraffic smartCore

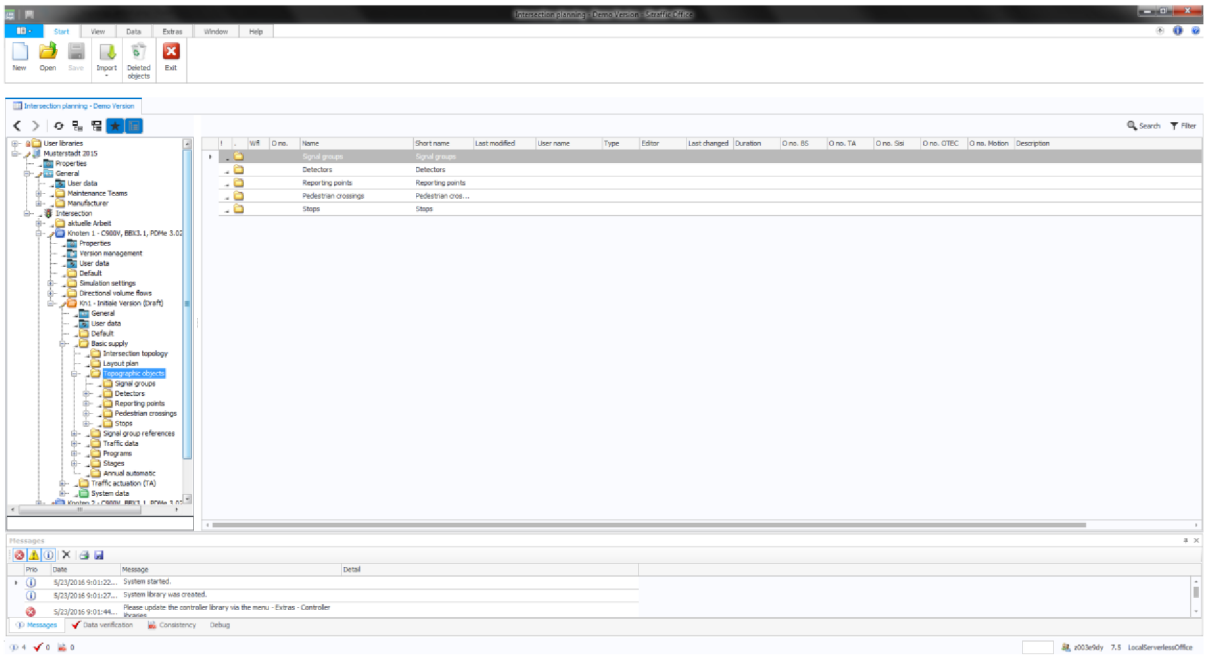
- **Systémová konfigurace (System data)** – Umožňuje nastavení technických parametrů, které se vztahují k použitému typu řadiče (napájení, přídavné moduly, detektory, výstupy, rozšiřující karty).
- **Kontrolní strana (Final page)** – Poslední záložka slouží zejména jako sumarizace nastavených informací.

Z uvedeného popisu nás zajímá především stránka zabývající se Signálními skupinami, kde se nastavuje tvar křižovatky, počet ramen, počet jízdních pruhů, směr jízdních pruhů, přechody pro chodce. Dalšími záložkami, které mohou být zajímavé v rámci této práce, jsou např. stránka s detektory, jejichž typ nebo minimálně umístění by bylo možné předem nastavit, pokud jsou uvedeny na vstupním plánku.

4.2 Sitraffic Office

Sitraffic Office (obr. 20) je software vyvinutý společností Siemens pro komplexní správu řízení a plánování samostatných i koordinovaných světelných křižovatek. Samotný software je postaven na modulárním konceptu a obsahuje mnoho funkcí, zde jsou některé příklady:

- Editace fází, plány fázování a tvorba fázových přechodů a signálních plánů
- Tvorba a editace základních dat signálního řízení jako např. mezičasy, progresivně řízené časy chodců
- Kalkulace a hodnocení signálních plánů
- Výpočet mezičasů
- Import a editace měřených hodnot ve formě grafů a analýza zatížení proudů
- Grafické editace pro časoprostorové diagramy včetně zobrazení tras veřejné dopravy a proměnných časů zelené při dynamickém řízení
- Více-uživatelský přístup (možnost pracovat paralelně na stejných projektech)
- Import/export/migrace/tisk dat

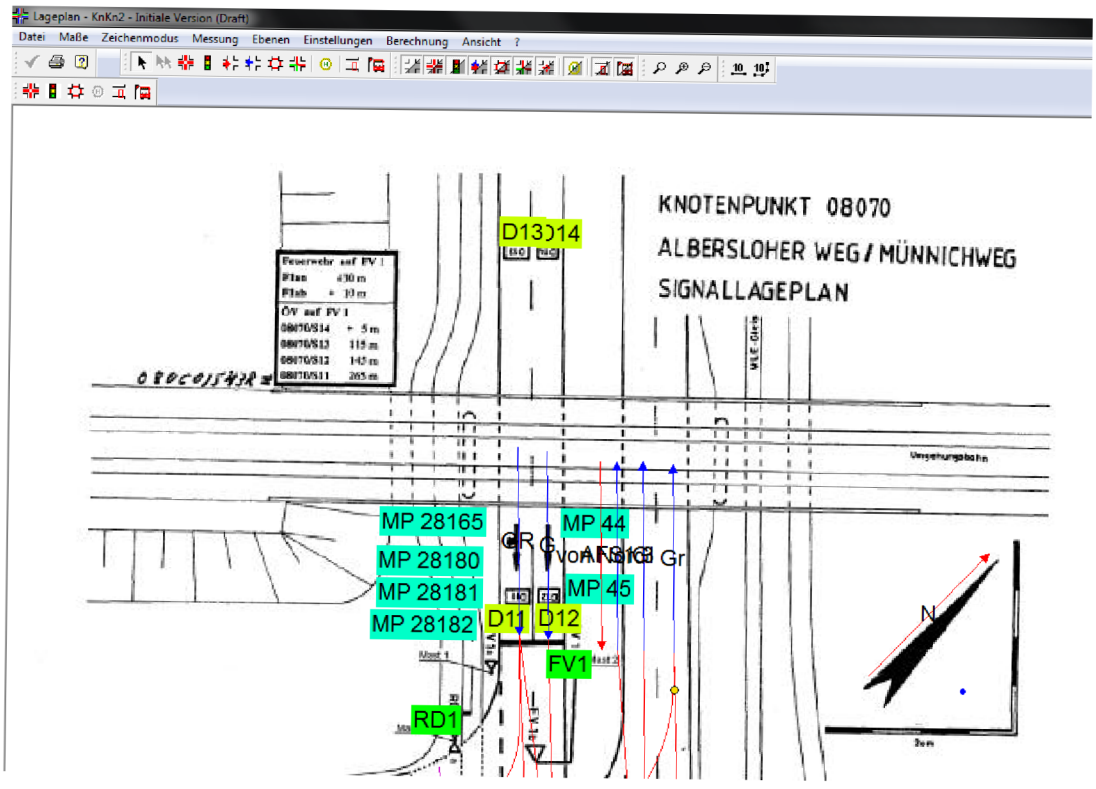


Obr. 20 Sitraffic Office

4.2.1 Popis funkcionality

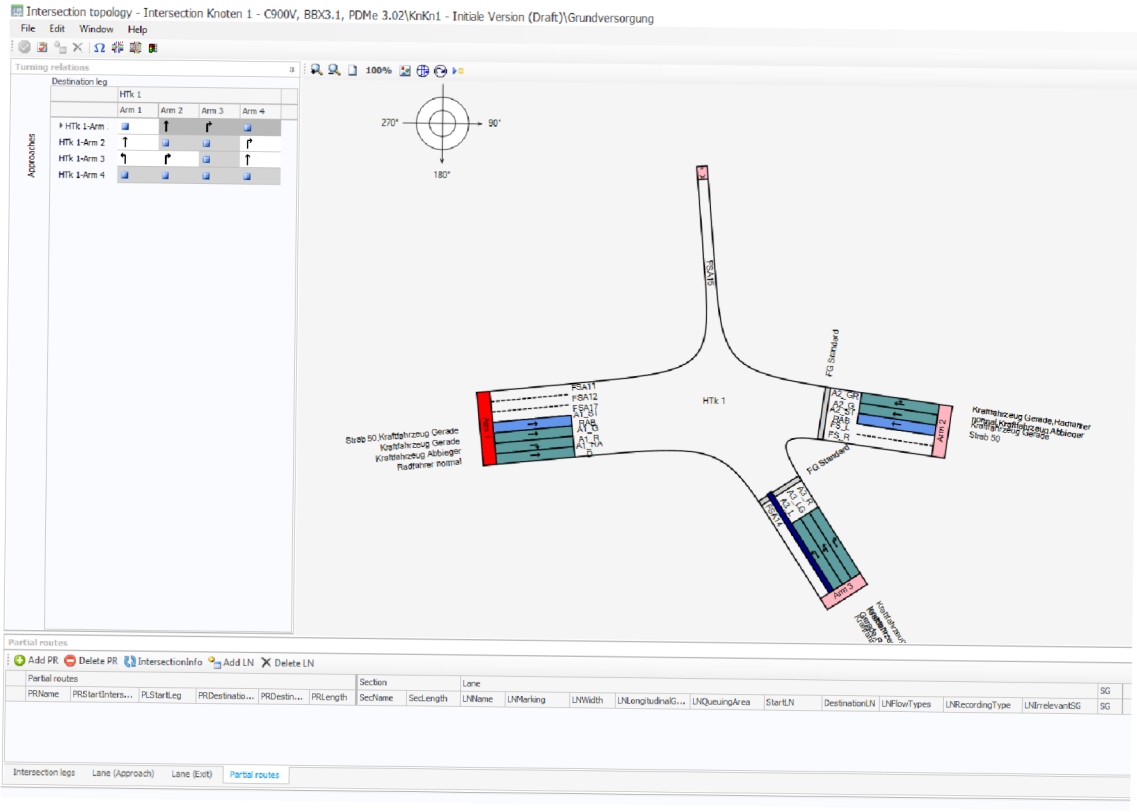
Sitraffic Office je komplexní nástroj obsahující mnoho funkcí a editorů. Pro potřeby této práce není nutné popisovat jeho celou funkcionalitu, je však vhodné zmínit editory, u kterých je předpokládáno využití dosažených výsledků.

- **Editor plánu křižovatky (Layout plan editor)** – umožňuje nahrát rastrový plán křižovatky a spravovat data, která se k němu mohou vázat (zakreslení údajů o ramenech křižovatky, jízdních pruzích, detektorech, lampách semaforů, průjezdech křižovatkou, konfliktních bodech,…).



Obr. 21 Editor plánu křižovatky

- **Topologický editor (Intersection topology editor)** – topologická reprezentace křižovatky, která popisuje křižovatku jako model obsahující informace o ramenech křižovatky, jízdních pruzích, přechodech,… a vztazích mezi nimi.



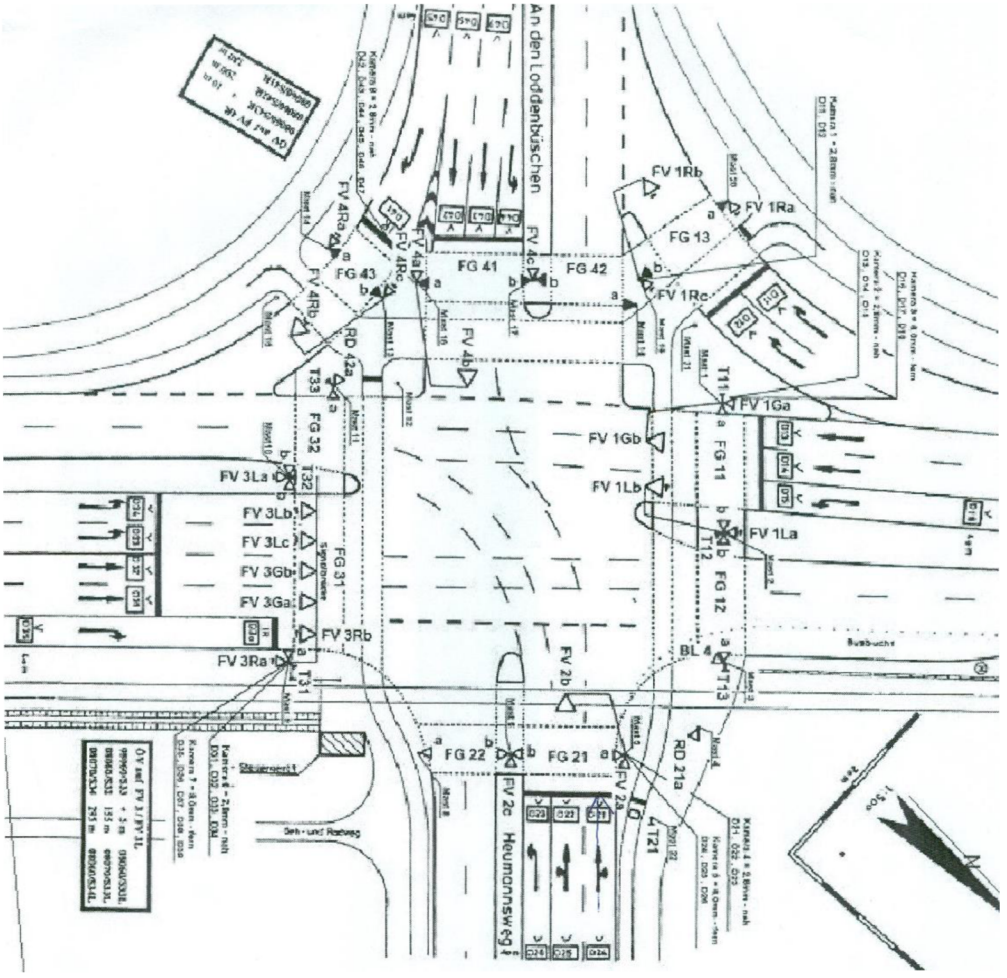
Obr. 22 Topologický editor

5 Analýza řešeného problému

Kapitola zabývající se analýzou řešených problémů má za úkol ujasnit si cíle a postupy k nim vedoucí ještě před samotnou implementací a posloužit jako vzor při vývoji praktické části práce. Uvažované řešení má hlavně za cíl z poskytnutých snímků plánů křižovatky vyextrahovat co nejvíce užitečných informací při zachování robustnosti a obecnosti postupů. V této kapitole bude postupně probráno, jaké cílové informace budou z plánů extrahovány, jaká vstupní data jsou poskytnuta, bude uveden příklad anotovaných vstupních dat, metody možného předzpracování obrazu a idea jak detekovat topologické objekty křižovatky a jejich vlastnosti.

5.1 Vstupní data

Jak je patrné již ze zadání, vstupní data jsou poskytnuta coby naskenovaný snímek křižovatky v podobě rastrového obrazu. Základním problémem vstupních dat, viz obr. 23, je, že obsahují příliš mnoho informací, které ve výsledku považujeme za šum. Je tedy patrné, že pro úspěšnou detekci zájmových objektů bude třeba řádného předzpracování.



Obr. 23 Příklad vstupních dat

5.2 Zájmové objekty

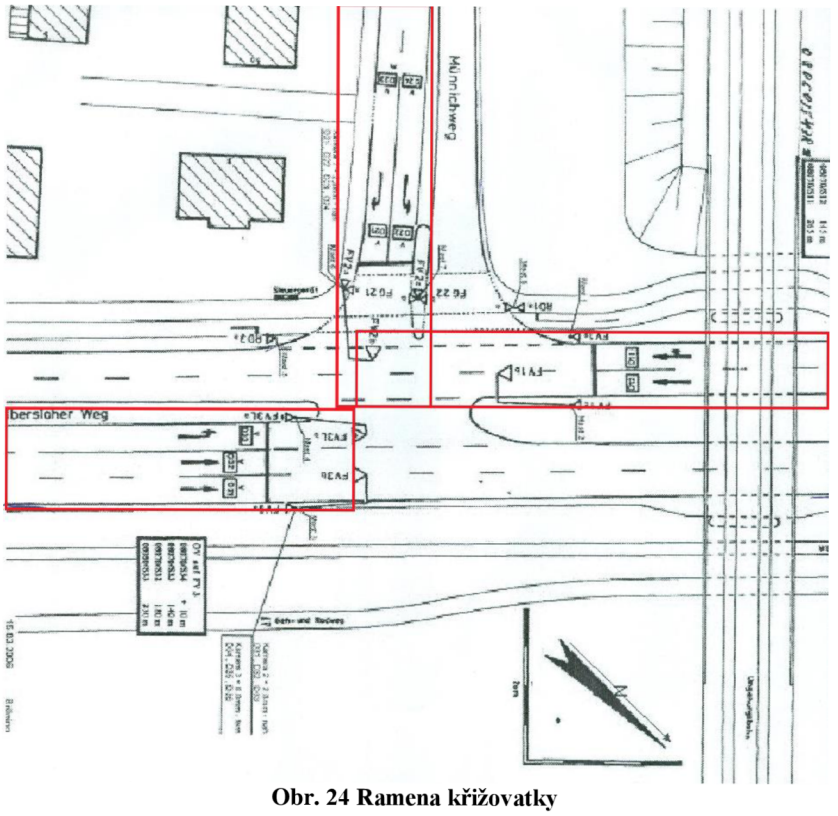
Z analýzy cílového softwaru Sitraffic smarCore a Sitraffic Office a jejich možných nastavení v rámci konfigurace kontrolérů řídících světelné signály vyplynulo, že ze vstupních obrazových dat by bylo vhodné extrahovat následující objekty:

- **Tvar křižovatky -** rozpoznat komplexní tvar křižovatky, její pozici (střed) a všechny okolní objekty ignorovat
- **Ramena křižovatky** - v křižovatce určit umístění jednotlivých ramen, zjistit jejich vzájemnou polohu (svírající úhel), polohu vůči středu křižovatky, šířku.
- **Jízdní pruhy -** v daném ramenu detekovat jízdní pruhy, především detekovat jejich pozici a směr.
- **STOP pozice –** pozice čar udávající pro každý jízdní pruh polohu, kde jsou vozidla povinna zastavit
- **Přechody pro chodce –** potvrdit/vyvrátit výskyt přechodů pro chodce v křižovatce, příp. detekovat jejich polohu v křižovatce

5.3 Anotace vstupních dat

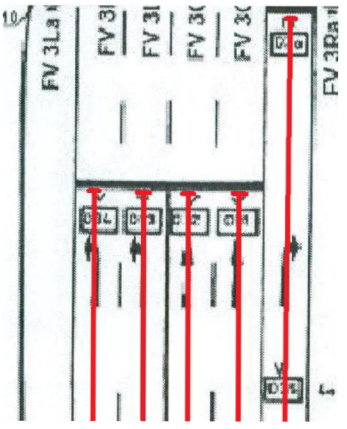
Za účelem stanovení cíle projektu a možného rychlého porovnávání výsledků se jeví jako vhodné vybrat z poskytnuté datové sady malou množinu snímků, na kterých bude provedeno manuální označení zájmových objektů a popis jejich vlastností. Takto označené objekty budou sloužit jako referenční jak při samotné implementaci, tak při prováděných experimentech a testování. Následuje příklad vybraného popsaného plánu křižovatky:

- **Vstupní obraz:**
  - `Width: Int (2550)` – šířka v pixelech
  - `Height: Int (1755)` – výška v pixelech
- **Křižovatka:**
  - `Position: Point (double 0.6863, double 0.5270)` – relativní pozice středu křižovatky
  - `Number of legs: Int (3)` – počet ramen
- **Každé Rameno křižovatky:**
  - `Angle: Double (5,34)` – úhel ve stupních svírající s osou X
  - `Width: Double (0.1384)` – šířka ramena v relativní hodnotě
  - `Length: Double (0.2949)` – délka ramena v relativní hodnotě od středu křižovatky po okraj plánu
  - `WidthP: Int (243)` – šířka zarovnaného výřezu obsahující rameno křižovatky
  - `HeightP: Int (752)` – výška zarovnaného výřezu obsahující rameno křižovatky
  - `Number of lines: Int(3)` – počet jízdních pruhů



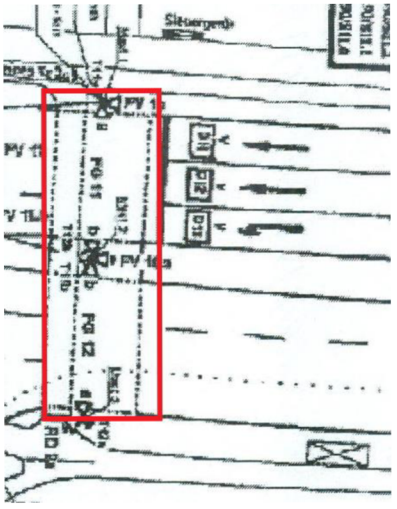
Obr. 24 Ramena křižovatky

- **Každý Jízdní pruh:**
  - `Way:` `Way (Left)` – směr jízdního pruhu
  - `Lenght:` `Double (0.654)` – relativní délka jízdního pruhu od spodní hrany v zarovnaném výřezu obsahujícího rameno křižovatky
  - `STOP line:` `Point (double 0.2836, double 0.654)` – relativní pozice STOP pruhu v zarovnaném výřezu obsahujícího rameno křižovatky



Obr. 11 Jízdní pruhy v zarovnaném ramenu křižovatky

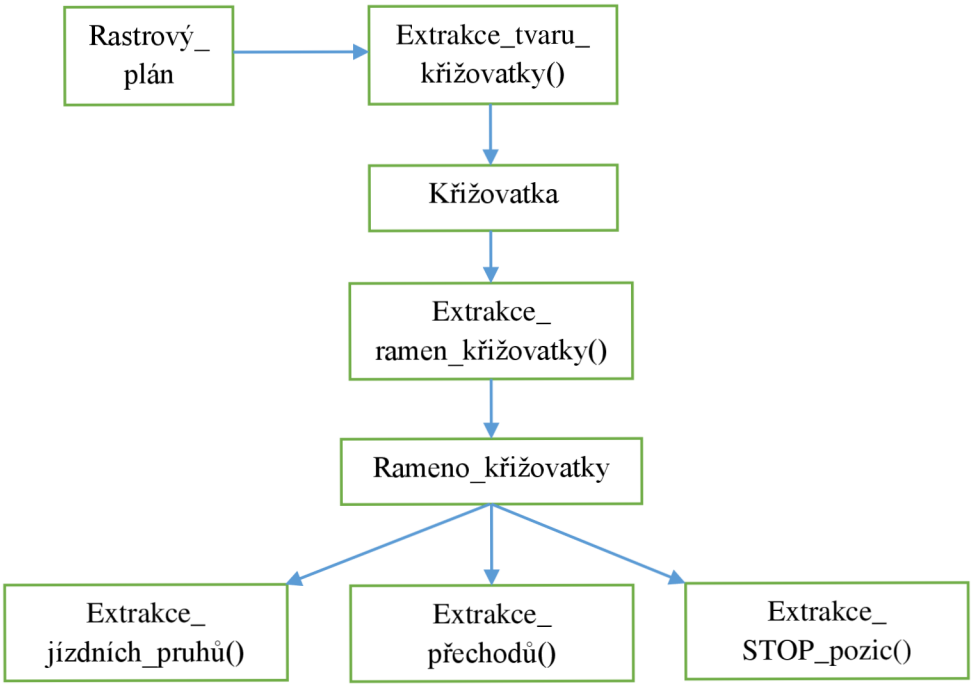
- **Přechod pro chodce:**
  - `TopLeft:` `Point (double 0.7756, double 0.4045)` – relativní pozice přechodu
  - `Boundary:` `Rect (double 0.7756, double 0.4045, double 0.8365, double 0. 6820)` – obepisující objekt zadaný v relativních souřadnicích



Obr. 12 Přechod pro chodce

5.4 Schématický návrh řešení

Pro lepší přehled celkové funkcionality systému a jasnější budoucí rozčlenění do funkčních bloků je celkový proces zpracování rastrového vstupu popsán diagramem. Každá funkční jednotka má na vstupu rastrový obraz a na výstupu datovou strukturu obsahující popis detekovaných objektů a případně rastrový obraz pro další zpracování.



Obr. 13 Schéma funkčních bloků řešení

V rámci implementační části práce bude každý blok představovat samostatnou jednotku, pro kterou bude vytvořena ukázková aplikace, očekávající adekvátní vstup v podobě předpřipraveného rastrového obrazu. Tím bude zajištěna možnost posoudit funkčnost jednotlivých bloků a jejich závěrečné vyhodnocení.

5.5 Předzpracování obrazu

Vstupní data jsou pořizována skenerem. Ten jako takový produkuje barevný obraz, který je pro další zpracování nejdříve nutné převést do obrazu ve stupních šedi. Dalším krokem je minimalizovat výskyt šumu. Za šum lze v našem případě považovat všechny informace, které nejsou relevantní pro dosažení úspěšné detekce zájmových objektů. Kromě objektů o minimální velikosti, které mohly vzniknout během skenovacího procesu, je nutné odstranit objekty typu budov, doprovodného textu, apod. Odstranění objektů, které nemají sémantický význam ve vztahu ke křižovatce, se tedy jeví jako nejdůležitější část celé práce. Samotné předzpracování obrazu je však nutné u každého funkčního bloku ze schématického náčrtu (obr. 27).

Předpokládejme následující úvahy. Pokud jde o budovy, lze z plánů vypozorovat, že se jedná o celkem velké objekty, často vyšrafované, a jejich značení tedy zaujímá značnou plochu v porovnání např. s dělicími čarami nebo šipkami. Nabízí se tedy myšlenka segmentovat vstupní obraz podle velikosti jednotlivých objektů. To lze udělat následovně: jednoduchou segmentací obrazu jej rozdělíme na vzájemně nepropojené (disjunktní) objekty, spočítáme jejich plochy, zvolíme vhodnou hodnotu prahu, vyřadíme největších z nich a dosáhneme tak tedy odstranění budov. Co se týče popisného textu, je možné použít detektor textu, konkrétně jednotlivých písmen, obraz celý postupně projít a nalezené písmena ukládat do masky, kterou následně ve finále použijeme pro odstranění textu.

Po odstranění nežádoucích informací je další předzpracování obrazu závislé na typu objektu, který chceme detekovat. V případě detekce ramen a jejich směrů je vhodné využít Houghovy transformace, před ní samotnou je vhodné použít morfologické operátory k získání hran obrazů a jejich následnému zesílení. V případě detekce šipek, přechodů nebo STOP čar pak spadá úvaha směrem k lokalizaci potenciálních kandidátů, oříznutí okna zájmu pouze na dané oblasti, kde se dané objekty mohou vyskytovat, a otestování těchto oblastí na jejich výskyt.

5.6 Odstranění doprovodných textů

Jak je patrné z obr. 23 vstupní data obsahují mnoho doprovodných textů, které nejsou pro základní rozpoznávání topologických informací potřebné. Tyto texty zde působí jako šum a navíc znemožňují přesnější vyhledávání zájmových objektů.

Nabízí se tedy možnost tyto doprovodné texty odstranit. Pro vyhledání těchto textů je možné využít metody MSER a místa obsahující text nahradit barvou pozadí.

5.7 Detekce šipek

Detekce šipek se jeví jako zásadní část v rámci získávání topologických informací z rastrového plánu křižovatky. Cílem je získání pozic šipek ze vstupního rastrového obrazu. Jejich nalezení bude třeba pro:

- Potvrzení nalezených ramen křižovatky
- Detekci jízdních pruhů
- Určení směrů jízdních pruhů

Při detekci šipek je třeba brát v úvahu:

- Tvary šipek popisující možné směry jízdy
- Velikosti šipek
- Možné úhly natočení šipek

Pro detekci šipek bude využito implementace kaskádových klasifikátorů využívající LBP příznaky. Za tímto účelem bude třeba vytvořit sadu trénovacích dat pro klasifikátory.

5.8 Detekce ramen

Cílem této detekce je z poskytnutého rastrového obrazu, obsahujícího pouze křižovatku, již oproštěnou od šumových informací, extrahovat dílčí informace o každém rameni křižovatky zvlášť. Jde tedy o určení jejich počtu, úhlu jejich sevření a jejich společného středu, coby středu křižovatky. Kromě těchto informací bude výstupem také rastrový obraz, který poslouží jako vstup pro další metody zpracování. Tento obraz bude za tímto účelem ořezán pouze na oblast samotného ramene a zarovnán jednotným způsobem, kdy bude orientován ve vertikálním směru rovnoběžně s osou Y a směr dopravních proudů bude orientován odspodu nahoru.

Na plánech křižovatek lze vypozorovat, že ve směrech dopravních proudů a tedy i jednotlivých ramen, se vyskytuje vzor čar rovnoběžných s těmito proudy. Základní myšlenkou detekce ramen je tedy nalezení takových čar a zaměření se na hustotu jejich výskytu. Pro detekci čar využijeme Houghovy transformace. Můžeme uvažovat, že nejvíce čar se bude vyskytovat právě ve směru ramen a tím tak určit jejich přibližný výskyt. Pokud již máme přibližný výskyt ramen, lze aplikovat ideu, že výskyt ramene je podmíněn výskytem šipek určujících směry dopravních proudů. Jak již bylo zmíněno v předchozí kapitole, za tímto účelem lze využit předzpracování obrazu využívající segmentace podle velikosti objektů – v tomto případě volíme práh takový, aby byly objekty, které jsou větší než šipky, odstraněny. Zbývá už tedy jen projít zbylé detekované objekty a pokusit se v nich detekovat šipky. V případě úspěšné detekce jsme pak schopni říct, zda v daném místě s vyšším výskytem detekovaných čar je skutečně rameno křižovatky nebo ne, a podle tvaru a orientace šipek také určit směr jízdních pruhů křižovatky. Samozřejmě se naskýtá otázka, co dělat v případě, že nalezneme více ramen značně podobného směru i umístění. V takových případech je vhodné porovnat jejich vzájemnou polohu, orientaci a umístění šipek, a pokud je nalezena shoda v rámci vhodně zvoleného prahu, tyto ramena sloučit do jednoho.

5.9 Detekce jízdních pruhů

Pro detekci jízdních pruhů lze předpokládat, že již máme získané informace o pozici jednotlivých ramen křižovatky a pracujeme tedy se vstupním rastrovým obrazem obsahujícím pouze jedno zkoumané rameno.

Jednotlivé jízdní pruhy v rámci ramene křižovatky jsou většinou odděleny dělicí čarou. Z tohoto faktu pak vystává samotný postup při detekci jízdních pruhů. Jednotlivé pruhy je možné oddělit a rozpoznat právě pomocí dělicích čar, které lze rozpoznat pomocí Houghovy transformace pro detekci přímek. Další postup pak spočívá v určení směru jednotlivých jízdních pruhů, tedy zda se jedná o vstupní, či výstupní jízdní pruh. Toto rozhodnutí lze provést na základě rozpoznání šipky v daném jízdním pruhu, zatímco vstupní jízdní pruh tyto šipky obsahuje, výstupní nikoliv. Poslední akcí při detekci jízdních pruhů, je určení směrů pro vstupní jízdní pruhy. Pokud již máme detekovány šipky, klasifikujeme jejich tvar a rozhodneme o směru.

5.10 Detekce přechodů pro chodce

Stejně jako pro detekci jízdních pruhů lze i zde předpokládat, že již máme získané informace o pozici jednotlivých ramen křižovatky a pracujeme tedy se vstupním rastrovým obrazem obsahujícím pouze jedno zkoumané rameno.

Pozici přechodu pro chodce lze předpokládat na konci ramene, který je nejblíže středu křižovatky. Z dostupných testovacích dat, lze předpokládat formát zakreslení přechodu, jak je zobrazen na obr. 26. Pro jeho detekci tedy bude potřeba detekovat čáry, které jsou co nejblíže ke středu křižovatky a svírají s osou ramene úhel přibližně 90°. Tyto čáry lze rozpoznat pomocí Houghovy transformace pro detekci přímek.

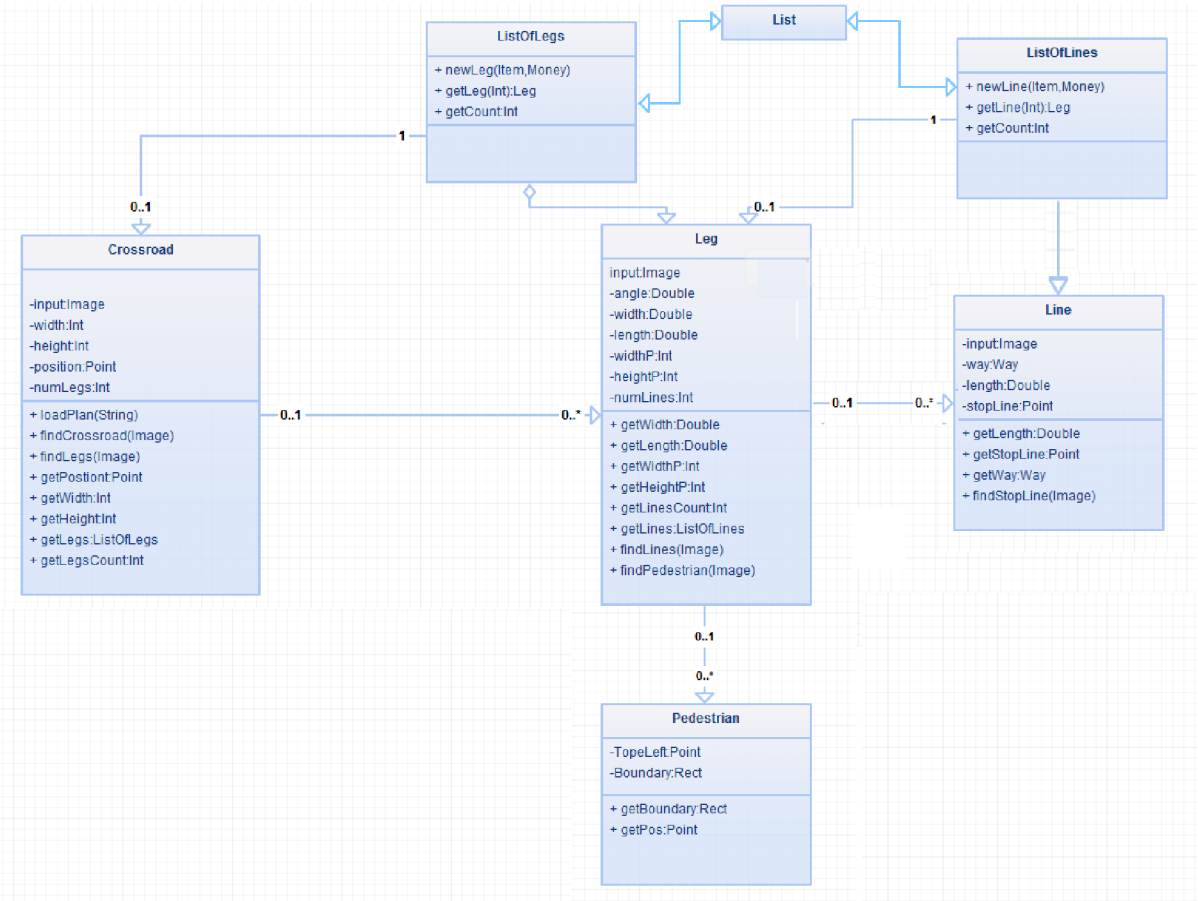
5.11 Výstup algoritmů

Zpracování veškerých souřadnic by mělo být v jejich relativních hodnotách, nezávislých na aktuálním reálném rozměru. To umožní přepočítání všech výsledků podle později zadaného měřítka. Výstup by měl být datová struktura reprezentující analyzovaný topologický objekt, ta se skládá z nalezených objektů a vlastností popisujících jejich stav

5.12 Datová struktura křižovatky

Pro budoucí možné propojení jednotlivých funkčních bloků detekce je navržen datový model (obr. 28) a to s ohledem na uchování veškerých vstupních a výstupních dat potřebných pro úspěšnou detekci topologických informací.

`Crossroad` slouží pro uložení globálních dat společných pro celou křižovatku, obsahuje vstupní obraz, metody pro detekci křižovatky a jejích ramen. `Leg` uchovává informace o ramenech křižovatky a poskytuje metody pro nalezení jízdních pruhů a přechodů pro chodce. Ty jsou uloženy v `Pedestrian`. Nakonec `Line` obsahuje informace o daném jízdním pruhu v rámci jednoho ramene a poskytuje možnost nalezení STOP čáry.



Obr. 14 Diagram datového modelu

5.13 Testovací aplikace

Za účelem cílové prezentace funkčního řešení je třeba navrhnout demo aplikaci s grafickým uživatelským rozhraním (GUI), která bude mít za úkol demonstrovat funkčnost jednotlivých přístupů a algoritmů. Aplikace by měla být zejména jednoduchá, a to jak po stránce intuitivního ovládání, tak po stránce designové. Jako prostředek pro vývoj a tvorbu GUI bude využito technologie Windows Forms [13]. Základním požadavkem na funkčnost a design vzhledu je:

- Zobrazení vstupního (zpracovávaného) obrazu
- Zobrazení detekovaných objektů a jejich promítnutí do vstupního obrazu
- Zobrazení podrobností o detekovaných objektech

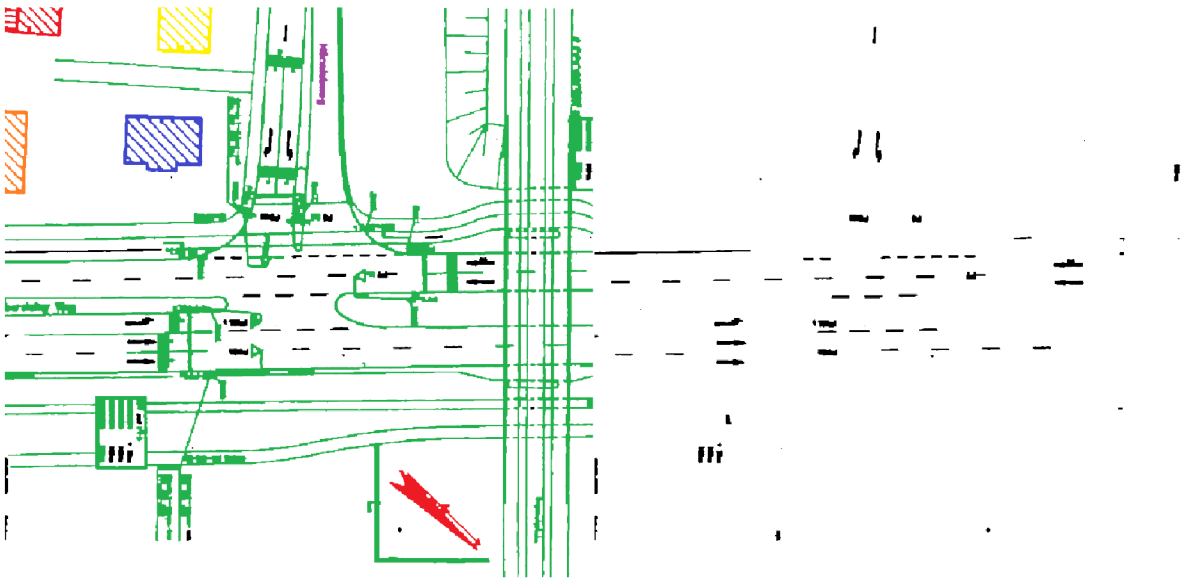
6 Realizace a implementace experimentů

Při implementaci je kladen důraz na co největší využití již existujícího kódu z knihovny OpenCV, která disponuje komplexními řešeními v oblasti zpracování obrazu. Dá se tedy uvažovat, že použité algoritmy jsou v optimalizované a odladěné verzi. Za implementační jazyk byl zvolen dle požadavku zadání diplomové práce jazyk C# s využitím knihovny EmguCV [14], což je C# verze knihovny OpenCV. Jednotlivé části realizace jsou v rámci řešení implementovány samostatně a následně propojeny ve funkční řešení. Aby bylo dosaženo maximální možné budoucí integrovatelnosti výsledného produktu do již existujícího řešení společnosti Siemens, probíhal vývoj ve vývojovém prostředí Microsoft Visual 2013 Ultimate s využitím vývojového nástroje ReSharper [15], které je rovněž využito při vývoji software ve společnosti Siemens.

V rámci diplomové práce bylo provedeno několik experimentů zaměřených na předzpracování vstupního obrazu a následnou detekci topologických objektů v křižovatce, ty budou popsány v následujících kapitolách.

6.1 Předzpracování obrazu

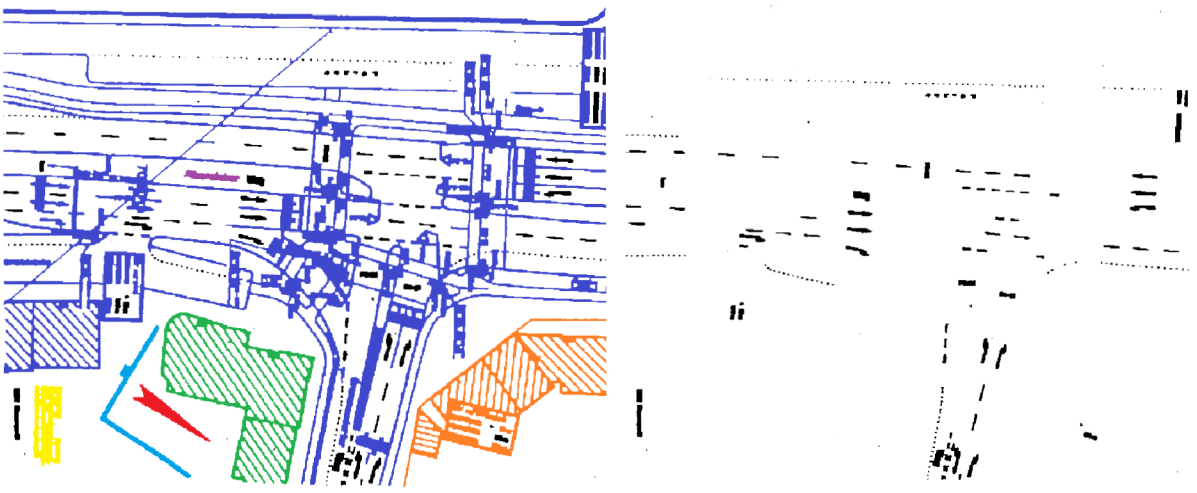
V rámci praktické části bylo provedeno několik experimentů zaměřených především na využití morfologických operátorů, segmentaci obrazu podle velikosti objektů a využití Houghovy transformace pro detekci přímek. To vše za účelem prozkoumání vhodného předzpracování obrazu, odstranění šumu a pozdější detekce ramen křižovatky.



Obr. 15 Ukázka 1 použití morfologických operátorů a segmentace

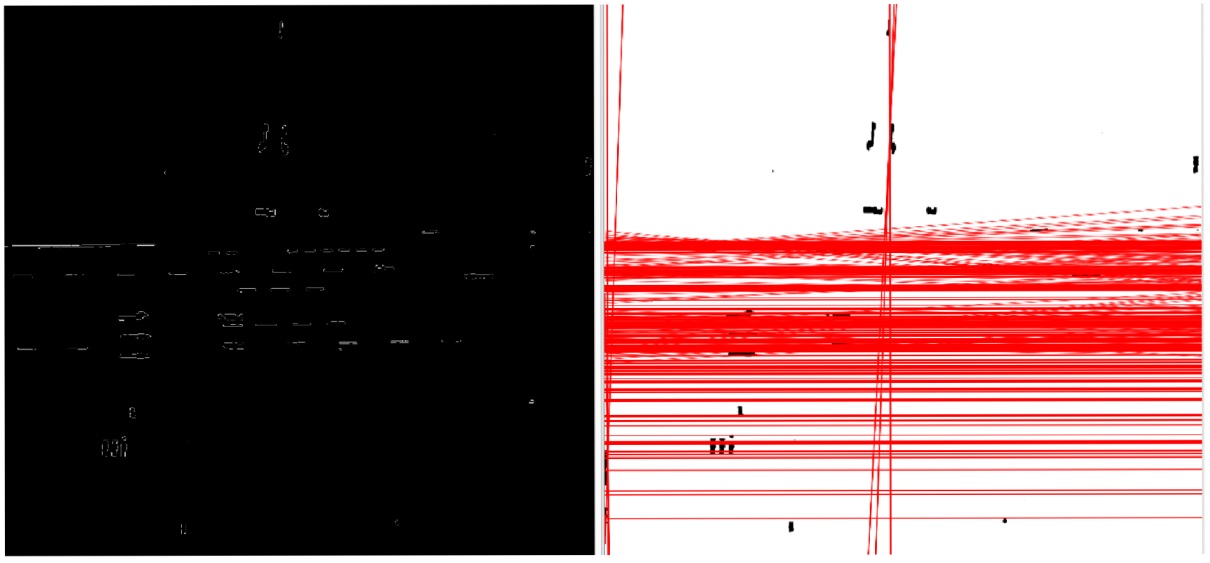
Na obr. 29 a 30 je ukázka použití morfologických operátorů a segmentace objektů podle velikosti. Jako vstupní obraz byl použit barevný sken plánu křižovatky, který byl převeden na obraz ve stupních šedi. Na tento obraz byly následně použity morfologické operátory pro uzavření a byla aplikována segmentace. Pro každý segmentovaný objekt byla spočítána plocha, která byla použita coby

rozhodující hodnota při mazání objektů, které nás nezajímají. Je patrné, že tímto přístupem lze označit možný výskyt šipek, což může být nápomocné v dalším rozpoznávání.

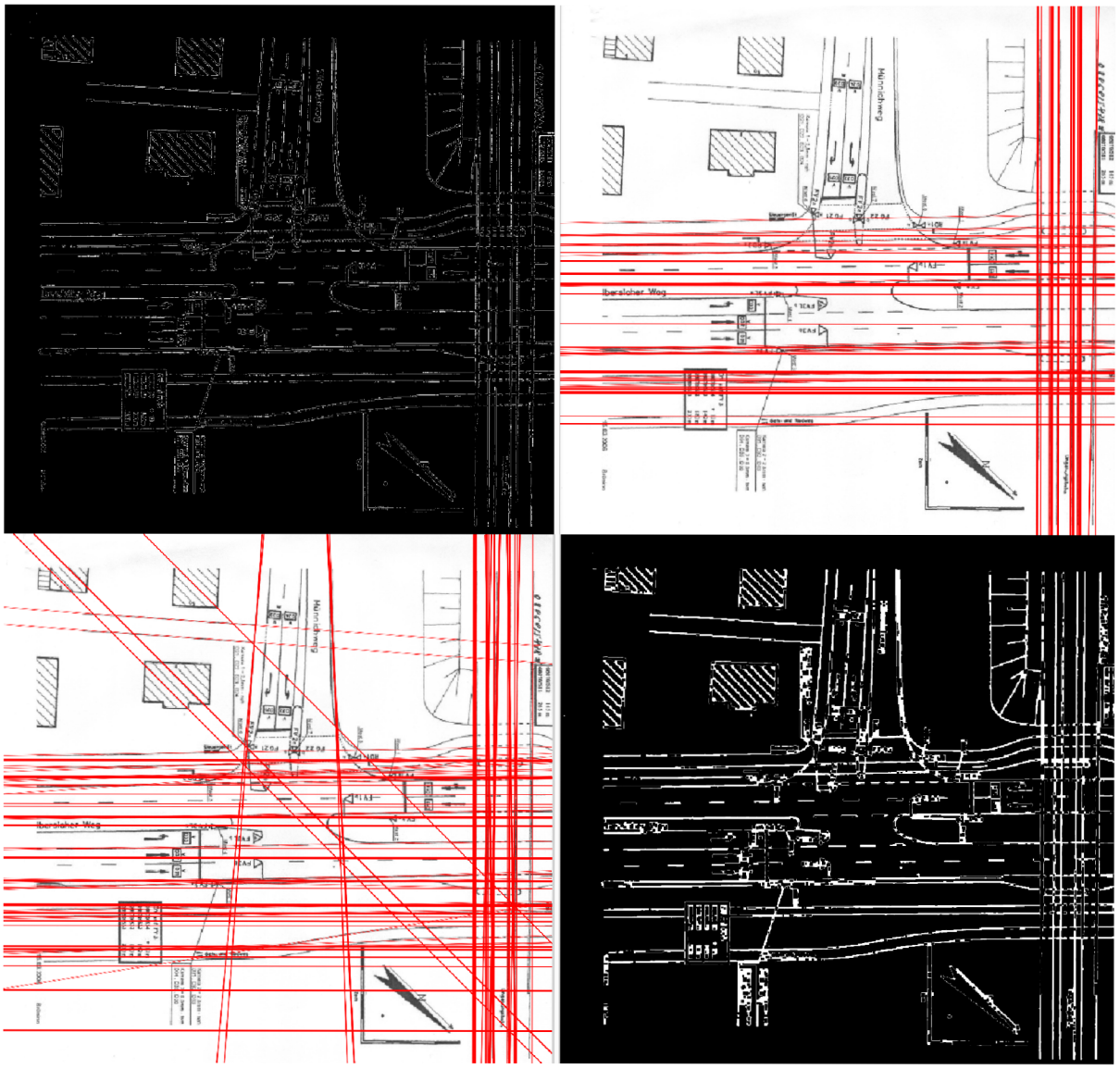


Obr. 30 Ukázka 2 použití morfologických operátorů a segmentace

Na obr. 31 a 32 je zobrazena ukázka detekce čar v křižovatce pro určení pozic a směru jednotlivých ramen. Pro detekci čar byla využita Houghova transformace s různými prahovými hodnotami, Tato detekce byla aplikována na různě předzpracované vstupy. Základním typem předzpracování byla detekce hran pomocí morfologických operátorů (gradient), následoval experiment detekce čar v obrazu upraveném pomocí uzavření a nakonec byla Houghova transformace aplikována i na výstupy segmentace objektů podle velikosti. Z experimentů vyplývá, že je výhodné v před samotnou detekcí čar provést převod na gradient a nejlépe jeho výsledky ještě zvýraznit vhodným morfologickým operátorem.



Obr. 31 Houghova transformace aplikována na výstup segmentace podle velikosti objektů

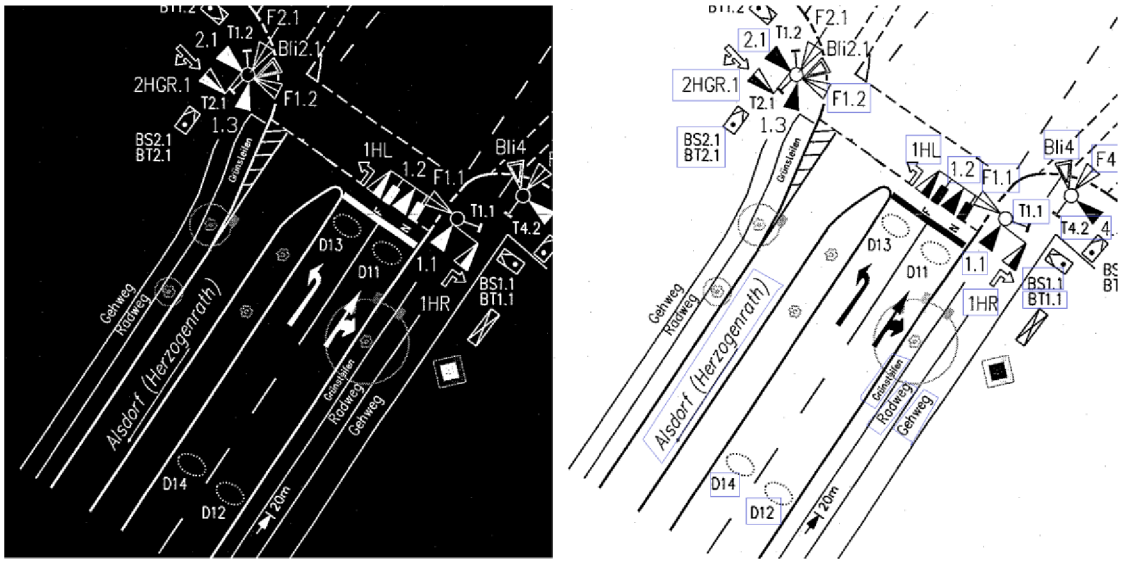


Obr. 32 Houghova transformace aplikovaná na gradient vstupu (nahoře) a vstup upravený operací uzavření (dole)

6.2 Detekce a odstranění textu

Pro detekci textu byla zvolena metoda MSER. Její implementace je součástí knihovny OpenCV. Spuštění algoritmu spočívá ve vytvoření filtrů extrémních regionů (ER filtrů), pro které se použijí základní kaskádové klasifikátory pro klasifikaci textu poskytnuté knihovnou OpenCV – metody `createERFilterNM1` a `createERFilterNM2`. Následuje aplikování kaskádových klasifikátorů na každý jednotlivý kanál vstupního obrazu – metoda `run`. A celý proces je ukončen detekováním skupin znaků metodou `erGrouping`.

Takto získané oblasti obsahující znaky jsou uloženy do odpovídajícího seznamu, který je později využit coby maska právě pro odstranění nežádoucího textu ve vstupním obrazu.

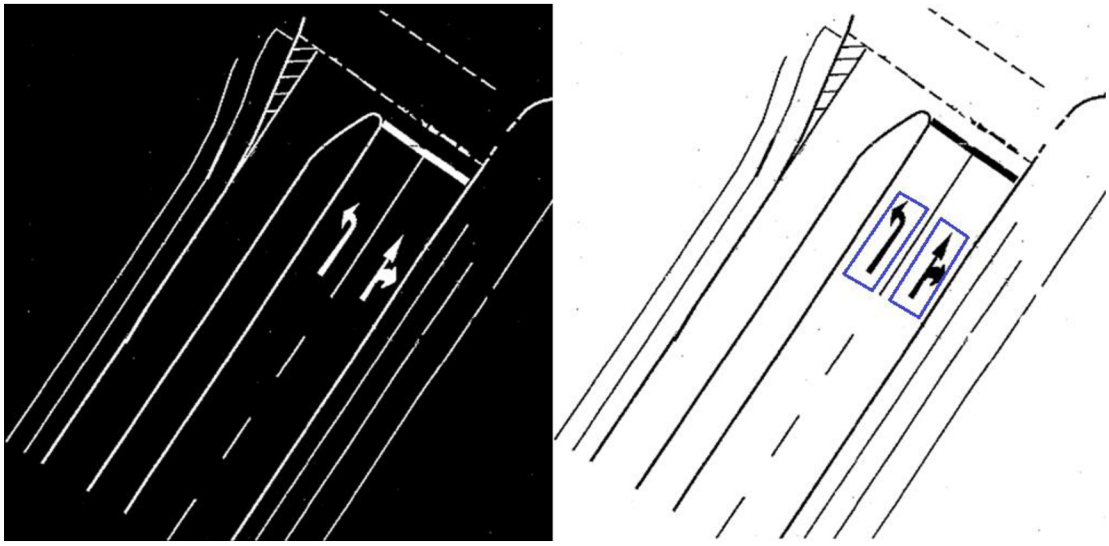


Obr. 33 Detekce textu pomocí metody MSER

6.3 Detekce šipek jízdních pruhů

Pro detekci šipek je využita implementace kaskádových klasifikátorů OpenCV využívající LBP příznaky.

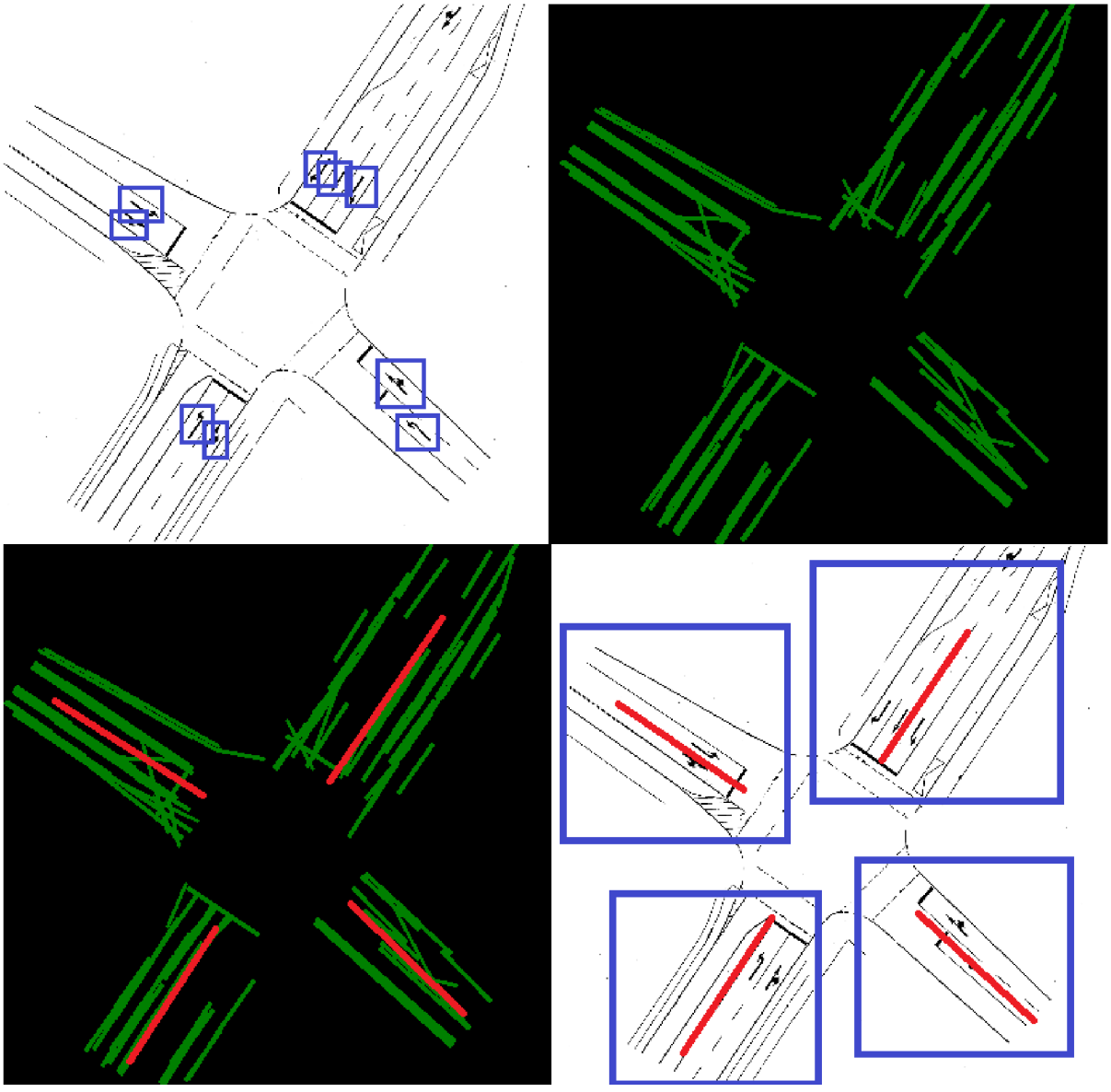
Za účelem využití kaskádových klasifikátorů byla vytvořena sada trénovacích dat pro trénování klasifikátorů. Pozitivní vzorky byly vybrány z dostupných šipek různých tvarů, velikostí a natočení. Negativní vzorky pak z různých výřezů z testovacích vstupních rastrů a zároveň neobsahující šipky. Samotné trénování proběhlo podle návodu dostupného [18]. Byly vytvořeny klasifikátory pro všechny dostupné tvary šipek. Je tedy nejen možné určit, zda se ve vstupním obraze šipky nacházejí, ale také jejich směr.



Obr. 34 Detekce a klasifikace šipek

6.4 Detekce ramen křižovatky

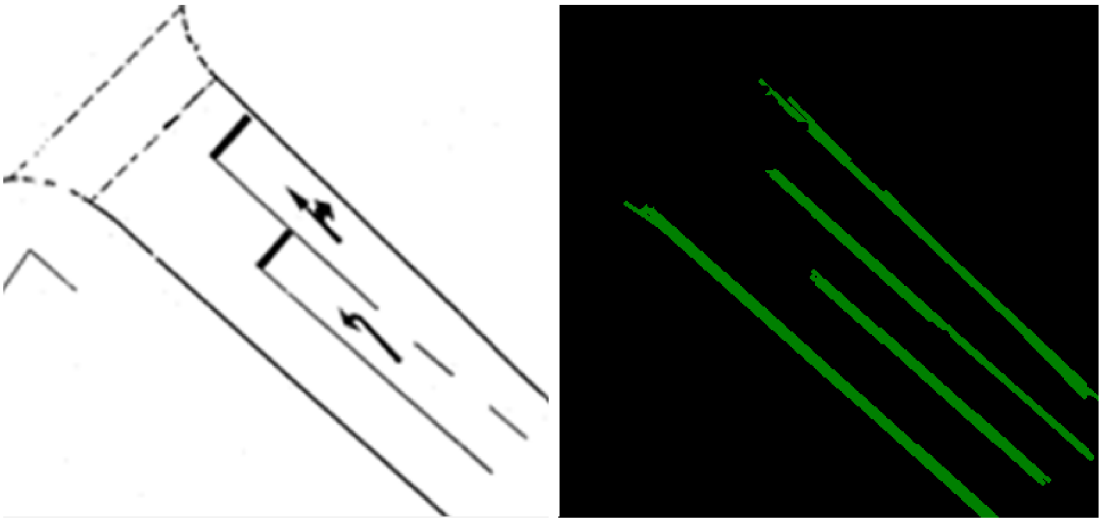
Ramena křižovatky jsou detekována na základě kombinace výstupu Houghovy transformace a detektoru šipek. Pomocí Houghovy transformace rozpoznáme potenciální oblasti, úhel natočení potenciální oblasti pak určíme výpočtem vektoru ze získaných detekovaných přímek v dané oblasti. V okamžiku kdy známe potenciální oblasti, porovnáme je na výskyt šipek, v případě, že se v dané oblasti šipka nachází, prohlásíme oblast za rameno křižovatky.



Obr. 35 Detekce ramen křižovatky

6.5 Detekce jízdních pruhů

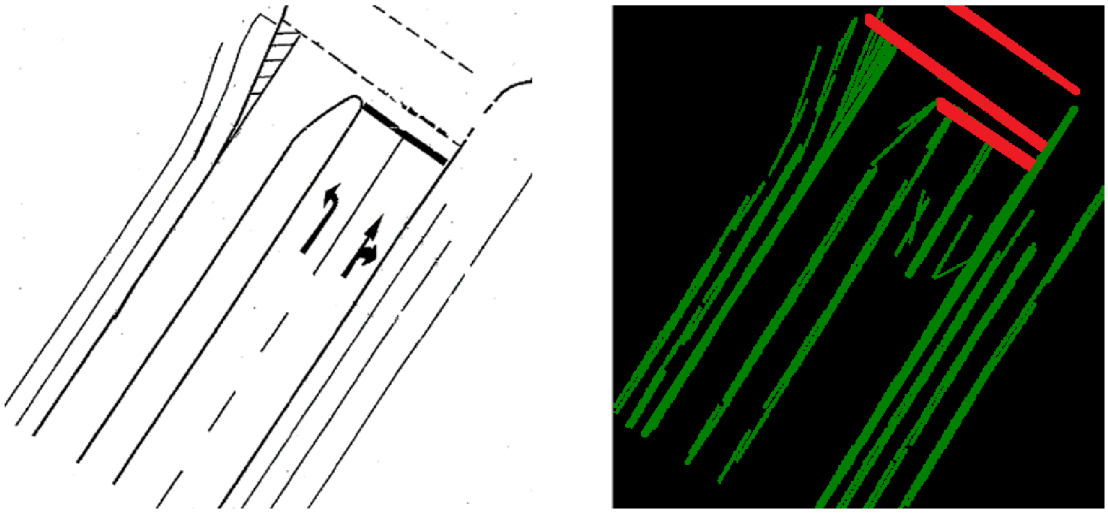
Na vstupu máme rameno křžovatky. Pomocí Houghovy transformace detekujeme přímky – zajímají nás přímky v přibližně stejném úhlu jako je osa ramene. Takto nalezené přímky označují hranice jízdních pruhů. Počet jízdních pruhů odpovídá $n - 1$, kde n je počet hranic jízdních pruhů. Mezi sousedními hranicemi zkusíme detekovat šipku. Pokud je detekce úspěšná určíme její tvar a tím směr daného jízdního pruhu. Pokud šipka detekovaná není, jde o výstupní jízdní pruh.



Obr. 36 Detekce přímek označujících hranice jízdních pruhů pomocí Houghovy transformace

6.6 Detekce přechodů

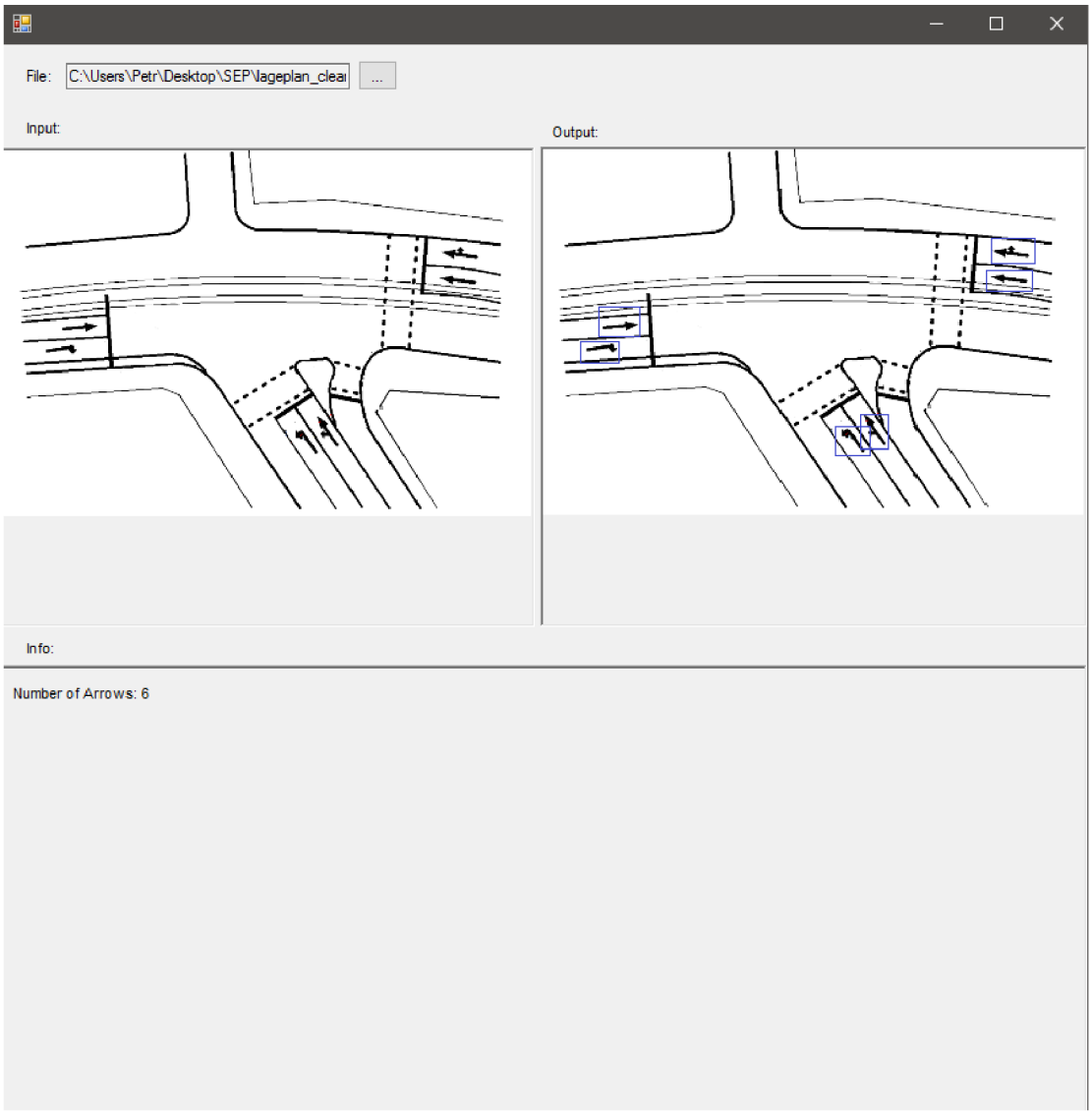
Na vstupu máme rameno křžovatky. Pomocí Houghovy transformace detekujeme přímky – zajímají nás přímky v úhlu přibližně 90° k ose ramene. Pokud nalezneme dvě přibližně stejné a zároveň v první polovině blíže ke středu křižovatky, prohlásíme, že jsme nalezli přechod pro chodce.



Obr. 37 Detekce přímek označujících přechod pro chodce pomocí Houghovy transformace

6.7 Testovací aplikace

Za účelem testování a prezentování výsledků práce byla vytvořena pro každou samostatnou jednotku detekce testovací aplikace. GUI bylo implementováno pomocí Windows Forms. Aplikace umožňuje nahrát a zobrazit vstupní soubor a zobrazit výstup a případné další informace o detekovaných objektech.



Obr. 38 Ukázka hlavního okna testovací aplikace

7 Zhodnocení výsledků

V rámci zhodnocení výsledků implementovaných a experimentovaných přístupů bylo pohlíženo na více možných hledisek. Většina testů byla prováděna během experimentování a to za účelem objevení, zjištění a ověření nejúčinnějších postupů pro rozpoznávání topologických informací z plánu křižovatky. Mimo samotné rozpoznávání topologický informací jsem se zaměřil také na postupy vedoucí k co nejlepšímu předzpracování vstupního obrazu pro každý funkční prvek detekce. Šlo hlavně o nastavení klíčových parametrů s ohledem na kvalitu i rychlost získaného výsledku.

Následující text popíše vykonané testy, jejich metody a popis výsledků. První podkapitola se zaobírá testováním předzpracování vstupního obrazu, následuje testování detekce a odstranění doprovodného textu a kapitola je uzavřena podkapitolou o testování samotné detekce topologických objektů křižovatky. V textu jsou uvedeny tabulky s dosaženými hodnotami úspěšnosti detekce. Tato úspěšnost vyjadřuje nejlepší dosažené výsledky v závislosti na experimentování s hodnotami vstupních parametrů jednotlivých algoritmů.

7.1 Zhodnocení předzpracování vstupního obrazu

Zhodnocení kvality předzpracování vstupního obrazu je poměrně složitý úkol. Není totiž možné snadno a přímo určit, zda výsledek předzpracování vstupního obrazu povede k lepší detekci objektů v obraze. Vyhodnocení tedy probíhalo v rámci každé samostatné funkční jednotky společně s vyhodnocením dané detekce.

Testování se v tomto případě tedy nejprve zaměřilo na optickou kontrolu dosažených výsledků, které bylo následováno kontrolou výsledků jednotlivých detekcí v závislosti na nastavení parametrů předzpracování obrazu. Byl především kladen důraz na úspěšnost detekce v návaznosti na volbě parametrů morfologických operátorů a volbě velikosti objektů u segmentace podle velikosti. Výsledky jednotlivých funkčních jednotek a detektorů jsou popsány v následujících kapitolách.

7.2 Zhodnocení detekce a odstranění textu

Za účelem zhodnocení úspěšnosti detekce a odstranění textu ve vstupních datech byl připraven data set obrázků obsahující různé výřezy z testovacích vstupních rastrů a zároveň obsahující text (Set_1 a Set_2). Tento text byl v obrázcích anotován, aby bylo později možné zjistit úspěšnost detekce.

V prvních fázích testování a ladění parametrů algoritmu bylo použito pouze optické kontroly výsledků filtrování textu. Na konci pak proběhlo testování, jehož výsledky lze vidět v následující tabulce:

Data set	Počet testovaných souborů	Úspěšnost (%)
Set_1	10	55.32
Set_2	20	38.50

Nižší dosaženou úspěšnost u druhého data setu si lze vysvětlit volbou obrázků s textem překrývajícím se s jinými objekty křižovatky. Toto je však poměrně častý jev, který tak reálně snižuje úspěšnost odstranění nepotřebných doprovodných textů.

7.3 Zhodnocení detekce topologických objektů

7.3.1 Detekce šipek

Za účelem zhodnocení úspěšnosti detekce šipek byl připraven data set obrázků obsahující různé výřezy z testovacích vstupních rastrů a zároveň obsahující šipky (Set_1). Další data sety byly vytvořeny z kompletních obrázků vstupních rastrů (Set_CompleteImages) a z obrázků jednotlivých ramen (Set_Legs). Všechny šipky v těchto obrázcích byly anotovány, aby bylo později možné zjistit úspěšnost detekce.

V prvních fázích testování a ladění parametrů algoritmu bylo použito pouze optické kontroly výsledků detekce šipek. Na konci pak proběhlo testování, jehož výsledky lze vidět v následující tabulce:

Data set	Počet testovaných souborů	Úspěšnost (%)
Set_CompleteImages	5	42.84
Set_Legs	20	55.30
Set_1	20	66.72

Je možné si povšimnout nižší úspěšnosti u obrázků obsahující kompletní obrázky plánů křižovatek. To je dáno nejspíše vysokým obsahem šumových informací, které se i přes využití filtračních algoritmů při předzpracování obrazu, nepodařilo dokonale odstranit. Nejlepšího výsledku je dosaženo u data setu obsahujícím výřezy šipek ze vstupních rastrů – šipky zde byly poměrně jasně oddělené od dalších objektů

7.3.2 Detekce ramen křižovatky

Za účelem zhodnocení úspěšnosti detekce ramen křižovatky byl připraven data set z kompletních obrázků vstupních rastrů (Set_CompleteImages). Všechna ramena v těchto obrázcích byla anotována, aby bylo později možné zjistit úspěšnost detekce.

V prvních fázích testování a ladění parametrů algoritmu bylo použito pouze optické kontroly výsledků detekce jízdních pruhů. Během tohoto experimentování se dospělo do stavu, kdy, nezávisle na vstupních parametrech, nebylo možné spolehlivě detekovat vyšší procento ramen. Pro ověření zvoleného postupu byly snímky manuálně vyčištěny od nadbytečných informací a byl z nich připraven data set (Set_CompleteFilteredImages). I přes to však výsledky nejsou příliš přesvědčivé. Důvodem je především vysoký obsah šumu ve vstupních datech a zřejmě ne příliš vhodně zvolený postup detekce. Výsledky testování lze vidět v následující tabulce:

Data set	Počet testovaných souborů	Úspěšnost (%)
Set_CompleteImages	10	5.72
Set_CompleteFilteredImages	10	28.73

7.3.3 Detekce jízdních pruhů

Za účelem zhodnocení úspěšnosti detekce jízdních pruhů byl připraven data set z obrázků jednotlivých ramen (Set_Legs). Všechny jízdní pruhy v těchto obrázcích byly anotovány, aby bylo později možné zjistit úspěšnost detekce.

V prvních fázích testování a ladění parametrů algoritmu bylo použito pouze optické kontroly výsledků detekce jízdních pruhů. Na konci pak proběhlo testování, jehož výsledky lze vidět v následující tabulce:

Data set	Počet testovaných souborů	Úspěšnost (%)
Set_Legs	20	65.30

7.3.4 Detekce přechodů

Za účelem zhodnocení úspěšnosti detekce přechodů byl připraven data set z obrázků jednotlivých ramen (Set_Legs). Všechny přechody v těchto obrázcích byly anotovány, aby bylo později možné zjistit úspěšnost detekce.

V prvních fázích testování a ladění parametrů algoritmu bylo použito pouze optické kontroly výsledků detekce přechodů. Na konci pak proběhlo testování, jehož výsledky lze vidět v následující tabulce:

Data set	Počet testovaných souborů	Úspěšnost (%)
Set_Legs	20	39.72

Detekce přechodů se i přes různé hodnoty vstupních parametrů ukázala jako velmi obtížná. Tato skutečnost je zřejmě dána faktem, že každý přechod je v poskytnutých datech reprezentován pouze dvěma přerušovanými čarami, které jsou snadno zaměnitelné s okolním šumem.

8 Závěr

V rámci diplomové práce byly nastudovány, popsány a vysvětleny metody pro zpracování obrazu za účelem rozpoznávání topologických informací z rastrového plánu křižovatky. Dále byly nastudovány stávající řešení zabývající se vektorizací map a detekcí vodorovného dopravního značení a byl popsán software Sitraffic smartCore a Sitraffic Office co by budoucí cílové platformy pro integraci výsledků práce. V návrhu řešení práce byly diskutovány přístupy jak efektivně detekovat křižovatku, její ramena, jízdní pruhy, přechody a relativní informace týkající se detekovaných objektů.

Praktická část se zaměřila na implementaci a experimenty s navrženými postupy. Tyto experimenty byly implementovány v jazyku C# s využitím knihovny EmguCV a AForge.NET. Byly naimplementovány postupy pro předzpracování vstupního obrazu, kdy byl kladen důraz na odstranění nerelevantních doprovodných textů a jiných šumových informací. V rámci samotné detekce bylo implementováno rozpoznávání šipek, ramen křižovatky, jízdních pruhů a přechodů. Ukázalo se, že samotné předzpracování obrazu je naprosto klíčové pro úspěšnou detekci. Bohužel to znamenalo, že úspěšnost rozpoznání u poskytnutých testovacích dat, která obsahují mnoho nadbytečných informací a šumu, je poměrně nízká a to i přes implementované přístupy předzpracování obrazu. Samotné algoritmy pro rozpoznání jízdních pruhů či přechodů jsou tedy v rámci možností úspěšné pouze pro kvalitní snímky plánů křižovatek, které neobsahují nadbytečné informace ani šum způsobený skenováním obrazu, případně pak pokud je během detekce využito akce uživatele např. pro označení ramen nebo některých šipek v jízdních pruzích. Ani jedna z možností však není vhodná pro zautomatizování
a ulehčení práce dopravních inženýrů.

Další vývoj práce je možné směrovat směrem k funkčnějším metodám detekci ramen křižovatky a k dokonalejšímu odstranění nadbytečných informací v obraze. Je však otázkou, zda se získané výsledky přiblíží očekávaným a zda by nebylo lepší v rámci automatické detekce zapojit interakci uživatele – dopravního inženýra. Tento směr dalšího vývoje by vyžadoval analýzu uživatelského rozhraní, přes které by uživatel pracoval a přes které by zadával potřebné vstupy. To vše za cenu že by již nešlo o plně automatickou detekci objektů křižovatky. Pokud se zmíněné nedostatky odstraní a získá se tak vyšší úspěšnost detekce a rozpoznávání, bude možné navrhované postupy integrovat do stávajícího řešení software Sitraffic smartCore nebo Sittrafic Office.

Literatura

[1] Szeliski Richard: Computer Vision: Algortihms and Applications [online]. [cit. 2016-05-23]. Dostupné na URL: < http://szeliski.org/Book/drafts/SzeliskiBook_20100903_draft.pdf >

[2] Wikipedia: Generalised Hough Tranform [online]. [cit. 2016-05-23]. Dostupné na URL: <http://en.wikipedia.org/wiki/Generalised_Hough_transform>

[3] OpenCV: Morphological transformations [online]. [cit. 2016-05-23]. Dostupné na URL: <http://docs.opencv.org/doc/tutorials/imgproc/opening_closing_hats/opening_closing_hats.html >

[4] Morfologické operace [online]. [cit. 2016-05-23]. Dostupné na URL: <http://midas.uamt.feec.vutbr.cz/ZVS/Exercise10/content_cz.php>

[5] The Watershed Transformation [online]. [cit. 2016-05-23]. Dostupné na URL: <http://cmm.ensmp.fr/~beucher/wtshed.html>

[6] Straka Stanislav: Segmentace obrazu, Brno, 2009 [online]. [cit. 2016-05-23]. Dostupné na URL: <https://is.muni.cz/th/72784/fi_m/dp.pdf>

[7] Houghova transformace pro detekci přímek na obrázku [online]. [cit. 2016-05-23]. Dostupné na URL: < http://petos.cz/2011/houghova-transformace-pro-detekci-primek-na-obrazku>

[8] Harvey Rhody: Hough Circle Transform [online]. [cit. 2016-05-23]. Dostupné na URL: <https://www.cis.rit.edu/class/simg782/lectures/lecture_10/lec782_05_10.pdf>

[9] Detekce geometrických tvaru, Houghova transformace [online]. [cit. 2016-05-23]. Dostupné na URL: < http://www.uamt.feec.vutbr.cz/~richter/pov/POV_HT_objekty.pdf >

[10] Wikipedia: Geographic information system [online]. [cit. 2016-05-23]. Dostupné na URL: < http://en.wikipedia.org/wiki/Geographic_information_system >

[11] Ojaswa Sharma: A methodology for Raster to Vector Conversion of Colour Scanned Maps [online]. [cit. 2016-05-23]. Dostupné na URL: <http://www2.unb.ca/gge/Pubs/TR240.pdf>

[12] Yao-Yi Chiang, Craig A. Knoblock: General Approach for Extracting Road Vector Data from Raster Maps [online]. [cit. 2016-05-23]. Dostupné na URL: <http://www.isi.edu/integration/papers/chiang11-ijdar.pdf >

[13] Microsoft: Windows Forms [online]. [cit. 2016-05-23]. Dostupné na URL: <https://msdn.microsoft.com/en-us/library/dd30h2yb%28v=vs.110%29.aspx >

[14] Emgu CV: OpenCV in .Net [online]. [cit. 2016-05-23]. Dostupné na URL: <http://www.emgu.com/wiki/index.php/Main_Page >

[15] ReSharper [online]. [cit. 2016-05-23]. Dostupné na URL: <https://www.jetbrains.com/resharper/>

[16] Juránek Roman: Rozpoznávání v obraze [online]. [cit. 2016-05-23]. Dostupné na URL: <http://www.fit.vutbr.cz/study/courses/IKR/public/prednasky/06_rozpoznavani_v_obraze/2013-IKR-Rozpoznavani%20v%20obraze%20-%20detection.pdf >

[17] Wikipedia: Local Binary Patterns [online]. [cit. 2016-05-23]. Dostupné na URL: <https://en.wikipedia.org/wiki/Local_binary_patterns>

[18] OpenCV: Cascade Classifier Training [online]. [cit. 2016-05-23]. Dostupné na URL: <http://docs.opencv.org/2.4/doc/user_guide/ug_traincascade.html >

[19] Wikipedia: AdaBoost [online]. [cit. 2016-05-23]. Dostupné na URL: <https://en.wikipedia.org/wiki/AdaBoost>

[20] Wikipedia: Statistical Classification [online]. [cit. 2016-05-23]. Dostupné na URL: < https://en.wikipedia.org/wiki/Statistical_classification>

[21] Wikipedia: Maximally Stable Extremal Regions [online]. [cit. 2016-05-23]. Dostupné na URL: < https://en.wikipedia.org/wiki/Maximally_stable_extremal_regions>

[22] Vacek S., Schimmel C., Dillmann R.: Road-marking analysis for autonomous vehicle guidance [online]. [cit. 2016-05-23]. Dostupné na URL: <http://ecmr07.informatik.uni-freiburg.de/proceedings/ECMR07_0034.pdf>

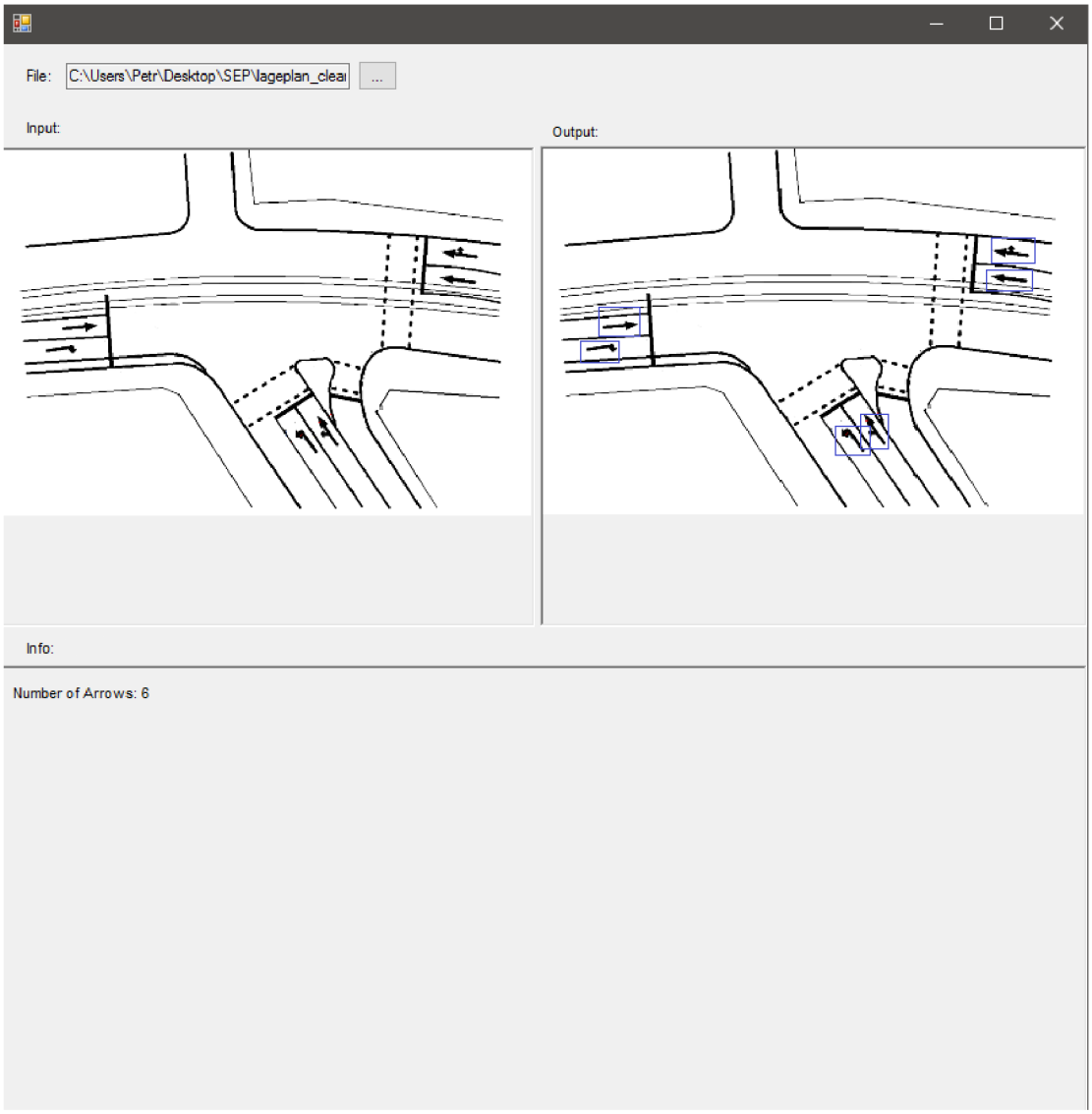
[23] Schreiber M., Poggenhans F., Stiller C.: Detecting Symbols on Road Surface for Mapping and Localization Using OCR. In: Proc. IEEE Int. Conf. Intelligent Transportation Systems, 2014, pp. 597-602

Příloha A – návod k použití

Aplikace byly testovány na 64-bitovém operačním systému Windows 10. Okno všech testovacích aplikací se skládá z:

- Vstupní soubor (File) – pro zadání vstupního rastrového obrazu.
- Levé okno (Input) – zobrazující vstupní soubor
- Pravé okno (Output) – zobrazující výstupní soubor
- Spodní okno (Info) – zobrazující dodatečné informace

Po načtení se automaticky spustí implementované algoritmy. Vstupní soubor se zobrazí v levém okně, výsledek operací v pravém okně a ve spodním okně se zobrazí dodatečné informace



Příloha B – kompilace zdrojových kódů

B.1 Požadavky

Pro úspěšnou kompilaci zdrojových kódů na systému Windows, je nutné mít nainstalováno vývojové prostředí Microsoft Visual Studio 2013 (a novější) a knihovna EmguCV a AForge.NET.

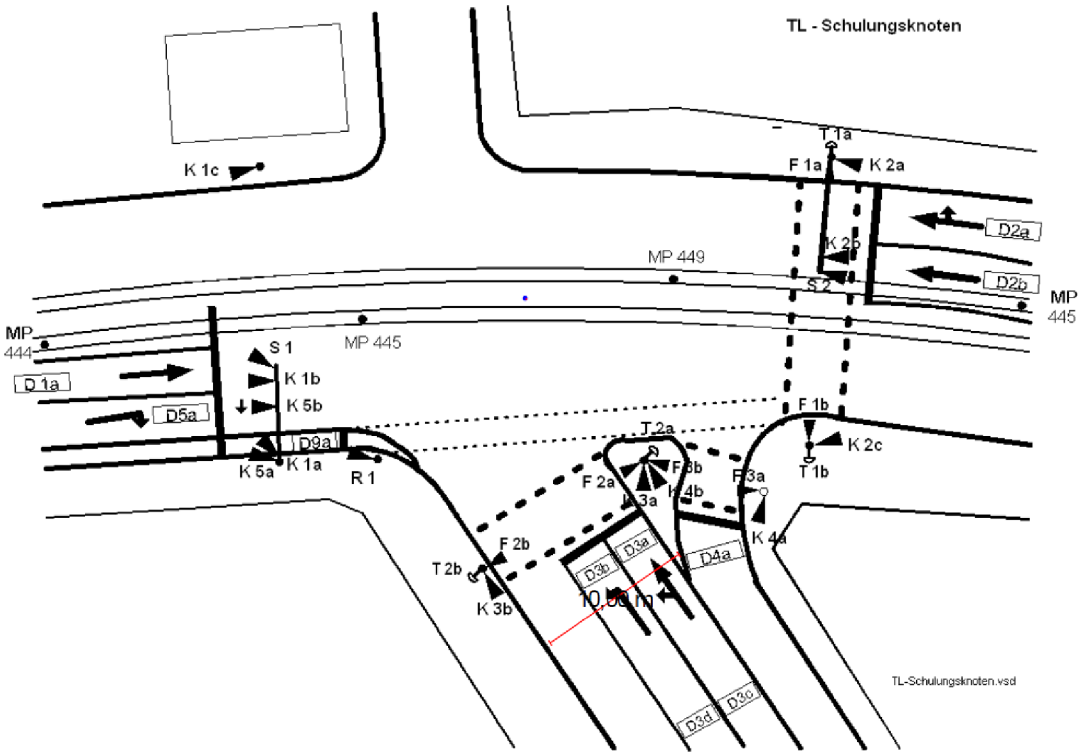
B.2 Kompilace

Projektové soubory testovacích aplikací lze otevřít ve vývojovém prostředí Microsoft Visual Studio 2013 (a novější), stejně tak zde lze spustit i jejich kompilaci.

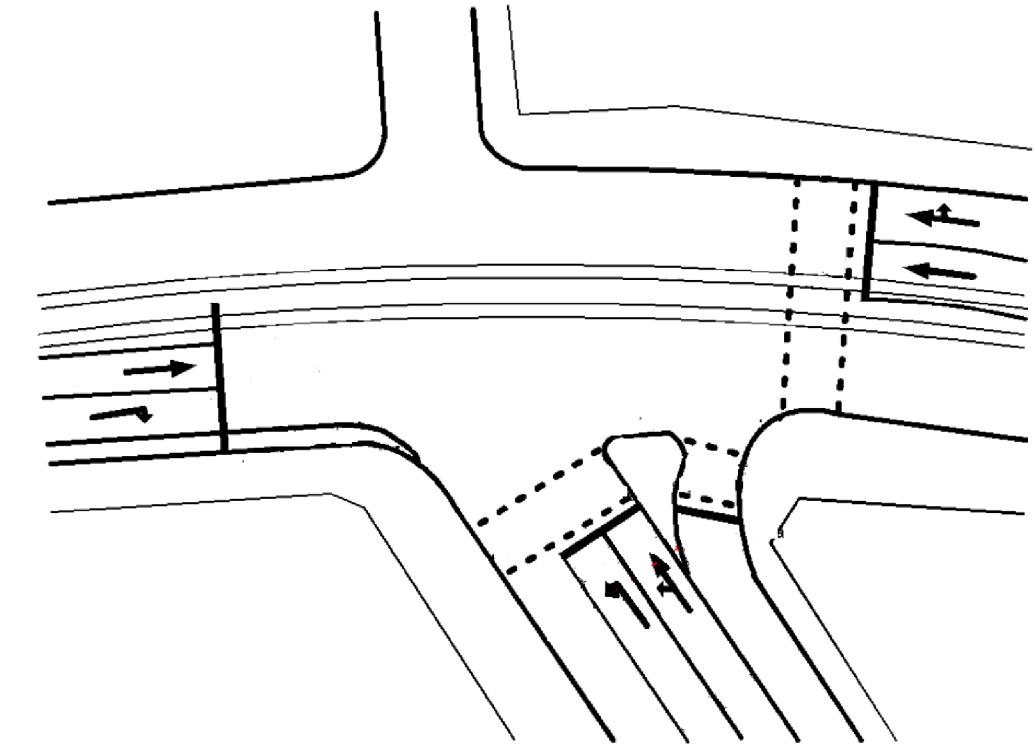
Příloha C – příklady testovacích dat

- Testovací data 1:

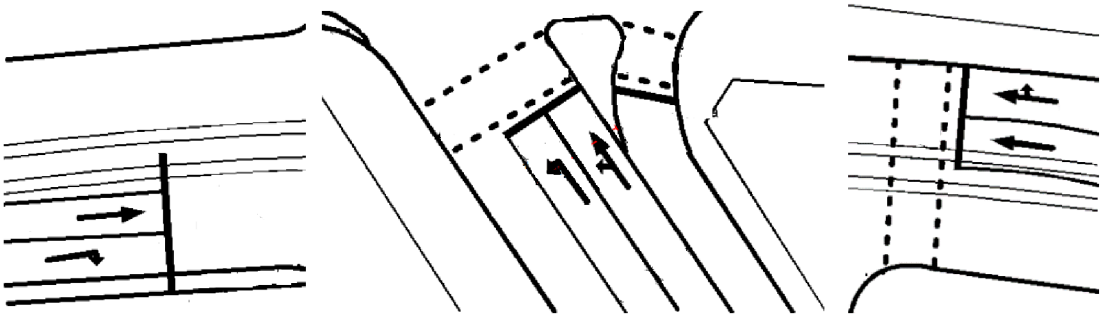
Originální vstupní rastrový obraz:



Obraz zbavený nadbytečných informací:



Ramena:



Anotace:

Křižovatka:

- Number of legs: 3

1. Rameno křižovatky:

- Angle: 355.00
- Number of lines: 3

Jízdní pruh A

- Way: Straight

Jízdní pruh B

- Way: Right

Jízdní pruh C

- Way: Out

2. Rameno křižovatky:

- Angle: 225.00
- Number of lines: 4

Jízdní pruh A

- Way: Left

Jízdní pruh B

- Way: StraightLeft

Jízdní pruh C

- Way: Right

Jízdní pruh D

- Way: Out

3. Rameno křižovatky:

- Angle: 185.00
- Number of lines: 3

Jízdní pruh A

- Way: StraightRight

Jízdní pruh B

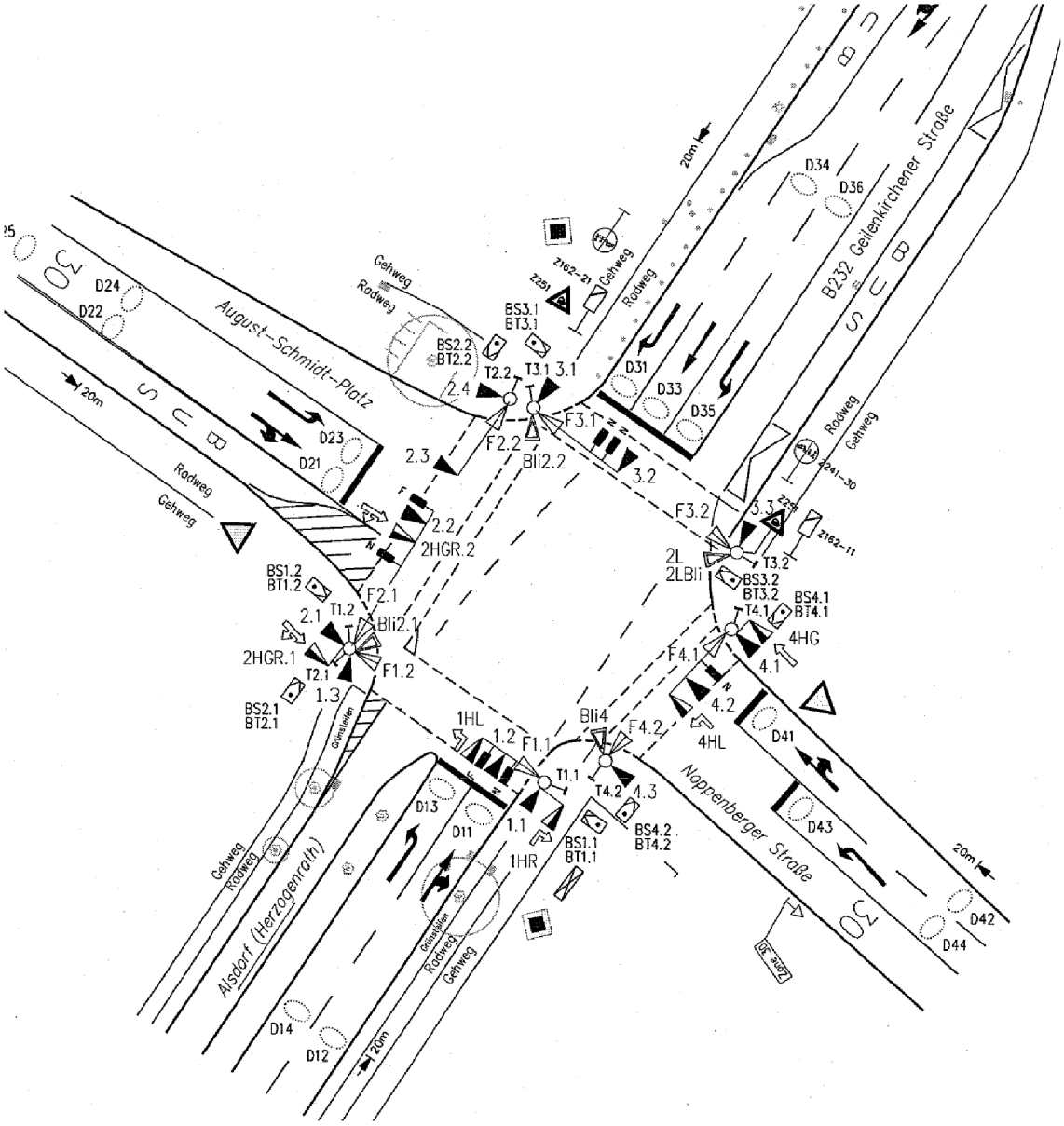
- Way: Straight

Jízdní pruh C

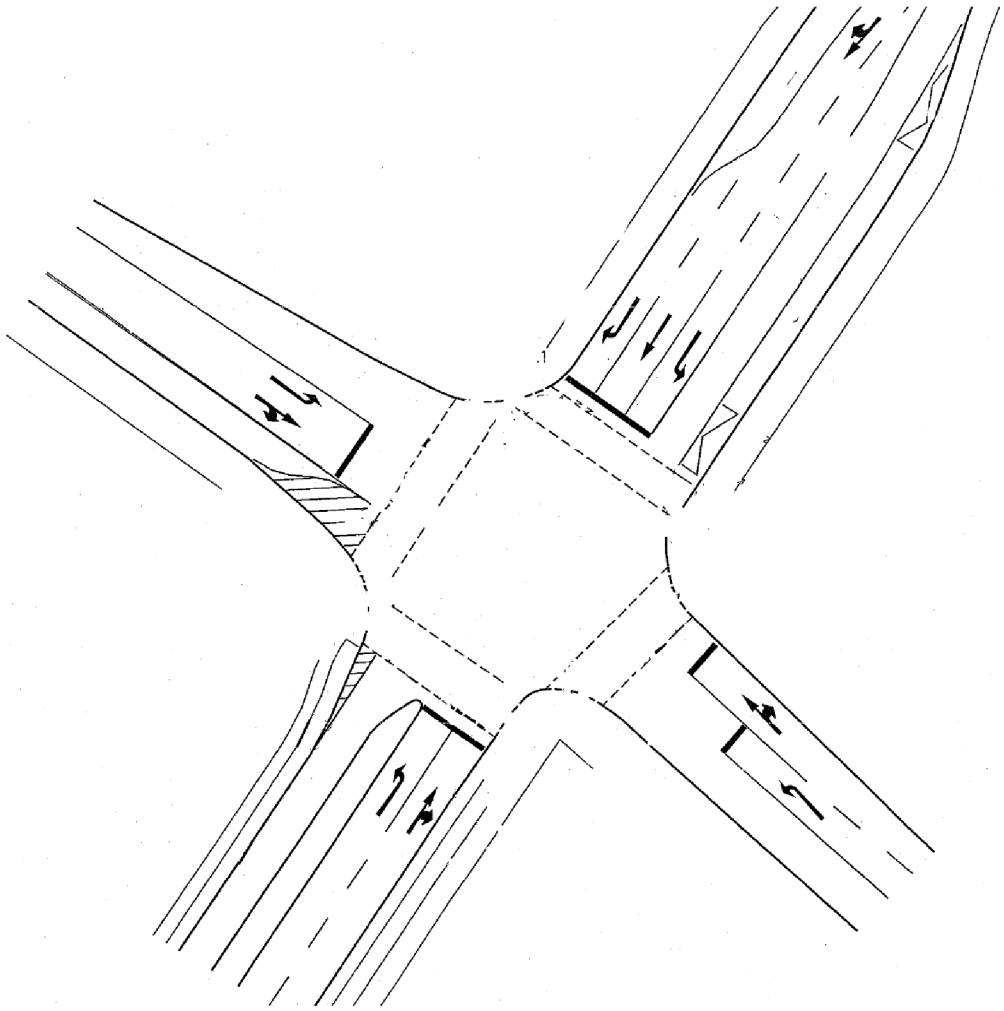
- Way: Out

- Testovací data 2:

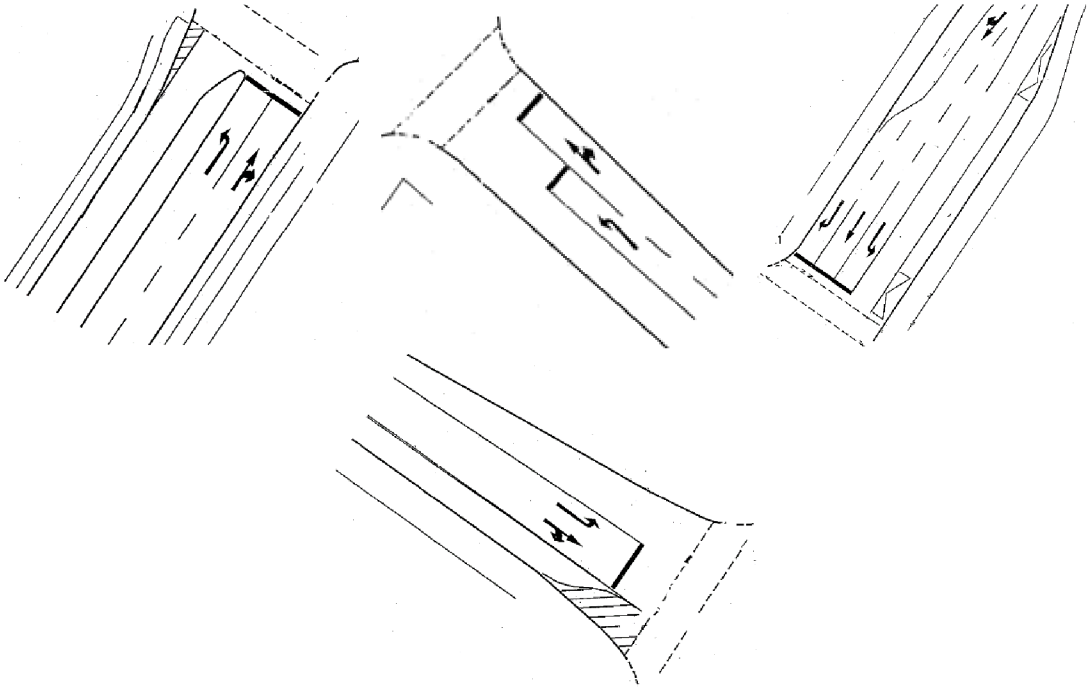
Originální vstupní rastrový obraz:



Obraz zbavený nadbytečných informací:



Ramena:



Anotace:

Křižovatka:

- Number of legs: 4

1. Rameno křižovatky:

- Angle: 310.00
- Number of lines: 3

Jízdní pruh A

- Way: Left

Jízdní pruh B

- Way: StrightRight

Jízdní pruh C

- Way: Out

2. Rameno křižovatky:

- Angle: 225.00
- Number of lines: 3

Jízdní pruh A

- Way: StraightRight

Jízdní pruh B

- Way: Left

Jízdní pruh C

- Way: Out

3. Rameno křižovatky:

- Angle: 130.00
- Number of lines: 4

Jízdní pruh A

- Way: Left

Jízdní pruh B

- Way: Straight

Jízdní pruh C

- Way: Right

Jízdní pruh D

- Way: Out

4. Rameno křižovatky:

- Angle: 40.00
- Number of lines: 3

Jízdní pruh A

- Way: Left

Jízdní pruh B

- Way: StraightRight

Jízdní pruh C

- Way: Out

Příloha D – obsah přiloženého CD

Soubory obsažené na přiloženém CD:

- `app` – spustitelné soubory testovacích aplikací
- `doc` – zdrojové soubory diplomové práce ve formátu DOCX
- `pdf` – text diplomové práce ve formátu PDF
- `source` – zdrojové soubory testovacích aplikací
- `testexample` – testovací data pro testovací aplikace