Vývoj nástroje zpracování obrazu pro vyčítání registračních značek silničních vozidel

Bc. Martin Mikel

Diplomová práce
2017

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2016/2017

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Martin Mikel**
Osobní číslo:	**A15195**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Bezpečnostní technologie, systémy a management**
Forma studia:	**prezenční**
Téma práce:	**Vývoj nástroje zpracování obrazu pro vyčítání registračních značek silničních vozidel**
Téma anglicky:	**The Developement of a Licence Plate Recognition Image Processing Device**

Zásady pro vypracování:

1. **Pojednejte o problematice zpracování obrazu.**
2. **Proveďte komparaci nástrojů pro vývoj aplikací zpracování obrazu.**
3. **Pojednejte o relevantních knihovnách pro manipulaci s obrazem.**
4. **Analyzujte kritické vlastnosti Licence Plate Recognition.**
5. **Vytvořte Licence Plate Recognition.**

Rozsah diplomové práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. **ŽÁRA, Jiří. Moderní počítačová grafika. 2., přeprac. a rozš. vyd. Praha: Computer Press, 2004. ISBN 80-251-0454-0.**
2. **RUSS, John C. a J. Christian. RUSS. Introduction to image processing and analysis. Boca Raton, FL: CRC Press, c2008. ISBN 0849370736.**
3. **FORSYTH, David a Jean PONCE. Computer vision: a modern approach. Upper Saddle River: Prentice Hall, c2003. Prentice Hall series in artificial intelligence. ISBN 0-13-085198-1.**
4. **SONKA, Milan., Vaclav. HLAVAC a Roger. BOYLE. Image processing, analysis, and machine vision. 2nd ed. Pacific Grove, CA: PWS Pub., c1999. ISBN 053495393X.**
5. **Communications of the ACM: a monthly publication of the ACM Publications Office. New York: Association for Computing Machinery. ISSN 0001-0782.**

Vedoucí diplomové práce: **Ing. Jiří Ševčík**
Ústav bezpečnostního inženýrství

Datum zadání diplomové práce: **3. února 2017**

Termín odevzdání diplomové práce: **24. května 2017**

Ve Zlíně dne 3. února 2017

doc. Mgr. Milan Adámek, Ph.D.
děkan

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

doc. RNDr. Vojtěch Křesálek, CSc.
ředitel ústavu

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 připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- 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/bakalářské 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, dne 18.5.2017

.....
podpis diplomanta

ABSTRAKT

Tato práce je zaměřena na zpracování a rozpoznání obrazu. V teoretické části jsou představeny základní pojmy, metody a postupy z problematiky analýzy obrazu. Dále se práce zabývá rozborem a implementací aplikace, která má za úkol vyčíst z obrazu osobního vozidla státní poznávací značku. Součástí jsou i výsledky testování a hodnocení spolehlivosti tohoto systému.

Klíčová slova: zpracování obrazu, předzpracování, prahování, segmentační techniky, srovnávání se vzorem, MS Visual Studio, C# .NET

ABSTRACT

This work is based on image processing and recognition. The theoretical part introduces the basic concepts, methods and procedures from the problem of image analysis. In addition, the thesis deals with the analysis and implementation of the application, which has the task to read from the image of a passenger vehicle a license plate. Also included are the results of testing and evaluating the reliability of this system.

Keywords: image processing, preprocessing, threshold method, segmentation techniques, template matching, MS Visual Studio, C# .NET

Poděkování

Touto cestou bych velice rád poděkoval svému vedoucímu diplomové práce, panu Ing. Jiřímu Ševčíkovi, za odborné vedení, rady, připomínky a čas, který mi věnoval při vypracovávání této diplomové práce.

Prohlašuji, že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

S aplikacemi využívajícími zpracování obrazu se v dnešní době můžeme setkat ve velkém množství odvětví lidské činnosti. Počítačové vidění se využívá v průmyslu například při kontroly kvality, pro odhalení vad výrobků. Naleznou uplatnění i v medicíně, jako součást kamerového systému (např. pohyb v monitorovaném prostoru), v dopravě, ve vojenství atd. Hlavním důvodem pro častější využívání těchto aplikací je zvýšení automatizace a efektivity práce.

S vývojem výpočetní techniky jde kupředu i zpracování obrazu. Výkonnější HW umožňuje vytvářet rychlejší, spolehlivější a efektivnější aplikace. Díky složitějším algoritmům je možné přenechat na systémech rozpoznání obrazu stále více a více operací.

Uplatnění však nepatří pouze do průmyslového odvětví. Prakticky v každodenním kontaktu s těmito aplikacemi jsou všichni uživatelé tzv. chytrých telefonů. Téměř každý je totiž vybaven fotoaparátem, který pomocí rozpoznání obrazu dokáže detekovat např. úsměv. Samozřejmostí už je sledování obličejů. Dalším příklad nalezneme u herních konzolí. Velké množství her je ovládáno pomocí gest nebo sledování pohybů celé postavy.

Cílem této práce je nahlédnou do problematiky zpracování obrazu a pokusit se vytvořit vlastní jednoduchý systém pro vyčítání registračních značek vozidel, jehož součástí je právě využití analýzy obrazu.

Práce je rozdělena do dvou hlavních částí. V té první, teoretické, jsou popsány základní pojmy a metody zpracování obrazu. Dále jsou zde vyhodnoceny dostupné nástroje pro zpracování obrazu. Nakonec je popsána problematika systému Licence Plate Recognition, základní operace, ze kterých se skládá a jeho využití v praxi.

Praktická část je zaměřena na vlastní vytvořenou aplikaci LPR. Popisuje výběr nástrojů a řešení konkrétní úlohy. Součástí je i podrobný návod na obsluhu vytvořeného nástroje pro vyčítání registračních značek a výsledky z jeho testování na zkušební sadě fotografií. Závěrem jsou vyhodnoceny dosažené výsledky s ohledem na uplatnění v praxi.

I. TEORETICKÁ ČÁST

1 ZPRACOVÁNÍ OBRAZU

Tato kapitola se věnuje problematice zpracování obrazu. Nejprve je důležité matematicky vyjádřit co je to obraz a barevné modely, pomocí kterých je vyjádřen.

Vlastní průběh zpracování a rozpoznání obrazu reálného světa je většinou rozčleněn do několika navazujících kroků. Rozdělení se může lišit vždy podle druhu aplikace. Není vždy nutné, aby byly provedeny všechny kroky postupu. [1]

Základní kroky rozpoznání obrazu:

- Snímání a digitalizace obrazu
- Předzpracování obrazu
- Segmentace obrazu
- Popis objektů
- Klasifikace

1.1 Obraz

Definice obrazu se může podle aplikační oblasti lišit, protože většina disciplín používá definice, které jim nejlépe vyhovují. Obraz lze chápat jako odraz reálného světa na sítnici oka, jako fotografii, obrazovku nebo např. odraz ve vodní hladině. Při formálním vymezení využijeme matematický model, kterým je následují spojitá funkce dvou proměnných. [1]

$$z = f(x, y)$$

1.2 Barevné modely

Všechny viditelné barvy světla lze vyjádřit pomocí chromatického diagramu CIE, ze kterého vycházejí i dále popisované barevné modely. *„Barva je vlastnost objektů spojená s jejich schopností odrážet elektromagnetické vlnění různých vlnových délek.“* [2]

K vyjádření všech barev viditelného spektra by bylo zapotřebí velice složitého modelu a z toho důvodu se využívá omezených modelů, které využívají míchání základních barev.

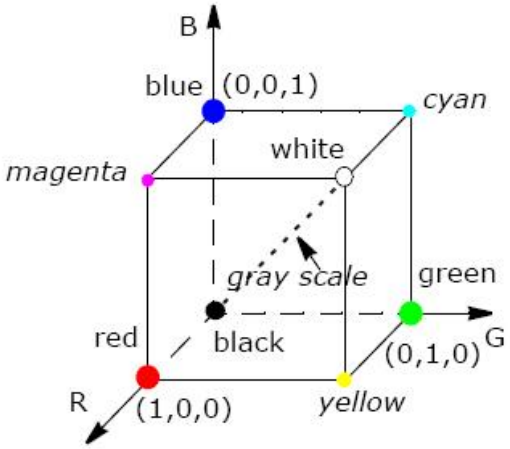
Ve většině případů se jedná o tři barvy, které vzájemně tvoří v chromatickém diagramu trojúhelník. Ten se nazývá gamut a zobrazuje škálu barev, které se dají určitým modelem popsat.

Mezi základní modely se řadí RGB a CMY, ty jsou závislé na zařízení. Oko ale reaguje citlivěji na změnu jasu než na změnu barvy. Přirozenější jsou proto lidskému oku modely, které oddělují jasovou a barevnou složku. Příkladem může být HSV model, který je reprezentovaný šestibokým jehlanem. [3]

1.2.1 RGB model

Při tvorbě obrazu, např. na barevné obrazovce, se využívají barvy, které jsou tvořeny ze tří základních barev z barevného spektra. Jsou to složky červená (Red), zelená (Green) a modrá (Blue). Kteroukoliv barvu lze vyjádřit vektorem, jehož složky obsahují hodnoty z intervalu <0,1>. Častěji však bývají uváděny v celočíselném rozsahu 0 – 255. Tyto hodnoty odpovídají intenzitě každé ze složek RGB v jednom bytu. Nula znamená, že složka není vůbec zastoupena a 255 indikuje, že je složka zastoupena v maximální intenzitě. Vyjádření barvy pomocí tří bytů je v současnosti nejběžnější. [3]

U aditivního modelu RGB se ve výpočetní technice běžně používá osmibitová representace každé barvy. Z toho vyplívá, že pro každou barvu je možné rozlišit 2^8 (256) úrovní. Celkově je tedy k dispozici 2^{24} možných kombinací, které dohromady představují 16 777 216 barev (TrueColors). [4]



Obr. 1 Model RGB [4]

Složením všech tří základních barev v plné intenzitě dostaneme barvu bílou. Rovnoměrným snižováním intenzity lze získat stupně šedi (GrayScale) až k samotné černé barvě. Pro převedení barevného obrazu na stupně šedi však není možné získat prostým průměrem ze tří základních barev. Lidské oko vnímá intenzitu jednotlivých barev různým způsobem. Proto se pro výpočet jasu aplikuje následující empirický vztah. [3]

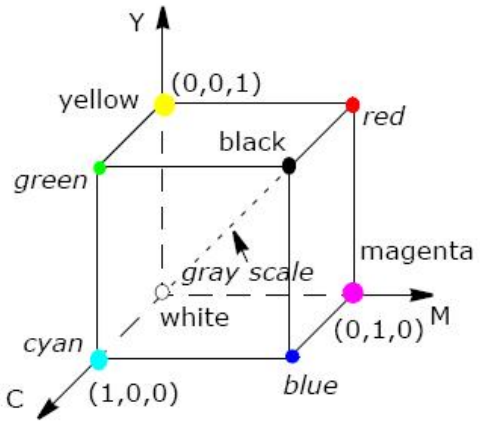
$$I = 0{,}299\ R + 0{,}587\ G + 0{,}114\ B$$

V počítačové grafice se často využívá rozšířený model RGB, kterým je RGBA (nebo RGBα). Jedná se o skutečnost, kdy je obraz zapsán v prostoru RGB a je navíc doplněn informací o průhlednosti. Každý bod sebou nese skalární údaj například v rozmezí 0 až 1 (nebo 0 až 255), který určuje, v jakém rozsahu barva pokrývá plochu obrazového bodu. Minimální hodnota znamená zcela neprůhledný barevný bod, maximální naopak průhledný.[3]

1.2.2 Modely CMY a CMYK

Technicky orientovaný model RGB je vhodný pro displeje. Ovšem lidská zkušenost (např. malíř) s mícháním barev je zcela jiná. Sloučením jednotlivých barevných pigmentů se vytváří nové barvy. Toto skládání se jmenuje subtraktivní. Naopak od aditivního skládání RGB, složením všech barev vznikne černá barva.

Toto složení barev je typické pro tiskařské techniky. CMY je model, který právě těmto účelům vyhovuje. Obsahuje tři základní barvy. Jsou to tyrkysová (Cyan), fialová (Magenta) a žlutá (Yellow). Kombinací těchto tří barev ovšem nevznikne dokonalá černá, a tak se k tomuto modelu přidává jako čtvrtá barva (blacK), kterou lze také využít ke ztmavení ostatních barev. V polygrafii se tak od modelu CMY přechází na model CMYK. [3]



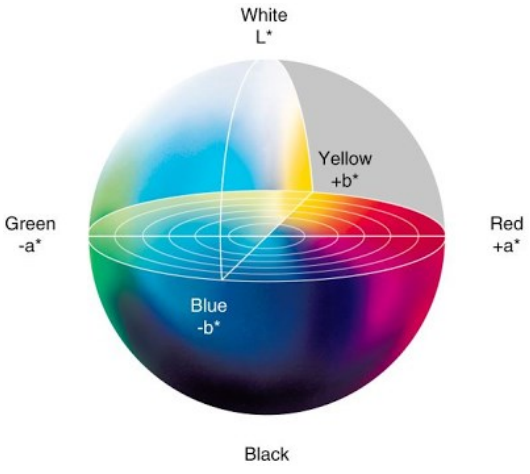
Obr. 2 Model CMY [4]

Pro převod mezi barevnými modely RGB a CMY se aplikuje jednoduchý vzorec. Vektor [c m y] v modelu CMY určíme odčítáním tříprvkových matic [r g b] vyjadřující barevný vektor v modelu RGB.[3]

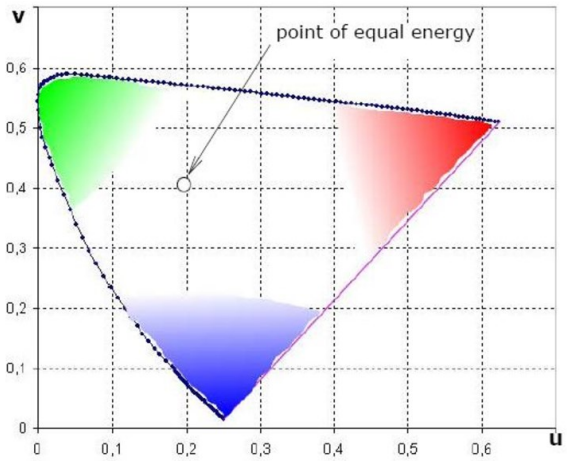
$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

1.2.3 Model CIELAB

V roce 1931 byl definován chromatický diagram CIE. Jelikož je ale nerovnoměrný, byl modifikován v roce 1976 a vznikl tak model CIELAB, reprezentovaný koulí. Jeho svislá osa *L* představuje jas, zatímco dvě horizontální osy *a* a *b* definují barvy. Právě z těchto os byl odvozen název celého modelu (CIE L*a*b). Hodnoty mezi červenou a zelenou barvou stanovuje osa *a*, osa *b* zase hodnoty mezi žlutou a modrou barvou. [3]



Obr. 3 barevný model CIELAB [5]

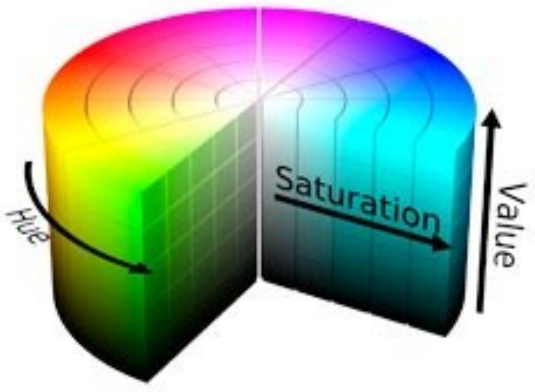


Obr. 4 Barevný prostor CIE (1931) [4]

1.2.4 Model HSV

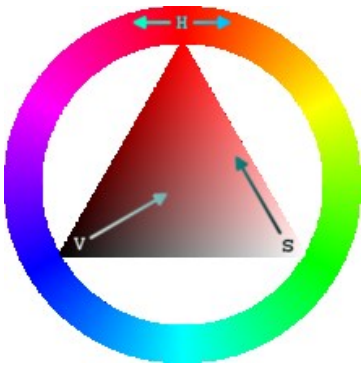
Model HSV (Hue - odstín, Saturation - sytost, Value – hodnota), známy taky jako HSB (Brigness - jas), je barevný model, který nejvíce odpovídá vnímání barev lidským okem. Skládá se ze tří složek, nejedná se však o základní barvy. U těchto složek je nutné hlídat hodnoty, jelikož by mohlo dojít k nesmyslným kombinacím. [4]

Převládající barevný tón (Hue), je barva, která je odražená nebo procházející objektem. Znázorněna je jako poloha na barevném kole (0° až 360°). Saturace představuje sytost barvy, nebo příměs barvy jiné (množství šedi v poměru k odstínu). Hodnota jasu udává množství bílého světla. [6]



Obr. 5 HSV model [6]

Nejčastější použití modelu HSV je v grafických aplikacích, kde je potřeba změnit barvu, která působí specifický grafický element. Za tímto účelem je používám HSV kruh (trojúhelník). [6]



Obr. 6 HSV kruh [7]

1.3 Snímání a digitalizace obrazu

Základním kamenem a zároveň prvním krokem v posloupnosti činností zpracování obrazu je jeho snímání a následný převod do digitální formy vhodné pro uložení a další zpracování. To má zásadní dopad na všechny následující kroky. Při vhodně zvolených podmínkách snímání, kam patří vlastnosti snímacího zařízení, jeho samotná poloha a v neposlední řadě i osvětlení scény, lze pozitivně ovlivnit náročnost následujících kroků. V opačném případě, kdy je obraz nasnímán nevhodně a jeho kvalita je nízká, může dojít k částečnému nebo až k úplnému nezdaru celého procesu.

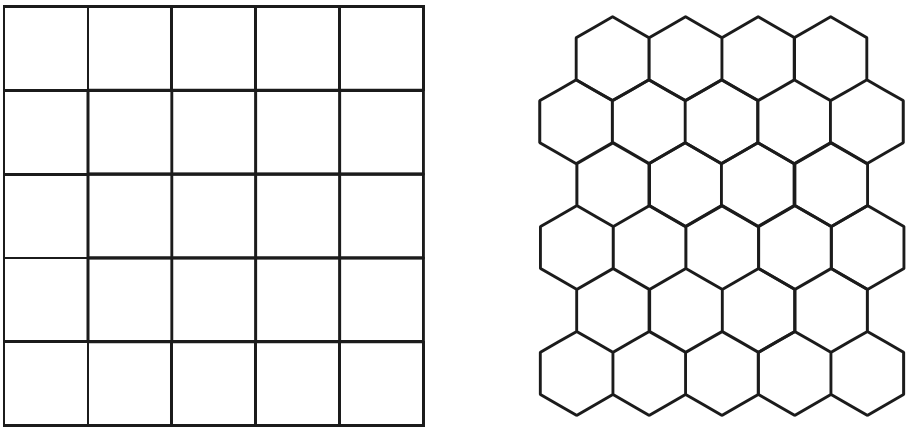
Snímáním obrazu rozumíme převod optické veličiny na elektrický signál, který je spojití v čase i úrovni. Pro další zpracování je nutná digitalizace tohoto signálu. Digitalizaci chápeme jako vzorkování spojité obrazové funkce do matice *m x n* bodů, (kde *m* představuje horizontální rozlišení a *n* vertikální rozlišení) a kvantování *k* do úrovní jasu. Analogová obrazová funkce se tedy převádí na konečný počet obrazových bodů s konečným počtem jasových úrovní. Nejběžnějším příkladem digitalizace snímků představují CCD nebo CMOS snímače, které využívají například digitální fotoaparáty. [1]

1.3.1 Vzorkování

Při vzorkování se postupuje podle Shannonovy věty, která říká, že vzorkovací frekvence musí být alespoň dvakrát větší než nejvyšší frekvence vzorkovacího signálu. Vzorkovací interval hraje důležitou roli. Při nízkém rozlišení (velká vzdálenost vzorkovacích bodů) může docházet ke ztrátě informací o detailech v obraze. Aby se podobným ztrátám předcházelo, měl by být vzorkovací interval menší nebo roven polovině rozměru nejmenších detailů ve snímané scéně. Naopak s rostoucím rozlišením bude stoupat i výpočetní náročnost při dalších operacích s obrazem.

Velikost obrazu se nejčastěji uvádí v obrazových bodech (pixelech). Každý jeden vzorkovací bod představuje právě jeden pixel na vzorkovací mřížce. Standardně se lze setkat se dvěma typy vzorkovacích mřížek. Jsou to mřížky čtvercové a hexagonální.

Čtvercová mřížka nabízí výhodu ve snadné implementaci a je tak součástí většiny snímacích prvků, ale na rozdíl od hexagonální mřížky, kterou tvoří pravidelné šestiúhelníky, je komplikovanější určit vzdálenost jednotlivých bodů. [1]



Obr. 7 Čtvercová a hexagonální mřížka [1]

1.3.2 Kvantování

Kvantování slouží k převodu analogových hodnot jasu obrazové funkce do diskrétního tvaru. Podle původní hodnoty jasu ve vzorkovacím bodě je přiřazena odpovídající kvantovací úroveň. Pokud se jedná o rovnoměrné rozložení kvantovacích

úrovní, hovoříme o tzv. uniformním kvantování. V opačné situaci jde o neuniformní kvantování.

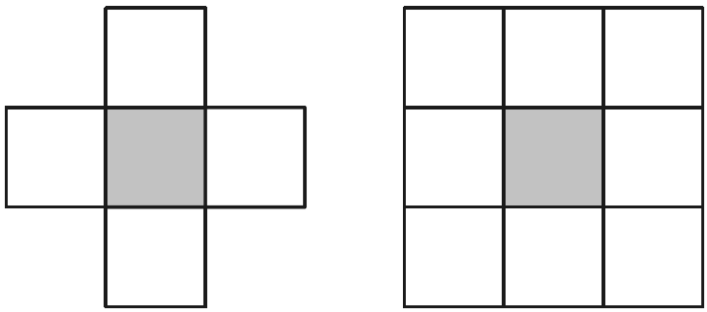
V případě, že by kvantovacích úrovní byl nedostatek, došlo by k hrubému rozložení jasu a velkému snížení rozlišení jasových úrovní. Hodnota počtu kvantovacích úrovní by neměla klesnout pod hranici padesáti úrovní jasu, jelikož přibližně tolik je schopno rozlišit lidské oko. Následující vzorec slouží k určení počtu jasových úrovní *k*, podle hodnoty *b*, která představuje počet bitů reprezentující jeden obrazový bod. Při reprezentaci jednoho pixelu pomocí 8 bitů je počet výsledných jasových úrovní 256. [2]

$$k = 2^b$$

Před další manipulací s obrazem je důležité definovat vzdálenost mezi dvěma obrazovými body *(i,j)* a *(x,y)*. Obecně je vzdálenost udávána podle Eukleidovské vzdálenosti D_E, která odpovídá následujícímu vztahu.

$$D_E = \sqrt{(x-i)^2 + (y-j)^2}$$

V případě, že je ale obraz diskretizován (nespojitý), je nutné nejprve definovat sousedství bodu a následně pak na základě sousedství vzdálenost bodů. U čtvercové mřížky lze počítat se sousedstvím čtyř nebo osmi bodů. [2]



Obr. 8 Možná sousedství ve čtvercové mřížce [2]

Vzdálenost dvou různých bodů poté vychází z následujících vztahů:

$$D_4 = |x - i| + |y - j|$$

$$D_8 = max\{|x - i|, |y - j|\}$$

1.4 Předzpracování obrazu

Při snímání obrazu a jeho následné digitalizaci dojde ve většině případů k poškození obrazu, nebo ke snížení jeho kvality. Jedná se především o šum nebo zkreslení obrazu. Tyto nedostatky se snaží minimalizovat pomocí předzpracování obrazu. Dá se tedy říct, že cílem je opětovně zvýšit kvalitu obrazu a to hlavně z hlediska následujícího zpracování. Důraz se klade na zvýraznění rysů, se kterými se má později pracovat, naopak se potlačují ty, které jsou k dosažení požadovaného výsledku nepodstatné. [8]

Základní metody pro předzpracování obrazu:

- Jasová transformace
- Geometrická transformace
- Filtrace a ostření
- Morfologické operace
-

1.4.1 Jasová transformace

Jednou z jednoduchých transformací sloužící k předzpracování obrazu je transformace jasová. Hlavním úkolem je pracovat s hodnotami jednotlivých obrazových bodů a podle transformační funkce jim přiřadit novou hodnotu jasu. Nejznámější příklad jasové transformace je negativ, kterého lze dosáhnout použitím následující funkce. [8]

$$g(x, y) = 1 - f(x, y)$$

Četnost zastoupení jasových úrovní v obraze se znázorní pomocí grafu, který se nazývá histogram. Jasovou úroveň v tomto grafu reprezentuje horizontální osa, zatímco osa vertikální znázorňuje počet pixelů v obraze, které odpovídají dané hodnotě jasu.

Hlavní využití histogramu je pro opravu nekvalitně exponovaných scén. Další uplatnění nalezne u automatického prahování obrazu. Pomocí tohoto grafu lze jistým způsobem popsat snímanou scénu, ale z hlediska rozpoznání obrazu nemá žádnou vypovídající hodnotu. Každý obraz má pouze jeden histogram, ovšem více snímků může mít shodný histogram.[8]

$$N = \sum_{k=0}^{l} h_k$$

Tento vzorec definuje histogram, kde N odpovídá celkovému počtu pixelů, h_k znázorňuje počet pixelů odpovídajících hodnotě jasu k a k až l je rozsah jasu.

Obr. 9 Příklad histogramu registrační značky [8]

1.4.2 Geometrická transformace

Aplikováním geometrických transformací se dosáhne cíleného geometrického zkreslení nebo opravy obrazu. Při nesprávné poloze snímacího zařízení, nebo při jeho nižší kvalitě, může dojít k deformaci objektů ve snímané scéně. Pro případnou korekci obrazu slouží právě geometrické transformace. [2]

Mezi základní geometrické transformace patří:

- Posunutí obrazu
- Změna měřítka obrazu
- Otočení obrazu

Výsledné hodnoty geometrických transformací jsou souřadnice obrazových bodů, ale jelikož nejsou v celočíselné podobě, je nutné je převést do diskrétní podoby interpolací. Nejsnadnější metodou je interpolace na nejbližšího souseda, mezi sofistikovanější patří bilineární nebo kubická interpolace. [2]

Posunutí bodu *A* o vektor t reprezentuje vztah:

$$x' = x + t_x,$$

$$y' = y + t_y,$$

kde *x, y* představují původní souřadnice bodu a *x´, y´* souřadnice nově vzniklého bodu, který e posunutý o vektor *t*.

Maticový zápis usnadní práci s transformacemi. Pro jejich skládání postačí vynásobit transformační matice jednotlivých transformací. Matici, která tímto násobením vznikne, se pak aplikuje na jednotlivé body. Výsledná transformace se tak počítá pouze jednou a předchází se tak zbytečnému počítání jednotlivých transformací samostatně. [9]

Pro **změnu měřítka** se uplatňuje níže uvedený vztah, kde s_x představuje změnu ve směru *x* a s_y ve směru *y*. [9]

$$x' = x * s_x,$$

$$y' = y * s_y,$$

Maticový zápis změny měřítka pak udává následující vztah:

$$[x'\ y'\ 1] = [x\ y\ 1]\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Otočení obrazu o úhel φ podle počátku definuje vztah:

$$x' = x\cos\varphi - y\sin\varphi,$$
$$y' = x\sin\varphi + y\sin\varphi.$$

Maticový zápis pak vypadá následovně:

$$[x'\ y'\ 1] = [x\ y\ 1]\begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1.4.3 Filtrace a ostření

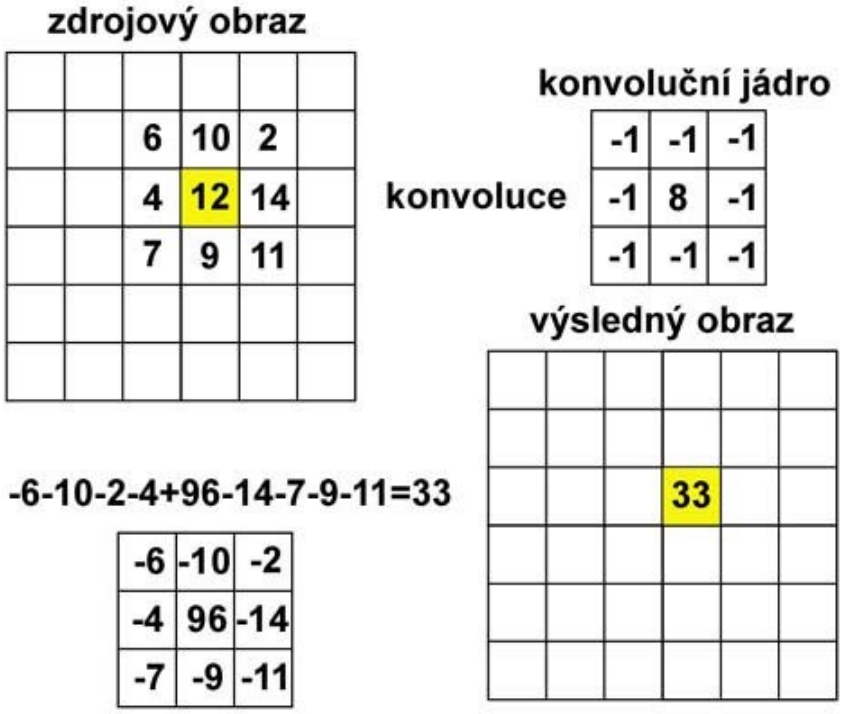
Nepříjemnou součástí obrazu je šum. Před jakoukoliv operací jako je například segmentace je vhodné tento šum odstranit, nebo alespoň potlačit. Do obrazu se šum v drtivé většině případů dostává už při snímání nebo při jeho přenosu a pro úplné odstranění tohoto šumu by bylo zapotřebí znát původní obrazovou informaci. [8]

Šum v obraze představují osamocené pixely, které se výrazným způsobem liší od svého okolí. Právě toho se snaží využít metody pro jeho potlačení. Tyto extrémní body nahrazují pravděpodobnějšími hodnotami, které získají na základě hodnot jasu obrazových bodů v blízkém okolí.

Nežádoucím jevem při potlačování nečistot v obraze může být rozmazání hran, které jsou v obraze reprezentovány také velkým rozdílem jasu. Opačný problém nastává při operacích pro ostření obrazu, s hranami se pochopitelně zvýrazní i šum. [9]

Diskrétní konvoluce je matematická operace, která pracuje se dvěma funkcemi. První funkcí je zpracovávaný obraz a druhou funkcí je jádro (maska), které představuje filtr aplikovaný na originální obraz. Maska je zapisována jako matice. Hodnoty této matice pak mají vliv na různé efekty u výsledného obrazu.

Hodnoty masky určují, jakou váhu budou mít hodnoty daných obrazových bodů na hodnotu výsledného pixelu. Maska se posouvá postupně po celém obraze a násobí tak hodnoty všech pixelů. Následně se takto vynásobené hodnoty sčítají. Je potřeba, aby se součet hodnot v matici rovnal jedné, aby nedocházelo k zesvětlení nebo ztmavování výsledného obrazu. Za tímto účelem je maska vynásobena tzv. koeficientem masky. [10]



Obr. 10 Princim konvolučního filtru [10]

Lineární metodou pro vyhlazování může být například metoda průměrování. Každému bodu je pomocí konvoluce vypočítána průměrná hodnota podle okolních bodů. V takovém případě ale dochází ke znatelnému rozmazání celého obrazu. Je proto vhodné, aby matice konvoluce neobsahovala hodnoty se stejnou váhou. [9]

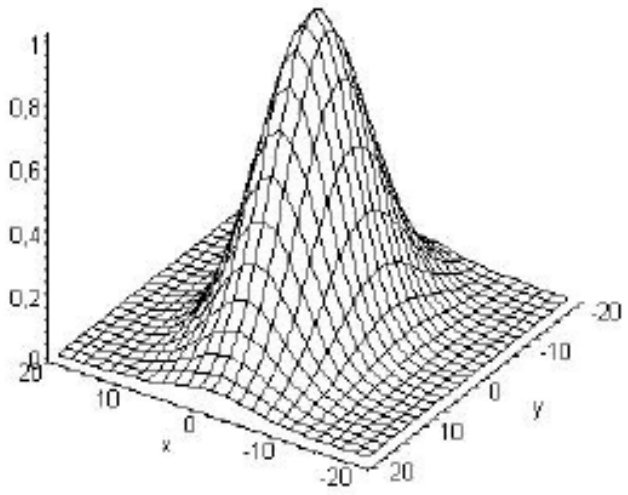
Příkladem vyhlazování s konvolucí, která má hodnoty s různou váhou jednotlivých bodů je **Gaussovo vyhlazování**. Konvoluční maska je odvozena z funkce dvourozměrného Gaussova rozdělení.

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

V tomto rozdělení x a y představují souřadnice obrazového bodu a σ je standardní odchylka (rozptyl). Konvoluční maska pak vypadá následovně: [9]

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Obr. 11 Konvoluční maska [9]



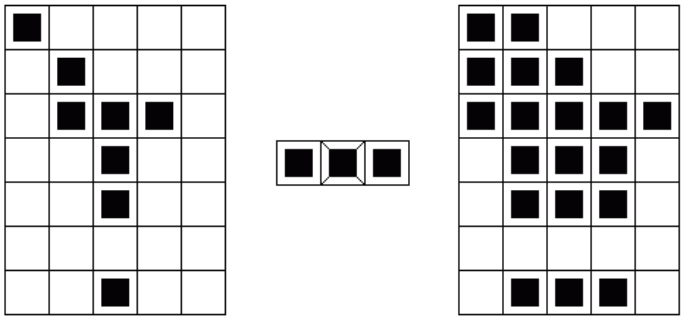
Obr. 12 Gaussovo rozdělení [11]

1.4.4 Morfologické operace

Matematické morfologické transformace pracují s tvary objektů v obraze. Jejich využití je jak v procesu předzpracování obrazu, tak i k jeho segmentaci. Operace pro transformace jsou prováděny s bodovou množinou *X*, která představuje původní obraz a s menší bodovou množinou *B*, představující strukturní element. Mezi základní morfologické operace se řadí dilatace a eroze. Dále třeba otevření nebo uzavření. [12]

Výsledkem dilatace je zvětšený původní objekt. Vyplní se tak například i malé díry. Definicí dilatace je vektorový součet dvou bodových množin. [13]

$$X \oplus B = \{d \in E^2 : d = x + b, x \in X, b \in B\}$$



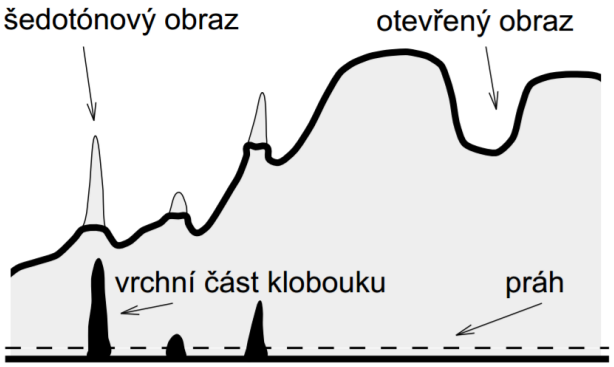
Obr. 13 Jednoduchý příklad dilatace [13]

Opakem dilatace je eroze. Všechny objekty, které jsou menší, než strukturní element zmizí. Následkem toho je celkové zjednodušení struktury objektů. Definicí eroze je rozdíl dvou bodových množin. [13]

$$X \oplus B = \{d \in E^2 : d + b \in X \; pro \; všechna \; b \in B\}$$

Transformace vrchní části klobouku (TopHat) je další příklad morfologické operace. Využívá se k segmentaci objektů v šedotónovém obraze, které se liší jasem a to i v případě, že se jas pozadí mění pomalu. Odstraní se části obrazu, které jsou větší než strukturní element. Po tomto odečtení zůstanou pouze odstraněné oblasti (objekty na

vyrovnaném pozadí). Následně se objekty hledají pomocí prahování. Transformace odstraní jenom vrchní části „klobouku“, tam kde je strukturní element větší než otvor pro hlavu. [14]



Obr. 14 Transformace vrchní části klobouku [14]

Pro vyhledávání naopak tmavých částí v obraze existuje modifikace této operace (*dark top hat transformation*). Provede se odečtením původního obrazu od toho uzavřeného. [12]

1.5 Segmentace obrazu

Úkolem segmentace obrazu je nalezení objektů, které jsou předmětem zájmu a jsou využívány v dalších krocích. Podstatou segmentace je výrazné rozlišení těchto objektů od pozadí. V situaci, kdy je výsledkem nalezení oblastí, které se navzájem nepřekrývají a odpovídají reálným objektům, jedná se o segmentaci kompletní. Pokud oblasti nesouhlasí přesně s objekty reálného světa, jde pouze o segmentaci částečnou. Aby mohla být výsledkem kompletní segmentace, je potřeba znát bližší znalosti reálné scény. Součástí jsou pochopitelně vyšší nároky na výpočty. Částečná segmentace je ve většině případů výsledkem operací, které pracují na principu homogenity oblastí. [8]

Další metody mohou pracovat například na základě hledání hranic objektů. Velice podstatným faktorem, který sebou ještě segmentace přináší, je významné snížení objemu dat, se kterými se následně pracuje. Výsledkem je znatelné snížení celkového času výpočtů.[8]

1.5.1 Prahování

Jedná se o nejstarší a nejjednodušší segmentační metodu. I přes to, že se jedná o metodu s velice omezenou použitelností, je široce používaná a oblíbená. Aplikuje se jak samostatně, tak jako součást jiných složitějších metod. Svou popularitu si získala právě díky své jednoduchosti, snadné implementaci a v neposlední řadě malé časové náročnosti. [9]

Samotná myšlenka metody spočívá v tom, že objekty a pozadí mají rozdílnou úroveň intenzity. Je tedy definována prahová hodnota (T). Každý pixel, který má menší hodnotu, než definuje tento práh, je považován za pozadí a všechny ostatní jsou považovány za pixely objektu. Výsledku metody prahování se tak dosáhne po jediném průchodu obrazu. [8]

$$f(x,y) = \begin{cases} 1\ pro\ g(x,y) \geq T \\ 0\ pro\ g(x,y) < T \end{cases}$$

Existuje několik způsobů, jak prahování využít. Nejznámějším z nich, je globální prahování, kde je pro celý obraz stanoven jeden práh. Nevýhodou tohoto způsobu je možnost nerovnoměrného osvětlení. Jako globální prahování by se dalo chápat i prahování procentní. Práh se v takovém případě neudává jako úroveň šedi, ale jako procentuální zastoupení bodů v obraze, které jsou vyšší nebo rovny nějaké vhodné prahové hodnotě. Nejčastěji se toho využívá při převodu skenovaných dokumentů na text (průměrné pokrytí stránky textem je asi 5%).

Další variantou je poloprahování. Jde v podstatě o klasické prahování, s tím rozdílem, že jednotlivým pixelům, které mají vyšší nebo rovnou hodnotu prahu T, se nepřiřazuje určitá hodnota, ale je jim ponechána jejich vlastní. [8]

$$f(x,y) = \begin{cases} g(x,y)\ pro\ g(x,y) \geq T \\ 0\ pro\ g(x,y) < T \end{cases}$$

Obraz jde taky rozdělit na několik stejných čtverců (obdélníků) a pro každou takto vzniklou oblast je stanoven práh zvlášť. Tento způsob se nazývá adaptivní prahování. Lze tímto postupem částečně kompenzovat nerovnoměrné osvětlení, ovšem problém vzniká v nevyzpytatelném chování na hranicích jednotlivých oblastí.

Další možnou variantou je vícestupňové prahování. Obraz se při tom rozdělí na více než jen dvě množiny. Každý pixel je přiřazen objektu *i* podle toho, mezi kterými prahy leží (T_i a T_{i+1}). [9]

$$f(x,y) = 1 \quad pokud\ g(x,y) \geq T_1\ \&\ g(x,y) < T_2$$

$$f(x,y) = 2 \quad pokud\ g(x,y) \geq T_2\ \&\ g(x,y) < T_3$$

$$f(x,y) = 3 \quad pokud\ g(x,y) \geq T_3\ \&\ g(x,y) < T_4$$

$$\vdots$$

$$f(x,y) = n \quad pokud\ g(x,y) \geq T_n$$

1.5.2 Detekce hran

Jako hranu lze chápat významnou změnu jasu v obraze. Při její detekci je tedy vyhledávána oblast, kde k takovým prudkým změnám dochází. Metody pro hledání hran jsou založeny na první a druhé derivaci obrazového signálu. V místě, kde se nachází tato výrazná změna jasu je lokální maximum první derivace obrazové funkce. Druhá derivace prochází nulou. V praxi je pak užívána konvoluce obrazu v kombinaci s vhodně zvolenou maskou pro aproximaci první nebo druhé derivace. [9]

Metoda, která využívá první derivace obrazové funkce *f(x,y),* popisuje její gradient. Jedná se o vektorovou funkci, která je definována jako vektor parciálních derivací.

V případě, že je potřeba detekovat hrany a nezáleží na jejich směru, může se využít metody založené na druhé derivaci obrazové funkce. V oblasti, kde první derivace dosahuje maxima, rovná se druhá derivace nule. Pokud lze v diskrétním obraze aproximovat první derivaci jako rozdíl hodnot jasu dvou bodů, které spolu sousedí, bude druhá derivace rovna rozdílu těchto dvou bodů. [9]

1.5.3 Srovnání se vzorem

Metoda založená na porovnávání obrazu s maskou. Základem je přiložení této masky na právě kontrolované místo. Pokud dojde k dostatečně velké odezvě (shodě) mezi touto oblastí obrazu a kontrolní maskou, je v daná oblasti detekována hrana. [1]

Tato metoda je dostatečně odolná vůči šumu, jeho vliv je ovlivněn velikostí masky. Za výhodu lze považovat možnost detekovat hrany, které vznikají na dvou odlišných texturách. Naopak nevýhodou tohoto způsobu vyhledávání je velká závislost na úhlu natočení a taky velikosti masky vůči hledané hraně. Pokud hledaná oblast obsahuje jiný úhel hrany, než jaký je zvolen ve výchozí masce, není možné dosáhnout požadovaného výsledku. Řešením tohoto problému jsou změny velikosti a úhlu masky, to ovšem výrazným způsobem ovlivňuje celkovou náročnost a tím pádem i prodlužuje celkový čas výpočtu. [1]

Časovou náročnost je možné z části snížit, při vhodné konstrukci masky a výpočtu její odezvy. Například nemusí v každém případě docházet k výpočtu celé masky, ale jen částí, které byly při otočení, či posunutí nějakým způsobem změněny.

V praxi se maska shoduje přesně s původním obrazem jenom v malém počtu případů. Z těchto důvodů se většinou hledání omezuje na dostatečnou podobnost. Tato podobnost se dá vyjádřit pomocí korelačního koeficientu o hodnotách <-1, 1> a má tvar:

$$r = \frac{\sum_x \sum_y (f(x,y) - f')(g(x,y) - g')}{\sqrt{\sum_x \sum_y (f(x,y) - f')^2 \sum_x \sum_y (g(x,y) - g')^2}}$$

kde *f(x,y)* je obrazová funkce objektu *A* a *f'* je střední hodnota jasu v tomto obraze. Obrazovou funkcí masky *B* reprezentuje *g(x,y)* a střední hodnotu jasu *g'*. [9]

2 NÁSTROJE PRO ZPRACOVÁNÍ OBRAZU

Následující kapitola pojednává o dostupných nástrojích, které slouží pro zpracování obrazu. Jedná se například o programovací jazyk Matlab, dále programovací jazyky C/C++ v kombinaci s knihovnou OpenCV a další.

2.1 Matlab

Matrix Laboratory, tedy Matlab, je vysokoúrovňový programovací jazyk pro řešení složitých matematických a fyzikálních algoritmů a jejich následnou vizualizaci. Jazyk je optimalizovaný k práci s maticemi a při správném zápisu algoritmu zpravidla podává lepší výsledky, z hlediska času výpočtů, než klasické programovací jazyky (C/C++/C# aj.).

Matlab se často využívá pro zpracování obrazu, analyzování a modelování finančních i biologických dat a spoustu dalších procesů. [15]

2.1.1 Výhody

Matlab obsahuje základní funkce, které se dají rozšířit pomocí specializovaných toolboxů, které přináší nové funkce na řešení konkrétních problémů. Jedním takový toolbox obsahuje i soubor funkcí určených právě pro zpracování obrazu (Matlab Image Processing Toolbox). Hlavními přínosy tohoto rozšíření je analýza obrazu (segmentace, morfologické operace, vyhledávání hran, statistiky, měření), úprava obrazu (filtrování, zaostření), geometrické transformace obrazu a další. [15]

Výhodou jazyka Matlab je i jeho možná integrace s jinými programovacími jazyky. Dále Matlab nabízí:

- Vysokoúrovňový programovací jazyk pro matematické výpočty
- Vývojové prostředí (správa kódu, souborů a dat)
- Matematické funkce – lineární algebra, statistiky, filtrování, optimalizace a numerické integrace
- 2D/3D vizualizace grafů
- Tvorba uživatelského rozhraní aplikace [15]

2.1.2 Nevýhody

Za největší nevýhodu jazyk Matlab se dá považovat nutnost zakoupení licence. V současné době je dostupný ve dvou verzích, standardní a školní. Pořizovací cena Standard verze 9.2 je 55 980 Kč. Cena školní verze je daleko přijatelnější, pohybuje se pod hranicí patnáct tisíc korun. Dále je nutné počítat se zakoupením příslušného toolboxu. V tomto případě jde o rozšíření v kategorii Zpracování signálu a obrazu. Cena Image Processing Toolboxu je 27 980 Kč ve standardní verzi a 5 780 Kč ve verzi školní. Ceny jsou uváděny podle ceníku platného od 10. března 2017. [16]

2.2 GNU Octave

Obdobou již zmiňované Matrix Laboratory je GNU Octave. Stejně jako u Matlabu jde o vysokoúrovňový programovací jazyk. Jeho zaměření je především na obecné numerické počty. Je možné nainstalovat GNU Octave jak na systémy Linux a Windows, tak i na Mac OS. Nabízí řadu funkcí pro řešení lineární algebry, výpočty nelineárních rovnic, integrálních a diferenciálních počtů. Dále je možné Octave rozšířit o moduly napsané přímo v GNU Octave nebo i o moduly psaných v jazycích C, C++ nebo Fortran. [17]

2.2.1 Výhody

Oproti konkurenčnímu Matlabu, software GNU Octave je zcela zdarma. Vývojáři mají navíc povoleno ho volně rozšiřovat.

V Octave je možné využít i vlastností objektově orientovaného programování, jelikož psané skripty umožňují tvorbu datových struktur. Při dodržování syntaxí, je možná napsat v GNU Octave skript, který bude spustitelný i v Matlabu. [17]

2.2.2 Nevýhody

GNU Octave neobsahuje všechny potřebné funkce pro zpracování obrazu a bylo by nutné tyto funkce implementovat nebo převzít od komunity. Hrozí tak neefektivní implementace těchto algoritmů a s tím spojená další rizika.

2.3 OpenCV

Vývoj této multiplatformní otevřené knihovny začal už v roce 1999, první vydání ovšem proběhlo oficiálně až v roce 2006. Samotná knihovna je zaměřena především na programové vidění a analýzu obrazu v reálném čase. Je napsána v jazycích C a C++. Podporuje platformy Linux, Windows i Mac OS. [18]

OpenCV spolupracuje s řadou programovacích jazyků, jako jsou například Java, Python, Ruby, nebo Matlab. Je kladen velký důraz na vysokou výpočetní rychlost procesů funkcí, jelikož se tato knihovna zaměřuje na aplikace, které běží v reálném čase. S tím je spojena optimalizace výpočetních procesů pro vícejádrové procesory.

V případě, že je potřeba ještě větší výpočetní optimalizace, umožňují procesory firmy Intel zakoupení knihoven IPP (Intel's Integrated Performance Primitives). Ty nahrazují optimalizovanými funkcemi některé funkce, které nedosahují tak vysoké úrovně pro daný procesor a tím docílí zvýšení rychlosti výpočtu většiny algoritmů. Po instalaci těchto IPP knihoven bude OpenCV využívat prioritně jejich funkce. [18]

2.3.1 Výhody

Nespornou výhodou knihovny OpenCV je její registrace pod licencí BSD. To znamená, že její užívání není nijak omezeno a je možné využít ji jak k akademickým tak komerčním účelů. To všechno bez dalších závazků vývojáře (např. sdílení kódu komunitě). Díky široké komunitě, která se kolem open source modelu vytvořila, je funkčnost knihovny dostatečně zdokumentována. [18]

OpenCV obsahuje:

- Více než 500 funkcí pro rozpoznání obrazu (detekce hran, ostření, prahování, vyhledávání kontur, klíčových bodů a další)
- Funkce pro kalibraci kamer
- Tvorbu uživatelského rozhraní
- Učící se algoritmy umělé inteligence v knihovně MML (Machine Learning Liberary)

2.3.2 Nevýhody

Nevýhodou OpenCV může být snad jedině nekompatibilita s programovacím jazykem C#. Některé funkce jsou u různých jazyků volány zcela rozdílným způsobem než v C/C++, nebo jsou součástí jiných balíků.

2.4 Emgu CV

Jedná se o multiplatformní knihovnu OpenCV s tím rozdílem, že je kompletně napsána v programovacím jazyce C#. Díky tomu rozšiřuje využití vlastností OpenCV i na programovací jazyky kompatibilní s .NET, jako jsou C#, Visual Basic, VC++, IronPython a další. [19]

2.4.1 Výhody

Implementace prakticky stejných funkcí, jaké obsahuje knihovna OpenCV, i do aplikací psaných v programovacích jazycích, které samotná knihovna OpenCV nepodporuje.

2.4.2 Nevýhody

Některé funkce jsou volány odlišným způsobem než u OpenCV.

3 LICENCE PLATE RECOGNITION

Systém LPR (Licence Plate Recognition) slouží k rozpoznání čísla poznávací značky projíždějícího vozidla. Součástí každého LPR systému je kamera, která zaznamenává obraz scény s vozidlem a dále software pro rozpoznávání obrazu z této kamery.

Základem systému LPR je rozeznávat a zpracovávat registrační značky snímané kamerou, která je propojena se serverem. Zaznamenaná SPZ je porovnána s databází a podle potřeby navazují příslušné operace. Využití tento systém nachází například při vjezdu a výjezdu na parkoviště (kontrola placení), při kontrole pohybu vozidel (např. firemní areály), u mýtných bran a u další spousty aplikací. [20]

Vlastnosti systému LPR:

- Monitoring v reálném čase
- Snadná instalace a konfigurace
- Automatizované řešení – komfort obsluhy
- Vysoká výkonnost zpracování SPZ
- Vyšší bezpečnost parkování a ochrana majetku
- Velká propustnost – 1 SPZ za 3 vteřiny
- Otevření závory na základě rozpoznání SPZ (bez parkovací karty)

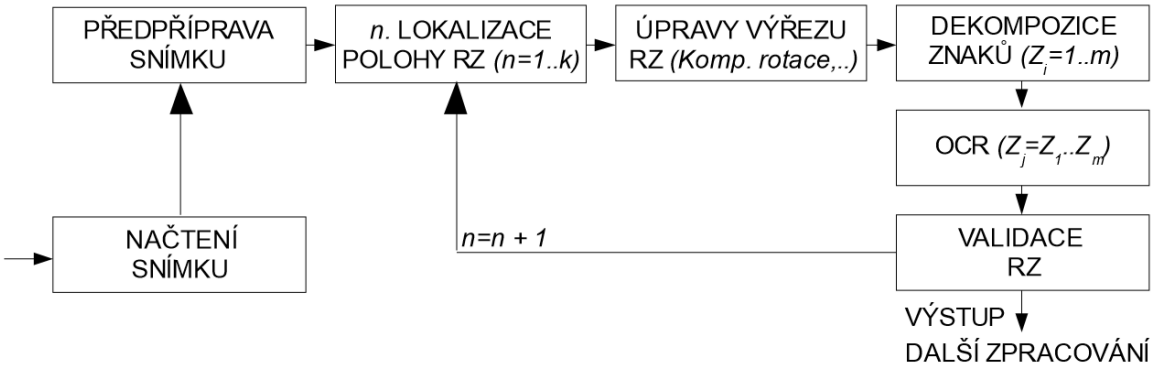
Obr. 15 Ukázka využití systému LPR (parkoviště) [21]

Aplikovat tento systém lze i pro úsekové měření rychlosti. Výhodou LPR ve srovnání s rozšířeným bodovým systémem měření rychlosti, je možnost vykonávat měření v podstatě v libovolném geografickém uspořádání měřeného úseku. Například i ve složitých městských aglomeracích s konečným počtem vstupních a výstupních bodů. Lze tento systém využívat i k bodovému monitoringu trasy pohybu vozidel. [21]

Detekovány mohou být buď celé obrazy vozidel, nebo jenom vybrané nejdůležitější charakteristické části (např. SPZ). Výstupy z této detekce jsou následně poskytnuty nadřazenému systému, kde dochází ke spárování dat vozidel při vstupu a při opuštění monitorované oblasti. Ze získaných informací lze například dopočítat průměrnou rychlost, kterou vozidlo daný úsek projelo, nebo jenom zaznamenávat čas vjezdu a výjezdu. [20]

3.1 Základní charakteristika systému LPR

Pro dosažení dostatečné spolehlivosti celého systému LPR je nutná správná dekompozice problému a vyřešení jednotlivých částí rozpoznávání. Systém je vhodné rozdělit do několika dílčích částí, které na sebe navazují. [20]



Obr. 16 Dekompozice systému LPR [20]

Prvním a zároveň nejdůležitějším krokem pro úspěšné rozpoznání RZ je správná lokalizace polohy značky ve vstupním obraze. Tím se velice usnadňují navazující kroky. Další důležitým faktorem, který je ovlivněn vhodným způsobem pravděpodobné lokalizace RZ, je celkový čas výpočtu. Právě tento krok je kritický pro významné zkrácení doby

výpočtu, kdy další navazující kroky algoritmu, jsou provedeny nejlépe v první pravděpodobné pozici RZ, popřípadě v několika málo dalších. [22]

Následujícím krokem je předúprava obrazu potencionální registrační značky. Jedná se především o co nejlepší oříznutí značky, její správné natočení, omezení vlivu rušivé jasové funkce obrazu a kompenzace dalších možných zkreslení. Tyto úpravy mají za úkol umožnit a především usnadnit, při znalosti možných typů RZ a pravděpodobných pozic jednotlivých znaků, následující analýzu a samotnou segmentaci znaků. [23]

Na takto upravený výřez navazuje krok, ve kterém dojde k dekompozici na jednotlivé znaky. Využívá se jak vlastního vyhledávání znaků, tak i některých dodatečných informací, které pomáhají určit vzájemnou polohu jednotlivých znaků. Díky tomu je často možné určit i lokaci znaků ve výrazně poškozeném obraze.

Po úspěšné segmentaci dochází ke zpracování jednotlivých znaků modulem OCR. Hlavní výhodou pro uplatnění systému OCR je omezená množina použitých vstupních znaků RZ (v ČR 32 znaků), navíc s jasně definovaným vzhledem fontu písma. V důsledku těchto vlastností je možné využít klasických korelačních postupů porovnávání neznámého znaku s předem definovanými vzorovými znaky (šablonami). Tím je zajištěna poměrně velká pravděpodobnost správného rozpoznání. [22]

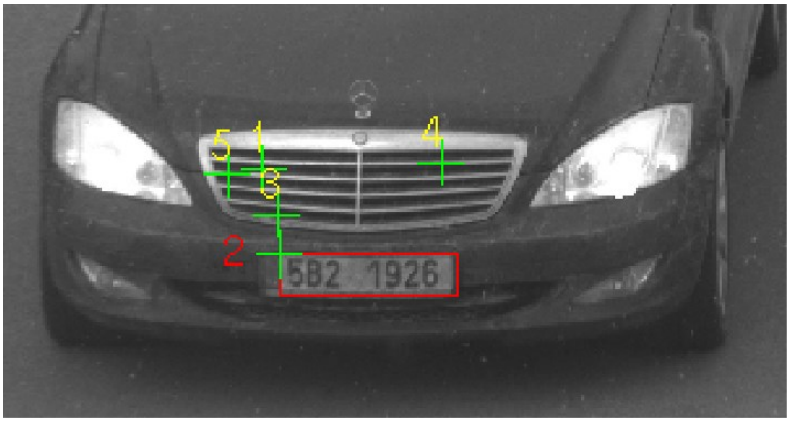
3.2 Určení pozice RZ ve vstupním obraze

Prvotním krokem systému LPR je lokalizace polohy registrační značky v kandidátním výřezu ze vstupního snímku. Na přesnosti a správnosti detekce značky závisí úspěšnost, složitost a v neposlední řadě rychlost následujících kroků. Při nedokonalém nebo příliš obecném určení polohy značky, dochází k potřebě dalších zpřesňujících postupů. Dále s nedostatečnou lokalizací polohy přichází i riziko opakovaných výpočtů v mnoha oblastech, nebo dokonce zamítnutí správné oblasti. [22]

Častým způsobem detekce RZ je nalezení předem definovaného bloku, který je charakterizován přítomností velkého počtu vertikálních hran. Takového obrazu se docílí využitím aplikace vertikálního hranového operátoru na vstupní obraz.

Tento postup je dostatečně spolehlivý a využívá se v mnoha systémech. Ve speciálních případech ale může dojít k jeho selhání. To zejména tehdy, kdy je na snímaném vozidle

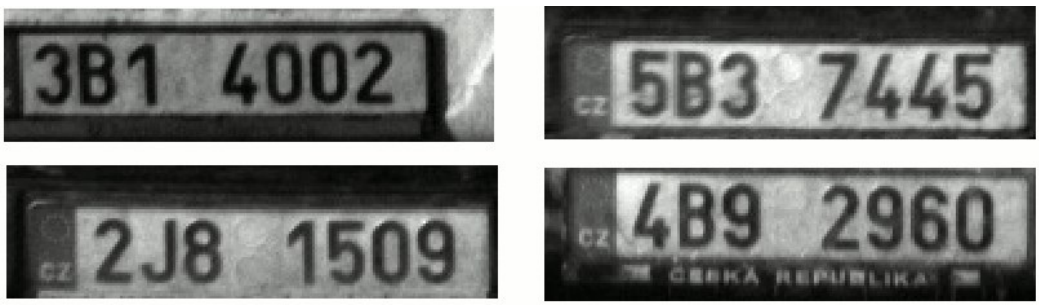
přítomna oblast s vlastnostmi podobnými RZ (např. mřížka chladiče se svislými hranami), nebo pokud nejsou vertikální hrany dostatečně výrazné (např. vinnou znečištění). [20]



Obr. 17 Příklad vozidla s členitými částmi prostoru RZ [20]

Tento způsob sebou nese jednu velkou nevýhodu v podobě poměrně velké nepřesnosti určení přesné pozice značky vzhledem ke kandidátnímu výřezu. Nelze tedy přesně určit pozici jednotlivých znaků. Dalším faktorem, který má také vliv na nepřesnost je modrý „EU“ proužek. Jeho velikost téměř splňuje maximální parametry boxu definovaného pro jeden znak.

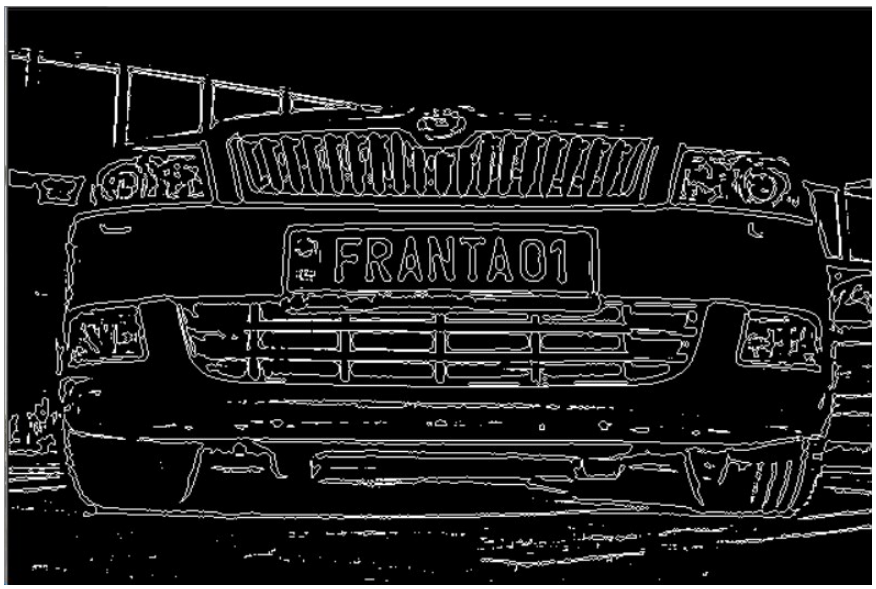
Tuto nepřesnost lze částečně kompenzovat zvětšením oblasti kandidátního výřezu. To znamená například rozšířit výběr o dvakrát 10% v horizontálním a dvakrát 20% ve vertikálním směru. Tím se zajistí, aby byla vybrána opravdu RZ se všemi jejími znaky. Cenou za to je ale pochopitelně zvýšení náročnosti odhadu předpokládané pozice znaků. [23]



Obr. 18 Příklad detekovaného umístění RZ vůči kandidátnímu výřezu [20]

3.2.1 Alternativní způsob lokalizace polohy RZ

Vzhledem k dalším charakteristickým vlastnostem snímané scény, je pro úspěšnou detekci vhodné využití hranové reprezentace obrazu. Hlavními důvody pro umožnění aplikace tohoto postupu jsou zejména předem známé, s určitou tolerancí, rozměry RZ a taky fakt, že lze, ve většině případů, počítat se spojitými okraji. Těchto vlastností se využívá pro upřesnění polohy RZ nebo pro samotné detekování lokace značky. [20]



Obr. 19 Aplikace obousměrné hranové detekce

Samotná detekce se zakládá na nalezení vhodné šablony, která reprezentuje vzorovou RZ ve vstupním obraze. Šablona musí být zvolena tak, aby byla shoda s potencionálními značkami co největší a to nezávisle na proměně jednotlivých znaků v ní. K porovnávání se uplatňují jak vodorovné a svislé hrany jednotlivých znaků, tak i hrany samotné RZ. Výhodou u této metody je využívání hranové detekce jak ve vodorovném tak i ve svislém směru. Oproti předchozímu způsobu je tak k dispozici dvakrát více informací. [20]

3.3 Segmentace znaků

Značný dopad na konečné rozpoznání znaku má určení jeho přesné pozice a následné ořezání. I malé posunutí nebo oříznutí jinak správně pruhovaného znaku má negativní dopad na celkovou úspěšnost. Proto je důležité věnovat segmentaci co největší pozornost.

V případě, že dojde k nedokonalé hranové detekci a nejsou všechny znaky tvořeny spojitým regionem, je možné využít obecných informací o RZ. Díky daným velikostem znaků a odstupů mezi nimi lze označit pravděpodobné pozice všech znaků poznávací značky. [23]

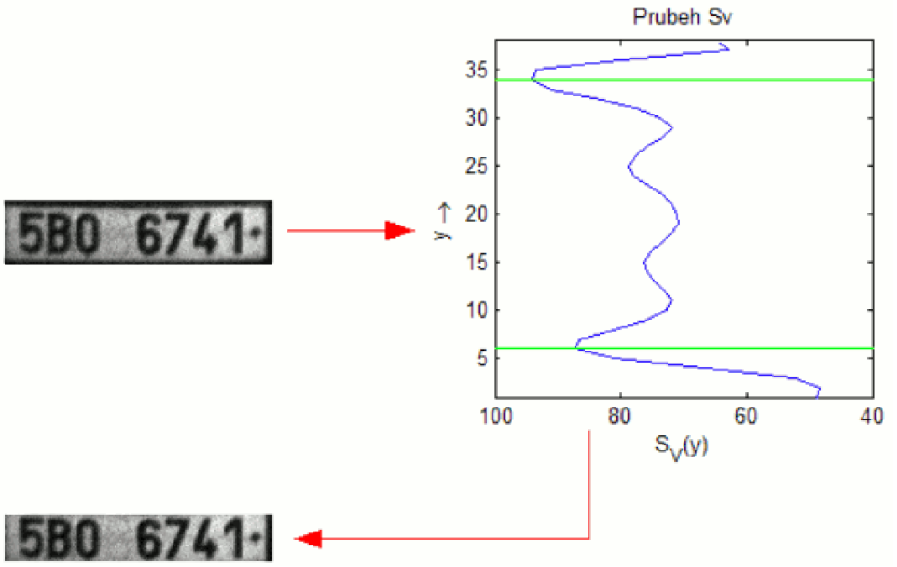
3.4 Odhad výšky znaků

Detekování správné výšky znaků je možné například vyhledáním horizontálního průmětu všech znaků RZ, které tvoří řádek a následná separace pozadí a okrajů RZ. Touto metodou je možné určit snadno i výšku více poškozených znaků.

Tento způsob se aplikuje přímo na vstupní obraz značky, který je převedený na stupně šedi. Je-li vstupní obraz neinvertovaný, funkce $S_v(y)$ dosahuje maxima právě v oblastech mezi horní, popřípadě spodní hranou znaků a samotného okraje RZ. Právě v těchto místech lze předpokládat největší jasovou intenzitu a minimální v místech okrajů značky, kde se nachází rámeček RZ. [20]

$$S_v(y) = \sum_{x=1}^{x=H} I(x, y)$$

Sumace $S_v(y)$ je provedena přes všechny řádky matice I, kde $I(x,y)$ je průběh jasové funkce ve vstupním obraze značky a x je potom horizontální souřadnice. Na následujícím obrázku je zřetelně vidět, kde se nachází maxima funkce $S_v(y)$. Z důvodu proměnných hodnot průběhu $S_v(y)$ v místech odpovídajícím vertikálnímu řezu znaku, je vhodné vyhledávat maxima jenom v oblastech, které odpovídají například 30% výšky RZ od okrajů.[20]



Obr. 20 Průběh $S_V(y)$ a řez na úrovni znaků [20]

Jak můžeme vidět, prostor mezi nalezenými maximy určuje prostor výšky znaků. Na základě toho je proveden řez a oddělí se znaky od spodního i horního okraje RZ. Stejný způsob je možné aplikovat i na vyhledání vertikálních okrajů jednotlivých znaků.

3.5 Extrakce znaků

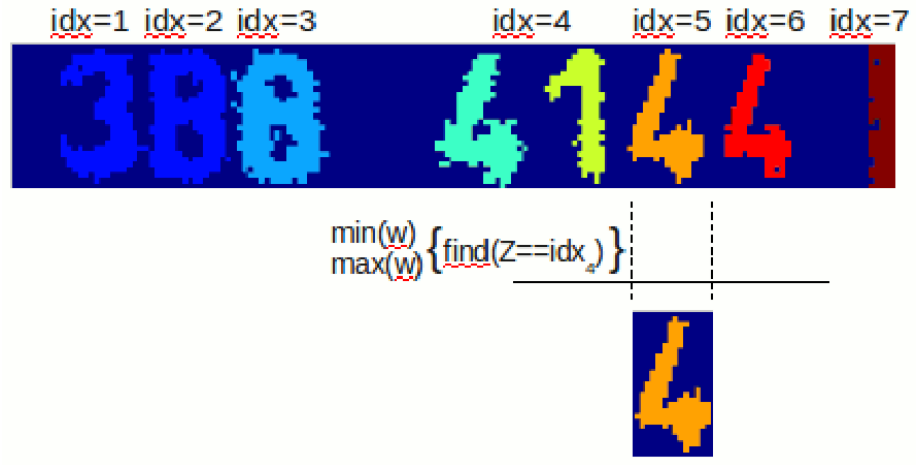
Takto ořezaný obraz už lze použít k samotné separaci jednotlivých znaků. Postup je totožný jako při získávání polohy RZ z původního vstupního obrazu. Nejdříve se převede do odstínů šedi a následně pomocí prahování až do černobílého vyjádření.

U takto předpřipraveného snímku už lze přistoupit k extrakci jednotlivých znaků. Jako první je provedeno indexování všech spojitých regionů v obraze algoritmem pro vyhledávání spojitých regionů v obraze. V následujícím kroku je zjištěna minimální a maximální horizontální souřadnice objektu a oblast, která je vyříznuta reprezentuje právě jeden znak.[22]

$$Z_h = \left(find(RZ = idx)\right)$$

$$Z_h min = \min(Z_h)$$

$$Z_h max = \max(Z_h)$$



Obr. 21 Extrakce znaků z obrazu [20]

V situaci, kdy dojde k nedokonalému prahování, například z důvodu poškozeného či znečištěného obrazu, mohou se jednotlivé znaky slít do jednoho obrazu. V těchto případech je potřeba z rozdílu $Z_h max$-$Z_h min$ vypočítat pravděpodobný počet znaků. Předpokládáme-li minimální šířku nejužšího znaku λ, je vhodné vyloučit symboly, pro které platí $Z_h max\ Z_h min \leq \lambda$. Tím dojde ke snížení chybovosti segmentace znaků.

Výstupem předešlých operací je *k* samostatných symbolů, které jsou následně podrobeny rozpoznávacím algoritmům. [20]

3.6 Rozpoznání znaků

Posledním krokem celého procesu, je rozpoznávání znaků. To je realizováno algoritmem, který je nazýván jako Template Matching. Vstupem tohoto algoritmu jsou obrazy segmentovaných znaků ze vstupního obrazu a šablony znaků. Každý obraz, který byl po segmentaci získán, je upraven tak, aby jeho velikost byla totožná s velikostmi porovnávaných šablon. Šablona, která nejvíce odpovídá danému obrazu je vybrána jako kandidátní znak.

Jediným kritériem pro porovnávání je vzájemná shoda překrývajících se obrazových bodů šablony a analyzovaného znaku. Z toho důvodu hraje velmi významnou roli správné překrytí těchto dvou obrazů. [20]

Dalším způsobem klasifikace znaků je tzv. OCR (Optical Charakter Recognition). Jedná se o technologii optického rozpoznávání znaků, která umožňuje převádět na text například naskenované dokumenty, PDF soubory, fotografie atd.

Aplikace OCR využívají odlišné metody na rozpoznání znaků. Jedním z příkladů může být software založený na principu rozložení bodů v obraze. Na definovaných místech v obraze jsou vyhledávány případné průsečíky s rozpoznávanými znaky. Podle nalezených průsečíků je určen konkrétní znak. Metoda OCR často využívá i metody srovnávání se vzorem. [22]

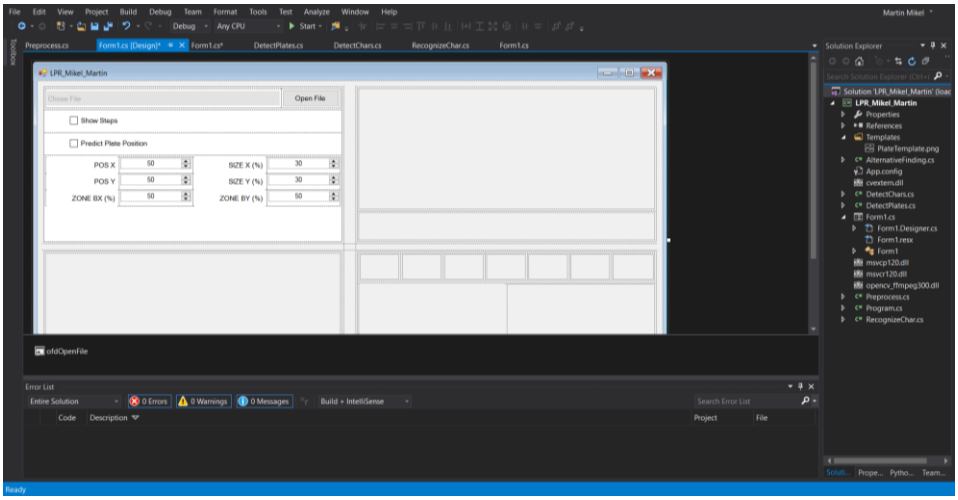
II. PRAKTICKÁ ČÁST

4 VYTVOŘENÍ NÁSTROJE LPR

Hlavním úkolem této práce bylo vytvořit vlastní systém pro rozpoznávání poznávacích značek (Licence Plate Recognition).

Jako nejvhodnější nástroj, pro zpracování obrazu byla vyhodnocena knihovna OpenCV. Jelikož ale tuto knihovnu nelze použít v kombinaci s vybraným programovacím jazykem C#, bylo nutné využít upravenou verzi, právě pro tyto případy. Jedná se o knihovnu EmguCV.

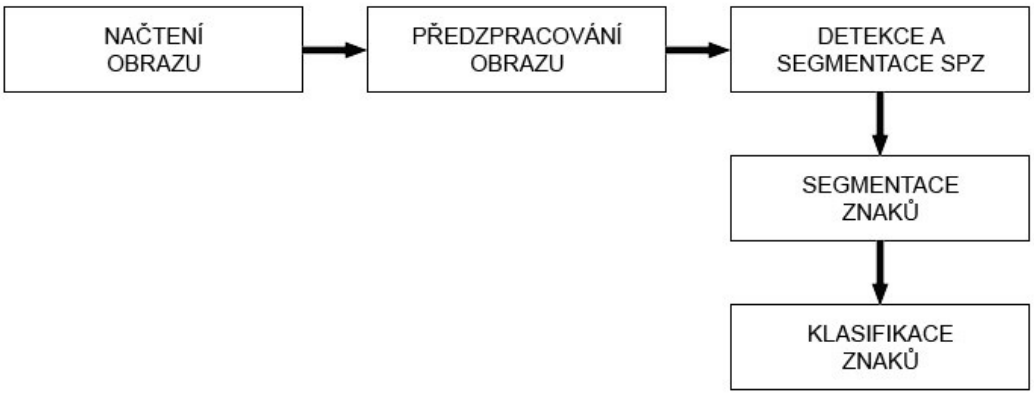
Vývojové prostředí, ve kterém byla aplikace vytvořena je MS Visual Studio 2015. Kompletní projekt byl vypracován jako Windows Forms Application v programovacím jazyce C#. Dále byla implementována knihovna pro rozpoznání obrazu EmguCV ve verzi 3.0.0.2157.



Obr. 22 MS Visual Studio 2015 (Windows Forms Application)

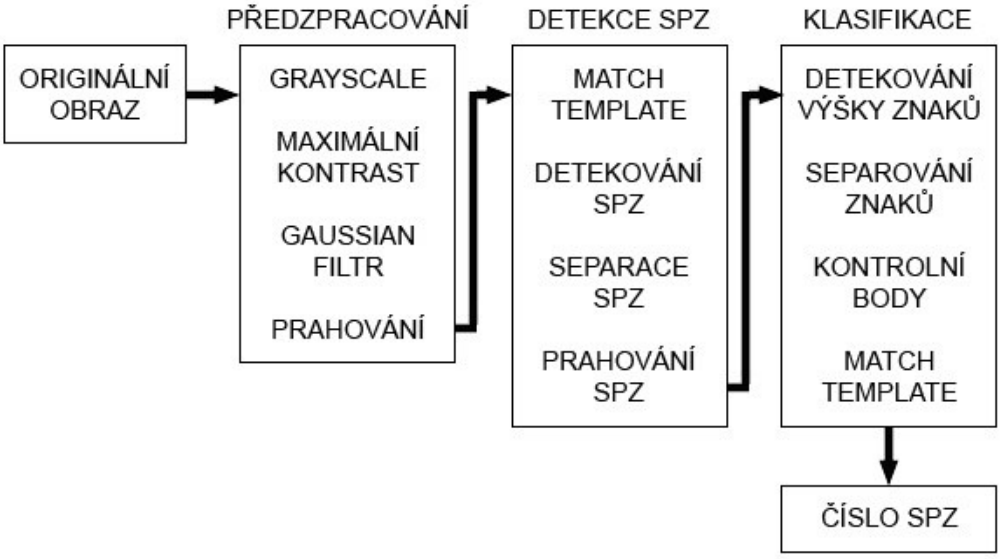
4.1 Základní struktura systému LPR

Systém LPR se skládá z pěti základních operací. Jako první je to samotné načtení scény respektive obrazu. Ten je následně upraven, aby výsledná detekce byla co nejspolehlivější. V dalších krocích je detekována pravděpodobná oblast, kde by se měla nacházet registrační značka. Pokud je tato oblast správná, dojde k segmentaci jednotlivých znaků. Navazuje jejich klasifikace.



Obr. 23 Základní struktura systému

Přehled základních operací, které svou návazností spolu tvoří celý systém pro rozpoznání registrační značky, je znázorněn na obrázku níže (*Obr. 24*).



Obr. 24 Přehled navazujících operací systému LPR

4.1.1 Státní poznávací značka

V České republice mají na starosti vydávání státních poznávacích značek obce s rozšířenou působností. Samotný vzhled a umístění řeší vyhláška č. 343/2014.

Standardní registrační značka je vyznačena na tabulce o rozměrech 520 x 110 mm formou prolisování. Znaky jsou psány velkými písmeny latinské abecedy, kterých je na

značce nejméně 5 a maximálně 8. Každá SPZ musí obsahovat minimálně 1 písmeno a nejméně 1 číslici. Druhý znak v pořadí určuje kraj, ve kterém byla daná SPZ vydána. [24]

Mezi jednotlivé znaky se nevkládá žádné interpunkční znaménko. Aby se předešlo možné záměně písmen při čtení SPZ, jsou z abecedy možných znaků vyřazeny písmena G, O, Q, a W. U standardního typu značky je mezi třetím a čtvrtým znakem mezera, která slouží k umístění nálepky o technické kontrole.

Speciální situace nastává u registrační značky na přání. Ta může obsahovat až 8 libovolných znaků. Navíc mohou být písmena i číslice použity úplně libovolně. Mezera mezi třetím a čtvrtým nemusí být dodržena. [24]

Umístění registrační značky podle vyhlášky:

- Na přední i zadní straně vozidla
- Uprostřed (pokud není povoleno jiné umístění)
- Kolmo na směr jízdy
- Vodorovně k vozovce, spodní hranou dolů [24]

Obr. 25 Státní poznávací značka [25]

4.2 Předzpracování obrazu

V první kroku je velice důležité předzpracovat obraz tak, aby následující operace pro samotné detekování polohy SPZ byly co nejpřesnější.

Originální obraz je převeden z RGB modelu na barevný model HSV, poté pomocí funkce *Split* rozdělen na samostatné kanály. Díky tomu je možné zbavit se barevných kanálů a docílit tak požadovaných odstínů šedi (Grayscale).

```
CvInvoke.CvtColor(imgOriginal, imgHSV, ColorConversion.Bgr2Hsv);

CvInvoke.Split(imgHSV, vectorOfHSVImages);

imgValue = vectorOfHSVImages[2];
```

Obr. 26 Převedení originálního obrazu na stupně šedi

Obr. 27 Originální obraz a obraz ve stupních šedi

Dále je vhodné zvýraznit kontrast tak, aby co nejvíce vynikly potencionální znaky na poznávací značce. K tomu je využito morfologických funkcí *TopHat* a *BlackHat*. Požadovaný obraz o maximálním kontrastu je výsledkem součtu obrazu *Grayscale* s výstupem funkce *TopHat* a následným odečtením výsledku funkce *BlackHat*.

```
Mat structuringElement = CvInvoke.GetStructuringElement(ElementShape.Rectangle,
new Size(3, 3), new Point(-1, -1));

CvInvoke.MorphologyEx(imgGrayscale, imgTopHat, MorphOp.Tophat, structuringEle-
ment, new Point(-1, -1), 1, BorderType.Default, new MCvScalar());

CvInvoke.MorphologyEx(imgGrayscale, imgBlackHat, MorphOp.Blackhat, structurin-
gElement, new Point(-1, -1), 1, BorderType.Default, new MCvScalar());

CvInvoke.Add(imgGrayscale, imgTopHat, imgGrayscalePlusTopHat);
CvInvoke.Subtract(imgGrayscalePlusTopHat, imgBlackHat, imgGrayscalePlusTopHatMi-
nusBlackHat);
```

Obr. 28 Aplikace morfologických funkcí TopHat a BlackHat

Před zahájením procesu prahování je nejlepší celý obraz rozostřit, čímž se sníží malé nepodstatné přechody a potlačí se šum. Jelikož poznávací značku tvoří černé znaky v bílém poli, je zabezpečen dostatečný rozdíl v přechodu mezi těmito barvami. Pokud je obraz a světelné podmínky dostatečně kvalitní ani tohle rozostření nezapříčiní ztrátu některého ze znaků.

Ke zmiňovanému rozostření je aplikován na obraz Gausianův filtr.

```
CvInvoke.GaussianBlur(imgMaxContrastGrayscale, imgBlurred, new
Size(GAUSSIAN_BLUR_FILTER_SIZE, GAUSSIAN_BLUR_FILTER_SIZE), 3);
```

Obr. 29 Gaissianův filtr

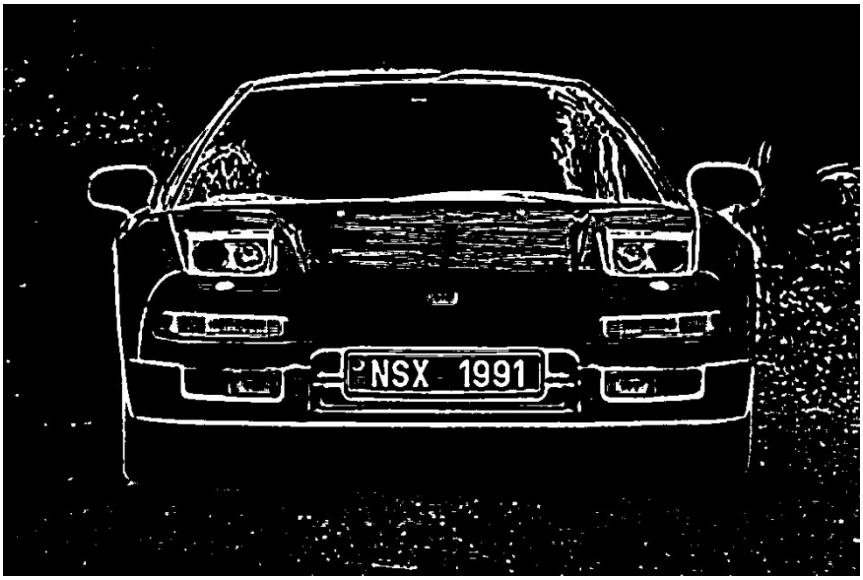
Obr. 30 Obraz se zvýšeným kontrastem a po aplikování Gaussianovu rozostření

Jak bylo už zmíněno, nyní následuje tzv. prahování. Jde o metodu segmentace, kdy je na základě hodnoty jasu každého obrazového bodu a vhodně zvoleného prahu rozhodnuto, zda je daný obrazový bod přiřazen popředí nebo pozadí.

K tomu slouží v knihovně Emgu CV funkce *AdaptiveThreshold*. Ta transformuje vstupní obraz ve stupních šedi na obraz binární.

```
CvInvoke.AdaptiveThreshold(imgBlurred, imgThresh, 255.0, AdaptiveThreshold-
Type.GaussianC, ThresholdType.BinaryInv, ADAPTIVE_THRESH_BLOCK_SIZE,
ADAPTIVE_THRESH_WEIGHT);
```

Obr. 31 Funkce AdaptiveThreshhold pro prahování



Obr. 32 Výsledný obraz získaný prahováním

4.3 Detekce polohy poznávací značky

Prvním předpokladem pro správné rozpoznání poznávací značky je úspěšná lokalizaci oblasti značky ve vstupním obrazu.

V této aplikaci byla zvolena pro nalezení SPZ metoda, která využívá univerzální masky (šablony) poznávací značky, která je postupně přikládána k jednotlivým obrazovým

bodům a hledá se místo s největší shodou. Šablona byla zvolena tak, aby co nejvíce odpovídala jakékoliv možné značce.

Obr. 33 Maska pro hledání polohy SPZ

Pro rychlejší a přesnější vyhledávání šablony, umožňuje aplikace přednastavit polohu na vstupním obraze, kde je předpokládaná poloha SPZ. Ta bývá ovlivněna úhlem pohledu kamery a lze tak předem odhadnout, ve kterých místech se nejspíš bude značka vyskytovat.

Za účelem vyhledávání předem zvolené šablony je v knihovně Emgu CV metoda *CvInvoke.MatchTemplate.* Vstupní parametry pro ni jsou obraz, ve kterém se bude hledat. V tomto případě se jedná o výsledný binární obraz po prahování. Dále je to šablona, která se bude přikládat a bude se hledat místo s největší shodou. Poslední parametr určuje, jakým způsobem se bude šablona s originálním obrazem srovnávat.

```
CvInvoke.MatchTemplate(imgThreshScene, templ, result, TemplateMatchingType.Cco-
effNormed);
```

Obr. 34 Funkce MatchTemplate pro hledání šablony

Výstupem této funkce jsou souřadnice bodů, kde byla nalezena největší shoda s přednastavenou šablonou. Pro lepší přehled o tom, zda byla lokalizována poloha SPZ správně, je toto místo zvýrazněno červeným obdélníkem.

Obr. 35 Poloha největší shody s šablonou

Z původního originálního obrazu je vyseparovaná oblast předpokládané poznávací značky.

Obr. 36 Vyseparovaný obraz SPZ

Ta je následně samostatně podrobena celému procesu předzpracování obrazu s tím rozdílem, že je na ni aplikován mírnější Gaussianův filtr, aby se docílilo menšího poškození nebo deformování jednotlivých znaků.

Obr. 37 Binární obraz SPZ

4.4 Segmentace jednotlivých znaků

Prvním krokem segmentování znaků SPZ je horizontální „oříznutí“ obrazu. Jedná se o dva řezy, které oddělí nepotřebnou část nad respektive pod hranou jednotlivých znaků.

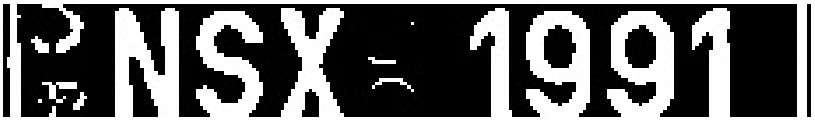
Od středu obrazu směrem nahoru jsou počítány v každém řádku počty bílých obrazových bodů. Dojde-li k velkému poklesu těchto pixelů, znamená to, že se jedná o horní hranu znaků. V případě této aplikace dojde k „oříznutí“ v místě, kde počet bílých pixelů na jednom řádku klesne pod hranici deseti procent.

Stejným způsobem se vypočítají souřadnice i pro spodní hranu. Změna je pouze ve směru počítání pixelů v řádcích. Postupuje se od středu obrazu směrem dolů.

Výsledkem je obraz, který se nachází právě mezi těmito dvěma souřadnicemi. Stejného výsledku by se dalo dosáhnout i automatickým oříznutím podle předpokládané výšky znaků. Jelikož se ale pozice snímané SPZ může mírně lišit, může dojít i k různým velikostem nalezených znaků. Snadno by tak mohlo dojít k odstranění části některého ze znaků, což by mohlo komplikovat jeho následnou klasifikaci.

```
// výpočet horní hrany pro oříznutí
            for (int y = startPoint; y > 1; y--)
            {
                whitePixels = 0;

                for (int x = 1; x < width; x++)
                {
                    p = bmp.GetPixel(x, y);

                    if (p.R == 255)
                    {
                        whitePixels++;
                    }
                }

                float percent = whitePixels / ((float)(width) / 100);

                if (percent < 10)
                {
                    croppY1 = y;
                    break;
                }
            }
```

Obr. 38 Ukázka kódu pro výpočet horní hrany oříznutí

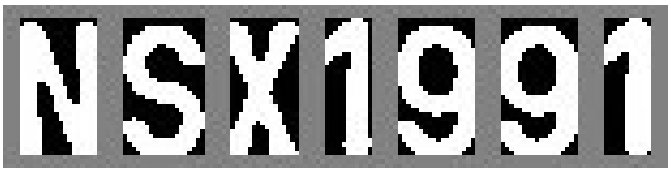


Obr. 39 Obraz po odstranění horních a dolních přesahů

Obdobným způsob jsou od sebe odděleny i jednotlivé znaky. Princip algoritmu je, stejně jako u předchozího segmentování, založen na porovnávání barvy jednotlivých obrazových bodů. Hledají se spojité bílé objekty, které reprezentují znaky SPZ.

Jelikož výška znaků už byla vypočítána, zde se hledá už pouze pozice pro vertikální hrany. Jak bylo vzpomenuto výše, i tady lze využít předpokládané šířky jednoho znaku a obraz rozčlenit na stejnoměrné objekty. I zde bude ale z důvodu menší chybovosti výhodnější postupovat podle hodnot bílých bodů ve sloupcích.

Ze znalostí, které víme o státní poznávací značce, lze určit polohu pro zahájení analýzy pixelů. Díky tomu je zajištěno, že za znak nebude považován například okraj rámečku SPZ.



Obr. 40 Segmentace znaků

4.5 Klasifikace znaků

Každý znak projde samostatně algoritmem, který je založený na principu metody srovnávání se vzorem. Každý nalezený a oddělený objekt, představující jeden znak, může mít odlišnou velikost. Vždy záleží na velikosti registrační značky v originálním obraze. Aby nedocházelo k chybným rozpoznáním, nenalezením žádné shody, nebo zdlouhavým výpočtům, je každý znak přizpůsobený velikosti vzorových šablon. Ty mají rozměry 33 x 57 obrazových bodů a právě na tyto rozměry se upraví i každá nalezená oblast se znakem. K tomu slouží metoda *Resize*. Její vstupní parametry jsou obraz, u kterého se budou měnit rozměry, objekt (obraz), do kterého bude uložen výstup metody, požadované rozměry (*33,*

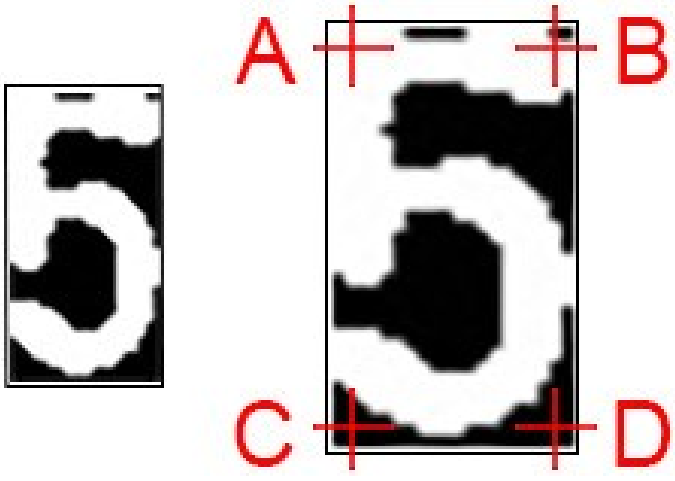
57), nebo poměr velikosti stran (*1, 1*), v jakém se mají změnit. A nakonec způsob, jakým má ke změně rozměrů dojít. Ten byl zvolen jako *Inter.Linear*. Jde o bilineární interpolaci. Klíčová myšlenka této metody pro změnu rozměrů, je lineární interpolace nejprve jedné strany a následně i té druhé.

```
CvInvoke.Resize(imgCharOriginal, imgCharOriginal, new Size(33, 57), 1, 1,
Inter.Linear);
```

Obr. 41 Metoda Resize pro změnu rozměrů

Po změně rozměrů následuje identifikace čtyř kontrolních bodů. Body se nachází ve všech rozích daného objektu. V každém z bodů je identifikována barva. Podle těchto získaných informací je zúžený okruh pravděpodobných znaků.

Příklad je znázorněný na obrázku níže. Na rozpoznávaném objektu, jehož rozměry byly upraveny, jsou vyznačeny body A, B, C, a D. V bodech A a B byla hodnota identifikována bílá barva. Z toho důvodu těmto dvou bodům byla přiřazena hodnota 1 (*true* - *pravda*). Naopak body C a D jsou barvy černé a tak jejich hodnoty budou 0 (*false* - *nepravda*).



Obr. 42 Změna velikosti objektu a kontrolní body

Podle hodnot těchto čtyř bodů jsou vzápětí přiřazeny pravděpodobné znaky, kterým tyto hodnoty odpovídají. Objekt je srovnáván právě s maskami těchto znaků. Díky tomu se

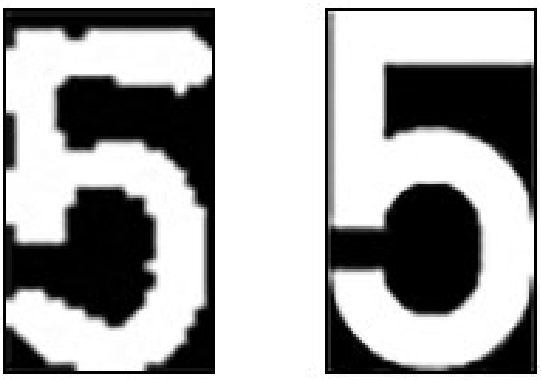
výrazně urychlí celý proces. Každý znak totiž není porovnáván se všema maskami, ale jen s těma nejpravděpodobnějšíma.

Tabulka č. 1 ukazuje, jaké skupiny znaků odpovídají určitým hodnotám kontrolních bodů. U tohoto ukázkového příkladu se jedná o hodnoty bodů 1100, což odpovídá možným znakům 5, T, U, V, a Y. Právě z této skupiny bude vybrán ten s největší shodou.

Tabulka 1 Hodnoty kontrolních bodů a jim odpovídající skupiny znaků

A	B	C	D	Odpovídající znaky
1	1	1	1	E, H, K, M, N, X, Z
1	1	1	0	7, F
1	0	1	1	L, R
1	0	1	0	B, D, P
1	1	0	0	5, T, U, V, Y
0	0	1	1	2, A
0	1	0	0	J
0	0	0	0	0, 1, 3, 4, 6, 8, 9, C, I, S

Ke znaku, podrobovanému detekci, je přikládána maska po masce všech znaků z dané skupiny. Jsou srovnávány vzájemně všechny obrazové body. Jejich shoda je vyjádřena v procentech. Právě za znak, jehož maska měla největší procento shody, je prohlášen i znak detekovaný. Je-li ovšem maximální nalezená shoda nižší než 50%, je daný znak označen jako nedetekovaný a ve výsledném vypsání čísla registrační značky je reprezentovaný jako pomlčka („-“).



Obr. 43 Znak a maska s největší shodou

Tato metoda srovnávání se vzorem je, pro její jednoduchost, velice využívaná. V její prospěch hraje taky fakt, že u registračních značek je předepsaný jednotný font písma a jeho velikost. Další výhodou je vyřazení znaků, u kterých by mohlo dojít lehce k záměně za jiný. Těmito znaky jsou G, O, Q a W.

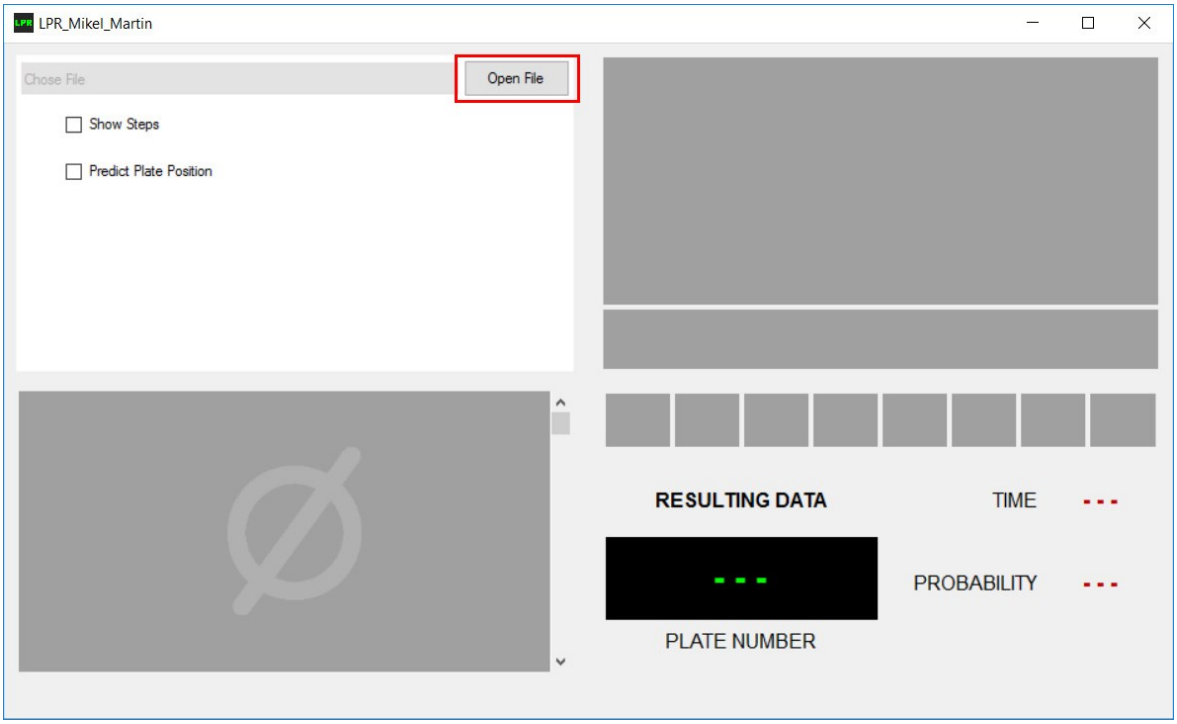
Obr. 44 Povolené možné znaky pro registrační značky

5 NÁVOD NA POUŽITÍ VYTVOŘENÉHO LPR

Tato kapitola slouží jako manuál pro práci s vytvořeným systémem LPR. Jako přílohy k této práci jsou adresáře s celým projektem systému, který byl zpracován ve Visual Studiu 2015. Dále adresář se samotnou finálovou verzí aplikace LPR a adresář s testovanými fotografiemi.

5.1 Aplikace LPR

Po spuštění souboru *LPR_Mikel_Martin.exe* se otevře okno aplikace. Pomocí jediného tlačítka (*Open File*), které je k dispozici, se otevře dialogové okno. V tomto okně si uživatel vybere požadovanou fotografii k detekci registrační značky.

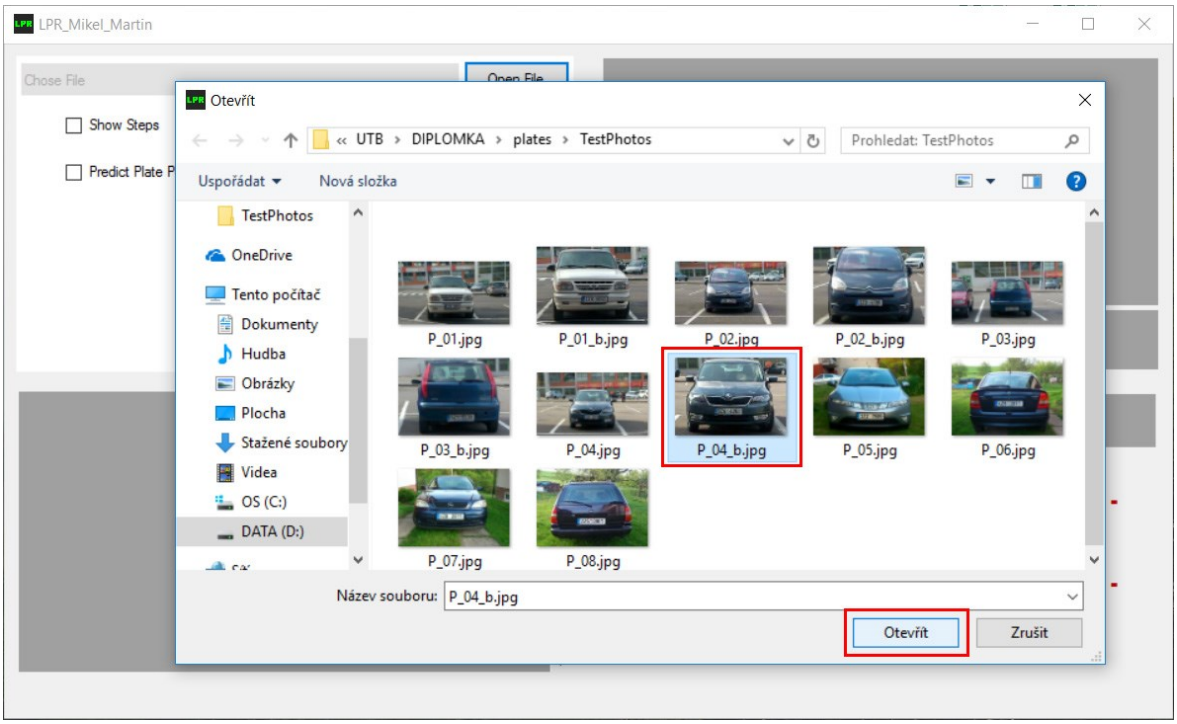


Obr. 45 Spuštěná aplikace LPR

Fotografie musí být ve formátu JPEG, JPG, PNG nebo BMP. V případě, že byl vybrán soubor, který není v požadovaném formátu a tím pádem se ho nepodařilo otevřít, vypíše se do textového pole, vedle tlačítka *Open File,* chybová zpráva. Pomocí ní je uživatel

upozorněn, že došlo k chybě při pokusu otevřít vybraný soubor a je nutné vybrat soubor nový.

Po zvolení odpovídajícího souboru (fotografie), potvrdí uživatel výběr tlačítkem *Otevřít*. Proběhne-li všechno v pořádku, otevře se v dolním panelu obraz v originální podobě.



Obr. 46 Výběr fotografie

V dalším kroku si uživatel zvolí vlastní požadované parametry nastavení. K dispozici je možnost *Show Steps*. Při jejím výběru bude v průběhu detekce proces zastavován a budou zobrazovány jednotlivé kroky, které vedou až k samotnému rozpoznání registrační značky. V tomto režimu ovšem není měřen čas, který byl potřebný k detekci značky. Důvodem je přerušování procesu zobrazováním kroků. Každá nově otevřená fotografie musí být

uzavřena nebo potvrzena stisknutím libovolného tlačítka, aby mohlo dojít k opětovnému spuštění procesu detekce.

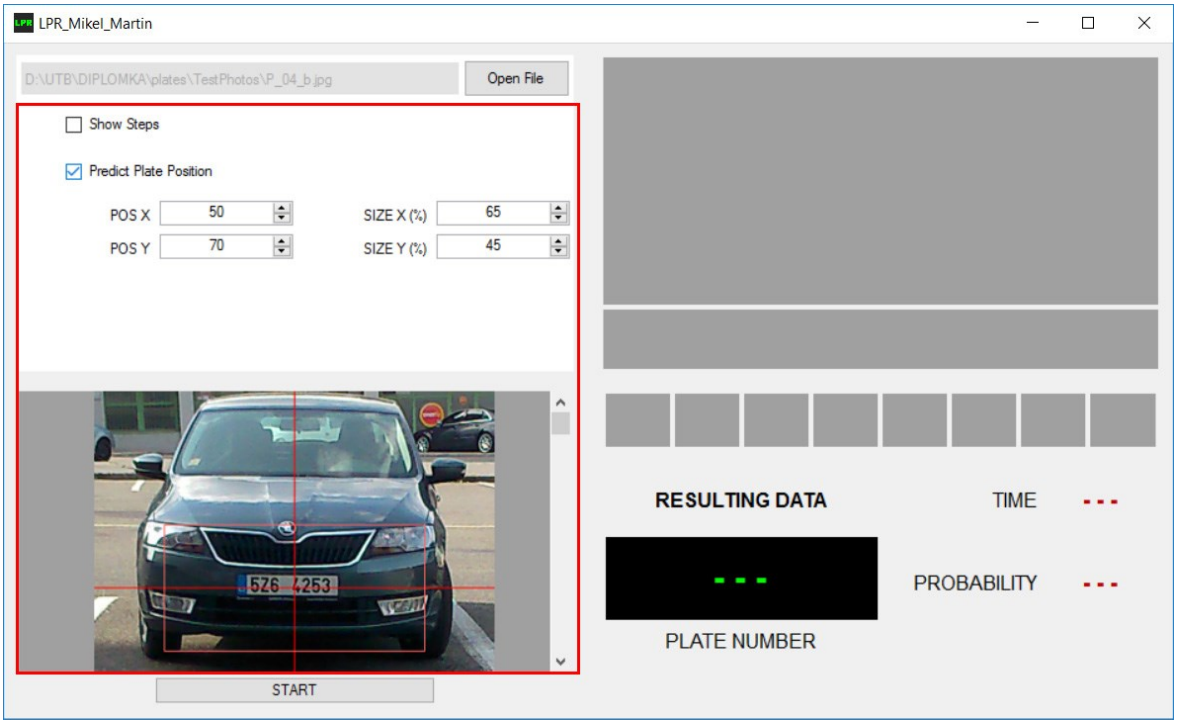
Obr. 47 Ukázka jednotlivých kroků

Druhou volbou v základním nastavení je *Predict Plate Position.* Po vybrání této možnosti se v originálním obraze v levém spodním rohu zobrazí červené osy s obdélníkovým výběrem. Ten slouží k výběru oblasti obrazu, ve které má být prováděna detekce.

Omezením oblasti, ve které je předpokládán výskyt registrační značky, ovlivní výrazným způsobem výsledný čas výpočtu algoritmů hledání. V praxi se takové oblasti dají jednoduchým způsobem definovat. Závisí na nastavení úhlu kamery směrem ke snímané scéně, směr přijíždějícího vozidla, rychlost, jakou vozidlo daným úsekem projede a další faktory. Každá kamera vyžaduje individuální nastavení a testování.

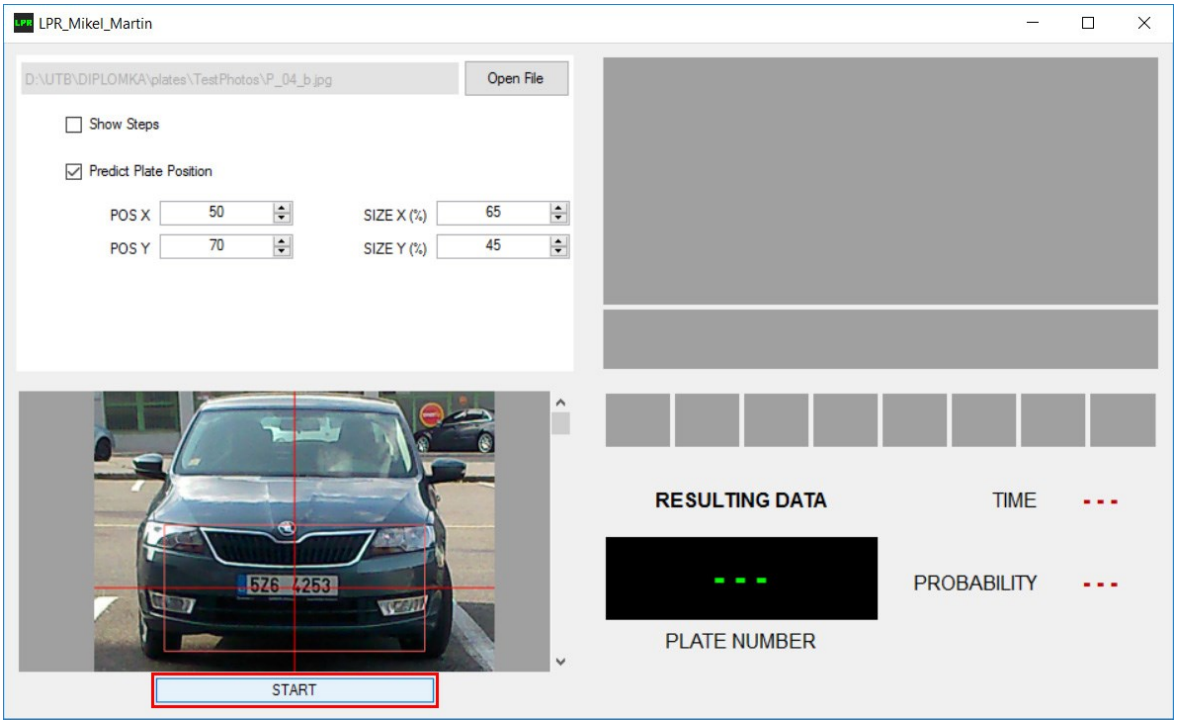
Pozici výběru lze měnit jednoduchou změnou hodnot *POS X* a *POS Y*. Stejný princip se uplatňuje i na změnu velikosti této oblasti. Ta je nastavitelná pomocí parametrů *SIZE X* a *SIZE Y*.

Nejsou-li zvolena žádná oblastní omezení pro detekci, proběhne detekce v celém obraze. Jak už bylo ale řečeno, čas výpočtu bude výrazně delší a může dojít i k neúspěchu nalezení SPZ vinou porovnávání velkého množství dat. Z těchto důvodů je doporučováno vždy oblast hledání zúžit.



Obr. 48 Nastavení požadovaných hodnot detekce

Po nastavení všech požadovaných parametrů se aplikace spustí tlačítkem *START*, které se nachází v dolní části obrazovky.

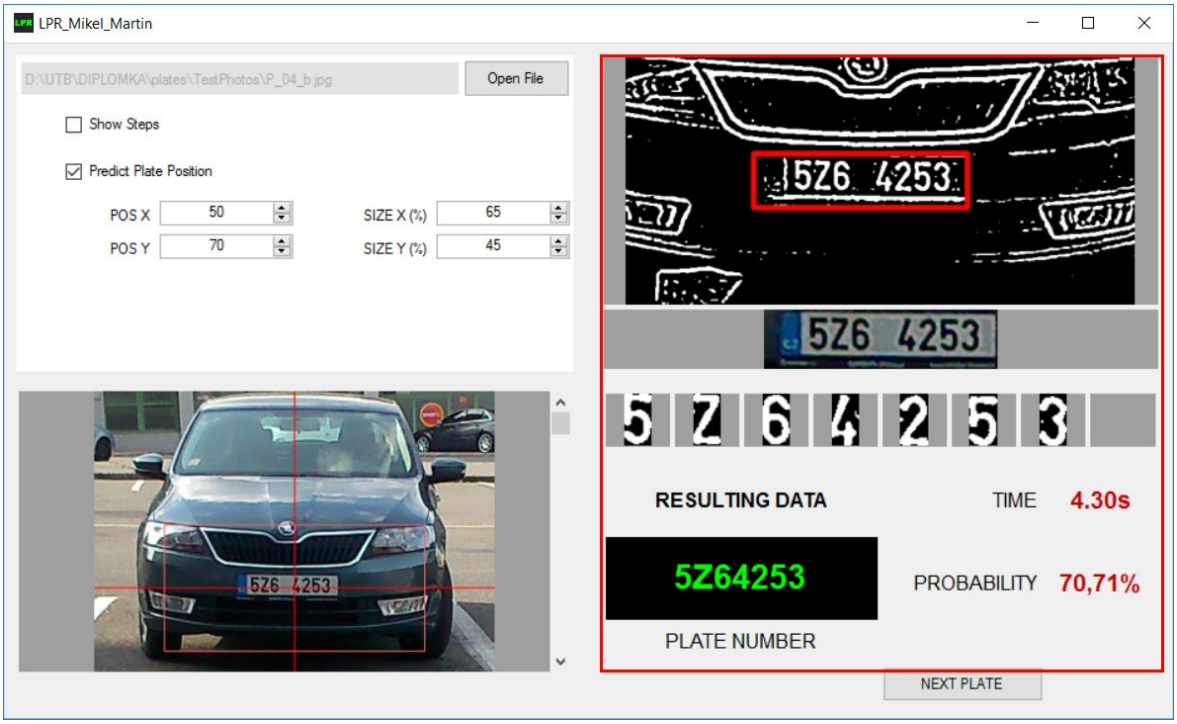


Obr. 49 Spuštění detekce registrační značky

Stisknutím startovacího tlačítka dojde k zahájení rozpoznání obrazu. Pokud nebyla vybrána možnost *Show Steps*, která už byla víše popisována, zobrazí se přímo výsledná data. Panel v pravém horním rohu je určen pro binární obraz. Ten je výstupem prahování. Pro lepší orientaci a ověření, zda systém pracuje správně, je oblast předpokládané registrační značky ohraničena červeným obdélníkem.

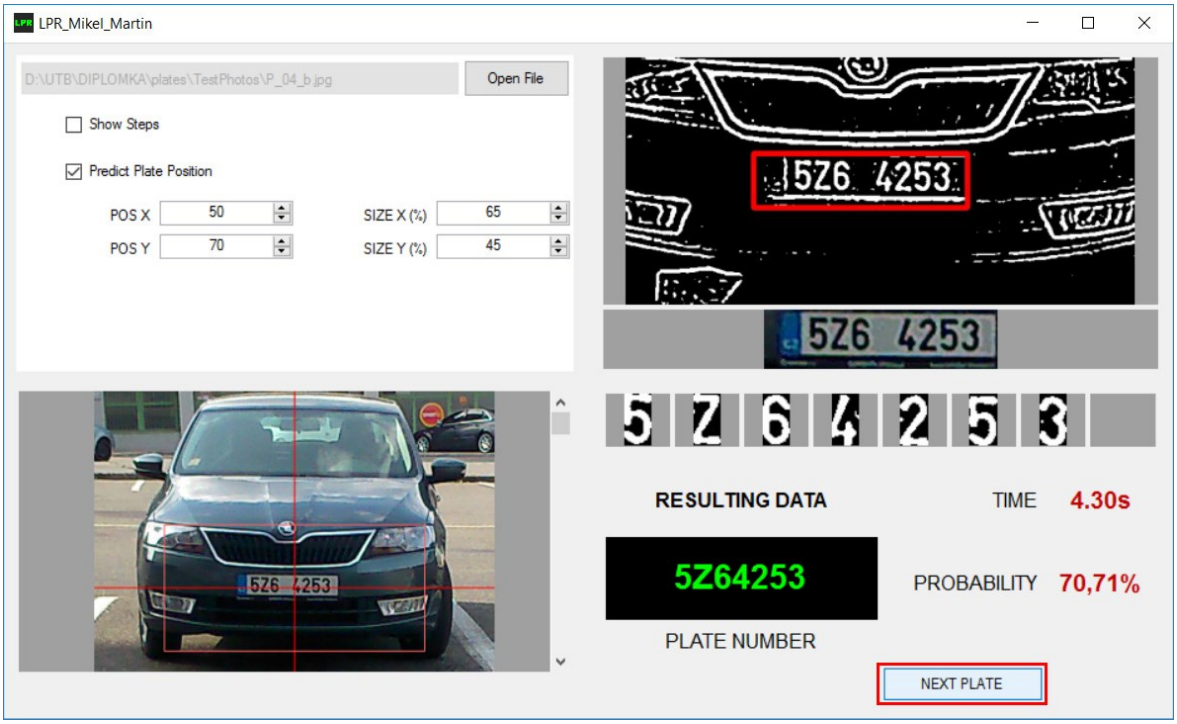
Pod tímto panelem je prostor pro vykreslení výřezu registrační značky z původního obrazu. Jako další jsou k vidění i jednotlivé znaky. Slouží to především k ověření správnosti segmentace těchto znaků. V případě, že by proběhla nesprávná segmentace, celý proces rozpoznávání by tím utrpěl na spolehlivosti.

V sekci *Resulting Data* se nachází informace o průběhu procesu. V tmavém poli jsou zeleným fontem vypsány znaky registrační značky po detekci. Dále celkový čas průběhu detekce (*Time*) a pravděpodobnost (*Probability*). Ta představuje hodnotu shody v procentech segmentovaných znaků a porovnávaných masek znaků.



Obr. 50 Výsledné udaje

Po ukončení rozpoznávání se v pravém spodním rohu aplikace zobrazí tlačítko *NEXT PLATE*. To slouží pro zadání nových vstupních údajů. Celý proces manipulace s LPR aplikací se od tohoto bodu opakuje.



Obr. 51 Detekovat další registrační značku

6 VYHODNOCENÍ SYSTÉMU LPR

V této kapitole je popsán průběh testování vytvořeného systému LPR. Součástí testování bylo vyhodnotit celkovou úspěšnost detekce registračních značek, správnost segmentování jednotlivých znaků a zároveň spolehlivost klasifikace těchto znaků.

6.1 Výsledky testování aplikace

Testování bylo provedeno na skupině celkem 36 fotografií. K jejich pořízení byly využity běžné digitální fotoaparáty. Proběhla úprava velikosti všech fotografií tak, aby rozměry registračních značek splňovaly požadované parametry. V případě, že by byla SPZ příliš velká nebo naopak malá, mohlo by dojít k výraznému poklesu v úspěšnosti jejího rozpoznání v obraze. Kromě velikosti ale fotografie nebyly žádným způsobem upravovány.

Testování proběhlo na všech fotografií se stejnými vstupními parametry. Z důvodu zvýšení rychlosti výpočtu algoritmů byla vybrána možnost *Predict Plate Position*. Hodnoty velikosti oblasti, ve které detekce probíhala, zůstaly podle přednastavených údajů. Tedy *SIZE X = 65* a *SIZE Y = 45*. Docházelo jenom k potřebné korekci hodnot *POS X* a *POS Y* tak, aby se registrační značka nacházela vždy ve vybrané oblasti.

V následující tabulce (*Tabulka 5*) je podrobně popsán průběh a hodnoty testování jednotlivých fotografií. První sloupec obsahuje skutečné číslo dané registrační značky. Dále je zde uvedeno, zda se podařilo SPZ v obraze detekovat. Pokud se to nepodařilo, byly upraveny vstupní podmínky upraveny tak, aby se zvýšili šance na správné nalezení lokality značky.

Ve sloupci „Znaky“ je vidět rozdíl mezi skutečným počtem a počtem úspěšně segmentovaných znaků SPZ (94%). Celkem bylo testováno 274 znaků, z toho jich 260 bylo úspěšně klasifikováno. To představuje úroveň spolehlivosti rozpoznání znaků 93,8%.

Výstupem každého procesu detekce je i výsledný čas a pravděpodobnost, s jakou byli jednotlivé znaky rozpoznány. Každý segmentovaný znak byl porovnáván s maskami pravděpodobných znaků. Pravděpodobnost (*Probability*) představuje průměr maximálních hodnot shody všech znaků SPZ.

Tabulka 2 Výstup z testování LPR

Skutečná SPZ	Nale-zení SPZ	Nalezení po změně nastavení	Znaky		Výstup	Chyby	Čas[s]	Pravd.[%]
			skut.	segm.				
2Z8 6650	NE	ANO	7	7	**2Z88650**	0	4,628	72,14
3Z9 4788	ANO	-	7	7	**3Z94788**	0	4,931	68,57
5Z1 5530	ANO	-	8	8	**-Z--5530**	2	5,151	48,12
5Z6 4253	ANO	-	7	7	**5Z64253**	0	4,677	70,71
3Z2 7595	ANO	-	7	7	**3Z27595**	0	4,63	75
4Z8 2811	ANO	-	8	8	**4Z8-2811**	0	5,353	65,62
4Z8 2811	ANO	-	7	7	**4Z82811**	0	4,73	76,46
2Z5 0901	ANO	-	8	8	**2Z5-0901**	0	4,904	71,25
4Z8 3010	ANO	-	8	8	**4Z8-3010**	0	5,218	68,12
7U0 6421	ANO	-	8	8	**7U0-6421**	0	5,13	70
3B9 9064	ANO	-	7	7	**3B99064**	0	5,106	65,23
3Z4 3149	ANO	-	8	8	**3Z4-3149**	0	5,233	66,25
6T2 2142	ANO	-	7	6	**6F221-2**	2	4,832	62,86
4T7 9313	ANO	-	7	6	**4F79313**	2	4,623	75
4L6 3733	ANO	-	8	8	**4L6-3733**	0	4,821	68,75
3Z8 8572	ANO	-	8	8	**3Z8-8572**	0	4,753	68,12
3Z8 7991	ANO	-	8	8	**3Z8-7991**	0	5,49	65,62
BZA 0356	NE	NE	7	-	**-**	-	-	-
3Z4 9927	ANO	-	8	8	**3Z4-9927**	0	4,801	66,87
3Z0 2252	ANO	-	8	8	**3Z0-2252**	0	5,3	69,37
5M6 4115	ANO	-	7	7	**5MB4115**	1	5,378	61,25
3Z3 4711	ANO	-	8	8	**3Z3-4711**	0	5,56	65,62
4Z8 2446	NE	NE	7	-	**-**	-	-	-
4Z8 6501	ANO	-	8	8	**-Z8-6501**	1	4,949	54,37
4Z6 6382	ANO	-	7	7	**4Z88382**	2	4,913	73,57
4Z3 9553	ANO	-	8	8	**4Z3-9553**	0	4,993	62,5
1Z0 7944	ANO	-	8	8	**1Z0-79--**	2	5,261	46,87
4Z9 1555	ANO	-	8	8	**4Z9--555**	1	4,941	53,75
5Z7 5888	ANO	-	7	7	**5Z75888**	0	4,603	81,43
4Z3 8724	ANO	-	8	8	**4Z3-8724**	0	5,94	65
2Z5 0924	ANO	-	8	8	**2Z5-0924**	0	5,144	68,75
6T0 5416	ANO	-	8	7	**6-0-5416**	1	5,274	55,62
5Z6 5545	ANO	-	8	8	**5Z8-5545**	1	4,897	59,37
3C4 5556	ANO	-	7	7	**3CA5556**	1	5,113	58,12
4Z4 2772	ANO	-	7	7	**4Z42772**	0	4,694	76,43
7A8 7777	ANO	-	8	8	**7A8-7777**	0	4,93	68,12

Jak už bylo uvedeno, bylo testováno celkem 36 fotografií. Z toho byla ve 34 z nich nalezena správná oblast registrační značky. Toto číslo představuje úspěšnost 94,4%. Následně v těchto nalezených lokacích byla provedena segmentace znaků se spolehlivostí 91,2%.

Tabulka 3 Úspěšnost detekování SPZ

	POČET	%
Celkem testovaných fotografií	36	100
Úspěšně nalezená oblast SPZ	34	94,4
Úspěšná segmentace všech znaků	31	91,2

Z celkového počtu lokalizovaných registračních značek bylo klasifikováno bez jediné chyby celkem 23 značek. S jednou chybou bylo vyhodnoceno 6 značek a 2 chyby nastaly v pěti případech. Pokud byla úspěšně nalezena oblast SPZ, nedošlo při klasifikaci k více než dvěma chybám.

Podle celkového skutečného počtu znaků registrační značky a podle počtu správně klasifikovaných znaků lze vypočítat úspěšnost, s jakou byla daná SPZ rozpoznána. Průměr všech takto získaných hodnot představuje celkovou spolehlivost systému LPR. Celková hodnota úspěšnosti je tedy 93,75%.

Tabulka 4 Úspěšnost rozpoznání znaků SPZ

	POČET	%
Celkem rozpoznaných SPZ	34	100
bez chyby	23	67,6
1 chyba	6	17,6
2 chyby	5	14,7
více chyb	0	0

Průměrná hodnota časů procesu hledání a klasifikování znaků je 5,027 sekund. Celková shoda mezi segmentovanými objekty a přikládanými maskami pravděpodobných znaků byla vyhodnocena jako 66,02%.

Tabulka 5 Průměrný čas a pravděpodobnost správného rozpoznání SPZ

	POČET	PRŮMĚRNÉ HODNOTY	
		ČAS	PRAVDĚP.
Celkem rozpoznaných SPZ	33	5,027 s	66,02%

6.2 Závěrečné shrnutí

Oblast registrační značky je realizována pomocí metody srovnávání se vzorem (Template Matching). Tento krok vyhledávání masky byl, na rozdíl od obdobných konkurenčních systému LPR, nasazen hned po prahování (binární obraz). U systému LPR popisovanému v teoretické části práce, je toto vyhledávání prováděno až po převedení bipolárního obrazu na kontury (hranovým detektorem). Důvodem vynechání tohoto kroku bylo větší množství bílých pixelů v oblasti masky. Bylo tak dosahováno lepších a spolehlivějších výsledků.

Aplikace je určena především pro hledání standardních registračních značek vydávaných v České republice. Ovšem využít by se dal i pro detekování poznávacích značek na přání, které můžou mít libovolný počet znaků (min. osm) a nemusí obsahovat mezeru pro značku STK. Pro vyšší spolehlivost by bylo vhodné aplikovat ještě jednu masku pro hledání a to ve formátu osmi znaků vedle sebe. Není ovšem vyloučeno, že by nebyla nalezena a správně segmentována SPZ na přání i současným systémem bez úprav.

Průměrný čas, který byl potřeba na provedení výpočtů pro nalezení a klasifikování jedné registrační značky byl 5 vteřin. Toto testování ovšem probíhalo na hardwarově slabším zařízení. Při testování na výkonnějším počítači byl průměrný čas stažen až na 1,4 s. Systém se tak dá označit za velice rychlý. U systémů s komerčním využitím byla uváděna časová náročnost kolem 3 vteřin.

Celková úspěšnost rozpoznání značky, segmentaci a následné klasifikace jednotlivých znaků dosahuje 93%, i z tohoto pohledu se dá vytvořený systém označit za poměrně spolehlivý.

Největší problémy při rozpoznávání konkrétního znaku byly zaznamenány u písmene T a čísla 4. Zatím co písmeno T bylo, většinou kvůli nesprávné segmentaci, zaměňováno za F, číslice 4 zůstala několikrát bez konkrétního výsledku.

Jelikož velké množství z testovaných fotografií obsahuje vozidlo focené ze zadní strany, jsou na registračních značkách často viděny značky STK. Tento fakt zapříčinil, že při segmentaci docházelo k označení této oblasti za jeden znak. Ovšem při klasifikaci byl zapsán jako pomlčka („-“). Z toho důvodu je uváděno, že některé SPZ obsahují místo 7 až 8 znaků. Číslo RZ bylo pak zapsáno např. jako 7U0-6421 místo 7U0 6421. Ve výsledcích nebylo toto považováno za chybu.

Po každé provedené detekci jsou výsledné obrazové materiály uloženy do adresáře „*Licence Plate Recognition*“ ve sožce „*Dokumenty“*. Je tak možné skontrovat, jak proběhl převod obrazu do binárního, zda byla oblast SPZ správně segnetována, nebo jak vypadali jednotlivé znaky po změně velikosti před srovnáváním se vzorem.

ZÁVĚR

Cílem této práce bylo pojednat o problematice zpracování obrazu a následné vytvoření vlastního nástroje pro vyčítání poznávacích značek vozidel.

V teoretické části jsou popsány metody a základní problematika práce s obrazem. Byl proveden výběr dostupných nástrojů pro práci s obrazem, popsána jejich základní vlastnosti, výhody a nevýhody. Dále byl charakterizován systém Licence Plate Recognition, vysvětleny kroky od předzpracování originální scény, přes způsoby detekce pravděpodobné oblasti registrační značky, metody segmentace jednotlivých znaků až po jejich samotnou klasifikaci.

Praktická část už se zaměřila na samotný vývoj nástroje pro vyčítání registračních značek vozidel. Byly popsány vybrané nástroje pro rozpoznání obrazu a důvody této volby. Součástí je i popis tvorby a principu, jakým daná aplikace pracuje. Obsahem praktické části této práce je i manuál, který vysvětluje práci s výslednou aplikací. Po vytvoření tohoto systému bylo provedeno testování jeho spolehlivosti a rychlosti. Výsledky a celkové hodnocení těchto výstupů tvoří závěrečnou kapitolu této práce.

Přílohy pak obsahují adresáře s kompletním projektem ve MS Visual Studiu 2015, samostatnou aplikaci LPR, sadu testovacích fotografií a jeden textový dokument s jednoduchým návodem k obsluze dané aplikace.

Výsledný systém slouží spíše jako ukázka, jak podobné analýzy obrazu pracují. V praxi by neobsahoval panely pro vykreslení jednotlivých kroků, ale jenom originální scénu (fotografii, video) a výstupem by bylo pouze číslo registrační značky v textovém formátu. Samozřejmě podle potřeb uživatele by mohl měřit čas zpracování, pravděpodobnost úspěšného detekování a podobně. Tyto důvody vedou k urychlení algoritmů a výsledné časy by byli výrazně kratší. Pochopitelně velký vliv nese i výkon HW.

Cílem práce bylo seznámit se s problematikou analýzy obrazu a vyzkoušet si aplikování těchto získaných informací aplikovat na konkrétní příklad.

SEZNAM POUŽITÉ LITERATURY

[1] HLAVÁČ, Václav a Milan ŠONKA. *Počítačové vidění*. Praha: Grada, 1992. ISBN 80-85424-67-3

[2] HLAVÁČ, Václav a Miloš SEDLÁČEK. *Zpracování signálů a obrazů*. Vyd. 2., přeprac. Praha: Vydavatelství ČVUT, 2005. ISBN 8001031101.

[3] JIŘÍ, Žára a Jiří ŽÁRA. *Moderní počítačová grafika*. Brno: Computer Press, 2004. ISBN 80-251-0454-0.

[4] Color Models. *Intel Software* [online]. Santa Clara: Intel, 2016 [cit. 2017-03-21]. Dostupné z: https://software.intel.com/en-us/node/503873

[5] CIE-LAB Color Space. *Interview Helper* [online]. 2013 [cit. 2017-03-14]. Dostupné z: http://sheriffblathur.blogspot.cz/2013/07/cie-lab-color-space.html

[6] HSL, HSB and HSV color. *Codeitdown.com* [online]. Costa Rica: Jose Vargas, 2013 [cit. 2017-03-14]. Dostupné z: http://codeitdown.com/hsl-hsb-hsv-color/

[7] MODELOS DE COLOR. *Gore HV* [online]. Spain: GOREHV, 2014 [cit. 2017-03-14]. Dostupné z: http://www.gorehv.es/photo/modelos-color-rgb-cmyk/

[8] DOBEŠ, Michal. *Zpracování obrazu a algoritmy v C#*. Praha: BEN - technická literatura, 2008. ISBN 9788073002336.

[9] *Image Processing Learning Resources* [online]. The University of Edinburgh: Fisher R., 2004 [cit. 2017-03-27]. Dostupné z: http://homepages.inf.ed.ac.uk/rbf/HIPR2/hipr_top.htm

[10] *Základní principy strojového vidění* [online]. Brno: Elektroprumysl.cz, 2011 [cit. 2017-04-03]. Dostupné z: http://www.elektroprumysl.cz/automatizace/zakladn-principy-strojoveho-videni-6-dil

[11] *Centre for Telematics and Information Technology* [online]. University of Twente: Keulen, Keijzer, 2007 [cit. 2017-04-10]. Dostupné z: https://www.researchgate.net/figure/254859348_fig3_Fig-3-2D-Gaussian-distribution-with-A-1-s-x-5-s-y-8-and-mean-0-0

[12] SOILLE, Pierre. *Morphological image analysis: principles and applications*. 2nd ed. New York: Springer, c2003. ISBN 3-540-42988-3.

[13] Morfologické operace. *Vutbr.cz* [online]. Brno: VUT, 2009 [cit. 2017-04-15]. Dostupné z: http://midas.uamt.feec.vutbr.cz/ZVS/Exercise10/content_cz.php

[14] VÁCLAV, Hlaváč. *Šedotónová matematická morfologie*. Praha, 2005. České vysoké učení technické v Praze.

[15] *MATLAB* [online]. Natick (United States): MathWorks, 2017 [cit. 2017-04-17]. Dostupné z: https://www.mathworks.com/products/matlab.html

[16] Ceník produktů systému MATLAB. *Humusoft* [online]. Praha: Humusoft, 2017 [cit. 2017-04-17]. Dostupné z: http://www.humusoft.cz/matlab/pricing

[17] GNU Octave. *GNU Octave* [online]. Texas, 2017 [cit. 2017-04-17]. Dostupné z: https://www.gnu.org/software/octave/

[18] OpenCV. *OpenCV* [online]. United States: OpenCV team, 2017 [cit. 2017-04-18]. Dostupné z: http://opencv.org/about.html

[19] Emgu CV. *EmguCV* [online]. United States, 2017 [cit. 2017-04-18]. Dostupné z: http://www.emgu.com/wiki/index.php/Main_Page

[20] DOBROVOLNÝ, Martin. *Rychlý algoritmus rozpoznání registračních značek vozidel*. Pardubice, 2009. Univerzita Pardubice.

[21] OpenPark Smart Parking. *Open Minds CIT* [online]. Cairo (Egypt): Open Minds CIT, 2016 [cit. 2017-03-10]. Dostupné z: http://www.openmindsys.com/index.php/portfolio/openpark-smart-parking

[22] EDITED BY B. RACHEV a A. SMRIKAROV. *Proceedings of the International Conference on Computer Systems and Technologies (e-Learning): CompSysTech '04, Rousse, Bulgaria, 17-18 June*. Bulgaria: Bulgarian Chapter of ACM, 2004. ISBN 9549641384.

[23] *Elektrotechnika a elektronika v dopravě: sborník semináře : [24.9.2009]* [online]. Pardubice: Univerzita Pardubice, 2009 [cit. 2017-03-10]. ISBN 978-80-7395-194-8.

[24] Vyhláška č. 343/2014 Sb. *Zakony pro lidi* [online]. 2017 [cit. 2017-05-14]. Dostupné z: https://www.zakonyprolidi.cz/cs/2014-343#cast5

[25] České registrační značky. *3260.cz* [online]. Hájek M., 2017 [cit. 2017-05-14]. Dostupné z: http://www.3260.cz/index.htm

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

HW Hardware – fyzické technické vybavení počítače.

LPR Licence Plate Recognition – systém rozpoznávání poznávacích značek.

RZ Registrační značka.

SPZ Státní poznávací značka.

OCR Optical Charakter Recognition

WFA Windows Forms Application

EU Evropská Unie

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

Součástí této práce jsou následující přílohy:

LPR_MSVS_Projekt – obsahuje celkový projekt systému LPR v MS Visual Studio

LPR_Aplikace – samostatná aplikace LPR s návodem

LPR_Photo – fotografie pro testování LPR

LPR_Návod k použití.txt – textový soubor s jednoduchým návodem k aplikaci LPR