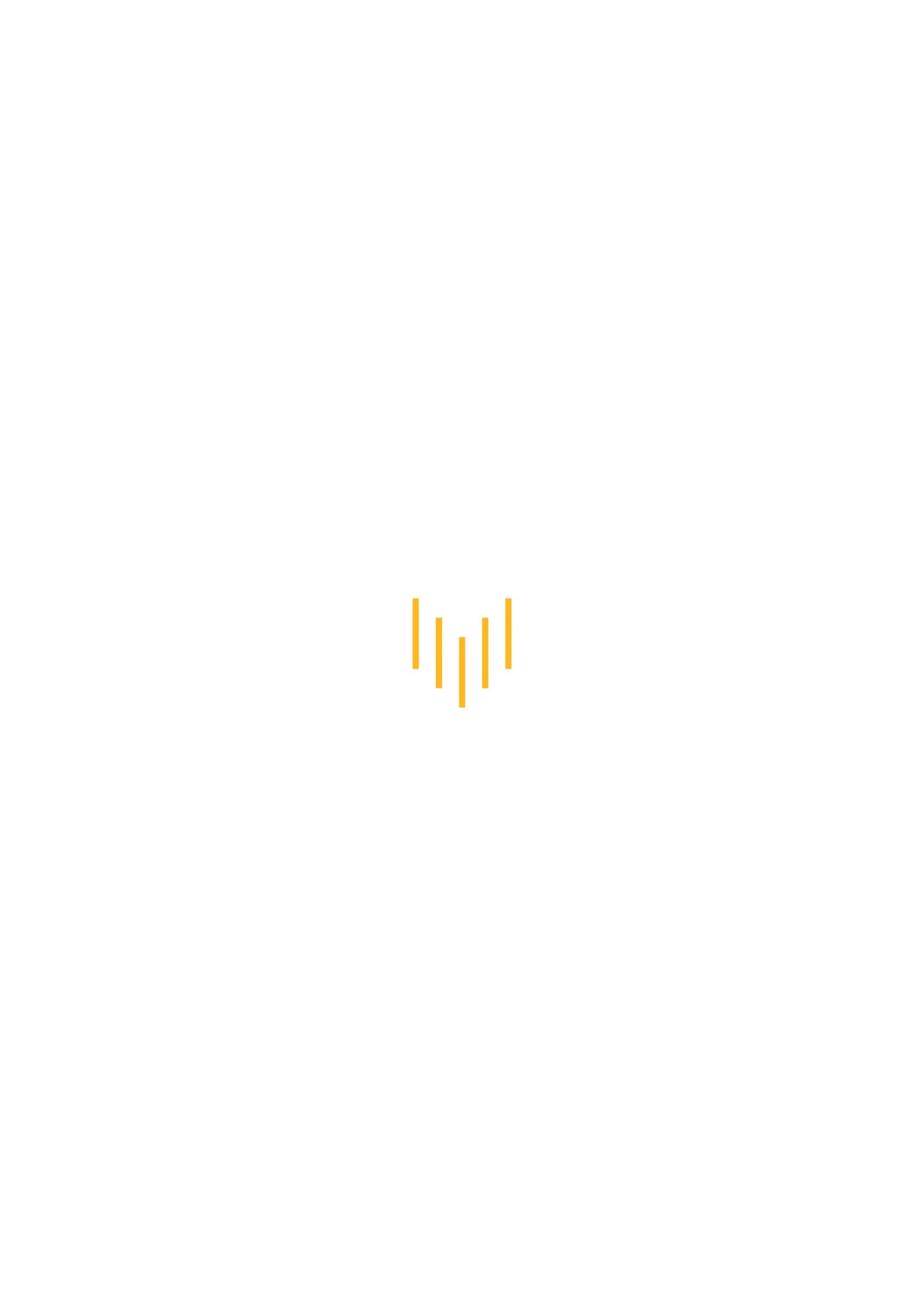


Bc. Adam Friedrich

**Rozpoznávání obrazu pro automatické brány**

Diplomová práce

Ostrava 2021





Bc. Adam Friedrich

**Rozpoznávání obrazu pro automatické brány**

The Image Recognition for Automatic Gates

Diplomová práce

Vedoucí práce:

Ing. Jiří Kulhánek Ph.D.

Ostrava 2021

**ZADÁNÍ SEMESTRÁLNÍHO PROJEKTU II.**

**Student:** Bc. Adam Friedrich

**Studijní program:** N2301 Strojní inženýrství

**Studijní obor:** 3902T004 Automatické řízení a inženýrská informatika

**Vedoucí diplomové práce:** Ing. Jiří Kulhánek, Ph.D.

**Téma:** Rozpoznávání obrazu pro automatické brány

The Image Recognition for Automatic Gates

**Zásady pro vypracování:**

1. Seznamte se se současnými technologiemi pro zpracování obrazu silničního provozu ze stacionárních kamer.
2. Navrhněte vhodnou SW a HW platformu pro rozpoznání obrazu ze silničního provozu s aplikací na automatickou bránu.
3. Navrhněte algoritmus, který detekuje přijíždějící vozidlo a jeho korektní polohu a unikátní parametry pro otevření brány.
4. Zhodnoťte dosažené výsledky a navrhněte směry dalšího řešení.

**Doporučená literatura:**

GONZALEZ, Rafael C. a Richard E. WOODS. Digital image processing. Fourth edition. New York: Pearson, [2018]. ISBN 978-1-292-22304-9.

ŠONKA, Milan, Václav HLAVÁČ a Roger BOYLE. Image processing, analysis, and machine vision. Fourth edition. [Austrálie]: Cengage Learning, [2015]. ISBN 978-1-133-59369-0.

BOVIK, Alan C., ed. Handbook of image and video processing [online]. Second edition. [Amsterdam]: Elsevier, 2007 [cit. 2020-12-03]. ISBN 978-0-12-119792-6.

HORNBERG, Alexander, ed. Handbook of machine and computer vision: the guide for developers and users [online]. Second, revised and updated edition. Weinheim, Germany: Wiley-VCH, [2017] [cit. 2020-12-03]. ISBN 9783527413409.

HOTAŘ, Vlastimil. Úvod do problematiky strojového vidění. Část 1, Základní principy a hardware. Liberec: Technická univerzita v Liberci, 2015. ISBN 978-80-7494-156-6.

………………………………………..

Ing. Jiří Kulhánek, Ph.D.

*vedoucí diplomové práce*

**Bibliografický záznam**

Autor: Kamila Homolíková, HOM0123

Název práce: Design předmětu pro pohybové aktivity dětí

Studijní program: B2341 Strojírenství

Vedoucí práce: MgA. Petr Nenička

Akademický rok: 2020/2021

ANOTACE DIPLOMOVÉ PRÁCE

FRIEDRICH, Adam. *Rozpoznávání obrazu pro automatické brány*. Ostrava, 2021. Diplomová práce. Vysoká škola Báňská - Technická univerzita Ostrava. Vedoucí práce Ing. Jiří Kulhánek, Ph.D.

Diplomová práce se zabývá metodikou zpracování digitálního obrazu. V úvodu jsou shrnuty metody a algoritmy pro zpracování a rozpoznávání obrazu. S využitím těchto informací je navržen postup pro aplikaci rozpoznávající vozidlo v obraze a nalezení regionu jeho registrační značky. Z registrační značky jsou poté extrahovány a identifikovány jednotlivé symboly. Výsledkem je informace o RZ vozidla pro autorizaci vjezdu. Je navrženo a následně testováno několik postupů, které jsou testovány na sadě vzorových fotografií, videozáznamu a na *realtime* záznamu z webkamery a je vyhodnocena jejich efektivita.

Klíčová slova: zpracování obrazu, počítačové vidění, Python, automatická brána,

ANNOTATION OF DIPLOMA THESIS

FRIEDRICH, Adam. *Image Recognition for Automatic Gates*. Ostrava, 2021. Diploma thesis. VSB - Technical University of Ostrava. Thesis supervisor Ing. Jiří Kulhánek, Ph.D.

Master thesis is dealing with methods for image recognition. In the introduction the current technologies and algorithms for image processing are described. With this information an algorithm for application is designed. An application which finds a vehicle in the image, locates the region of its license plate and extracts and reads the symbols from the license plate. Several methods are designed, tested and evaluated.

Key words: image processing, computer vision, Python, automatic gate

Obsah

[1 Úvod 11](#_Toc61459237)

[2 Zpracování obrazu 12](#_Toc61459238)

[2.1 Reprezentace obrazu 12](#_Toc61459239)

[2.2 Barevné modely 13](#_Toc61459240)

[2.2.1 RGB 13](#_Toc61459241)

[2.2.2 CMY a CMYK 15](#_Toc61459242)

[2.2.3 Stupně šedi 16](#_Toc61459243)

[2.3 Předzpracování obrazu 16](#_Toc61459244)

[2.3.1 Jasové transformace 17](#_Toc61459245)

[2.3.2 Vyhlazení obrazu 17](#_Toc61459246)

[2.3.3 Histogram 18](#_Toc61459247)

[2.4 Morfologické operace 19](#_Toc61459248)

[2.4.1 Dilatace 19](#_Toc61459249)

[2.4.2 Eroze 20](#_Toc61459250)

[2.5 Segmentace obrazu 21](#_Toc61459251)

[2.5.1 Prahování 21](#_Toc61459252)

[2.5.2 Hledání hran 22](#_Toc61459253)

[2.5.3 Hledání oblastí 25](#_Toc61459254)

[2.6 Klasifikace 25](#_Toc61459255)

[2.6.1 Kontury, skelet 25](#_Toc61459256)

[2.7 Optické rozpoznávání znaků (OCR) 25](#_Toc61459257)

[3 Prostředí NI Vision Builder 25](#_Toc61459258)

[4 Návrh řešení 27](#_Toc61459259)

[5 Implementace 27](#_Toc61459260)

[6 Testování 27](#_Toc61459261)

[6.1 Testování na sadě vzorových fotografií 27](#_Toc61459262)

[7 Výsledky 27](#_Toc61459263)

[8 Závěr 27](#_Toc61459264)

[9 Použitá literatura 28](#_Toc61459265)

[10 Přílohy 28](#_Toc61459266)

# Seznam použitého značení a symbolů

* OCR Optical Character Recognition
* RZ Registrační značka
* B Bajt
* b bit
* Pi pixel
* RGB Red, Blue, Green (Červený, Modrá, Zelená) - barevný model
* CMY Cayen, Magenta, Yellow (Azurová, urpurová, Žlutá) - barevný model
* CMYK Cyan, Magenta, Yellow, blacK (Azurová, Purpurová, Žlutá, Černá) – barevný model
* HSI Hue, Saturation, Intensity barevný model
* 2D dvourozměrný prostor
* 3D trojrozměrný prostor
* SW Software
* HW Hardware

# Úvod

Schopnost vnímat okolí zrakem je pro nás jednou z nejzákladnějších vlastností. Pomocí zraku získáváme největší množství informací z okolního světa. Lidské oko zachycuje světlo, které se odráží od okolních předmětů a přenáší tyto informace do mozku ke zpracování a vyhodnocení. Tento proces se s vývojem technologií snažíme naučit počítač a říkáme mu počítačové vidění. Zpracování obrazu a s tím úzce spjatý pojem počítačové vidění je v dnešní době čím dál aktuálnějším tématem, a setkáváme se s ním denně všude kolem nás. Tyto dva pojmy spolu souvisí, avšak neznamenají to samé. Zpracování obrazu si můžeme představit jako soubor technologických úprav obrazu a jeho vylepšení. Počítačové vidění je proces, při kterém stroj, většinou digitální počítač, automaticky zpracuje obrázek a vyhodnotí „co se na daném obrázku nachází“. Neexistuje přesně stanovená hranice, která by říkala, kde jedno začíná a druhé končí. Obecně se ale zpracování obrazu definuje jako disciplína, kde vstupem i výstupem je obraz. Výstupem aplikace počítačového vidění už pak není obraz, ale nějaká informace o obraze a objektech, které se na něm nachází. Málokteré odvětví průmyslu se dnes obejde bez nějaké formy počítačového vidění. Důvody a způsob použití můžou být různé, může se jednat například o kontrolu kvality výrobků od kontroly barvy nebo přítomnosti po kontrolu tvaru, rozpoznávání obličejů na letištích nebo pro odemykání našich chytrých telefonů pomocí ověření obličeje majitele, monitorování vozidel v dopravě a čtení registračních značek, a v neposlední řadě třeba skenování čárových kódů zboží optickými skenery. Vždy však nemusí jít o klasický obraz z kamery nebo fotoaparátu. Zpracování obrazu je hojně využíváno i v medicíně na rentgenové snímky, mikroskopické snímky nebo při měření teploty pomocí termokamery.

Osobní motivace a důvod proč: brána na VŠB atd…..

V této práci bude cílem seznámit se s možnostmi zpracování obrazu a zejména technologiemi pro detekci objektů v obraze a detekci a čtení textu. Tyto metody bude následně třeba využít pro návrh aplikace pro detekci a čtení registračních značek (RZ) automobilů.

Úvodní část práce bude obsahovat seznámení s možnostmi zpracování obrazu. Bude zde popsán digitální obraz a jeho reprezentace skutečné scény z reálného světa, popis barevného obrazu pomocí barevných modelů a obrazu ve stupních šedi. Dále budou popsány metody předzpracování obrazu a jeho přípravy pro detekci objektů. Následně budou popsány metody segmentace obrazu, hledání a klasifikace objektů zájmu, kterými jsou v této práci vozidla, registrační značky a také písmena a číslice na RZ. Pro rozpoznání těchto znaků budou popsány a použity vhodné metody optického rozpoznávání znaků (OCR). Druhá část práce bude obsahovat návrh a postup řešení, pro který bude využito Open Source programovacího jazyka Python a prostředí Visual Studio Code. Budou zde popsány použité knihovny a popis možností, které tato platforma nabízí. Na základě poznatků z první části práce budou navrženy algoritmy řešení a ty budou implementovány pomocí jazyka Python. Následně budou tyto metody testovány na sadě vzorových fotografií a bude vyhodnocena efektivita a možnost využití. V poslední části práce budou navržené postupy testovány na reálných datech a bude navrženo Hardware a Software řešení pro reálnou aplikaci. Nakonec budou zhodnoceny dosažené výsledek a navrženy kroky dalšího postupu.

# Zpracování obrazu

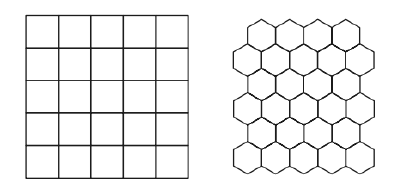
V této kapitole bude obsažen teoretický podklad a informace, potřebné pro pochopení možností práce s digitálním obrazem. Budou zde popsány metody a principy počítačového zpracování obrazu, co to vůbec je digitální obraz a jak z něj získat informace.

## Reprezentace obrazu

K tomu, abychom mohli s obrazem provádět nějaké operace, je nejprve nutné porozumět tomu, co představuje a čím je tvořen obraz v našem počítači. Jde o převod pohledu na okolní svět, který lze z pohledu pozorovatele chápat jako analogový obraz na digitální podobu, se kterou můžeme pracovat pomocí matematických operací. Obraz si můžeme představit jako spojitou funkci f(x,y), kde x a y představují souřadnice bodu v obraze a f je funkční hodnota v tomto bodu. Tato hodnota může představovat hodnotu jasu v obraze ve stupních šedi. Může se jednat také o tři čísla představující hodnoty tří barevných složek modelu RGB, který bude podrobněji popsán později. Rozdělením souřadnicové plochy na celočíselné hodnoty dostaneme matici o rozměrech x krát y. Tato matice pak představuje klasický obrázek složený z pixelů. Proces, kterým takto obraz převedeme na matici se nazývá vzorkování. Rovněž hodnoty jasu v jednotlivých pixelech je třeba digitalizovat a k tomuto slouží obdobný proces, kterému se říká kvantování. (Citaci)

**Vzorkování a kvantování**

Vzorkování je proces, při kterém rozdělíme spojitý signál na posloupnost vzorků, které jsou zaznamenané ve zvolených intervalech. O tom, jak velký interval se použije je třeba rozhodnout na základě požadavků na vzorkování. Velikost tohoto intervalu se nazývá vzorkovací perioda a má zásadní vliv na kvalitu výsledku. Podle zvolené velikosti vzorkovací periody je ovlivněno výsledné množství vzorků, a tudíž i velikost, jakou zabere obraz v paměti počítače. Zároveň je docíleno vyšší přesnosti rekonstrukce původního signálu. Je tedy třeba volit vzorkovací periodu podle účelu zpracovaní daného obrazu. Obecně k tomuto slouží tzv. *Shannon – Kotelnikův* teorém, podle něj je vhodné volit velikost vzorkovací frekvence alespoň dvakrát větší, než je maximální frekvence původní signálu. Toto platí pro zpracování signálů obecně, a v případě zpracování obrazu to znamená, že velikost vzorkovací periody by měla být alespoň dvakrát menší než nejmenší detail, který nás v obraze zajímá. Výsledné množství pixelů je pak udáváno jako pojem rozlišení obrazu. Volba rozlišení je tedy důležitá pro ostrost výsledného obrazu, čím vyšší je, tím více detailů jsme schopni na obraze zachytit. Ve většině případů je obraz tvořen čtvercovou mřížkou, existují ale i jiné typy jako například hexagonální mřížka, která může být vhodnější při jiných typech operace s obrazem jako je například měření vzdálenosti.



Obrázek 1 – čtvercová a hexagonální mřížka dostupné z (zcu.cz 2020)

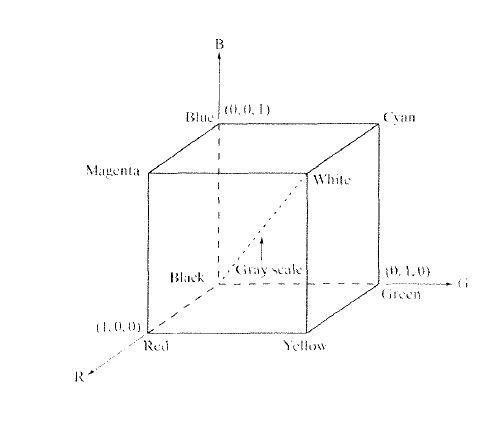
Kvantování je ekvivalentní proces rozdělení signálu do úrovní, ale zaznamenáváme zde úrovně jasu. Podle typu obrazu můžou hodnoty nabývat různého rozmezí. Můžeme zaznamenávat pouze hodnoty 0 a 1, čímž dostaneme čistě černobílý obraz. Pro nejběžnější velikost pixelu se používá jeden bajt, tedy 8 bitů, což odpovídá rozsahu 0-255. (Fiřt, Holota 2020), (Gonzales, Woods 2020)

## Barevné modely

Účel barevných modelů je shrnutí specifikace rozdělení barev do určitých standardů, které budou obecně platit. Většina dnes používaných modelů je orientovaná buďto na hardware v podobě monitorů nebo barevných tiskáren, nebo na aplikace, kde je práce s barvami hlavním cílem. V praxi se při digitálním zpracování obrazu používá nejčastěji model RGB (Red, Blue, Green), modely CMY (cyan, magenta, yellow) a CMYK (cyan, magenta, yellow, black) zejména pro barevný tisk a model HSI (hue, saturation, intenstity), který úzce souvisí se způsobem, jakým barvy vnímá lidské oko.

### RGB

Jedná se o nejvíce používaný barevný model, ve kterém je každá barva vyjádřena hodnotami jasu tří složek a to červené (Red), zelené (Green) a modré (Blue). Tento model je založen na Kartézském souřadnicovém systému, kde barevný prostor znázorňuje krychle na obrázku (Obr. 2) a kde nejvyšší hodnoty RGB barev jsou v rozích na zaznačených osách. V protilehlých rozích jsou barvy modelu CMY a černá barva je v bodě, kde všechny osy mají hodnotu 0. Bílá barva je pak logicky naopak v bodě, který je od počátku nejdále. Osa spojující bílou a černou barvu je množinou bodů, kde jsou hodnoty RGB barev shodné a přestavuje škálu stupňů šedi.



Obrázek 2 – Schéma RGB barevné krychle (Gonzales, Woods 2020)

Každá ze tří barevných složek může nabývat hodnot zpravidla 0 až 255. Složením těchto hodnot vznikne výsledná barva. Hovoříme o tzv. aditivním skládání barev. Jak lze vidět na obrázku (4), jsou-li všechny barvy v nulové intenzitě, vznikne černá barva. Nabývají-li všechny barvy maximální intenzity vznikne bílá. Takový obraz představuje trojrozměrnou matici, kde v každém bodě jsou tři barevné hodnoty a kombinace těchto hodnot určuje výslednou barvu pixelu. Na obrázku (Obr. 3) jsou vykresleny všechny tři jednotlivé barevné vrstvy obrazu. Každý obraz tvoří trojrozměrná matice, stejně jako výsledný barevný obraz s tím, že jsou použity hodnoty jedné z barevných složek, a zbylé dvě vrstvy mají hodnotu 0 ve všech bodech obrazu.

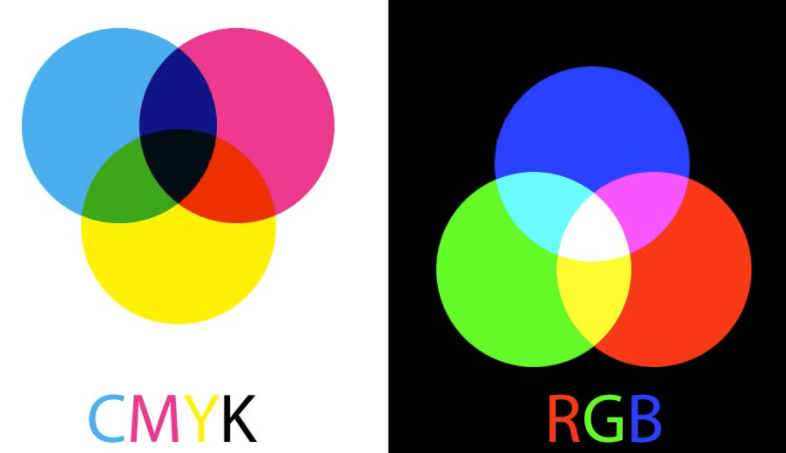


Obrázek 3 – Barevné vrstvy obrazu formátu RGB

### CMY a CMYK

Zkratka modelu rovněž představuje názvy základních barev. Cyan (Azurová), Magenta (purpurová) a Yellow (žlutá). Tento model se velmi často používá v inkoustových tiskárnách, protože používá míchání reálných barev. Bílá barva zde vzniká nepoužitím žádné barvy, černá naopak použitím všech. Protože úplně dokonalé černé barvy se smícháním nikdy nedocílí je model CMYK doplněn o písmeno K představující černou barvu (blacK). Z obrázku (Obr. 4) vyplývá, že se jedná o inverzní model k modelu RGB. Rovněž pro převod hodnot mezi modely platí vztah

kde platí předpoklad, že všechny barevné hodnoty byly normalizovány do rozmezí 0 až 1. Z rovnice je patrné, že například povrch pokrytý pouze žlutou barvou neodrazí žádné modré světlo. Dosadíme-li totiž za Y hodnotu 1, dostaneme pro B hodnotu 0. (Gonzales, Woods 2020)



Obrázek 4 – barevné schéma modelu RGB a CMYK dostupné z (creativepro.com, 2020)

### Stupně šedi

Podle účelu zpracování je potřeba volit vhodný model. V případě, že barvy samotné hrají v úloze velkou roli, bude potřeba pracovat s barevným modelem. V této práci půjde ale převážně o detekci objektu v obraze, kde není potřeba pracovat s barevným obrazem. Obraz ve stupních šedi je oproti barevnému tvořen pouze dvourozměrnou maticí, kde každý bod představuje hodnotu jasu. Na takovouto matici se pak mnohem snáze aplikují postupy pro hledání hran nebo prahování. Tento formát je vhodnější i z důvodu rychlosti zpracování. Vzhledem k tomu, že okem vnímáme jednotlivé barevné složky jinak intenzivně se pro přepočet používá poměr

,

kde f je hodnota jasu daného pixelu a R, G, a B jsou hodnoty jasu jednotlivých barevných složek. Na obrázku (Obr. 5) je znázorněna aplikace převodu barevného obrázku do stupňů šedi.



Obrázek 5 – převod barevného obrazu na obraz ve stupních šedi

## Předzpracování obrazu

Předzpracování obrazu je zcela nezbytná příprava, kterou je nutno provést předtím, než začneme obraz dále zpracovávat. Protože se nám nikdy nepodaří pořídit obraz tak, aby neobsahoval žádný šum a byl dokonale ostrý a nasvětlený, je potřeba z něj tyto nežádoucí vlivy odstranit. Je zároveň nutné myslet na to, co chceme s obrazem dále dělat a jak si ho tedy nejvhodněji připravit. Metod, které lze použít je celá řada, můžeme použít například jen jednu z barevných složek, převést obraz na binární nebo použít různé typy transformace obrazu.

### Vyhlazení obrazu

Šum může v obraze způsobovat řada faktorů a rozlišujeme různé druhy. V závislosti na tom je třeba zvolit vhodné metody pro jeho potlačení. Mezi základní metody pro odstranění šumu patří vyhlazování (*smoothing)* pomocí různých filtrů.

**Průměrování**

Nejjednodušším způsobem vyhlazení obrazu je použití průměrového filtru. Pro použití většiny uvedených filtrů je třeba definovat tzv. masku. Maskou rozumíme oblast pixelů v obraze, která se posouvá postupně po všech pixelech v obraze a pro každý je vypočtena nová hodnota. Pokud například zvolíme masku o velikosti 3x3 pi, bude hodnota pro prostřední pixel vypočítána z 8 okolních pixelů. V případě průměrování je hodnota jednoduše vypočtena jako aritmetický průměr hodnot všech okolních pixelů. Velikost masky zásadním způsobem ovlivňuje také výpočetní náročnost programu, čím větší masku totiž použijeme, tím složitější je výpočet.

**Gaussův filtr**

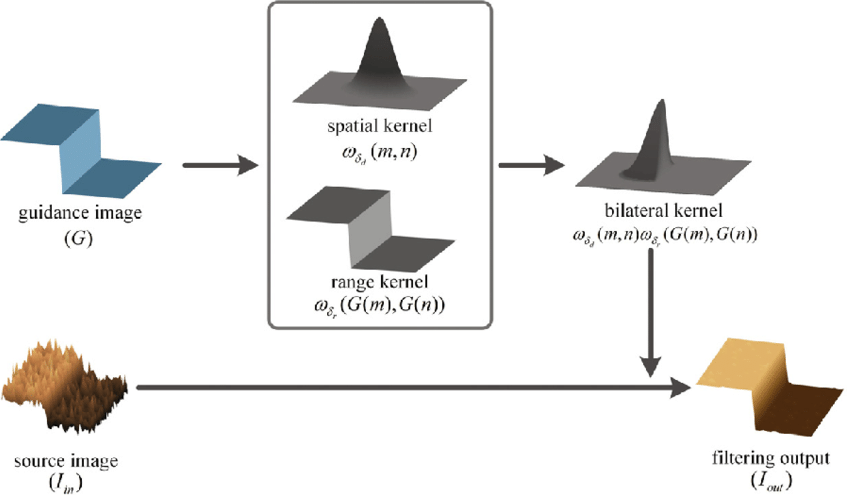
Jedná se o specifický případ průměrového filtru. Postup pro výpočet je stejný, ale každému pixelu v okolí je přiřazena určitá váha. Typicky je nejbližším sousedním pixelům přiřazena nejvyšší váha a vzdálenějším pixelům váha menší. Výsledkem pro hodnotu jasu prostředního pixelu je pak vážený průměr hodnot okolních pixelů. Prodloužit lehce popis filtrů (tak jako ten bilaterální), přidat obrázky principů filtrů.

**Mediánový filtr**

Opět se jedná o obdobný postup výpočtu jako při filtraci s tím rozdílem, že místo průměru hodnot okolních pixelů se pro hodnotu prostředního pixelu použije medián. Tento filtr je velmi výhodný pro odstranění šumu označovanému jako „sůl a pepř“ (Bovik, 2007), (Šebela, 2010).

**Bilaterální filtr**

Ve většině případů je možné říct, že bilaterální filtrace je nejefektivnější metodou vyhlazení obrazu. Rozdíl oproti ostatním metodám filtrace spočívá hlavně v tom, že tyto metody mají nežádoucí vedlejší efekt, a to že rozmazávají hrany v obraze, které jsou téměř vždy důležitými prvky obrazu pro získání informací. Oproti ostatním metodám požívá bilaterální filtrace jak prostorovou závislost, tak závislost intenzity prostorových bodů. Tvar masky pro prostorovou závislost je stejný, jako je tvar Gaussova filtru a vyhlazuje šum stejným způsobem. Druhá funkce bere v potaz podobnost hodnot zpracovávaných obrazových bodů. Jsou tedy v podstatě zvlášť filtrovány skupiny pixelů, které mají podobnou intenzitu a patří tudíž, pravděpodobně, do stejné oblasti v obraze. Při použití bilaterální filtrace je dosaženo maximálního zachování důležitých, což je velmi dobře vidět na obrázku (Obr.7), (Nejezchleb, 2009).



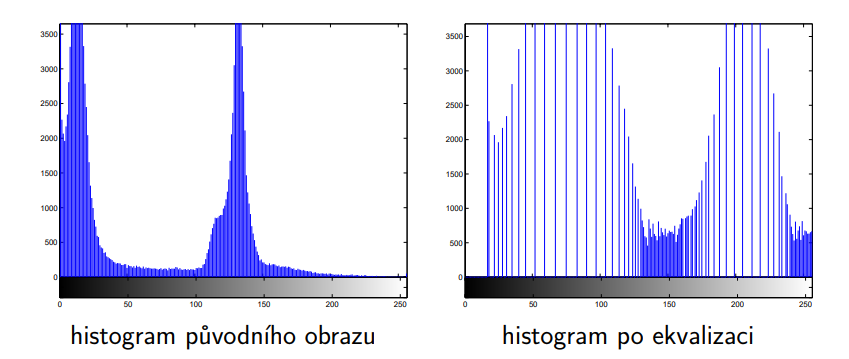
Obrázek 6 – Princip funkce bilaterálního filtru (researchgate.net, 2020)



Obrázek 7 – Aplikace filtrů při použití stejné velikosti masky. Vlevo nahoře – původní šedý obraz, vpravo nahoře – průměrový filtr, vlevo uprostřed – Gaussův filtr, vpravo uprostřed – mediánový filtr, vlevo dole – bilaterální filtr, vpravo dole –Dvakrát po sobě použitý bilaterální filtr

### Histogram

Histogram vyjadřuje četnost hodnot, tedy množství výskytu jedné hodnoty stupně šedi v daném souboru hodnot. Nijak nesouvisí s prostorovými souřadnicemi obrazu. Graficky představuje histogram charakteristiku, kde na vodorovné ose jsou hodnoty jasu pixelů obrazu v rozmezí 0 až 255 a na svislé ose počet výskytů dané úrovně v matici. Histogram sám o sobě nám o obraze hodně říká a je možné z něj vyčíst mnoho informací, jako například zvolit vhodnou hodnotu pro prahování, nebo jej můžeme použít pro úpravu obrazu jakou je například vyrovnání histogramu. To se používá hlavně v případech nekontrastních obrazů, kde rozložení četnosti jednotlivých stupňů šedi je nerovnoměrné jako na obrázku (Obr. 8). Vyrovnáním histogramu se zrovnoměrní zastoupení hladin šedi a zvýrazní se tak lokální kontrast obrazu.



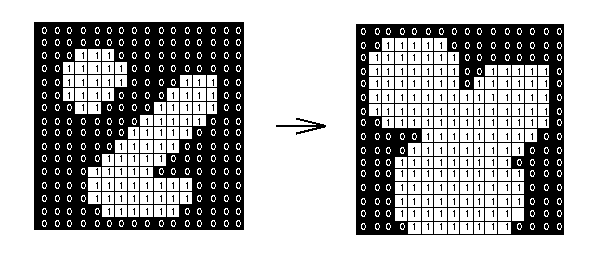
Obrázek 8 – Příklad vyrovnání histogramu dostupné z (ciirc.cvut.cz, 2020)

## Morfologické operace

Další skupinou metod, které je možné použít pro předzpracování obrazu jsou morfologické operace. Princip použití úzce souvisí se vzorkováním obrazu a je podobný jako při použití různých typů filtrace uvedených výše. Základním prvkem je množina souřadných bodů, tzv. strukturní element (*structuring element)* někdy také označovaný jako tvořící element, ze kterého tyto metody vychází. Tento element je obdobou masky používaní při filtraci. Tímto elementem postupně pohybujeme po obraze a aplikujeme jej na každý bod. Velikost elementu, stejně jako volba metody je stejně jako u většiny ostatních metod pro předzpracování obrazu věc konkrétního případu a vždy je zvolení vhodného kompromisu. Větší maska sice odstraní větší množství šumu, ale zase obraz více rozmaže. Stejně tak nevhodně použitá metoda na nevhodný typ šumu v obraze ho rovněž může jen zhoršit. Morfologických operací existuje celá řada, mezi ty základní patří *dilatace, eroze* a dále také *otevření a uzavření*, které je od nich odvozeno.

### Dilatace

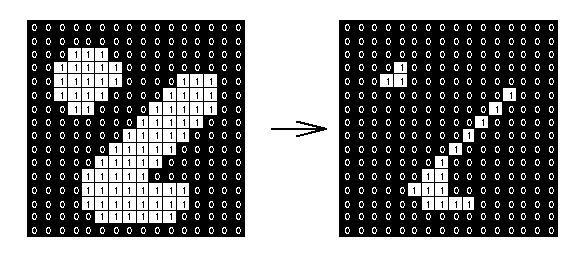
Principem této metody je postupné rozšiřování hranic regionů pixelů. Tuto operaci je vhodné použít, když chceme rozšířit a sjednotit skupiny pixelů v obraze, které představují oblasti zájmu. Použitím dilatace např. pro bílou barvu v binárním obraze se zbavíme výskytu menších skupin černých pixelů a rozšíříme bílé oblasti. Prvním vstupem pro tuto metodu je obraz, který chceme upravit. Tím druhým je již zmíněný strukturní element, který definuje míru efektu dilatace na vstupní obraz. Pokud tedy zvolíme strukturní element o rozměrech 3x3 pixely s počátkem ve středu, dostaneme výsledek jako na obrázku (Obr. 9). Na každý pixel, který má hodnotu 1 (odpovídá tedy bílé barvě), aplikujeme element a hodnoty jeho okolních pixelů se přepíšou podle hodnot odpovídajících pixelů elementu. V tomto případě se všechny okolní body přepíšou na hodnotu 1.



Obrázek 9 – Jednoduchý příklad dilatace při použití čtvercového strukturního elementu 3x3

### Eroze

Jedná se o opačnou metodu, někdy také označovanou jako duální transformaci k dilataci. Jak dilatace, tak eroze se typicky používá pro binární obrazy, je ale možné je použít pro obraz v odstínech šedi. Základním efektem eroze je odstranění hranic regionů pixelů (typicky bílých pixelů). Bílé oblasti se tudíž zmenší a černé díry v nich obsažené se naopak rozšíří. Princip fungování je prakticky totožný jako u dilatace, jen inverzní. Strukturní element je posouván po pixelech v obraze stejným způsobem, ale když dojde na odpovídající pixel s hodnotou 1, nepřepíše okolní pixely, ale naopak daný pixel smaže a přepíše na hodnotu 0, pokud nesplňuje podmínku a neodpovídá přesně tvaru elementu. Použijeme-li tedy stejný strukturní element o rozměrech 3x3 a se všemi pixely hodnoty 1, tak ve výsledném obraze jsou tedy zachovány pouze ty bílé pixely, které mají všech 8 okolních pixelů bílých. (Kolouchová, 2008), (Gonzales, Woods, 2018)



Obrázek 10 - eroze

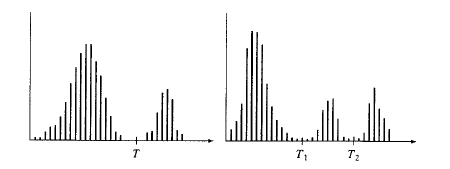
## Segmentace obrazu

Segmentace obrazu představuje dělení obrazu na jednotlivé segmenty (oblasti). Účelem segmentace je nalézt objekt y zájmu a oddělit je od pozadí. V případě této práce bude segmentace využívána zejména pro nalezení automobilů na snímku, lokace jednotlivých prvků vozidla jako jsou světla, registrační značka vozidla a pak i jednotlivé symboly na značce. Postup provedení segmentace je velmi komplexní a je třeba postupovat podle konkrétního problému. Obecně lze ale způsob segmentace rozdělit na několik metod, z nichž zde uvádím ty nejdůležitější pro tuto práci.

### Prahování

Prahování je jednou z nejstarších a nejjednodušších metod zpracování obrazu. Díky svým intuitivním vlastnostem a jednoduchému použití je velmi hojně používáno pro segmentaci. V případě vhodného, velmi dobře kontrastního obrazu, kde je například velmi světlý objekt jasně oddělený od tmavého pozadí (např. světlá součást na černém dopravním páse pod kamerou) může samo o sobě stačit k segmentaci objektu. Hodnotu prahu lze nejjednodušeji určit z histogramu obrazu, kde je vidět rozložení intenzit pixelů v obraze. Podle toho se zvolí práh T, který rozdělí všechny pixely v obraze na černé a bílé, a vznikne tak binární obraz. Pro každý pixel pak platí vztah (123456)

kde *f(x,y)* je hodnota jasu pixelu. Tento vzorec platí jednoduché prahování. Existuje více komplexnějších prahovacích metod jako například adaptivní prahování nebo prahování s hysterezí, které bude popsáno v další kapitole v rámci detekce hran.



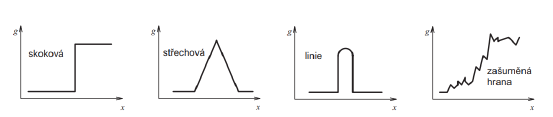
Obrázek 11 – Histogram obrazu ve stupních šedi pro a) jednoduché prahování a b) vícestupňové prahování

Existuje také vícestupňové prahování, které může být vhodné v případě, že na pozadí je více objektů různých intenzit jako na obrázku (obr.11) Hodnota pro každý pixel je pak určena ze vztahu (123).

Oblasti s hodnotami 1 tedy odpovídají jednomu objektu, oblasti s jinou hodnotou jinému objektu a oblasti s hodnotou 0 představují pozadí (Straka, 2009), (Gonzales, Woods, 20188)

### Hledání hran

Hledání hran neboli *edge detection* je zásadní metoda pro nalezení objektů v obraze. Předpokladem nalezení objektu je fakt, že ho lze identifikovat pomocí jeho 2D projekce, počítáme-li se zpracováním obrazu snímaného jednou kamerou. Hranou se v obraze rozumí řada pixelů ležících na hranici dvou oblastí. Hranu identifikujeme tak, že nalezneme oblast, kde se hodnoty sousedních pixelů prudce mění (stoupají nebo klesají). Takováto hrana může mít různý sklon a orientaci. Podle průběhu sklonu může mít různý tvar, jak je znázorněno na obrázku (Bovik, 2007)



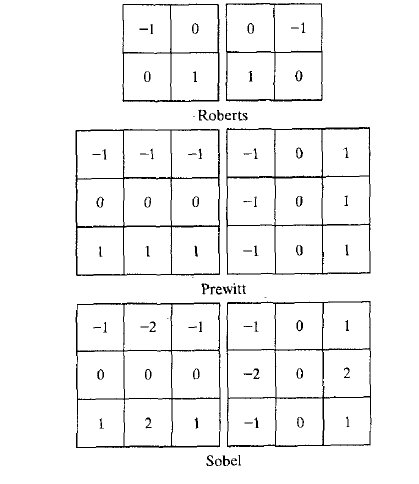
Obrázek 12 – Příklady typů hran v obraze dostupné z (ciirc.cvut.cz, 2020)

V případě hledání hrany v binárním obraze by byla tvořena pouze přechodem hodnot 0 a 1 a šlo by tzv. ideální hranu. Takový průběh je obrázku (obr.12) na první pozici. Nejlépe lze průběh růstu, respektive poklesu jasu pixelů sledovat pomocí první derivace. K výpočtu derivace je nejčastěji používán gradient. Gradient obrázku daného funkcí *f(x,y)* v bodě *(x,y)* je definován jako vektor

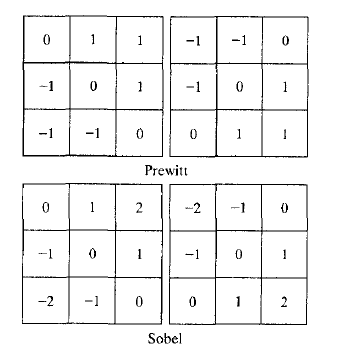
Vektor gradientu ukazuje ve směru nejvyšší změny hodnot funkce *f(x,y)* v bodě *(x,y).* Stejně důležitá je velikost vektoru, kterou určíme podle vztahu

Tento tvar bývá často zjednodušen takto

Tento vztah je mnohem přívětivější, co se týče výpočetní náročnosti při zachování relativních změn stupňů šedi. Nevýhodou je, že filtry vycházející z tohoto vztahu nejsou izotropické, tedy nejsou invariatní v rotaci. Existuje mnoho druhů operátorů, při použití například Sobelova nebo operátoru Prewittové je dosaženo invariantnosti pouze v horizontálním a vertikálním směru. Tyto operátory pak lze modifikovat i pro diagonální směr, jak je ukázáno na obrázku (obr. 13), (obr.14). (Hlaváč, 2020) (Gonzales, Woods, 2020)



Obrázek 13 – Robertsův, Sobelův a operátor Prewittové (Gonzales, woods)



Obrázek 14 –Sobelův a operátor Prewittové pro diagonální směr (Gonzales, woods)

**Cannyho hranový detektor**

Cannyho hranový detektor je pravděpodobně nejefektivnější algoritmus používaný k hledání hran a je to jisté završení v oblasti hledání ideálního hranového detektoru. Jeho cílem je nalezení všech skutečných hran v obraze, a naopak žádných falešných hran tvořených šumem. V kostce lze tohoto docílit tak, že se bude snažit splnit tyto tři požadavky:

1. Spolehlivá detekce (nalézt co nejvíce existujících hranových bodů a žádné falešné)
2. Správná lokalizace (minimální chyba mezi skutečnou a nalezenou pozicí hranového bodu)
3. Jednoznačná odezva (minimum nalezených neexistujících bodů)

Protože čím přesnější je lokalizace hranových bodů, tím méně je jich nalezeno, je třeba opět najít vhodný kompromis mezi jednotlivými body. Dalo by se říct, že Cannyho detektor funguje na principu gradientu a výpočtu první derivace. V každém bodě obrazu je třeba najít gradient a poté maximum první derivace ve směru tohoto gradientu. Body v nalezených maximech, které leží v liniích, tvoří hrany. Nalezené hrany jsou zpravidla vždy velmi tlusté a nevhodné pro další použití, proto se v Cannyho algoritmu používá proces zvaný ztenčení hran. V principu je pro každý pixel ve hraně porovnávána jeho hodnota se sousedními dvěma pixely, ležícími ve směru gradientu, dokud není nalezeno lokální maximum. Abychom vybrali pouze významné hrany používáme proces zvaný hysterezní prahování. To znamená, že jsou zvoleny dvě hranice pro prahování, vše nad horní hranici je ponecháno, vše pod spodní hranicí je smazáno. Body, ležící v oblasti mezi prahy jsou ponechány pouze v případě, že jsou s něčím spojeny. (Straka, 2008), (Hlaváč, 2020)

## Klasifikace

Klasifikace objektů je poslední krok při procesu získávání informace z obrazu. Z vlastností segmentovaných objektů jako je tvar, velikost, pravoúhlost atd. lze určit hodně o daném segmentu. Výstupem je pak informace o tom, jaký objekt se na obraze nachází.

### Kontury, skelet

Jedním ze způsobů, jak identifikovat objekt na obraze je podle jeho kontury. Konturu lze jednoduše popsat jako křivku, která spojuje všechny sousedící body nalezených hran v obraze, které tvoří uzavřené křivky. Zatímco izolované hrany mohou být použity pro různé aplikace, jejich spojování do plynulých kontur bývá velmi užitečné a vytváří obrysy objektů v obraze. Podle vlastností nalezených kontur můžeme o objektech v obraze zjistit několik různých typů informací jako je kruhovost, poměr stran, orientace nebo třeba konvexnost či konkávnost. Zejména pak poměr stran a jejich úhly, které svírají, jsou klíčové vlastnosti pro nalezení obdélníku představujícího registrační značku vozidla. Stejně tak obrysy znaků na RZ (Szeliski, 2020).

the **Douglas–Peucker algorithm**

## Optické rozpoznávání znaků (OCR)

Zkratka OCR pochází z anglického *Optical Character Recognition*, což v je v doslovném překladu optické rozpoznávání znaků.

# Návrh SW a HW

Porovnat srovnání ideálně několika SW i HW možností a variant. Popsat třeba i MATLAB – nějakou diplomku, technickou zprávu.

USB webkamery, Inteligentní kamery…

HW – Počítač s Windows, Raspberry PI…

Ideálně tabulka výhody a nevýhody jednotlivých řešení. Plus cíl – robustní drahé řešení nebo levné atd..

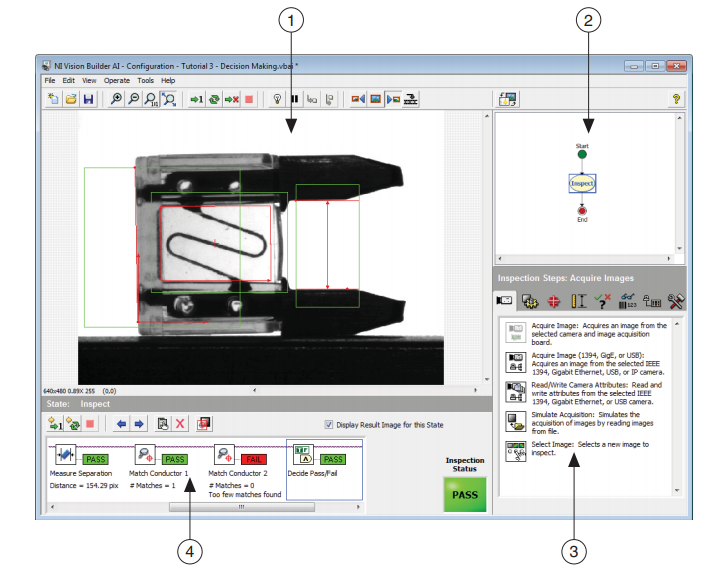
Kritéria volby vlastní – ale úvaha proč byl který HW a SW vybrán

PROČ JSEM VYBRAL PYTHON A NE VB nebo MATLAB

Pro volbu Software prostředí jsem první návrhy předzpracování a segmentace provedl v prostředí NI Vision Builder a v prostředí Microsoft Visual Studio Code, které jsem použil pro tvorbu textového souboru v jazyce Python. Obě řešení měly své výhody i nevýhody, pro tuto práci jsem se nakonec rozhodl použít *Python*, protože vykázal lepší výsledky po prvních pokusech. Zároveň je pro tento jazyk dostupnější větší množství informací a zdrojů a poskytuje komplexnější možnosti.

## NI Vision Builder

NI Vision Builder je Software, který umožňuje vyvíjet systémy strojového vidění zejména pro inspekci strojních součástí. Obsahuje několik knihoven funkcí, které je možné konfigurovat a kombinovat pro dosažení vhodné kontroly. Uživatelské rozhraní je složeno ze 4 oken (Obr. 15), ve kterých je možné nastavit proces inspekce kontrolovaného obrazu pomocí skládání připravených funkcí ve formě bloků a nastavování hodnot parametrů daných funkcí. Tento software poskytuje většinu známých metod, uvedených v této práci ve formě předem připravených funkcí s nastavitelnými parametry.



Obrázek 15 – prostředí NI Vision Builderu

* **1) Hlavní okno** – Zobrazuje obrázek, který je zpracováván, stránky s vlastnostmi některých kroků nebo stavový diagram inspekce. Toto okno je využíváno k nastavení oblastí zájmu na obrázku, nastavení jednotlivých parametrů daného kroku a tvorbu stavového diagramu.
* **2) Přehledové okno** – Toto okno zobrazuje obvykle miniaturu zpracovávaného obrázku nebo stavový diagram
* **3) Paleta kontrolních funkcí** – Obsahuje seznam a popis funkcí, které lze použít pro inspekci.
* **4) Okno konfigurace stavu** – Zobrazuje seznam kroků použitých v dané inspekci.

## Python

Python je v poslední době pravděpodobně jeden z nejpoužívanějších jazyků, a to hlavně pro svou jednoduchost a intuitivitu. Díky tomu existuje spousta dostupných zdrojů, informací a řešení. Pro tento jazyk zároveň existuje řada knihoven, které je možné využít. Pro navržení algoritmu byly použity tyto veřejně dostupné knihovny: *Opencv (funkce pro operace s obrazem)* a *Numpy (matematické funkce pro práci s vektory a maticemi).* Pro vývoj zdrojového kódu bylo použito prostředí Visual Studio Code.

# Návrh řešení

Klidně využít jeden řádek v kódu s jedním příkazem z knihovny, ale vědět co proč a jak funguje. Vědět důvod proč jsem ho použil, ostatní možnosti, proč jsem zvolil tuhle.

Klidně použít cokoliv je dostupné, ale dotáhnout to k optimalizaci mého konkrétního problému. Nedělat znovu to, co už existuje.

Když jsem to nastavil takhle, fungovalo to takhle, když jinak, fungovalo to hůř. Zhodnocení atd.. proč jsem se rozhodl co použít co ne.

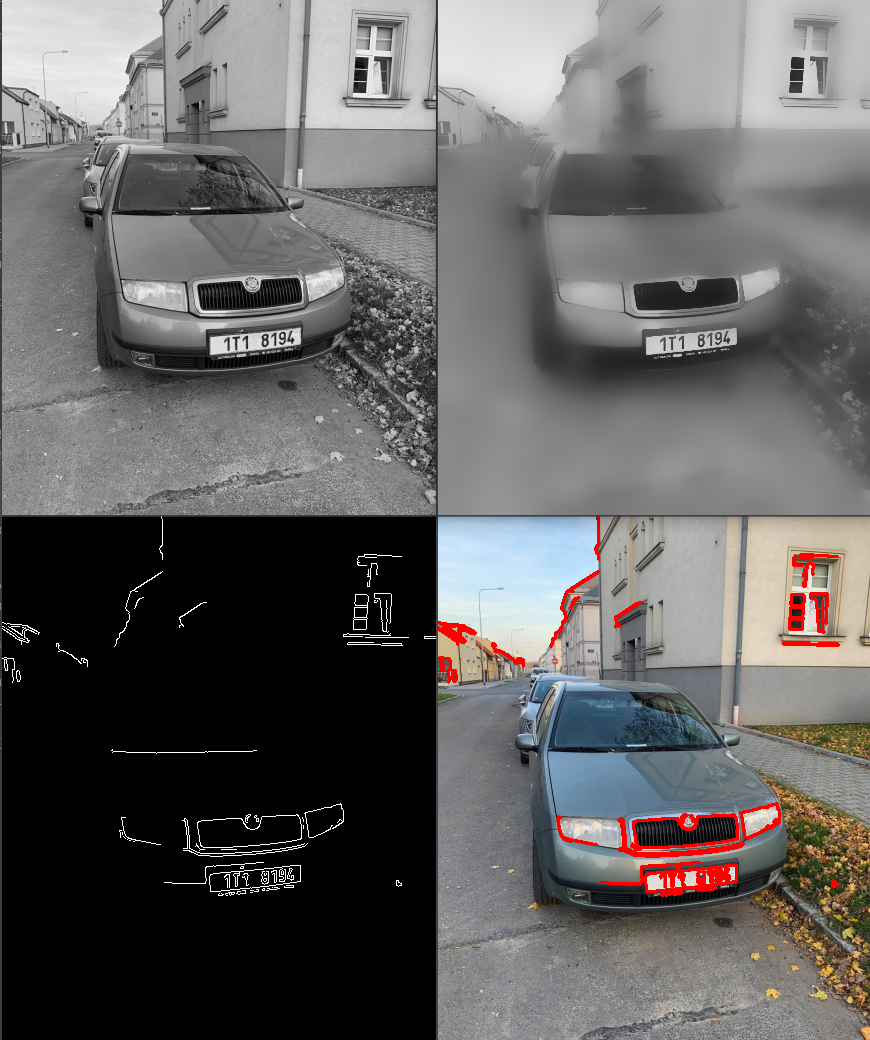
Použil jsem takovou a takovou kameru, která funguje jen za ideálních podmínek, pro horší podmínky by bylo potřeba koupit jinou kameru….

Bilaterállní filtr pro větší masky než 5 je pomalý a nehodí se pro real time aplikace, pro statickou ale má nejlepší výsledky.

Pro návrh řešení byly použity teoretické znalosti, popsané v první části této práce. Pro provedení některých metod byly použity funkce z knihovny *OpenCv* a pro některé výpočty funkce z knihovny *Numpy*. Pro úlohu nalezení Registrační značky vozidla bylo navrženo několik variant s použitím různých metod a jejich kombinací. Začátek je pro všechny metody stejný, je vybrán a načten vstupní 3D obraz, který je převeden do odstínu šedi, abychom dostali pouze 2D matici.

## Předzpracování obrazu

V první variantě byl použit dvakrát po sobě *bilaterální filtr*, pro který byly postupně odladěny parametry a výsledná velikost použité masky byla zvolena na 40pi. Takto velká maska může být výpočetně náročnější a je možné, že výpočetní náročnost bude příliš velká pro *realtime* zpracování snímků z kamery. Na filtrovaný obraz byl aplikován *Cannyho hranový detektor.* Výstupem je binární obraz, na který je použita funkce pro nalezení kontur. V rámci první varianty bylo testováno i použití prahování, které ale mělo špatné výsledky (Obr. 16).

Obrázek 16 – Vlevo použití bilaterálního filtru, vpravo použití prahování

Ve druhé variantě bylo použit bilineární filtr pouze jednou a jinak byl postup totožný. Úspěšnost první varianty na použité sadě 40 fotek byla okolo 55 %, úspěšnost druhé byla velmi podobná, kolem 57 %, avšak distribuce úspěšně rozpoznaných snímků byla rozdílná. Některé fotky tedy prošly při první variantě, a ne při druhé, a jiné zase naopak. Při třetí variantě byl použit Cannyho detektor přímo na šedo tónový obraz. Úspěšnost byla trochu menší než u obou předchozích variant, ale opět doplnila některé zatím neidentifikované případy.

Výsledná varianta tedy využívá kombinace všech tří předchozích. Všechny tři výsledné binární obrazy obsahující nalezené hrany jsou sečteny. Kontury jsou pak hledány jak na každém z těchto obrazů hran zvlášť, tak na tomto výsledném obraze a jsou poté rovněž sečteny.

## Nalezení RZ

Pro nalezení RZ byl vytvořen cyklus, který z nalezených kontur vybírá pouze takovou, která odpovídá daným kritériím. Konturou je proložena křivka, (the **Douglas–Peucker algorithm).** aproximovaná pro nejjednodušší tvar. Pokud je výsledným tvarem čtyřúhelník, je splněna podmínka. Danému čtyřúhelníku je poté opsán obdélník s minimálním obsahem a jsou uloženy souřadnice vrcholů tohoto obdélníku. Tyto souřadnice jsou použity pro výpočet šířky a délky jednotlivých stran obdélníku pomocí *Pythagorovy věty.* Následuje podmínka poměru, která byla po odladění nastavena na poměr stran v rozmezí od 3,5 do 5,5. Registrační značka používaná v ČR má definované rozměry, ne všechna vozidla ale mají novou značku opatřenou modrým pruhem a zkratkou dané země na levé straně a jejich bílá plocha je tedy horizontálně delší. Z tohoto byl nastaven zvolený rozsah možného poměru délek. Nakonec je porovnán obsah plochy výplně dané kontury a obsah opsaného obdélníku, kde je nastaven povolený rozdíl 5 %.



Obrázek 17 – Nalezená RZ

Navržené varianty byly testovány na sadě vzorových fotografií a byla porovnávána jejich efektivita. Některé značky však nebyly nalezeny při použití žádné z metod. Jedním z důvodů jsou světlé segmenty na reklamním pásku umístěném na držáku RZ, které splývají s plochou RZ (Obr. 18). Výsledná kombinační metoda dosáhla na vzorové sadě úspěšnosti **73,8 %** (Tab. 1).



Obrázek 18 – rušivý element na RZ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Metoda / Objektu na fotce | Metoda 1 | Metoda 2 | Metoda 3 | Kombinovaná metoda |
|  |
|  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 1 | 0 | 1 | 1 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 0 | 0 | 0 | 0 |  |
| 1 | 1 | 0 | 0 | 1 |  |
| 1 | 0 | 0 | 0 | 0 |  |
| 1 | 1 | 0 | 1 | 1 |  |
| 1 | 0 | 1 | 0 | 1 |  |
| 1 | 0 | 0 | 0 | 0 |  |
| 1 | 0 | 0 | 0 | 0 |  |
| 1 | 1 | 1 | 0 | 1 |  |
| 1 | 1 | 0 | 0 | 0 |  |
| 1 | 0 | 0 | 1 | 1 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 0 | 1 | 0 | 1 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 1 | 1 | 0 | 1 |  |
| 1 | 0 | 0 | 0 | 0 |  |
| 2 | 0 | 1 | 2 | 2 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 0 | 0 | 1 | 0 |  |
| 1 | 0 | 1 | 0 | 1 |  |
| 1 | 1 | 0 | 1 | 1 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 0 | 1 | 1 | 1 |  |
| 2 | 0 | 1 | 1 | 1 |  |
| 2 | 0 | 1 | 1 | 1 |  |
| 1 | 1 | 0 | 0 | 1 |  |
| 1 | 0 | 0 | 1 | 0 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 2 | 2 | 1 | 1 | 2 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 0 | 0 | 0 | 1 |  |
| 1 | 1 | 0 | 0 | 0 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| 1 | 1 | 1 | 1 | 1 |  |
| Správně | 23 | 22 | 24 | 31 |  |
| Celkem | 42 | 42 | 42 | 42 |  |
| Skóre (%) | **54,7619** | **52,381** | **57,1429** | **73,8095** |  |

Tabulka 1 - Úspěšnost použitých metod

# Výsledky

# Závěr

V první části semestrálního projektu byly popsány základní principy přenesení obrazové scény do digitální podoby a rovněž byly vysvětleny relevantní metody zpracování digitálního obrazu pro účely této práce. Ve druhé části bylo navrženo několik funkcí ve dvou SW platformách, a to v prostředí NI Vision Builderu a v prostředí Visual Studio Code s využitím jazyka *Python*. Pro větší paletu možností a dostupnost zdrojů byl pro další postup použit *Python.* V tomto jazyce poté byly navrženy funkce s využitím volně dostupné knihovny *OpenCv*, obsahující základní funkce pro *preprocessing* a práci s obrazem a knihovny *numpy,* pro matematické operace s daty ve formě vektorů a matic. V těchto funkcích byly pro konkrétní účel nalezení RZ vozidla testovány metody a kombinace metod uvedených v teoretické části této práce. Metody byly testovány na sadě fotografií, pořízených pro účely práce, byly postupně odladěny parametry a vyhodnocena efektivita použití pro danou aplikaci. Pro výslednou aplikaci, dosahující nejvyšší efektivity bylo nakonec použito kombinace několika metod. Výsledná kombinovaná metoda dosahuje úspěšnosti 73,8 % při testování na 40 fotografiích. Pro výslednou aplikaci bude platit předpoklad, že vozidlo bude přijíždět do bezprostřední blízkosti a RZ bude možné hledat v malém regionu očekáváného výskytu. Lze tedy předpokládat vyšší úspěšnost.

V pokračování diplomové práce budou navrženy bloky pro rozpoznání a čtení znaků na RZ vozidla a pro následné ověření povolení vozidla pro vjezd. Funkčnost aplikace bude rozšířena pro videozáznam a zpracování dat v reálním čase. Bude testována výpočetní rychlost navrženého algoritmu jeho možnost použití v reálném čase. Kvůli použití velkých masek bilaterálního filtru je možné, že výpočetní čas nebude aplikaci stačit a algoritmus bude třeba upravit. Bude navržena SW a HW platforma pro demonstrační model, který bude prezentovat funkčnost celého systému na reálných datech.

Motivace – chtěli jsme udělat to a to, povedlo se nepovedlo se.

# Použitá literatura

FIŘT, Jaroslav a Radek HOLOTA. *Digitalizace a zpracování obrazu* [online]. 5 [cit. 2020-10-10]. Dostupné z: http://home.zcu.cz/~holota5/publ/DigZprO.pdf.

Szeliski R.: Computer Vision: Algorithms and Application, Springer, Berlin, 2020.

GONZALEZ, Rafael C. a Richard E. WOODS. *Digital image processing.* Fourth edition. New York: Pearson, [2018]. ISBN 978-1-292-22304-9.

HLAVÁČ, Václav. Hledání hran. *Http://people.ciirc.cvut.cz* [online]. [cit. 2020-01-08]. Dostupné z: http://people.ciirc.cvut.cz/~hlavac/TeachPresCz/11DigZprObr/22EdgeDetectionCz.pdf

ŠEBELA, Miroslav. *Detekce objektu ve videosekvencích*. Brno, 2010. Diplmová práce. Vysoké učení technické v Brně.

KOLOUCHOVÁ, Michaela. *Morfologické operace ve zpracování obrazu*. Brno, 2008. Diplomová práce. Vysoké učení technické v Brně.

BOVIK, Alan C., ed. *Handbook of image and video processing* [online]. Second edition. [Amsterdam]: Elsevier, 2007 [cit. 2020-12-03]. ISBN 978-0-12-119792-6.

FESSL, Lukáš. *Knihovna pro automatickou detekci a měření objektů v obrazu z mikroskopu*. České Budějovice, 2012. Bakalářská práce. Jihočeská univerzita v Českých Budějovicích.

HORNBERG, Alexander, ed. Handbook of machine and computer vision: the guide for developers and users [online]. Second, revised and updated edition. Weinheim, Germany: Wiley-VCH, [2017] [cit. 2020-12-03]. ISBN 9783527413409.

STRAKA, Stanislav. *Segmentace obrazu*. Brno, 2009. Diplomová práce. Masarykova univerzita.

CANNY, John. A Computational Approach to Edge Detection. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*. **1986**(6), 40.

NEJEZCHLEB, Ivan. *Tone-Mapping pro HDR obrazy*. Brno, 2009. Diplomová práce. Vysoké učení technické v Brně.

*Https://www.researchgate.net/figure/Schematic-diagram-of-the-joint-bilateral-filter\_fig4\_322893461* [online]. [cit. 2021-01-13].

# Přílohy