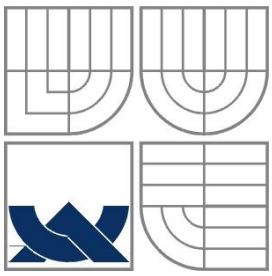
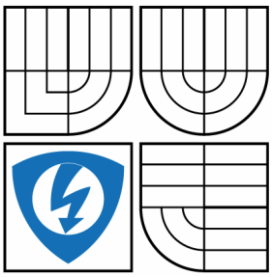


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

DETEKCE OBJEKTŮ

OBJECT DETECTION

BAKALÁŘSKÁ PRÁCE

AUTOR PRÁCE — PETR STRAŠEK

AUTHOR

VEDOUCÍ PRÁCE — Ing. MILOSLAV RICHTER, Ph.D.

SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ
Fakulta elektrotechniky
a komunikačních technologií
Ústav automatizace a měřicí techniky

Bakalářská práce

bakalářský studijní obor
Automatizační a měřicí technika

Student: Petr Strašek ***ID:*** 164406
Ročník: 3 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Detekce objektů

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte prostředí pro porovnávání kontrastních objektů ze snímků/videí a stanovení jejich odlišnosti. Úkolem je na kontrastním obraze určit pohybující se objekt a porovnat ho s definovanými mezními stavy. Dalším úkolem je zhodnocení kvality symetrie objektu vůči zadané ose.
1) Vytvořte databázi snímků a videí pro testování algoritmů.
2) Navrhněte algoritmus pro vyhledání objektů, stanovení mezních stavů pohybujících se objektů. Navrhněte postup pro zjištění odchylky aktuálně snímaných objektů od mezních stavů a pro stanovení symetrie. Navrhněte vhodný způsob vizualizace odchylek podle zadaných parametrů nebo vzorových stavů.
3) Otestujte navržené algoritmy a zhodnoťte jejich kvalitu a přesnost.

DOPORUČENÁ LITERATURA:

Kraus K.: Photogrammetrie 1 und 2, Ummler / Bonn, 1996
Žára J., Beneš B., Sochor J., Felkel P.: Moderní počítačová grafika, Computer Press, 1998, ISBN 80-251-0454-0
Hlaváč V., Šonka M.: Počítačové vidění,Grada, Praha 1992, ISBN 80-85424-67-3
Faugeras O.: Three-Dimensional Computer Vision, The MIT Press 1993

Termín zadání: 8. 2. 2016 ***Termín odevzdání:*** 1. 8. 2016

Vedoucí práce: Ing. Miloslav Richter, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

ABSTRAKT

Tato bakalářská práce řeší problematiku detekce kontrastních objektů, hodnocení jejich symetrie a mezních stavů. Za tímto účelem byla vytvořena databáze snímků, na kterých byly algoritmy testovány. Práce dále popisuje konkrétní implementaci těchto algoritmů a jejich výsledky.

KLÍČOVÁ SLOVA

Symetrie, mezní stav, prahování, projektivní transformace

ABSTRACT

This thesis solves the problem of detecting contrast subjects, assessing their symmetry and limit states. For this purpose, a database of photographs had been created, that were used for testing algorithms. The thesis also describes the specific implementation of these algorithms and their results.

KEYWORDS

Symmetry, limit states, thresholding, projective transformation

Bibliografická citace mé práce:

STRAŠEK, P. *Detekce objektů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2016. 52 s. Vedoucí bakalářské práce Ing. Miloslav Richter, Ph.D..

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma Detekce objektů jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ....

(podpis autora)

PODĚKOVÁNÍ

Děkuji svému vedoucímu práce Ing. Miloslavovi Richterovi, Ph. D. za trpělivost, konzultace a udělení klíčových rad.

V Brně dne ....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Počítačové vidění obecně, je perspektivní a rychle se rozvíjející odvětví zpracování obrazových dat, řízení strojů a podobných. Tato práce se zabývá detekcí kontrastních objektů, určení symetrie a porovnávání s mezními stavy. Tato oblast je vhodná pro jakostní zkoumání kvality výrobků, kde může nahradit lidského pracovníka a zvýšit rychlost zpracování s menší náchylností na chyby.

Dalším uplatněním může být kontrola, zda-li se nějaká pohyblivá součást stroje pohybuje v předepsaných mezích nebo jestli došlo k poruše či havárii.

V následující kapitole jsou popsané teoretické základy, popisující principy užité při tvorbě algoritmů plnících zadání. V kapitole 2 je popsána implementace programu, jeho principy a ideje a ve třetí kapitole se nachází informace o pořízení snímků a videí spolu se zhodnocením. V závěru je shrnutí výsledků.

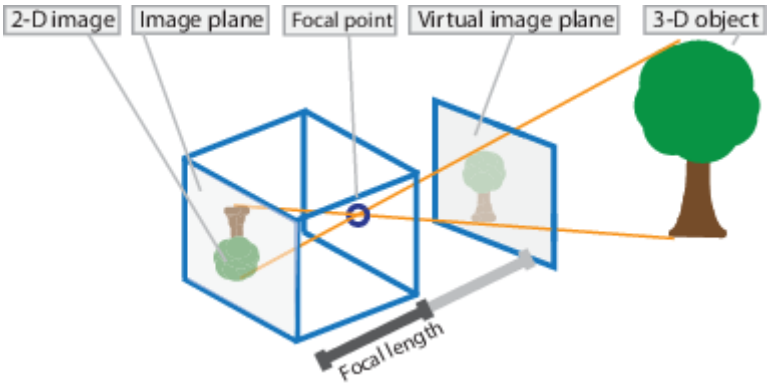
1 TEORETICKÝ ROZBOR

Pro řešení dané problematiky je zapotřebí snímat daný předmět kamerou a data zpracovat. V následujících podkapitolách jsou popsány principy částí snímacího a vyhodnocovacího řetězce a odstraňování nežádoucích jevů.

1.1 Kamera

Prostředkem ke snímání obrazu scény s měřeným objektem je kamera. Kamera se skládá z optické soustavy a snímače. Optická soustava (objektiv, zrcadla, clona) slouží k nasměrování světelných paprsků na snímač, který převádí (v případě digitální kamery) v daných bodech intenzitu osvětlení na elektrický signál. Snímač je nejčastěji typu CCD nebo CMOS. Oba tyto typy snímačů jsou citlivé pouze na jas, ale samy o sobě nepodávají informaci o barvě. Pro získání informace o barvě se pomocí polopropustných hranolů a barevných filtrů rozdělí snímaný obraz na tři barevné složky (červená, zelená a modrá) a každá z nich dopadá na jeden ze tří snímačů. Druhým řešením je nanesení optických filtrů na jednotlivé pixely jednoho snímače a výsledný obraz vypočíst. Výstupem ze snímače a jeho elektroniky jsou tři dvojrozměrné signály reprezentující obraz tří barevných složek.

Model kamery je matematický popis mapování 3D prostoru do 2D obrazu. Základním modelem kamery je dírková kamera a dírková kamera představuje čistou perspektivní projekci (ale převrácenou) viz Obr. 1.1.



Obr. 1.1: Principiální obrázek dírkové kamery [3]

Dírková kamera má následující parametry, které tvoří tzv. vnitřní parametry ideální kamery:

- Souřadnice hlavního bodu (optický střed)
- Ohnisková vzdálenost
- Zkosení
- Velikost pixelu v jednotkách scény (souvisí s velikostí snímače)

Tyto parametry tvoří matici kamery **K**, která vypadá takto:

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1.1}$$

kde c_x a c_y jsou souřadnice hlavního bodu, f_x a f_y je ohnisková vzdálenost v pixelech pro obě osy a „s“ je koeficient zkosení (anglicky skew), který je nulový, pokud jsou body snímače v pravoúhlé mřížce. Ohnisková vzdálenost v pixelech se spočte jako:

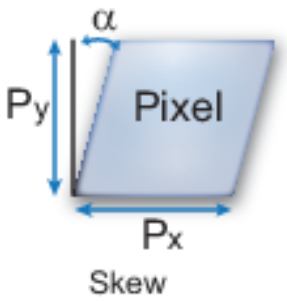
$$f_x = \frac{F}{p_x} \tag{1.2}$$

$$f_y = \frac{F}{p_y} \tag{1.3}$$

kde F je ohnisková vzdálenost v jednotkách scény (typicky v mm) a p_x a p_y je velikost pixelu v jednotkách scény. Koeficient zkosení se spočte jako:

$$s = f_y \cdot \tan \alpha \tag{1.4}$$

kde α je úhel zkosení jak ukazuje Obr. 1.2.



Obr. 1.2: Zkosení snímače/pixelu o úhel α

Vnější parametry kamery tvoří další část modelu kamery, které popisují translaci/polohu a rotaci/natočení kamery. Translace je vektor o třech prvcích, který udává počátek souřadnic scény v kamerových souřadnicích a značí se **t** a vypadá následovně:

$$\mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{1.5}$$

Rotace je matice 3x3 popisující natočení kamery ve všech třech osách, značí se **R** a má tvar:

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{1.6}$$

Vnější parametry kamery se zapisují dohromady v jedné matici a to následujícím způsobem:

$$[\mathbf{R}|\mathbf{t}] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \tag{1.7}$$

Zkombinováním matice vnitřních a vnějších parametrů vznikne projekční matice **P**, pomocí které se přemapovávají body scény do obrazu, který snímá kamera. Projekční matice se vypočte jako:

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} = \mathbf{K} \cdot [\mathbf{R}|\mathbf{t}] \tag{1.8}$$

$$\mathbf{P} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \tag{1.9}$$

Následné přemapovávání bodů pomocí projekční matice **P** probíhá podle následující rovnice:

$$w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{P} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{1.10}$$

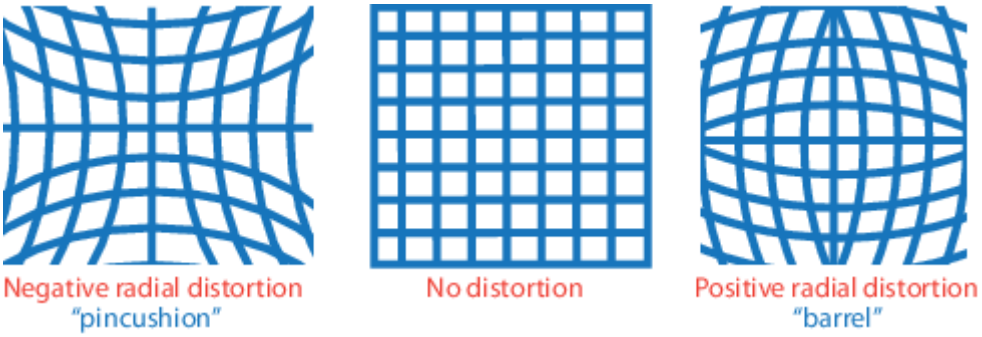
kde „w“ je měřítko, *x* a *y* jsou body obrazu a *X, Y* a *Z* jsou body reálné scény. [3]

I sebedokonalejší kamera či fotoaparát vždy do určité míry výsledný obraz zkresluje. Zkreslení je několik druhů. Mezi nejvýznamnější patří tyto:

- Radiální distorze (soudkovité a poduškovité zkreslení)
- Chromatická aberace
- Šum snímače
- Tangenciální zkreslení
- Zkosení snímače

„Radiální distorze je zkreslení objektivu způsobené souhrnem geometrických nepřesností při jeho výrobě. Úhel vystupujícího paprsku není naprosto stejný, jako u vstupujícího paprsku a poloha zobrazeného bodu se mírně liší od správné polohy. Velikost odchylky se mění s radiální vzdáleností od středu objektivu. Posuny bodů o radiální vzdálenosti r′ na snímku o hodnotu Δr′ označujeme jako radiální distorzi (zkreslení).“[1]

Jak již bylo zmíněno v dělení výše, radiální distorze má dvě varianty. Jsou jimi soudkovité (anglicky barrel/positive) a poduškovité zkreslení (anglicky pincushion/negative). Následující obrázek ilustruje tento typ zkreslení.



Obr. 1.3: Principiální znázornění radiální distorze (vlevo poduškovité, uprostřed bez zkreslení, vpravo soudkovité). [3]

Rovnice souřadnic bodů pro výpočet zkreslených bodů při znalosti koeficientů zkreslení a původních bodů [3]:

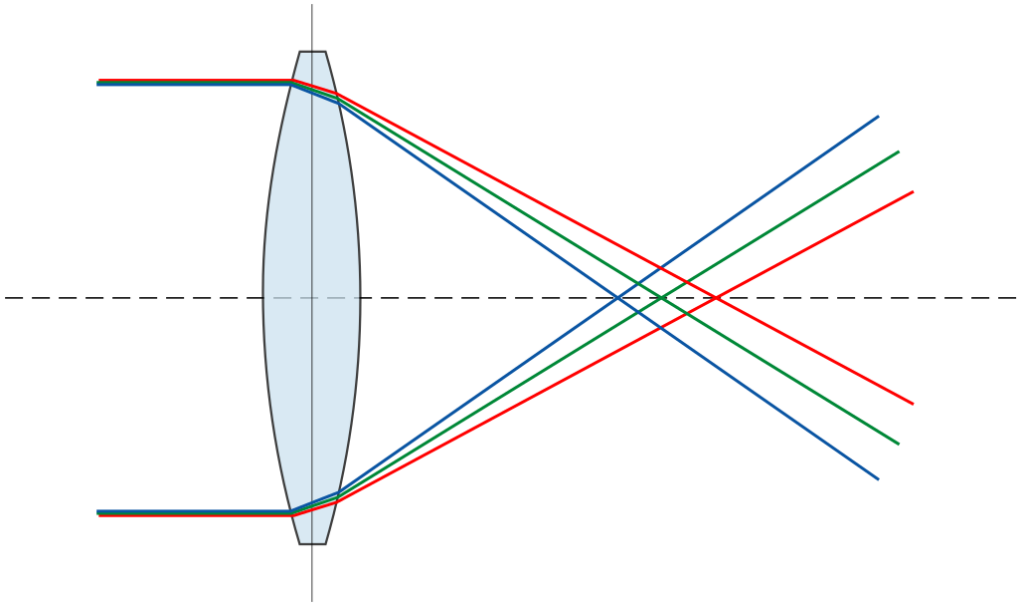
$$x_{zkr} = x\,(1 + k_1 \cdot r^2 + k_2 \cdot r^4 + k_3 \cdot r^6) \qquad (1.11)$$

$$y_{zkr} = y\,(1 + k_1 \cdot r^2 + k_2 \cdot r^4 + k_3 \cdot r^6) \qquad (1.12)$$

kde x_{zkr} a y_{zkr} jsou souřadnice zkresleného bodu, x a y jsou souřadnice bodu po odstranění zkreslení, k_1 a k_2 a k_3 jsou koeficienty tangenciálního zkreslení a r^2 je substituce za $(x^2 + y^2)$. Tyto vzorce používá kalibrátor kamery v Matlabu pro zjištění a potlačení radiální distorze. [3]

Chromatická aberace (též chromatická vada nebo barevná vada) je barevná vada čočky, i složitější optické soustavy čoček (např. objektivu), způsobená závislostí ohniskové vzdálenosti čoček na vlnové délce světla. Fyzikální podstatou tohoto jevu je závislost indexu lomu u všech průhledných látek na vlnové délce. Čočky pak lámou světlo každé barvy jinak (záření dlouhovlnné, tedy červené, nejméně, krátkovlnné, tedy fialové, nejvíce), což se na snímku projeví jako barevné lemování ostrých přechodů mezi světlem a stínem. [2]

„Všeobecně se dá říci, že vhodnou kombinací dvou k sobě nepřiléhajících čoček a to i ze stejného optického materiálu lze sestavit takovou optickou soustavu, u které je pro dvě zvolené vlnové délky (obvykle pro červenou a modrou barvu) celková efektivní ohnisková vzdálenost stejná. V takovém případě hovoříme o achromatické soustavě (objektivu).“ [2]



Obr. 1.4: Chromatická aberace [2]

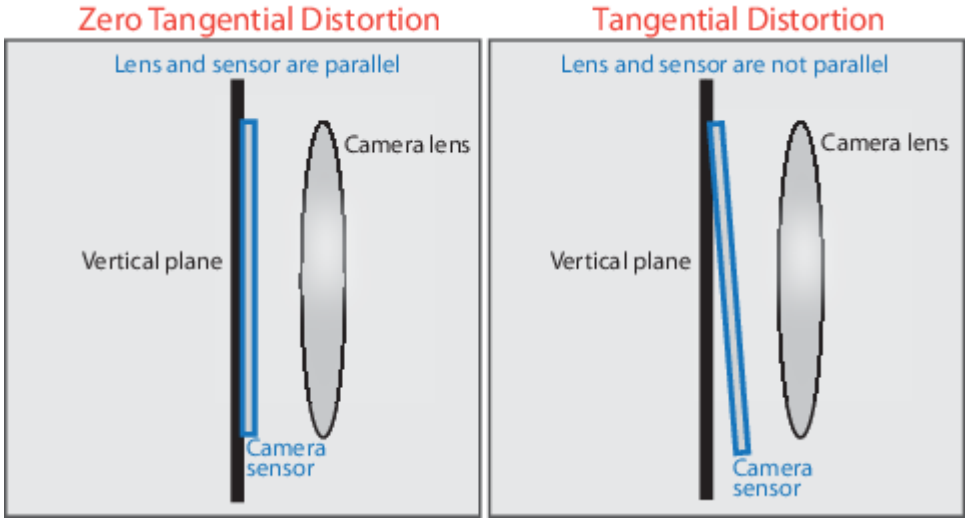
Protože jsou snímače v digitálních kamerách a fotoaparátech polovodičové a nepoužívají se při teplotě 0K, tak se v krystalické mřížce polovodiče snímače (vlivem tepelného kmitání) uvolní nosiče náboje i když nedojde k osvitu snímače. Tento parazitní jev se nazývá šum snímače a náhodně zvyšuje hodnotu signálu, který je z buněk snímače získán. Nejvíce se projevuje při focení za nízkých světelných podmínek, protože velikost šumu vznikající ve snímači se mění málo, ale při vyšším zesílení signálu snímače se zesílí i onen šum.

Tangenciální zkreslení kamery je způsobeno špatnou polohou snímače vůči optice kamery (objektivu). Projevuje se zanesením chyby perspektivy pohledu. Rovnice souřadnic bodů tangenciálního zkreslení jsou podle [3]:

$$x_{zkreslené} = x + [2 \cdot p_1 \cdot x \cdot y + p_2 \cdot (r^2 + 2 \cdot x^2)] \qquad (1.13)$$

$$y_{zkreslené} = y + [p_1 \cdot (r^2 + 2 \cdot y^2) + 2 \cdot p_2 \cdot x \cdot y] \qquad (1.14)$$

kde $x_{zkreslené}$ a $y_{zkreslené}$ jsou souřadnice zkresleného bodu, x a y jsou souřadnice bodu po odstranění zkreslení, p_1 a p_2 jsou koeficienty tangenciálního zkreslení a r^2 je substituce za $(x^2 + y^2)$. Toto je jedna z vad, které umí vykompenzovat kalibrátor kamery v Matlabu. Následující obrázek zobrazuje polohu snímače vůči optice kamery, která netvoří a tvoří tangenciální zkreslení obrazu. [3]



Obr. 1.5: Poloha snímače vůči optice kamery nepůsobící (vlevo) a působící tangenciální zkreslení. [3]

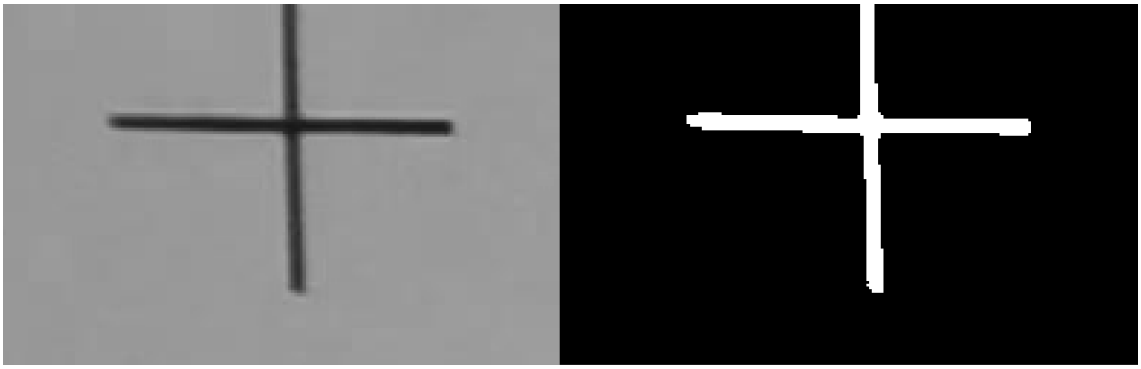
1.2 Binární obraz

V minulé kapitole bylo zmíněno, že výsledný obraz z kamery či fotoaparátu je v podobě tří dvojrozměrných matic, kde každá z nich reprezentuje jednu barevnou složku. V dnešní době se nejčastěji používá bitová hloubka 24 bitů. To znamená, že každý pixel každé barvy (a následně element každé matice) je tvořen osmibitovým číslem (0 až 255). Ovšem pro získání a uchování informace, zda a kde se vyskytuje pozorovaný objekt, je vhodné mít pouze jednu obrazovou matici, kde každý element bude nabývat hodnot nula (značí, že se zde objekt nevyskytuje) nebo jedna (objekt se zde vyskytuje). Takový obraz se nazývá binární. [4]

Binární obraz se vytváří metodou zvanou prahování. Spočívá v tom, že určitému intervalu hodnot ve vstupním obraze se přiřadí nula a zbytku jednička podle vzorce (1.15):

$$x_{\text{výst}} = \begin{cases} 0 \text{ pro } x_{\text{vstup}} < p \\ 1 \text{ pro } x_{\text{vstup}} \geq \text{p} \end{cases} \tag{1.15}$$

kde $x_{\text{výst}}$ je hodnota výstupního bodu, x_{vstup} je hodnota vstupního bodu a p je práh. Jako vstupní obraz může posloužit kterákoliv barevná složka barevného obrázku, ale například i rozdíly jednotlivých barevných složek. Dále je možné například použít různou hodnotu prahu pro každou barvu a tyto výsledné binární obrazy sloučit do jednoho použitím vhodné logické operace. Například pro vybrání pouze společných částí poslouží logický součin a pro přidání objektů a jejich částí ze všech je vhodný logický součet. [4] Praktický příklad viz Obr. 1.6.



Obr. 1.6: Ukázka prahování obrazu. (Vlevo červená složka obrázku a vpravo vyprahovaný obraz)

1.3 Vyplnění děr v binárním obraze

Po vytvoření binárního obrazu se může stát, že objekt (ať už vinou vady na objektu samém, při snímání nebo prahování) obsahuje tzv. „díru“ tzn., že uvnitř objektu tvořeného jedničkami jsou oblasti, kde jsou hodnoty nulové. Tato situace může mít neblahý vliv například na vyšetřování symetrie zkoumaného objektu, protože díra na jedné straně osy symetrie bude vyhodnocena, jako chybějící část oproti druhé straně a míra nesymetrie neoprávněně vzroste.

Nabízí se dvě základní možnosti řešení daného problému. Matematická morfologie a morfologická rekonstrukce.

1.3.1 Matematická morfologie

Matematická morfologie je technika zpracování geometrických struktur. K úpravě obrazu používá několik operací a to[5]:

- dilatace
- eroze
- otevření
- uzavření
- tref či miň
- ztenčování
- zesilování.

Z těchto operací je pro řešení výše zmíněného problému důležité uzavření, které přímo vychází z dilatace a eroze.

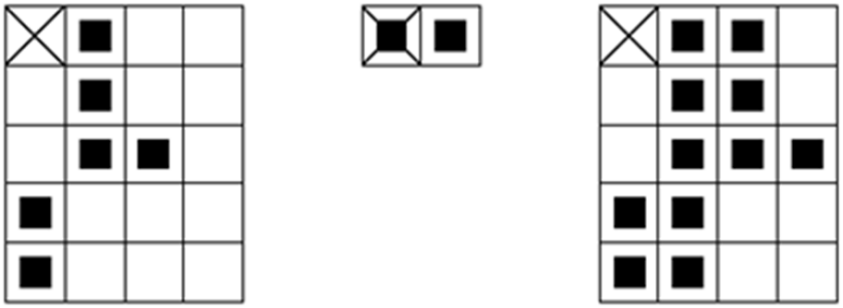
Každá z těchto operací/metod vyžaduje kromě vstupního binárního obrazu **B** tzv. strukturní element **S**, což je pomocný binární a menší obraz, který určuje výsledný binární obraz. Tento strukturální element je posouván nad obrazem B tak, aby jeho střed ležel na souřadnicích *x*, *y* (pak je značen $\mathbf{S}_{x,y}$) a podle vzorce dané operace byl vypočítán výsledek. [5]

Morfologická dilatace zvětšuje původní velikost objektů ve vstupním obraze. Díry a zálivy, které jsou menší než strukturní element, jsou dilatací odstraněny/zaplněny.

Vzorec pro výpočet dilatace je [4]:

$$\mathbf{D} = \mathbf{B} \oplus \mathbf{S} = \{(x, y) | \mathbf{S}_{x,y} \cap \mathbf{B} \neq \emptyset\} \quad (1.16)$$

Kde D je výsledek dilatace. Dilataci přibližuje následující obrázek:



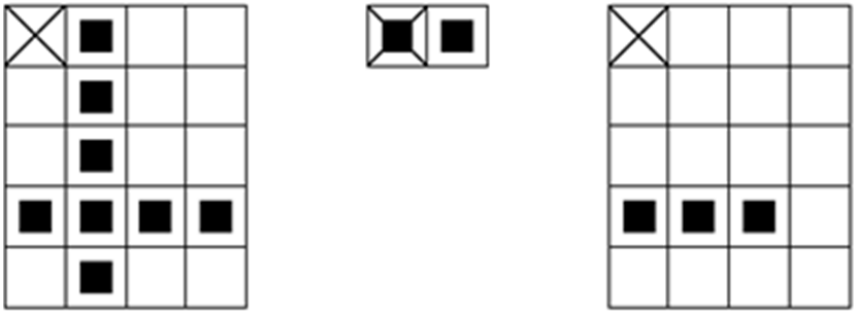
Obr. 1.7: Dilatace obrazu (vlevo) pomocí strukturního elementu (uprostřed) a výsledku (vpravo). Křížek v jednom z polí každého obrazu značí jeho střed. [5]

Principem dilatace je, že strukturní element je posouván nad vstupním obrazem a pokud je na daných souřadnicích *x*, *y* vstupního obrazu jednička, tak se do výstupního obrazu „obtiskne“ (pomocí logické operace or) strukturní element jeho středem na souřadnice *x*, *y*.

Morfologická eroze zmenšuje původní velikost objektů v obraze. Odstraňuje čáry tenčí než strukturní element a tím může oddělit tence spojené objekty. Vzorec pro výpočet dilatace je [4]:

$$\mathbf{E} = \mathbf{B} \otimes \mathbf{S} = \{(x, y) | \mathbf{S}_{x,y} \subseteq \mathbf{B}\} \quad (1.17)$$

kde **E** je výsledek eroze. Erozi přibližuje následující obrázek:



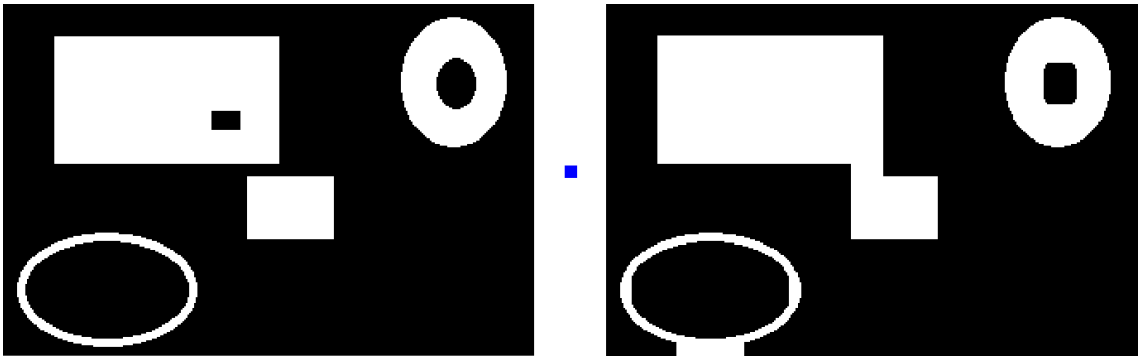
Obr. 1.8: Eroze obrazu (vlevo) pomocí strukturního elementu (uprostřed) a výsledku (vpravo). Křížek v jednom z polí každého obrazu značí jeho střed. [5]

Principem eroze je, že strukturní element je posouván nad vstupním obrazem tak, aby se na vyhodnocovaných souřadnicích *x*, *y* nacházel jeho střed. Pokud jsou ve vstupním obraze jedničky tam, kde je má posunutý strukturní element $\mathbf{S}_{x,y}$, tak na souřadnice *x*, *y* výstupního obrazu je zapsána jednička. [4]

Morfologické uzavření kombinuje morfologickou dilataci následovanou morfologickou erozí. Obě operace využijí jeden a tentýž strukturní element. Výsledkem bývá zaplnění děr (menších jak strukturní element) a spojuje blízké objekty. Vzorec pro morfologické uzavření je [4]:

$$\mathbf{I} = \mathbf{B} \bullet \mathbf{S} = (\mathbf{B} \oplus \mathbf{S}) \otimes \mathbf{S} \qquad (1.18)$$

Kde **I** je výstupní obraz. Ukázka morfologického uzavření s čtvercovým strukturním elementem je na následujícím obrázku:

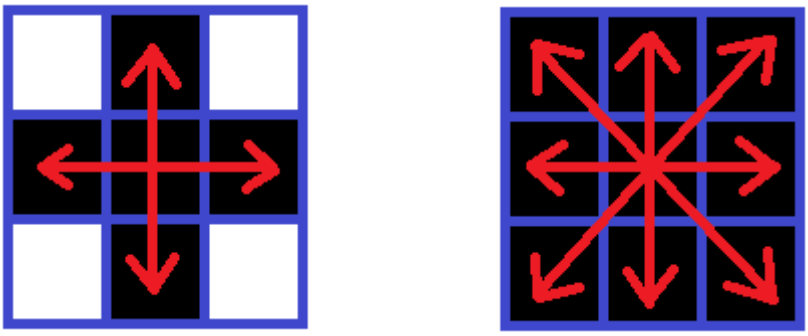


Obr. 1.9: Uzavření obrazu (vlevo) pomocí strukturního elementu (uprostřed, vybarven modře, v měřítku) a výsledku (vpravo)

Jak je z Obr. 1.9 patrné, tak se nezaplnily díry, které jsou větší než strukturní element. Došlo ke spojení čtverců, které byly blízko ale i ke spojení spodní elipsy s okrajem obrázku.

1.3.2 Flood-Fill

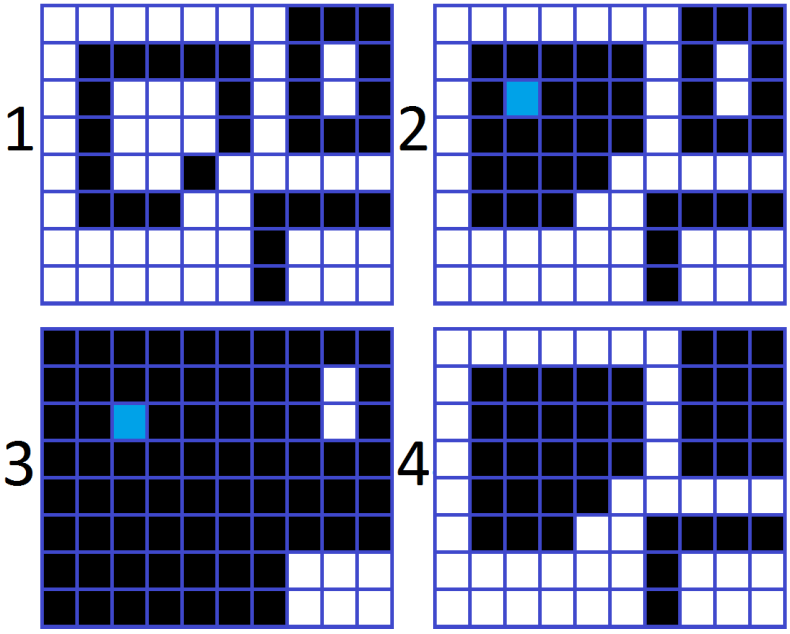
Dalším řešením zaplnění děr v binárním obraze je algoritmus Flood-Fill. Algoritmus postupně mění nuly za jedničky, až narazí na okraje objektu. Jinými slovy mění pixely pozadí (nuly) za pixely popředí (jedničky - pixely oněch prahováním nalezených objektů) dokud je pravda, že stále vyplňuje danou díru. Aby algoritmus posoudil, zda-li se ještě jedná o vyplňovanou díru nebo už o jinou, tak má předepsanou konektivitu. Konektivity pro 2D obraz jsou dvě a to 4okolí a 8okolí a říkají, které pixely od již zjištěných má algoritmus zkoumat.[3] Viz Obr. 1.10.



Obr. 1.10: 4 okolí (vlevo) a 8 okolí (vpravo)

Tento algoritmus je implementován v Matlabu a to ve funkci `imfill`. Pro danou problematiku lze relevantně spustit se zadaným bodem uvnitř prostoru, který má vyplnit a to ve variantách 4okolí a 8okolí posuzujících propojení vyplňované oblasti. Další možností je spustit jej v režimu, kde sám vyhledává díry a zaplňuje je. Aby

nezaplnil všechno, tak nezaplňuje oblasti, které se dotýkají okrajů zpracovávaného obrazu. V tomto režimu posuzuje propojení oblastí pomocí 4okolí. [4] Na následujícím obrázku jsou příklady vstupu a třech výstupů z funkce pro různá vstupní nastavení:

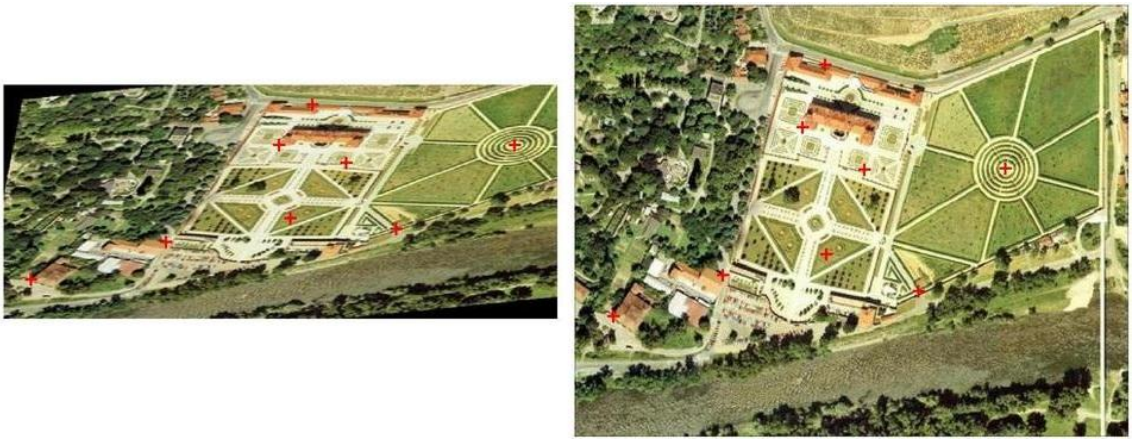


Obr. 1.11: Příklady práce algoritmu Flood-Fill (vstupní obraz (1), výplň se startovacím bodem (modrý bod) 4okolí (2) a 8okolí (3) a automatické vyplňování 4okolí (4))

Flood-Fill jsem zvolil jako výhodnější variantu, protože nemusím předem znát nebo zkoumat, jakou velikost mají díry objektu.

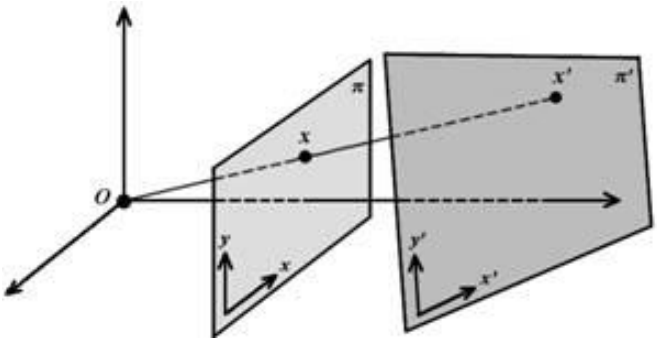
1.4 Korekce perspektivy při snímání

Při snímání objektu se může stát, že kamera nesnímá zkoumanou scénu se správnou perspektivou, což vede k tomu, že je do obrazu vneseno perspektivní zkreslení, které by vneslo chybu do výsledků následujícího zpracování. Toto zkreslení lze odstranit pomocí projektivní transformace. Tato transformace dokáže změnit perspektivu obrazu na jinou pomocí dvojic bodů, které si odpovídají v jedné i druhé perspektivě. Minimální množství těchto dvojic jsou čtyři. [6] Na následujícím obrázku je ukázka dvou perspektiv s vyznačením odpovídajících si bodů:



Obr. 1.12: Ukázka záběru jedné scény ve dvou perspektivách s vyznačenými body, které si odpovídají [7]

Náčrt bodu x jedné roviny mapovaného do druhé roviny jako x' je na Obr. 1.13.



Obr. 1.13: Ukázka jednoho bodu mapovaného do dvou rovin [6]

Výpočet obrazu jiné perspektivy se provádí mapováním bodů ze vstupního obrazu do výstupního pomocí transformační matice **H** o rozměru 3x3. Matice má tvar:

$$\mathbf{H} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \tag{1.19}$$

a rovnice má tvar:

$$\boldsymbol{x'}_{\mathrm{i}} = \mathbf{H}\boldsymbol{x}_{\mathrm{i}} \tag{1.20}$$

Kde $\boldsymbol{x}_{\mathrm{i}}$ a $\boldsymbol{x'}_{\mathrm{i}}$ jsou vektory vstupních a výstupních bodů. Znázorněny jsou v rovnici (1.21) a (1.22).

$$\boldsymbol{x}_{\mathrm{i}} = \begin{pmatrix} x_{\mathrm{i}} \\ y_{\mathrm{i}} \\ 1 \end{pmatrix} \tag{1.21}$$

$$\boldsymbol{x'}_{\mathrm{i}} = \begin{pmatrix} x'_{\mathrm{i}} \\ y'_{\mathrm{i}} \\ w'_{\mathrm{i}} \end{pmatrix} \tag{1.22}$$

Důvodem, proč v rovnici (1.22) není na posledním místě 1 ale w'_{i} je ten, že by se jinak vektor $\boldsymbol{x'}_{\mathrm{i}}$ číselně nerovnal výsledku $\mathbf{H}\boldsymbol{x}_{\mathrm{i}}$. Prvek w'_{i} nese informaci, jak normalizovat souřadnice x'_{i} a y'_{i}, aby odpovídaly správnému výsledku. [6][7]

Pro vytvoření výstupního obrazu se změněnou perspektivou je třeba vypočíst právě onu transformační matici **H**, pro kterou platí rovnice (1.20). Vyřešením maticové rovnice (1.23) dojde ke zjištění všech koeficientů h_{11} až h_{33}, které tvoří matici **H**. Aby bylo možné podle rovnice (1.23) nalézt vektor koeficientů **h**, je nutné aby matice **A** měla alespoň 8 lineárně nezávislých řádků, z čehož plyne, že je potřeba alespoň čtyři páry bodů, které si odpovídají v obou obrazech a kde žádné tři body neleží na stejné přímce. Aby bylo nalezeno jedno řešení vektoru koeficientů h, tak se často právě koeficientu h_{33} přiřazuje hodnota 1.

$$\mathbf{Ah} = \begin{pmatrix} \mathrm{x}_1 & \mathrm{y}_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 & -x'_1 \\ 0 & 0 & 0 & \mathrm{x}_1 & \mathrm{y}_1 & 1 & -x_1y'_1 & -y_1y'_1 & -y'_1 \\ \mathrm{x}_2 & \mathrm{y}_2 & 1 & 0 & 0 & 0 & -x_2x'_2 & -y_2x'_2 & -x'_2 \\ 0 & 0 & 0 & \mathrm{x}_2 & \mathrm{y}_2 & 1 & -x_2y'_2 & -y_2y'_2 & -y'_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathrm{x}_{\mathrm{n}} & \mathrm{y}_{\mathrm{n}} & 1 & 0 & 0 & 0 & -x_{\mathrm{n}}x'_{\mathrm{n}} & -y_{\mathrm{n}}x'_{\mathrm{n}} & -x'_{\mathrm{n}} \\ 0 & 0 & 0 & \mathrm{x}_{\mathrm{n}} & \mathrm{y}_{\mathrm{n}} & 1 & -x_{\mathrm{n}}y'_{\mathrm{n}} & -y_{\mathrm{n}}y'_{\mathrm{n}} & -y'_{\mathrm{n}} \end{pmatrix} \begin{pmatrix} \mathrm{h}_{11} \\ \mathrm{h}_{12} \\ \mathrm{h}_{13} \\ \mathrm{h}_{21} \\ \mathrm{h}_{22} \\ \mathrm{h}_{23} \\ \mathrm{h}_{31} \\ \mathrm{h}_{32} \\ \mathrm{h}_{33} \end{pmatrix} = 0 \tag{1.23}$$

Poté je možné koeficienty h_{11} až h_{33} a body vstupního obrazu postupně dosadit do rovnice (1.24) a poskládat výstupní obraz jiné perspektivy [6][7].

$$\boldsymbol{x'}_{\mathrm{i}} = \mathbf{H}\boldsymbol{x}_{\mathrm{i}} = \begin{pmatrix} \mathrm{h}_{11} & \mathrm{h}_{12} & \mathrm{h}_{13} \\ \mathrm{h}_{21} & \mathrm{h}_{22} & \mathrm{h}_{23} \\ \mathrm{h}_{31} & \mathrm{h}_{32} & \mathrm{h}_{33} \end{pmatrix} \begin{pmatrix} x_{\mathrm{i}} \\ y_{\mathrm{i}} \\ 1 \end{pmatrix} = \begin{pmatrix} x'_{\mathrm{i}} \\ y'_{\mathrm{i}} \\ w'_{\mathrm{i}} \end{pmatrix} \tag{1.24}$$

1.5 Detekce natočení objektu

Zjištění orientace objektu je důležité, protože se může ve scéně měnit a detekce

symetrie objektu je jednodušší na objektu orientovaném stále stejně (a to s natočením 0°), protože svislá osa symetrie vede kolmo obrazem objektu.

Úhel natočení, který vrací Matlab je získán jako úhel mezi vodorovnou osou dotýkající se tzv. hlavní osy objektu (v rozmezí ±90°) jak naznačuje následující obrázek, kde α je úhel natočení objektu:[3]

Obr. 1.14: Detekce natočení (Upraveno [3])

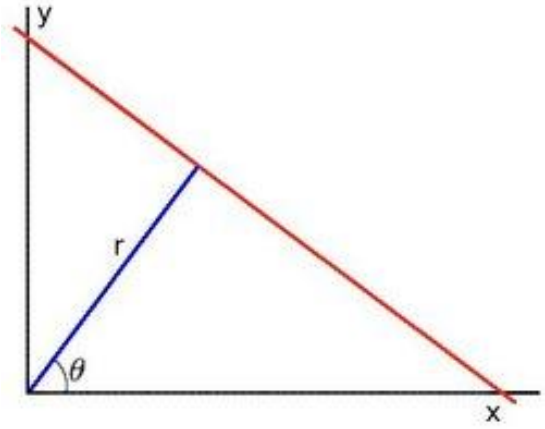
Toto v Matlabu defaultně naimplementované řešení však neposkytovalo uspokojivé výsledky a proto byla detekce natočení realizována za pomoci Houghovy transformace pro hledání přímek.

1.5.1 Houghova transformace

Pro Houghovu transformaci je využíván polární tvar rovnice přímky a to:

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right) \cdot x + \frac{r}{\sin\theta} \tag{1.25}$$

kde θ je úhel kolmice směřující od středu souřadného systému na přímku a kde r délka této kolmice, jak ukazuje následující obrázek:



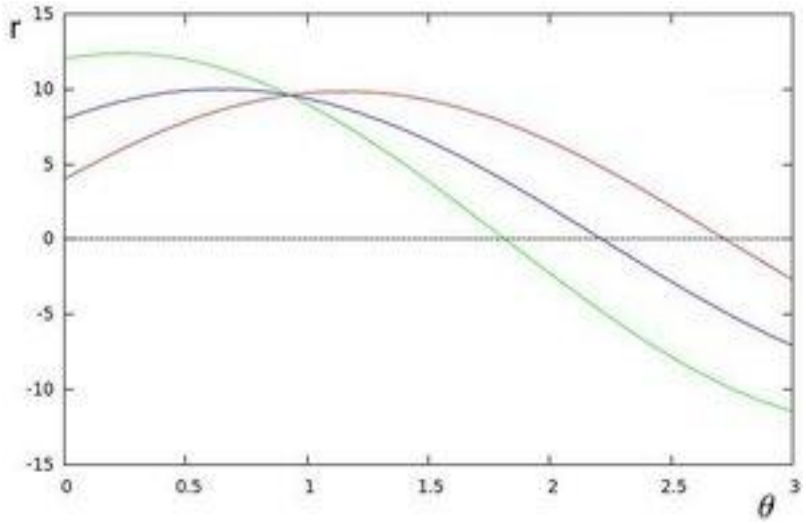
Obr. 1.15: Znázornění úhlu θ a délky kolmice r určující přímku [10]

Po úpravě vzorce (1.25) je získán následující tvar:

$$r = x \cdot \cos\theta + y \cdot \sin\theta \tag{1.26}$$

Tento vzorec umožňuje převod souřadnic bodu (x, y) do roviny „θ – r“, ve které měněním úhlu θ vznikne harmonická křivka, která popisuje všechny přímky, které mohou vést převáděným bodem. Tímto způsobem se převedou všechny body obrazu.

Pokud nějaký počet bodů obrazu leží na stejné přímce, tak se v „θ – r“ rovině jejich křivky protnou. Pro tři body ležící na jedné přímce může „θ – r“ rovina vypadat následovně:



Obr. 1.16: Průsečík křivek bodů protínající se v „θ – r“ rovině ukazující parametry θ a r nalezené přímky procházející všemi třemi body [10]

Průsečík v „θ – r“ je následně převeden na rovnici přímky pomocí vzorce (1.25). [10]

Pokud každý snímaný objekt bude obsahovat kontrastní obdélník, tak je možné v jeho obrysu nalézt přímku odpovídající jedné z jeho delších stran, jejíž úhel natočení odpovídá úhlu natočení celého objektu.

1.6 Afinní transformace

Geometrické transformace jsou vektorové funkce, které mají za úkol mapovat body vstupního obrazu do výstupního obrazu. Častokrát jsou obecné transformace aproximovány bilineární formou mající čtyři dvojice koeficientů (pro x i y osu). Viz rovnice (1.27) a (1.28).

$$\boldsymbol{x'} = \mathrm{a}_0 + \mathrm{a}_1 x + \mathrm{a}_2 y + \mathrm{a}_3 xy \tag{1.27}$$

$$\boldsymbol{y'} = \mathrm{b}_0 + \mathrm{b}_1 x + \mathrm{b}_2 y + \mathrm{b}_3 xy \tag{1.28}$$

Pokud se vypustí z bilineární transformace poslední členy $\mathrm{a}_3 xy$ a $\mathrm{b}_3 xy$, tak dostaneme jednodušší afinní transformaci. I tato transformace je schopna těchto operací: translace, rotace, měřítko a zkosení. Rovnice afinní transformace znějí[8]:

$$\boldsymbol{x'} = \mathrm{a}_0 + \mathrm{a}_1 x + \mathrm{a}_2 y \tag{1.29}$$

$$\boldsymbol{y'} = \mathrm{b}_0 + \mathrm{b}_1 x + \mathrm{b}_2 y \tag{1.30}$$

Pokud je potřeba provést nad obrazem kombinaci výše zmíněných operací, tak se jednotlivé operace zkombinují do jedné afinní transformace, aby se obraz transformovával pouze jednou a výpočet byl méně náročný. Pro moji práci je klíčové hlavně otočení. Využívá pouze koeficienty a_1, a_2, b_1 a b_2. Obraz je možné otočit o libovolný úhel α v protisměru hodinových ručiček. Základní rovnice otočení pak

vypadají následovně[8]:

$$\boldsymbol{x'} = x \cdot \cos\alpha - y \cdot \sin\alpha \tag{1.31}$$

$$\boldsymbol{y'} = x \cdot \sin\alpha + y \cdot \cos\alpha \tag{1.32}$$

Aby nevznikaly ve výstupním obraze díry, tak souřadnice výstupního obrazu jsou zadány a počítají se souřadnice vstupního obrazu. Protože souřadnice bodů v digitálním obrazu jsou celočíselné, tak se téměř vždy vyskytne stav, kdy vypočtené souřadnice bodu ve vstupním obrazu nebudou celočíselné a nebudou jasně odpovídat jeho jednomu konkrétnímu bodu. V tom případě je nutné aproximovat hodnotu zanesenou do výstupního obrazu interpolací okolních bodů nebo alespoň zaokrouhlit souřadnice na celá čísla. Otočení je ilustrováno na Obr. 1.17. [8]

Obr. 1.17: Ukázka otočení obrazu[8]

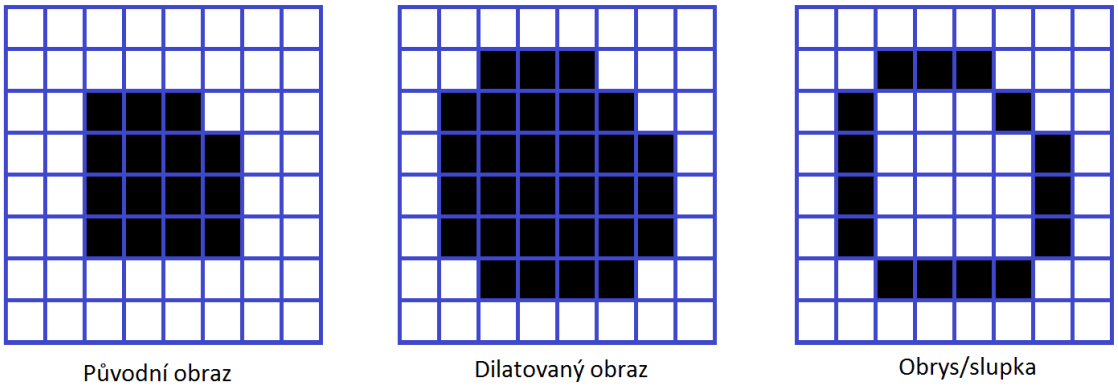
Původně jsem zamýšlel, že bych užil pro korekci perspektivy právě afinní transformaci vytvořenou pomocí čtyř párů souřadnic reálného světa a korespondujících bodů v obraze, ale projektivní lineární transformace dávala mnohem lepší výsledky.

1.7 Získání obrysu objektu

Pro vyznačení plochy do snímku, která vyznačuje mezní vzorový pohyb, je zapotřebí získat obrys/slupku plochy, ve kterém se objekt při vzorovém pohybu vyskytoval. K získání tohoto obrysu je možné užít morfologické operace nebo hranové detektory.

1.7.1 Získání slupky pomocí matematické morfologie

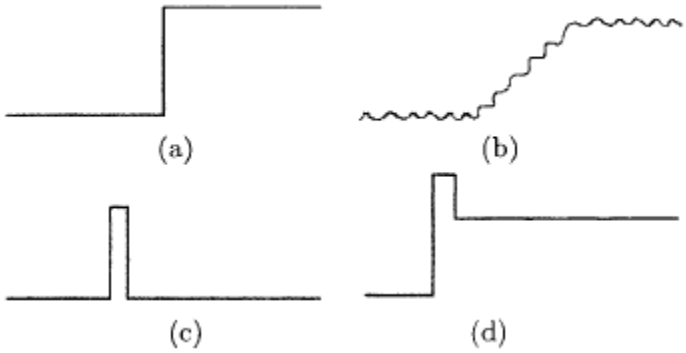
Principy matematické morfologie jsou vysvětleny v kapitole 1.3.1. S využitím těchto znalostí je možné dilatovat objekt (ve formě binárního obrazu) a od tohoto dilatovaného obrazu objektu odečíst původní obraz objektu. [5] Princip je ukázán na následujícím obrázku:



Obr. 1.18: Ilustrace výsledku a mezikroku tvoření slupky matematickou morfologií

1.7.2 Získání slupky za pomoci hranových operátorů

Hrana je linie v obraze, která odděluje oblasti s různou jasovou intenzitou. Přechod jedné jasové intenzity do druhé může mít u šedotónového obrazu různý průběh.



Obr. 1.19: (a) skoková změna, (b) rampa/pozvolná změna (zde zašuměná), (c) čára, (d) skoková čára [9]

V mém případě je potřeba obrys získávat z binárního obrazu, a proto se jedná o skokovou změnu (z nuly na jedničku).

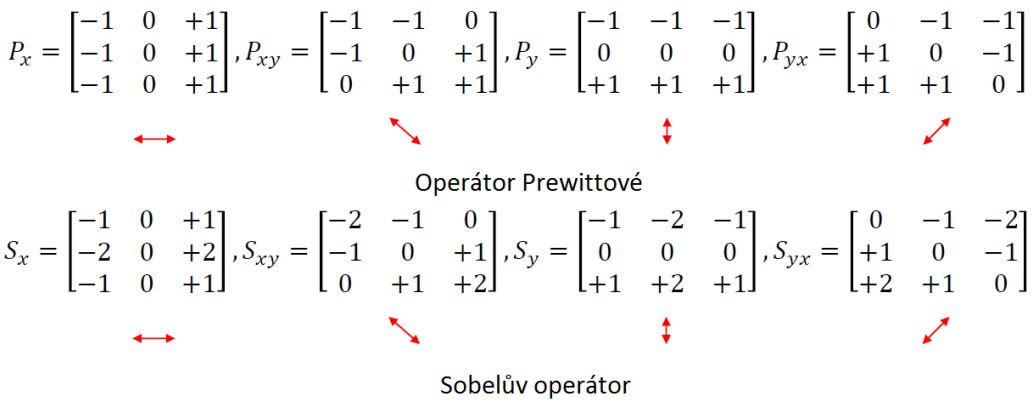
Tyto změny jsou hledány a měřeny gradientem obrazu. Ve dvourozměrném obrazu se jedná o vektor o dvou prvcích (pro složku ve směru osy *x* a *y*). U diskrétního obrazu se k hledání hran používají diference podle vzorců:

$$\Delta_x f(x,y) = f(x,y) - f(x-\mathrm{n},y) \tag{1.33}$$

$$\Delta_y f(x,y) = f(x,y) - f(x,y-\mathrm{n}) \tag{1.34}$$

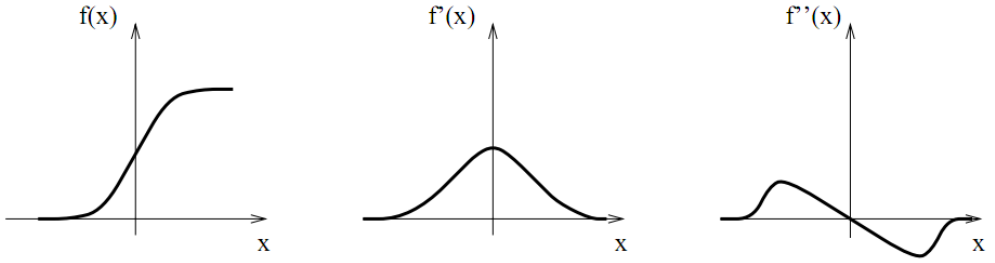
kde n bývá 1, ale může být i vyšší. V podstatě se vypočítávají rozdíly sousedních pixelů pro danou osu. Z těchto dvou složek lze vypočíst velikost i úhel gradientu. [8]

Dále je možné hledat hrany pomocí konvoluce a hranových operátorů. Tyto operátory aproximují derivace obrazové funkce. Příklady operátorů aproximující první derivaci následují na obrázku:



Obr. 1.20: Operátor Prewittové a Sobelův operátor ve variantách pro čtyři směry [8]

Lepší než operátory aproximující první derivaci jsou ty aproximující druhou. Výhoda použití druhé derivace místo první je, že ve výsledném obrazu nemusíme hledat lokální maxima, ale průchody nulou, což je výpočetně jednodušší a tudíž výhodnější. Jejich další výhodou je, že rozpoznávají hrany ve všech směrech a proto stačí pouze jeden a nemusí se kombinovat se svými otočenými verzemi. [8] Ukázka na jednorozměrné funkci je na Obr. 1.21.



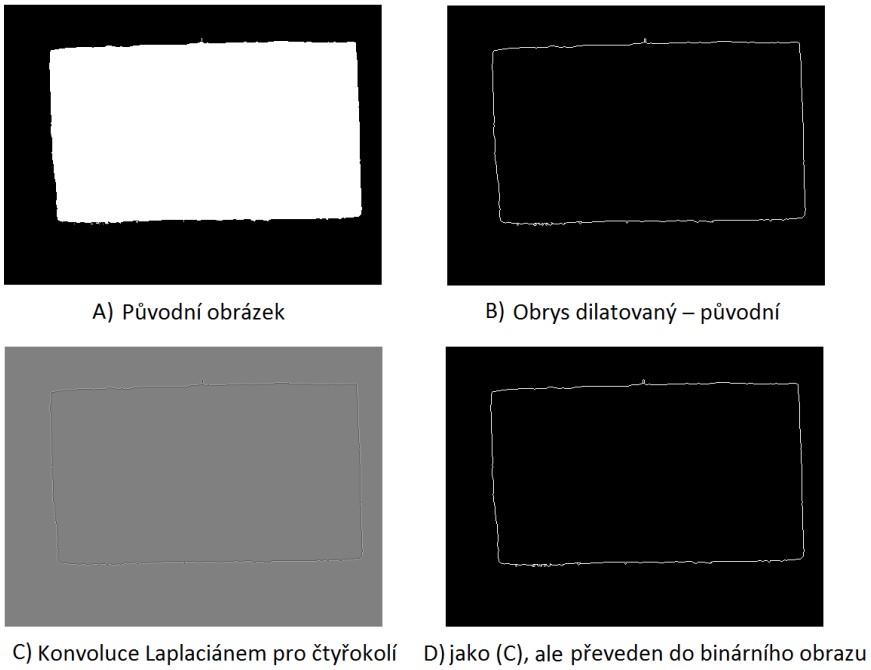
Obr. 1.21: Náběžná hrana, první derivace a druhá derivace [8]

Nejdůležitějším operátorem aproximujícím druhou derivaci je Laplaceův operátor ve verzi pro čtyř-okolí a osmi-okolí.[8]

$$L_4 = \begin{bmatrix} 0 & +1 & 0 \\ +1 & -4 & +1 \\ 0 & +1 & 0 \end{bmatrix}, \qquad L_8 = \begin{bmatrix} +1 & +1 & +1 \\ +1 & -8 & +1 \\ +1 & +1 & +1 \end{bmatrix}$$

Obr. 1.22: Laplaceův operátor ve verzi pro čtyř-okolí a osmi-okolí [8]

Když je výsledek konvoluce Laplaciánem pro čtyř-okolí převeden do binárního obrazu, tak se výsledek shoduje s výsledkem dosaženým morfologickou dilatací, od které je odečten původní binární obraz, jak ukazuje obrázek Obr. 1.23.



Obr. 1.23: Porovnání metod tvorby obrysu binárního objektu

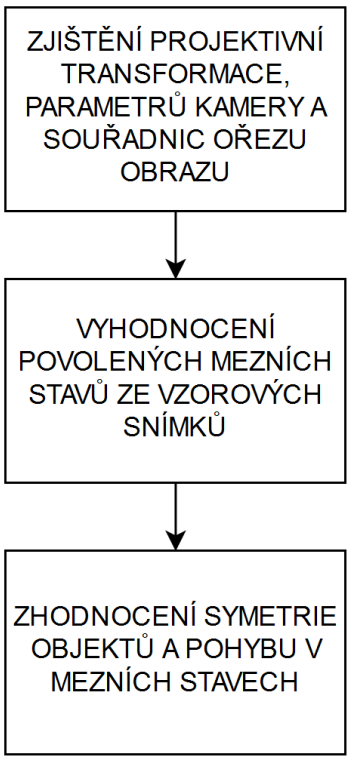
2 IMPLEMENTACE PROGRAMU

Tato kapitola pojednává o výsledném programu, jeho činnosti a principech, ale i požadavcích na vstupní data.

Jedním z požadavků, aby program pracoval, jak bylo zamýšleno, je nutnost, aby snímaná scéna byla dobře nasvícena kvůli dobrému kontrastu a také z důvodu eliminace stínů, protože stíny se mohou stát částmi objektů při prahování. Nízký kontrast může znemožnit funkci hledající práh určit správně práh nutný pro tvorbu binárního obrazu, která je popsána níže.

Scéna musí mít bílé pozadí se čtyřmi černými kalibračními značkami, které jsou v podobě křížků a jsou rozmístěny v rozích bílého pozadí. Požadavky na zkoumaný objekt jsou, že je černý, je největším objektem v celé snímané scéně a nepřekrývá žádnou z kalibračních značek. Dalším požadavkem je, aby se neměnila pozice kamery ani scény.

Základní idea činnosti programu je, že z jednoho snímku vyhodnotí, jakou perspektivní transformaci je nutné provést nad obrázky a která část snímků se bude zachovávat v dalším zpracování. Z dalších několika prvních snímků určí, kde se smí zkoumaný objekt vyskytovat a nakonec s předešle získanými znalostmi ohodnotí symetrii snímaného objektu a jeho setrvávání ve vymezených mezích. Další obrázek ukazuje základní diagram této myšlenky:



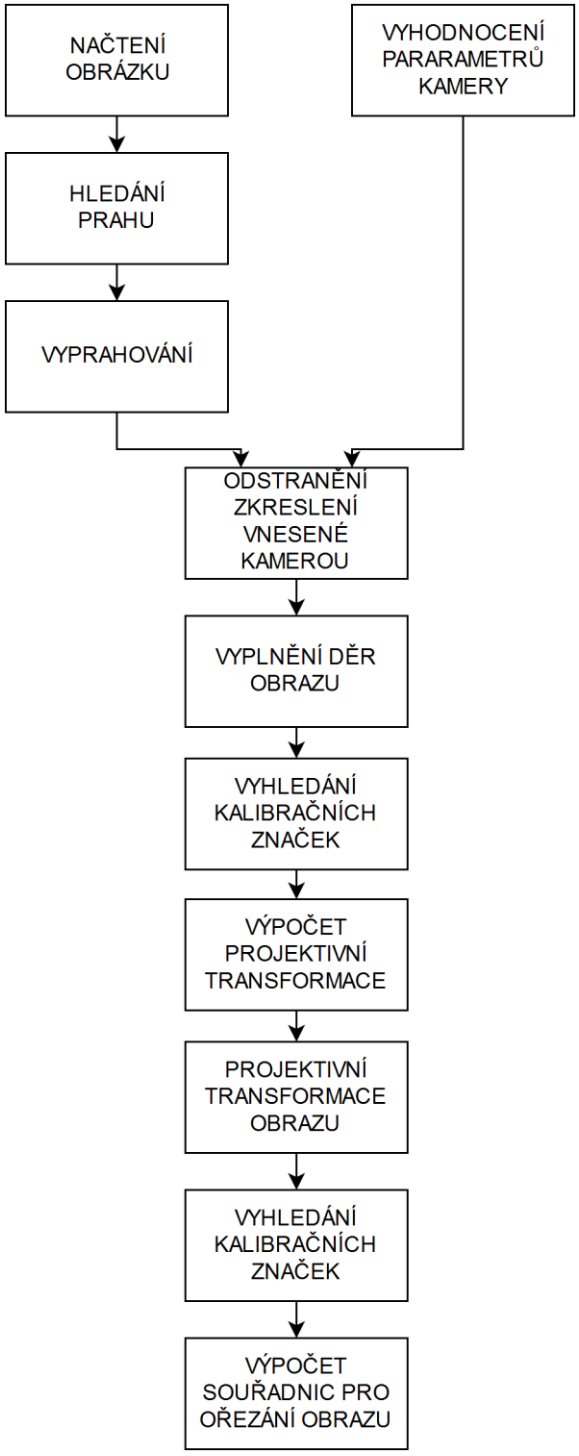
Obr. 2.1: Diagram základní myšlenky programu

2.1 Zjištění projektivní transformace a souřadnic ořezu obrazu

Tato část programu realizovaná funkcí se nazývá `urciTfotmAOrez`. Je jí předán první snímek a za pomocí objektu s parametry kamery a souřadnic kalibračních bodů v reálné scéně vyhodnocuje potřebnou projektivní transformaci a souřadnice ořezu vymezující plochu mezi kalibračními křížky.

Díky zavedení požadavku na nehybnost scény a kamery je postačující, aby byla určena projektivní transformace a prostor vymezený kalibračními značkami pouze jednou a pak byly aplikovány na ostatní snímky. Výhodou je, že pokud by došlo na nějakém snímku k chybnému nalezení kalibračních značek, tak mohlo dojít i k pádu programu. Pokud na prvním snímku nejsou správně detekovány kalibrační značky, je funkci předán další snímek.

Diagram činnosti této funkce je:



Obr. 2.2: Diagram činnosti funkce `urciTfotmAOrez`

Následující oddíly se budou věnovat jednotlivým blokům.

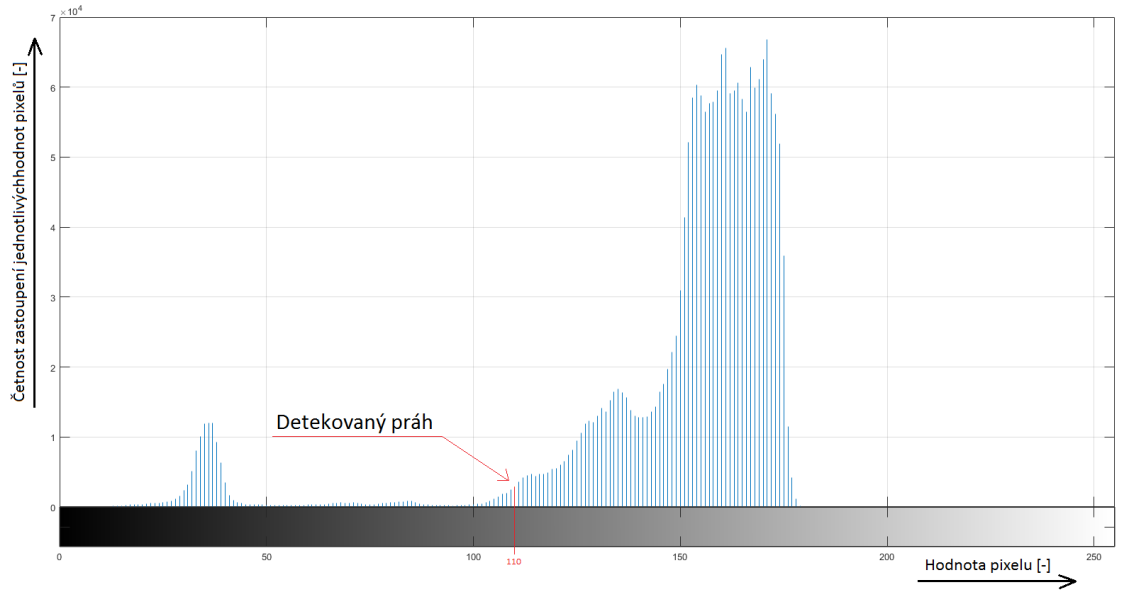
2.1.1 Načítání snímků

Program byl nastaven, aby prohledal předem danou složku a vyhledal v ní všechny soubory s příponou „jpg“. Poté, se znalostí jejich počtu a názvů, si je postupně načítá ke zpracovávání.

2.1.2 Získání prahu

Z histogramu červené barevné složky obrazu, se za pomoci funkce `prah4` získává práh. Funkce `prah4` si z histogramu červené barevné složky obrazu, který obdrží, zjistí globální maximum. Funkce prochází vektor histogramu od nejsvětlejšího k nejtmavšímu odstínu a za výsledek označí první bod, který leží za globálním/lokálním maximem (je třeba, aby dosáhlo 70% globálního maxima) a současně je menší než 5% (po některých úpravách i méně).

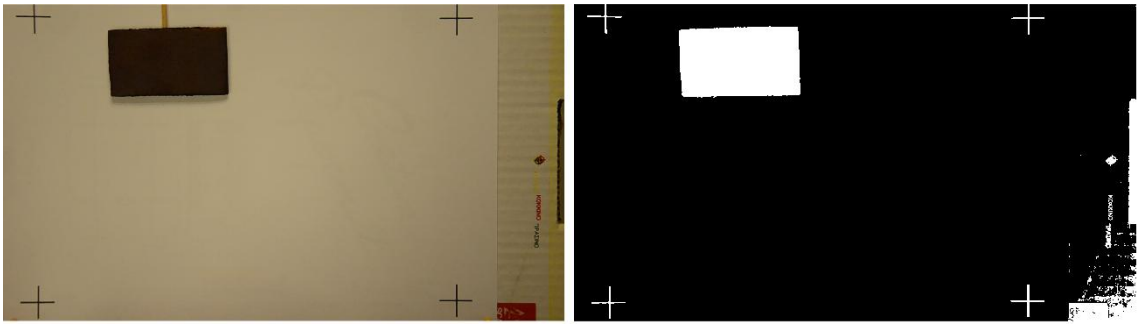
Důvodem nezvolení modré barevné složky jako vstupní bylo to, že její histogram měl nejméně jasně odlišitelné části, které by odpovídaly bílému pozadí černým značkám (křížkům) a černému zkoumanému objektu. Červená a zelená barevná složka měly tyto oblasti histogramu lépe odlišené než modrá. Průběh histogramu červené a zelené barevné složky měly obdobný tvar. Na následujícím obrázku je histogram červené barevné složky, ve kterém je práh, nalezený funkcí `prah4`.



Obr. 2.3: Histogram červené složky obrazu s vyznačeným detekovaným prahem

2.1.3 Vyprahování

Implementace samotného prahování se znalostí prahu je již snadná. Všechny pixely, které mají hodnotu nižší než práh, jsou ve výsledném binárním obrazu nastaveny na hodnotu jedna.



Obr. 2.4: Obraz získaný kamerou (vlevo) a vyprahovaný binární obraz

2.1.4 Vyhodnocení parametrů kamery

Parametry kamery se v Matlabu ukládají do `cameraParameters` objektu. Tento objekt je generován Camera Calibratorem Matlabu. Camera Calibrator načte obrázky vyfoceného šachovnicového vzoru a se znalostí délky strany jednoho políčka onoho vzoru, tvoří právě potřebný `cameraParameters` objekt. Do hlavního programu je načítán z předem vytvořeného objektu `cameraParameters`.

2.1.5 Odstranění zkreslení zanesené nedokonalostmi kamery

Ke korekci tohoto zkreslení je používána knihovní funkce Matlabu. Je to funkce `undistortImage`, která odstraňuje zkreslení zaviněné kamerou a jejím objektivem. Využívá k tomu objekt `cameraParameters`, který je vytvořen při kalibraci kamery.

2.1.6 Vyplnění děr v obraze

Protože při prahování může dojít k situaci, kdy se vlivem odlesku popřípadě jiného nežádoucího efektu stane, že vnitřní část/oblast objektu nebyla prahováním označena na hodnotu logická jedna a tudíž že není objektem. Takováto „díra“ může mít za následek špatné vyhodnocení symetrie. V tomto programu je k jejímu odstranění použita knihovní funkce `imfill`.

2.1.7 Vyhledání kalibračních značek v obrazu (křížků)

Pro detekci objektů a jejich parametrů/vlastností v binárním obraze je použita knihovní funkce Matlabu a to `regionprops`. Tato knihovní funkce nalezne všechny objekty a zjistí mnoho parametrů těchto objektů. S pomocí této funkce je možné zjistit informaci o obdélníku (BoundingBoxu), který obklopuje daný objekt. Následně program zpracovává výsek obrazu, ve kterém se daný objekt nachází. Protože křížek se sestává ze dvou čar, jejichž délky vymezují délky stran BoundingBoxu, tak počet pixelů, které mají v tomto výseku hodnotu jedna je mnohem menší, než celkový počet pixelů výseku (v případě scény pořízené v této práci). Typicky se jednalo o hodnoty kolem 11 % pixelů. I s rezervou bylo nastaveno 25%.

Protože se v některých snímcích objevovaly i malé nežádoucí objekty, které také splňovaly výše uvedené kritérium, ale současně nebyly křížky, tak bylo zavedeno další kritérium. Aby byl objekt posouzen, že je křížkem, tak musejí být splněny podmínky dané následujícími nerovnicemi:

$$5 < \frac{l_{ox}}{l_{vx}} < 20 \tag{2.1}$$

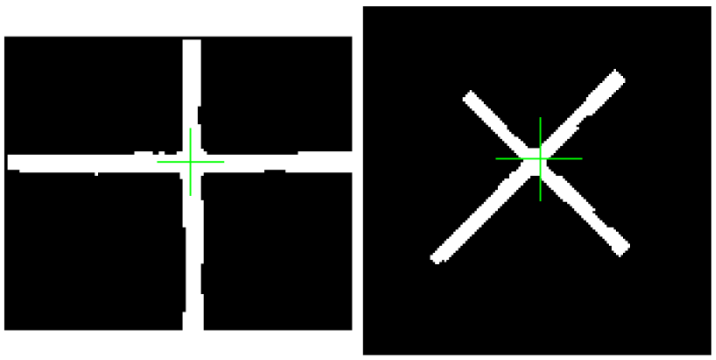
$$5 < \frac{l_{oy}}{l_{vy}} < 20 \tag{2.2}$$

kde l_{ox} a l_{oy} jsou délky celého binárního obrazu (v pixelech) v osách *x*, *y* a l_{vx} a l_{vy} jsou délky zkoumaného objektu/výseku opět v osách *x*, *y* (opět v pixelech).

Pro hledání středů křížků je naimplementovaná funkce `souradniceKrizku`. Z důvodu, že střed křížku (průsečík obou čar tvořící křížek) nemusí být totožný se středem BoundingBoxu, tak je nutné nalézt ve zpracovávaném výseku onen průsečík čar. Za předpokladu, že je křížek ve zkoumaném výseku rovně (jedna z čar je vodorovně a druhá svisle), tak pokud se vytvoří dva vektory sumací řádků (spočte se počet jedniček na každém řádku) a druhý sumací sloupců, tak každý z nich bude mít přibližně uprostřed (závisí na místě průsečíku čar křížku) výraznou špičku, protože právě zde se vyskytla druhá čára, jejíž body tuto špičku vytvořily. Indexy odpovídající středům těchto špiček jsou souřadnicemi středu křížku v obou osách.

Algoritmus má naimplementovanou také variantu detekce křížku, jehož čáry jsou v diagonálním směru. Zde je princip podobný, ale při sumaci řádků a sloupců se v obou vektorech vyskytnou dvě lokální maxima, mezi nimiž je lokální minimum. Důvodem je to, že v místě, kde průsečík není, se v řádcích a sloupcích vyskytují dvě čáry, zatím co v místě průsečíku se překrývají a tloušťka objektu tu značně klesá, proto středy těchto lokálních minim tvoří souřadnice křížku v obou osách.

Následuje ukázka obou variant křížku s vyznačenými středy nalezené touto implementovanou funkcí:



Obr. 2.5: Výseky s křížky s vyznačenými nalezenými středy

Protože v pořízených snímcích jsou křížky otočeny normálně, tak je funkce více optimalizovaná pro hledání právě takto natočených křížků. Odchylka takto nalezeného středu je asi jeden až dva pixely oproti ručnímu hledání člověkem.

Protože i přes výše zmíněná kritéria se může stát, že je detekováno jako křížek i něco jiného, tak je třeba vybrat z nalezené množiny bodů čtyři souřadnice, které budou odpovídat zadaným souřadnicím reálného světa. Pro tuto úlohu je naimplementovaná funkce `spravPoradiIP`. Tato funkce ke každému zadanému světovému bodu hledá nejbližší nalezený bod (pro jednoduchost implementace Manhattanskou metrikou). Dále

je ve funkci zabudované dvě jednoduché ochrany. První je, že funkce vrátí chybu, pokud je počet nalezených bodů menší, než počet bodů světa. Druhou ochranou je, že absolutní hodnota rozdílu hodnot nalezených a zadaných souřadnic bodů, nesmí být více, než 30 % maximální hodnoty světových souřadnic v každé ose. Pokud není podmínka dodržena, opět funkce vrátí chybu.

Před implementací základních ochran na méně kvalitních (například hůře nasvícených) snímcích se stávalo, že bylo nalezeno méně bodů a funkce souřadnice jednoho nalezeného bodu vyhodnotila jako nebližší pro dva světové body anebo bylo nalezeno dostatek bodů, ale nebyly nalezeny správné (neodpovídaly světovým bodům) a následně program spadl při počítání transformace s nesmyslnými požadavky.

2.1.8 Získání projektivní transformace

K odstranění zkreslení, které zavinila špatná poloha kamery vůči snímané scéně je zapotřebí užít dvě knihovní funkce. Jednou z nich je `fitgeotrans`, která se nastaví, aby počítala projektivní transformaci a ze souřadnic bodů scény a odpovídajících bodů v obraze vypočte transformační matici pro odstranění projektivního zkreslení. Díky faktu, že se pozice scény ani kamery během snímání neměnila, tak se tato transformační matice počítá jen jednou.

2.1.9 Projektivní transformace obrazu

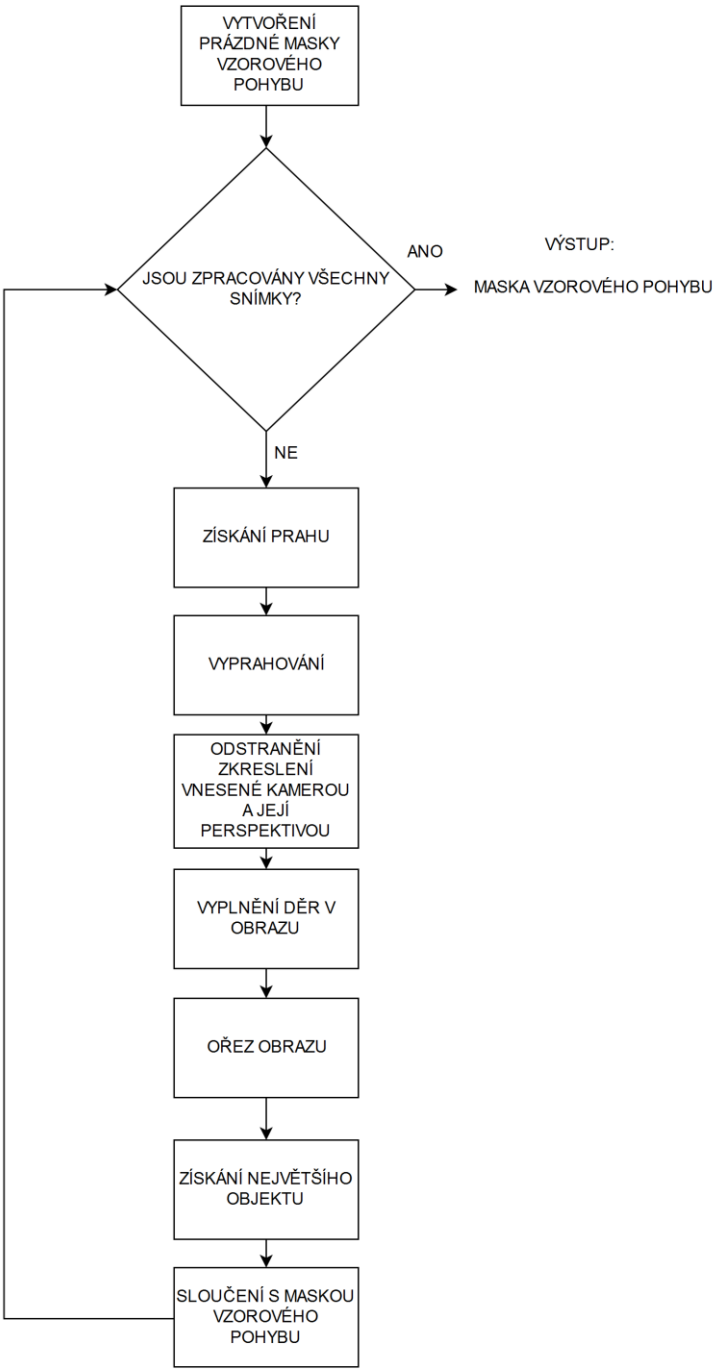
Na výstup funkce `fitgeotrans` následně navazuje funkce `imwarp`, která provede získanou transformaci nad obrazem.

2.1.10 Výpočet souřadnic pro ořezávání obrázků

Po projektivní transformaci a znovunalezení souřadnic kalibračních značek vymezujících vyhodnocovanou část scény, se právě ze souřadnic křížků nalézá ten, jehož součet obou souřadnic je nejmenší. Tento bod vymezuje levý horní roh výřezu. Následně je nalezen bod, jehož součet souřadnic v obou osách je největší. Tento vymezuje pravý dolní roh výseku. Souřadnice výseku se ale udávají ve formě levého horního bodu a přírůstků v obou osách, kterého se dosáhne odečtením levého horního bodu od pravého dolního bodu (v příslušných osách).

2.2 Vyhodnocení povolených mezních stavů ze vzorových snímků

Pokud má program vyhodnocovat, jestli a jak moc se zkoumaný objekt vychyluje z dráhy, je třeba mu ukázat vzorové snímky, kde se objekt vyskytovat smí. Principem této myšlenky je to, že každý snímek ve vzorové množině obsahuje objekt, který je na povoleném místě. Pokud se tato místa spojí, vznikne oblast definující mezní polohu/stav. Tato část programu je zapouzdřena ve funkci `tvorMaskuPohybu` a tvoří „masku“, která udává, na kterých místech je objektu povoleno se vyskytovat. Zde následuje diagram činnosti této funkce:



Obr. 2.6: Diagram činnosti funkce `tvorMaskuPohybu`

V následujících oddílech jsou popsány bloky, které nebyly použity v minulé funkci.

2.2.1 Vytvoření prázdné masky vzorového pohybu

Zde se pouze vytvoří binární matice o hodnotách nula s rozměry odpovídajícími rozměrům oblasti vymezené kalibračními značkami (křížky).

2.2.2 Ořez obrazu

K ořezu obrazu je využívána naimplementovaná funkce `vysekZBB`, která vrátí výsek obrazu specifikovaný ve tvaru souřadnice levého horního rohu (v obou osách) a rozměrů daného obdélníku. K ořezu jsou využita data získaná dříve.

2.2.3 Získání největšího objektu

Pro hledání objektů v binárním obrazu je použita výše zmíněná knihovní funkce `regionprops`, která objekty hledá a zjišťuje jejich vlastnosti a pro každý objekt generuje vektor souřadnic, ze kterých je možné každý jednotlivý objekt zrekonstruovat. Protože zkoumaný objekt není jediným vyskytujícím se objektem v obraze scény ani za ideálních podmínek (jsou zde kalibrační křížky a dále nežádoucí objekty), tak je třeba zvolit, který z nich je právě tím zkoumaným objektem. Nejjednodušším kritériem se prokázalo být to, že zkoumaný objekt je objektem největším, tzn., že má nejvíce pixelů a proto byly takto nastaveny i požadavky na scénu.

Díky implementaci ořezu obrazu (aby obsahoval jen část vymezenou křížky) se odstraňuje nežádoucí vliv nasnímaného okolí na okraji snímaného obrazu, který nevyhovuje požadavkům na scénu.

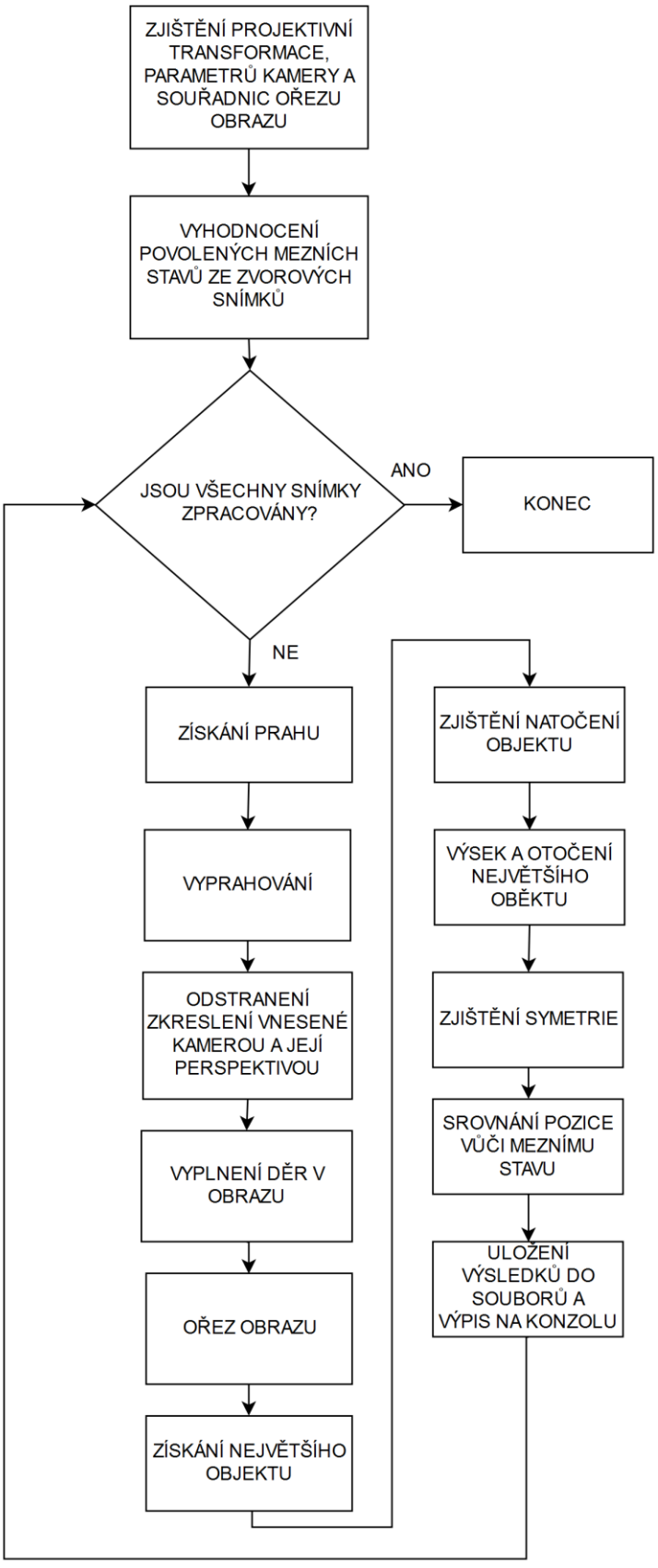
Výstupem tohoto bloku je binární matice, ve které je jen největší objekt vytcořený z údajů poskytnutých funkcí `regionprops`.

2.2.4 Sloučení s maskou vzorového pohybu

Aktuální binární obraz obsahující pouze hledaný (největší) objekt je spojen s maticí masky pomocí logického součtu a výsledek je opět uložen do masky. Po zpracování posledního cvičného snímku obsahuje matice masky jedničky všude tam, kde je povoleno, aby se objekt vyskytoval.

2.3 Zhodnocení symetrie objektu a pohybu v mezních stavech

V této části se provádí samotné vyhodnocování symetrie a překračování předem definovaných stavů. Toto je obsahem skriptu s názvem `scriptA.m`, který využívá právě obě předešlé stěžejní funkce a to `urciTfotmAOrez` a `tvorMaskuPohybu`. Informace z funkcí získané určují používanou projektivní transformaci, určený výsek scény a povolené mezní stavy pohybu/výskytu sledovaného objektu. Činnost popisuje následující diagram:

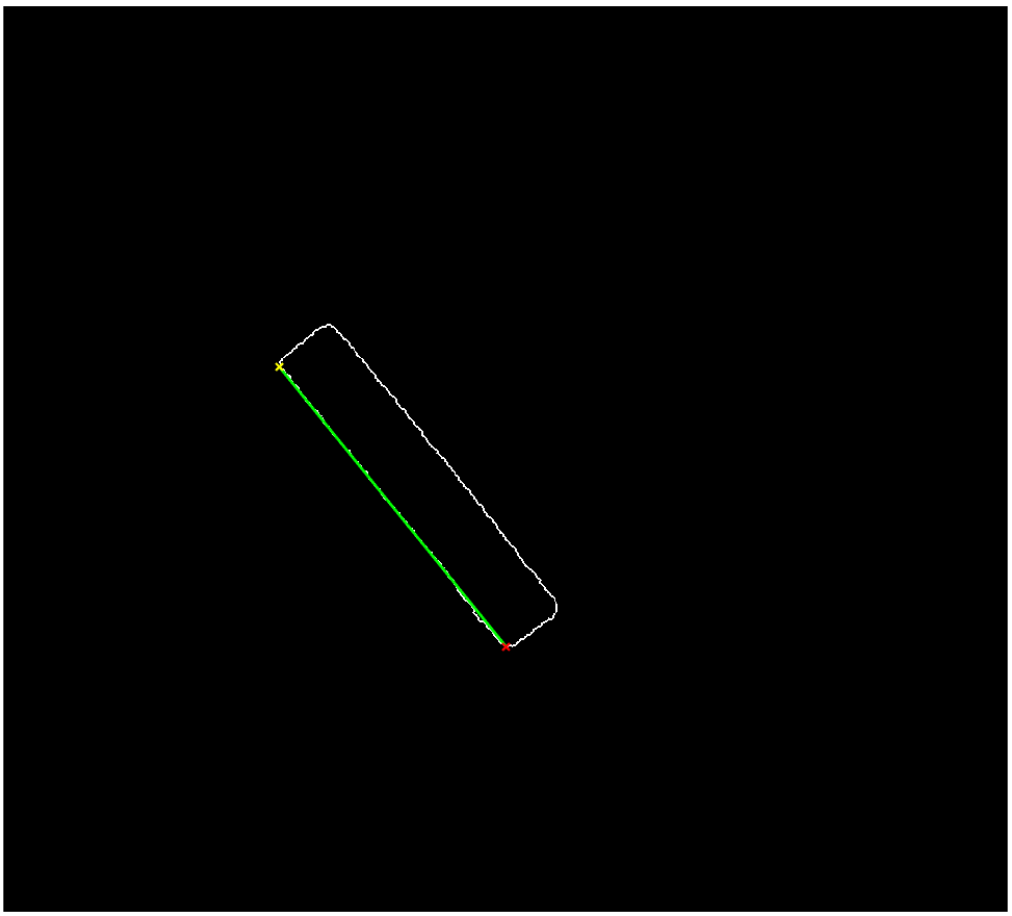


Obr. 2.7: Digram činnosti hlavního skriptu

Následující oddíly popisují nové bloky.

2.3.1 Zjištění natočení objektu

Pro zjištění natočení objektu je umístěn na každém testovacím objektu červený obdélník, jenž je detekován prahováním hledajícím značně vyšší hodnotu jasové funkce červeného barevného kanálu, než u jednoho ze zbývajících. Takto vzniklý binární obraz obsahující obdélník nesoucí informaci o natočení je zpracováván obdobně, jako hlavní binární obraz. Po detekování největšího objektu je stejný výsek vybrán i z binárního obrazu s obdélníkem, nad kterým je provedena morfologická dilatace a následně odečtení původního výseku, čím vznikne obrys onoho obdélníku. Na tomto obryse se hledá nejdelší úsečka pomocí Houghovy transformace (použity knihovní funkce). Spolu s nejdelší úsečkou, která se (v ideálním případě) kryje s jednou z delších stran onoho obdélníka, se získají body na jejích okrajích a z těch je pomocí goniometrických funkcí vypočítán náklon objektu. Hledání natočení objektu je implementováno ve funkci `houghUhelNatoceni`. Vizualizace nalezené úsečky na obrysu obdélníku ja na následujícím obrázku:



Obr. 2.8: Zobrazení hledané přímky ležící na jedné z delších stran obrysu hledaného obdélníku

2.3.2 Výsek a otočení největšího objektu

Se znalostí úhlu natočení a BoundingBoxu poskytnutých knihovní funkcí `regionprops` a implementovanou funkcí `houghUhelNatoceni`, není otočení zpět díky knihovní funkci `imrotate` obtížné. Výsek s objektem se vytvoří pomocí naimplementované funkce `VysekZBB` pomocí BoundingBoxu a funkce `imrotate` jej se zadanou zápornou hodnotou úhlu natočení otočí do vodorovné pozice.

2.3.3 Zjištění symetrie

Zhodnocování symetrie je zvoleno jako porovnávání symetrie pouze výseku binárního obrazu obsahujícího zkoumaný objekt, kde osa souměrnosti je svislice tohoto výseku, která je v jeho středu.

Tato úloha je implementována funkcí `symetrieAuto`. Osa symetrie v podobě svislé čáry je identifikovatelná jediným indexem (sloupcem) matice výseku, který je získán jako polovina šířky matice výseku dělená dvěma a zaokrouhlen dolů (aby výsledek byl celočíselný a dal se použít k indexaci matice). Z této matice výseku je třeba vytvořit dvě dílčí submatice, mezi nimiž se budou hledat rozdíly poukazující na nesymetrii. První submatice obsahuje část matice výseku od prvního sloupce až po sloupec indexu osy souměrnosti včetně a druhá submatice obsahuje zbytek vstupní matice. Následně se provede s druhou submaticí tzv. „flip“, což je otočení pořadí sloupců matice. Problém, který se vyskytl, byl, že dvě nově vzniklé matice neměly ve většině případů stejné rozměry, které pro maticové porovnávání vyžaduje Matlab. Řešením bylo, zvětšit menší matici (přidat nulové sloupce), ale ne na straně, na které byl střed objektu, ale na okraji.

Tímto existují dvě submatice o stejných dimenzích, které jsou připraveny, aby se hledaly jejich rozdíly značící výskyt nesymetrií. Z těchto submatic se vytvoří dvě matice nesymetrií, kde do jedné se pomocí logických operací vyznačí, co v jedné submatici je a zároveň v druhé není a do druhé matice nesymetrií se vyznačí, co v první není a současně je v druhé. Druhá matice nesymetrií musí být opět „flipnuta“, aby korespondovala s pravou částí vstupní matice výseku.

Výsledná matice je již trojrozměrná (aby uložila i barvy). Je vytvořena překopírováním vstupní matice výseku do všech tří barevných složek. Následně jsou do příslušných barevných složek pomocí logických operací přidány matice nesymetrií, které ve výsledném obrazu vyznačují, co na jaké straně od osy symetrie chybí, vůči té druhé. Svou implementací je funkce připravena, aby mohla být poloha osy symetrie volena. Velikost vstupní a výstupní matice (až na přírůstek ve třetím rozměru kvůli nesení barev) je stejný a nemění se ani v případě, kdy by kvůli posunuté ose souměrnosti vůči středu vznikala vyznačovaná nesymetrie za hranicí vstupní matice.

Funkce dále vypočítavá procento nesymetrie podle následujícího vzorce:

$$\text{Nesymetrie} = \frac{\text{Počet pixelů označujících nesymetrii}}{\text{Celkový počet pixelů objektu}} \cdot 100\ [\%] \tag{2.3}$$

Vzhledem k možnosti funkci zadat souřadnici (číslo sloupce) osy souměrnosti, tak by se dala použít ke zhodnocování souměrnosti v celém výseku obrazu, kde by se vyskytoval pouze zkoumaný objekt a to by mohlo sloužit ke správnému nastavení nebo ověření pozice objektu. Například by bylo možné kontrolovat, zda li je výrobek na

dopravníku ve správné poloze, aby mohl být uchopen dalším strojem.

Na následujícím obrázku je ukázka vyznačení nesymetrie:

Obr. 2.9: Ukázka detekování nesymetrie na syntetickém obrazu

2.3.4 Srovnání pozice vůči mezním stavům

Výstupem tohoto bloku programu je RGB obraz. Jeho sestavení se provádí následovně.

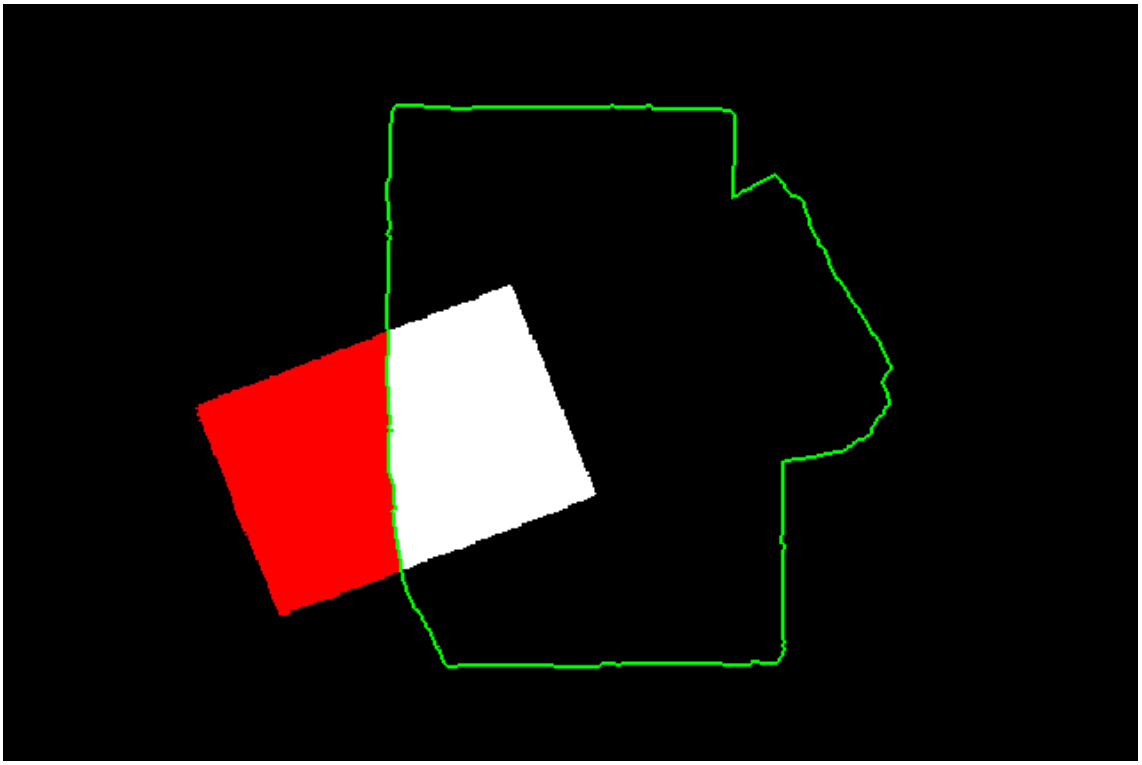
Nejdříve se vytvoří černý obrazu a do jeho červené barevné složky se překopíruje zkoumaný objekt. Do ostatních barev se uloží logický součin obrazu objektu a masky. Tímto vznikne obraz, ve kterém části mimo povolené mezní stavy budou červeně a části uvnitř budou bílé.

Dále se přidá do výstupního obrázku pomocí logických operací zelený rámeček, který vyznačuje místo/místa povoleného výskytu objektu. Tento rámeček vznikl morfologickou dilatací masky a následným odečtením masky.

Tento blok programu také vypočítává procento části, kterou je mimo vymezený mezní stav podle vzorce:

$$\text{Část objektu mimo mezní stav} = \frac{\text{Počet pixelů mimo vymezený stav}}{\text{Celkový počet pixelů objektu}} \cdot 100\ [\%] \qquad (2.4)$$

Na následujícím obrázku je ukázka vyznačení odchylky od mezního stavu generovaná touto částí programu:



Obr. 2.10: Ukázka zvýraznění odchylky objektu od dovoleného mezního stavu

2.3.5 Výstup

Program vypisuje na konzolu a exportuje soubory. Do konzoly vypisuje informaci o procentu nesymetrie zkoumaného objektu a o procentu zkoumaného objektu, který je mimo povolený mezní stav.

Z každého snímku ukládá čtyři obrázky, kde jedním je vyprahovaná oblast mezi kalibračními značkami (po aplikaci projektivní transformace), druhým je stejná oblast, ve které je už pouze sledovaný objekt s vyznačením mezních stavů a jeho polohy vůči nim. Dalším obrázkem je výsek vodorovně natočeného objektu a poslední je stejný objekt doplněný o vyznačení symetrie.

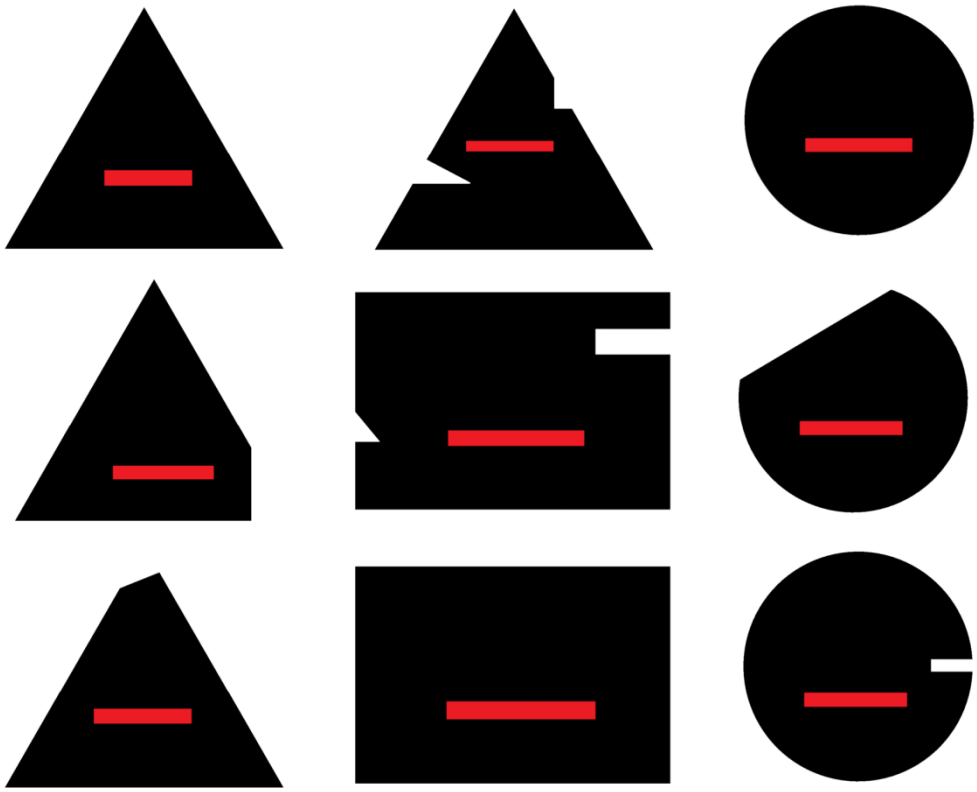
3 MĚŘENÍ A ZHODNOCENÍ

Tato kapitola pojednává o testovacích objektech, samotném pořízení snímků a videí, jejich vlastnosti a zhodnocení přesnosti navržených algoritmů.

3.1 Testovací objekty

Pro ověření činnosti navržených algoritmů bylo vytvořeno deset testovacích objektů. Jsou to tři základní geometrické útvary (kruh, obdélník a trojúhelník) a jejich verze, které byly asymetricky seříznuty nebo do nich byl utvořen jeden nebo dva výřezy. Testovací objekty jsou na následujících dvou obrázcích:

Obr. 3.1: Jeden z deseti testovacích objektů – obdélník se seříznutým rohem



Obr. 3.2: Zbylých devět testovacích objektů

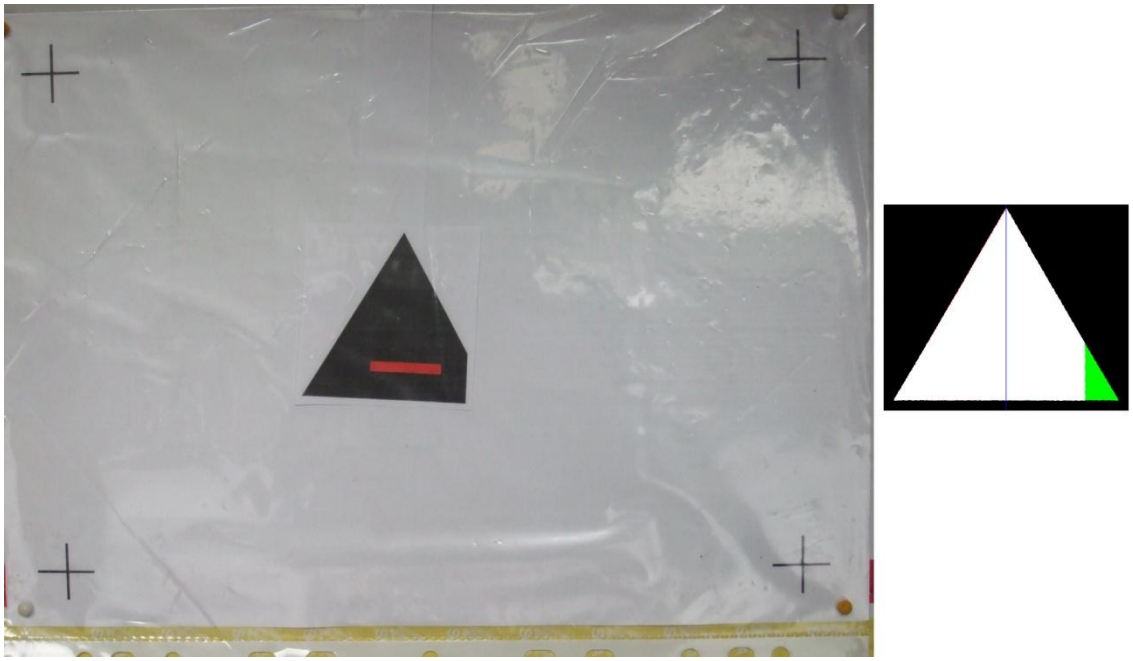
Na testovacích objektech se nachází červený obdélník, který pomáhá při snímání určit správné natočení objektu, než je analyzován funkcí pro detekci nesymetrie.

3.2 Pořízení snímků a videí

Podkladem snímané scény byla kartonová krabice, na níž byl připevněn papír obsahující kalibrační značky a průhledná eurosložka. Ta zde byla umístěna z důvodu, aby udržovala posouvající se testovací objekty přitlačené na papíru s kalibračními značkami. Kalibrační značky jsou na rozích pomyslného obdélníku o rozměrech 25,5x17 mm. Testovací objekty byly vytištěny na papír, vystřihnuty (i s bílým okrajem, aby střihem nebyly zmenšeny) a byly opatřeny pruhem papíru, aby bylo možné s nimi po snímané scéně pohybovat. Velikost testovacích objektů se pohybovala v rozmezí 5 až 8 cm. Vzdálenost fotoaparátu byla přibližně 60 cm od snímané scény.

S každým testovacím objektem bylo pořízeno jedno video a přibližně pět fotografií. Celkem tedy deset videí a 54 fotografií. Fotografie a videa byly pořizovány fotoaparátem FUJIFILM FinePix F31fd. Fotoaparát ukládal fotografie ve formátu JPG s rozlišením 2848x2136 pixelů a bitové hloubce 24 bitů. Videa měla mnohem nižší rozlišení a to 640x480 pixelů. Videa mají 30 snímků za sekundu, bitovou hloubku 24bitů a jsou ve formátu AVI. Z videí byla odstraněna malá část z počátku i konce každého videa, aby byla splněna podmínka, že se ani fotoaparát ani snímaná scéna v rámci videa nepohybuje (odstraněny otřesy způsobené stisknutím spouště). Videa byla následně uložena jako skupina obrázků ve formátu JPG, aby mohla být předložena ke

zpracování vytvořenému programu. Na následujícím obrázku je ukázka fotografie z množiny vstupních dat a výsledek po zpracování a vyznačení nesymetrie:



Obr. 3.3: Jedna ze vstupních fotografií (vlevo), výsledný obraz zobrazující nesymetrii (chybějící pravý roh trojúhelníku) (vpravo)

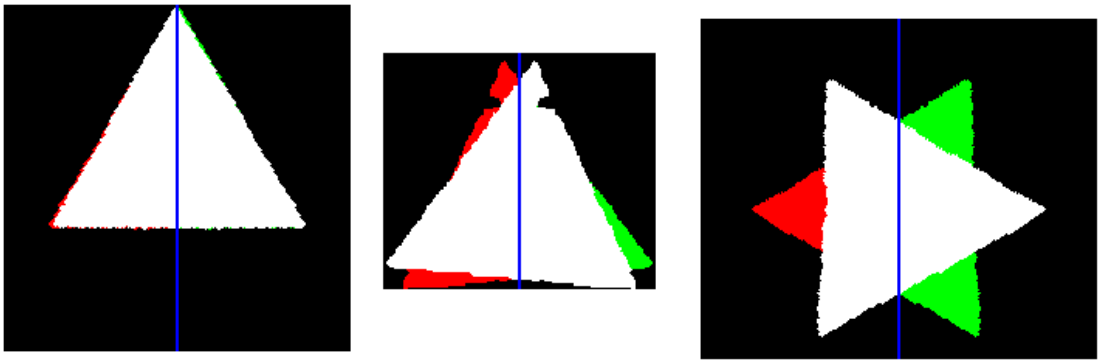
3.3 Metrika a zhodnocení

Program má za úkol na nasnímaném obrazu najít kontrastní objekt, ten správně otočit za pomocí červeného obdélníku, který je uvnitř objektu a na výsledném obrazu najít nejmenší nesymetrii daného objektu pomocí svislé osy. Stejný postup je uplatněn i na syntetické objekty (před tiskem – v PC). Při zpracovávání snímků ukládá do souboru typu Excel nesymetrii každého zpracovávaného snímku. Do následující tabulky je pro porovnání zanesena nejmenší, střední a největší hodnota nesymetrie a k ní odpovídající absolutní chyba.

Tab. 3.1: Tabulka správných a změřených hodnot nesymetrie deseti testovacích objektů pro fotografie i videa

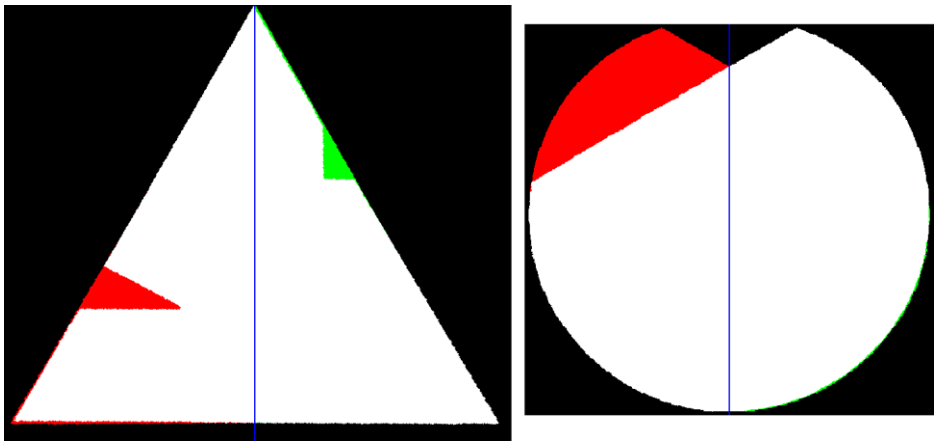
Objekt	Video/ Fotografie	Procento nesymetrie				Absolutní chyba		
		Původní obraz	Nasnímaný obraz					
			min	medián	max	min	medián	max
Kruh – seříznutý	V	10,572	5,139	12,406	17,622	-5,433	1,834	7,050
	F	10,572	10,030	10,392	11,397	-0,542	-0,180	0,825
Kruh s výřezem	V	1,336	1,585	2,153	3,450	0,249	0,817	2,114
	F	1,336	1,605	2,202	2,753	0,270	0,866	1,417
Kruh	V	0,071	0,567	1,147	8,587	0,496	1,077	8,517
	F	0,071	0,321	1,059	1,285	0,250	0,989	1,215
Obdélník se dvěma výřezy	V	3,469	3,432	4,503	9,230	-0,037	1,034	5,761
	F	3,469	4,224	4,476	5,512	0,755	1,007	2,043
Obdélník – seříznutý roh	V	5,093	3,342	5,159	14,157	-1,751	0,066	9,064
	F	5,093	4,982	5,224	6,195	-0,111	0,131	1,102
Obdélník	V	0,040	0,276	1,358	5,956	0,236	1,318	5,915
	F	0,040	0,616	0,782	1,551	0,575	0,742	1,510
Trojúhelník se dvěma výřezy	V	3,200	3,408	6,917	19,854	0,208	3,717	16,654
	F	3,200	3,702	4,833	5,127	0,502	1,632	1,927
Trojúhelník – seříznutý horní roh	V	0,627	1,192	3,975	14,172	0,565	3,349	13,546
	F	0,627	1,644	3,925	6,557	1,017	3,298	5,930
Trojúhelník – seříznutý pravý roh	V	4,834	4,534	6,555	14,189	-0,300	1,721	9,355
	F	4,834	4,528	4,979	6,866	-0,306	0,145	2,032
Trojúhelník	V	0,202	0,912	4,907	34,189	0,711	4,706	33,988
	F	0,202	1,561	4,789	8,009	1,360	4,588	7,807

Tabulka ukazuje, že program musí minimálně v několika případech některých videí fatálně selhat ve fázi předzpracování. Největší chyby má za vinu špatná detekce natočení při hledání červeného obdélníku definujícího správný náklon. Je to patrné z následujícího obrázku:



Obr. 3.4: Zobrazení tří výsledků. Správně natočený i vyprahovaný (vlevo), špatně vyprahovaný i natočený (uprostřed) a velmi špatně otočený a obstojně vyprahovaný zřídka se vyskytující – vpravo)

V menší míře nese odpovědnost nejasné hranice objektu při prahování. Nakonec v ještě menší míře nese odpovědnost kvalita transformací (hledání značek, kalibrace kamery). Z tabulky je dále vidět, že pro většinu testovacích objektů jsou výsledky fotografií lepší, protože mají větší rozlišení, což má za následek lepší otočení objektu díky přesnější detekce červeného obdélníku. Vyšší rozlišení má také pozitivní dopad na výsledek prahování, protože pokud se špatně vyprahuje stejné množství pixelů na okrajích objektu, objekt zaznamenáván ve vyšším rozlišení se sestává z více bodů a ve výsledném poměru je chyba menší a s tím jde ruku v ruce nižší chyba vlivem diskretizace, protože samotný algoritmus realizuje vzorec (2.3). Z tabulky ale též plyne, že ve většině z obrazových množin program dosáhl velmi malých chyb a v mnohých není ani maximální chyba moc závratná. Příklad pěkných výsledků je na následujícím obrázku:



Obr. 3.5: Příklad přesného výsledku algoritmu při detekci nesymetrie

Algoritmu realizujícímu hodnocení přítomnosti objektu v mezním stavu se týkají stejná pravidla, co se chyb týče. Také realizuje pouze matematický vzorec a proto je také přesnost jeho samotného ovlivněna jen rozlišením. Jeho největší nepřesnosti tedy mohou vznikat hlavně ve fázi předzpracování, kde jsou vlastnosti stejné. Typicky může jít o rozdíl o dva pixely na povrchu objektu. Výjimku tvoří to, že se ho netýkají jakékoliv chyby vlivem natočení, protože zde k němu nedochází.

I přes všechnu snahu o zdokonalení algoritmů, modifikovaná verze programu pro hromadné zpracovávání snímků odmítne zpracovat jeden snímek z 54.

4 ZÁVĚR

V této bakalářské práci byly shromážděny informace, které jsou nutné k vytvoření algoritmů ale i k objasnění jejich principů.

Dle zadání byla vytvořena databáze 54 snímků/fotografií a deseti videí, na kterých byly algoritmy testovány. Tyto algoritmy jsou implementované v programu, který je schopen vyhledat kontrastní objekt na snímku a z definovaného množství snímků zkonstruovat informaci o povolených/žádoucích mezních stavech. Následně je program schopen tyto informace využít a spočítat odchylky aktuální pozice objektu od povolených mezních stavů. Dále program určí objektovou symetrii. Program kromě textového výstupu podávajícího informaci o procentu nesymetrie a procentu velikosti objektu mimo mezní stav generuje obrázky, v nichž jsou tyto odchylky a hranice mezních stavů barevně vyznačeny.

Navržené algoritmy byly otestovány na pořízené množině 54 snímků a deseti videích, kde program úspěšně zpracoval všechna data, kromě jedné fotografie. Zhodnocením přesnosti algoritmů se zabývá kapitola 3.

Do budoucnosti bych rád zrobustnil program, aby lépe a spolehlivěji vyhledával kalibrační značky, například adaptabilním prahováním nebo automatickými pokusy o opravu kalibračních značek morfologickými operacemi.

LITERATURA

[1] Slovník VÚGTK [online]. – [cit. 15. květen 2016]. Dostupné z: https://www.vugtk.cz/slovnik/6390_radialni-distorze

[2] Chromatická aberace. *Wikipedia: the free encyclopedia.* [online]. 2001- [cit. 2016-05-15]. Dostupné z: *https://cs.wikipedia.org/wiki/Chromatick%C3%A1_aberace*

[3] Matlab. Programový systém pro provádění matematických výpočtů. Komerční software, verze R2015b. [online]. <http://www.mathworks.com>. The MathWorks, 2016.

[4] SOJKA, Eduard, Jan Gaura a Michal Krumnikl. *Matematické základy digitálního zpracování obrazu* [online]. 2011. [cit. 2016-05-17]. Dostupné z: *http://mrl.cs.vsb.cz/people/sojka/dzo/mzdzo.pdf*

[5] Horák Karel, Matematická morfologie [online]. Dostupné z: *http://midas.uamt.feec.vutbr.cz/ZVS/Lectures/10_Matematicka_morfologie.pdf*

[6] BENDA, Martin. Homografie a epipolární geometrie. *Trilobit.* [online]. 31.10.2010 [cit. 2016-05-18]. Dostupné z: *http://trilobit.fai.utb.cz/homografie-a-epipolarni-geometrie*

[7] DRBOHLAV, Ondrej. Výpočet projektivní transformace. *Center for Machine Perception (CMP).* [online]. 06.12.2002 [cit. 2016-05-18]. Dostupné z: *http://cmp.felk.cvut.cz/cmp/courses/383ZS/ZSO2003-4/cvic3/node4.html*

[8] HORÁK, Karel. Cvičení 5 – Geometrické transformace. *Computer Vision.* [online]. [2010] [cit. 2016-05-19]. Dostupné z: *http://midas.uamt.feec.vutbr.cz/ZVS/Exercise05/content_cz.php*

[9] Horák Karel, Detekce hran a Rohů [online]. Dostupné z: *http://midas.uamt.feec.vutbr.cz/ZVS/Lectures/07_Detekce_hran_a_rohu.pdf*

[10] Hough Line Transform. OpenCV 2.4.13.0 documentation. [online]. © 2011-2014 [cit. 2016-07-31]. Dostupné z: http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/hough_lines/hough_lines.htm

SEZNAM PŘÍLOH

Příloha 1: DVD – obsahuje:

1. Soubor: Detekce objektu.pdf
2. Složka: program a snimky