



**FAKULTA APLIKOVANÝCH VĚD  
ZÁPADOČESKÉ UNIVERZITY  
V PLZNI**

**KATEDRA  
KYBERNETIKY**

## **Bakalářská práce**

# **Odhad polohy a orientace robotu pomocí fiduciálních značek**

**Andrea Kadlecová**





FAKULTA APLIKOVANÝCH VĚD  
ZÁPADOČESKÉ UNIVERZITY  
V PLZNI

KATEDRA  
KYBERNETIKY

## **Bakalářská práce**

# **Odhad polohy a orientace robotu pomocí fiduciálních značek**

Andrea Kadlecová

### **Vedoucí práce**

Ing. Miroslav Flídr, Ph.D.

© Andrea Kadlecová, 2023.

Všechna práva vyhrazena. Žádná část tohoto dokumentu nesmí být reprodukována ani rozšiřována jakoukoli formou, elektronicky či mechanicky, fotokopírováním, nahráváním nebo jiným způsobem, nebo uložena v systému pro ukládání a vyhledávání informací bez písemného souhlasu držitelů autorských práv.

**Citace v seznamu literatury:**

KADLECOVÁ, Andrea. *Odhad polohy a orientace robotu pomocí fiduciálních značek*. Plzeň, 2023. Bakalářská práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra kybernetiky. Vedoucí práce Ing. Miroslav Flídr, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd  
Akademický rok: 2022/2023

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: **Andrea KADLECOVÁ**  
Osobní číslo: **A20B0584P**  
Studijní program: **B0714A150005 Kybernetika a řídicí technika**  
Specializace: **Automatické řízení a robotika**  
Téma práce: **Odhad polohy a orientace robotu pomocí fiduciálních značek**  
Zadávací katedra: **Katedra kybernetiky**

## Zásady pro vypracování

1. Seznamte se ze základními metodami odhadu polohy robotu z obrazových dat.
2. Seznamte se s metodami odhadu orientace objektu v obraze s využitím fiduciálních značek.
3. Navrhňte systém odhadu polohy a orientace robotu pomocí fiduciálních značek.





Rozsah bakalářské práce: **30-40 stránek A4**  
Rozsah grafických prací:  
Forma zpracování bakalářské práce: **tištěná**

Seznam doporučené literatury:

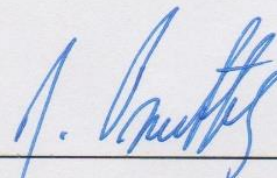
- P. Corke, Robotics, Vision and Control: Fundamental Algorithms In MATLAB, Second Edition, ISBN 978-3319544120, 2017
- Krajník, T., Nitsche, M., Faigl, J. et al. A Practical Multirobot Localization System. J Intell Robot Syst 76, 539–562 (2014)
- Lightbody P., Krajník T., Hanheide M., An efficient visual fiducial localisation system, ACM SIGAPP Applied Computing Review, 17(3), pp. 28–37, (2017)
- Yu, J., Jiang, W., Luo, Z., Yang, L. Application of a Vision-Based Single Target on Robot Positioning System. Sensors 2021, 21, 1829
- Kalaitzakis, M., Cain, B., Carroll, S. et al. Fiducial Markers for Pose Estimation. J Intell Robot Syst 101, 71 (2021)
- Cruz-Hernández H. and de la Fraga L. G., A fiducial tag invariant to rotation, translation, and perspective transformations, Pattern Recognition, Volume 81, (2018), pp. 213-223, ISSN 0031-3203

Vedoucí bakalářské práce: **Ing. Miroslav Flídr, Ph.D.**  
Katedra kybernetiky

Datum zadání bakalářské práce: **17. října 2022**  
Termín odevzdání bakalářské práce: **22. května 2023**



**Doc. Ing. Miloš Železný, Ph.D.**  
děkan



**Prof. Ing. Josef Psutka, CSc.**  
vedoucí katedry

# Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Západočeská univerzita v Plzni má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Dobřanech dne 10. srpna 2023

.....

Andrea Kadlecová

V textu jsou použity názvy produktů, technologií, služeb, aplikací, společností apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

## Abstrakt

Tato bakalářská práce se zabývá sledováním polohy a orientace fiduciální značky za použití běžně dostupné kamery. Nejdříve se bude řešit problematika se zpracováním obrazu. Dále se bude zabývat fiduciálními značkami, které lze v této úloze použít. Následně se zde bude probírat transformace souřadných systémů. Budou zde diskutovány metody odhadu polohy a orientace objektu na základě obrazových dat. Jedna část bude věnována experimentům, kde se bude sledovat statický marker a určovat jeho poloha a orientace v obraze a poloha v lokálním souřadném systému. Porovnájí se dva typy běžných kamer na přesnost odhadu s reálnými naměřenými daty. Také se zde bude zkoumat vliv velikosti markru, výška položení kamery, úhel naklonění a vzdálenost položení kamery od detekované plochy. Na konci se vše vyhodnotí v závěru.

## Abstract

This bachelor's thesis deals with the tracking of the position and orientation of a fiducial marker using a commonly available camera. The first focus will be on image processing issues. It will further address the fiducial markers that can be utilized in this task. Subsequently, transformations of coordinate systems will be discussed. Methods for estimating the position and orientation of an object based on image data will be deliberated upon. One section will be dedicated to experiments involving the tracking of a static marker, determining its position and orientation in the image, and its position in the local coordinate system. Two types of common cameras will be compared for the accuracy of estimation using real measured data. The impact of marker size, camera elevation, tilt angle, and camera-to-surface distance will also be examined. The conclusion will provide an evaluation of all findings.

## Klíčová slova

ArUco marker • AprilTag marker • STag marker • detekce fiduciální značky • kalibrace kamery • odhad polohy z obrazových dat • transformace 3D souřadných systémů



## Poděkování

Zde bych chtěla poděkovat svému vedoucímu bakalářské práce panu Ing. Miroslavu Flídřovi Ph.D. za velkou trpělivost, vstřícnost, poskytnutí studijních materiálů a zdrojů, ze kterých jsem mohla čerpat.



# Obsah

<b>1</b>	<b>Úvod</b>	<b>3</b>
<b>2</b>	<b>Zpracování obrazu pro detekci objektů v prostoru</b>	<b>5</b>
2.1	Teorie zpracování obrazu . . . . .	5
2.2	Kalibrace kamery . . . . .	8
2.3	Implementace v OpenCV . . . . .	17
<b>3</b>	<b>Metody detekce objektu v obraze s využitím fiduciálních značek</b>	<b>19</b>
3.1	Příklady druhů fiduciálních značek . . . . .	19
3.1.1	AprilTag markery . . . . .	19
3.1.2	S-Tag markery . . . . .	21
3.1.3	ArUco markery . . . . .	23
3.2	Detekce ArUco markeru . . . . .	24
3.2.1	Implementace v OpenCV . . . . .	29
<b>4</b>	<b>Základní transformace v souřadných systémech pro lokalizaci v obraze</b>	<b>31</b>
4.1	Transformace 3D souřadných systémů . . . . .	31
4.2	Souřadný systém ArUco markeru . . . . .	35
<b>5</b>	<b>Metody odhadu polohy a orientace objektu na základě obrazových dat</b>	<b>37</b>
5.1	Metoda estimatePoseSingleMarkers . . . . .	37
5.2	Implementace v OpenCV . . . . .	38
<b>6</b>	<b>Experimenty</b>	<b>42</b>
6.1	Použité ArUco markery . . . . .	42
6.2	Použité kamery . . . . .	43
6.3	Průběh experimentů . . . . .	43
6.3.1	Poloha a orientace ArUco markeru v souřadném systému kamery . . . . .	44

6.4	Poloha ArUco markeru v lokálním souřadném systému . . . . .	57
<b>7</b>	<b>Závěr</b>	<b>63</b>
<b>A</b>	<b>Knihovna OpenCV</b>	<b>65</b>
<b>B</b>	<b>Diadické operace pro zpracování obrazu</b>	<b>67</b>
<b>C</b>	<b>Metoda RANSAC pro odhad polohy a orientace objektu na základě obrazových dat</b>	<b>75</b>
	<b>Seznam obrázků</b>	<b>77</b>
	<b>Seznam tabulek</b>	<b>79</b>
	<b>Seznam výpisů</b>	<b>81</b>
	<b>Bibliografie</b>	<b>82</b>

V reálných situacích se setkáváme s případy, kdy potřebujeme sledovat nějakého robota. Chceme znát jeho polohu a natočení v prostoru, z čehož můžeme usuzovat, kam se bude následně pohybovat. Vzhled robotů je různorodý, proto by před každým sledováním muselo proběhnout učení programu pro rozpoznání daného typu robota. A ani to by nevyřešilo všechny problémy. Když už bychom měli program, který nám našeho robota rozpozná, musí se definovat, odkud se vzdálenost od kamery bude měřit, kde je jeho přední a zadní část. Zní to zbytečně složitě. My jednoduše může umístit na robota fiduciální značku na námi zvolené místo, které je velmi dobře viditelné. Místo robota budeme určovat polohu a orientaci markeru na něm umístěném. Fiduciálních značek je velké množství. Pár z nich si v této práci představíme. Jedná se o binární matice, které jsou podle svého vzhledu roztržiděné do různých knihoven. Jejich identifikace je tedy velmi jednoduchá a rychlá. V experimentech využíváme ArUco marker, u kterého se vzdálenost určujeme od jeho středu ke kameře a jeho natožení je udáváno podle polohy horního levého rohu.

V článku *Detection of Binary Square Fiducial Markers Using an Event Camera* [1] je popsán algoritmus detekce ArUco značek, který využívá neuromorfni kameru, jenž zaznamenává události v podobě světelných změn místo klasických snímků. Po předzpracování obrazů jsou detekovány úsečky, které korespondují s okraji značky. Tyto úsečky jsou poté přidány do skupiny kandidátů pro značku. Následně jsou kandidáti rozvinuti do čtvercového obrazu se standardními rozměry. Pro každou buňku značky jsou použity Gaussovy filtry k určení barvy (bílá nebo černá) uvnitř každé buňky. Výsledkem jsou dekodované barevné kódy v každé buňce značky. Nakonec je provedena extrakce binárního kódu z rekonstruované značky a je ověřena jeho platnost a identifikátor v souladu s ArUco specifikacemi. Pokud je kód nalezen ve slovníku značek, je přijat jako validní, jinak je odmítnut jako falešný kandidát.

V článku *ArUco Marker based localization and Node graph approach to mapping* [2] je popsáno, jak získat hodnoty stočení (yaw) aniž bychom museli kalibrovat kameru. Knihovna OpenCV vrací seznam čtyř bodů s pixelovými souřadnicemi, které označují rohy markeru:  $(x_a, y_a)$ ,  $(x_b, y_b)$ ,  $(x_c, y_c)$ ,  $(x_d, y_d)$ . Tento postup umož-

ňuje vypočítat stočení markeru  $\Theta_z$  na základě zdánlivé horizontální délky strany  $s_a$  a ideální vertikální strany  $s_i$ . Implementace této metody poskytuje konzistentní a přesné měření  $\Theta_z$  s malou chybou, zvláště při vzdálenosti do 3 metrů od markeru o velikosti  $20 \times 20$  cm. Pro spolehlivé výsledky je však nutné zarovnat kameru na střed snímku.

Tato bakalářská práce se zabývá sledováním polohy a orientace fiduciální značky za použití běžně dostupné kamery. Nejdříve se bude řešit problematika se zpracováním obrazu, která je popsána v části 2. Část 3 je věnovaná fiduciálním značkám, které lze v této úloze použít. Dále se zde bude probírat transformace souřadných systémů v části 4. Problematice odhadu polohy a orientace objektu na základě obrazových dat je věnována část 5. V části 6 se budou řešit experimenty. Bude se sledovat statický marker a určovat jeho poloha a orientace v obraze a v lokálním souřadném systému. Porovnájí se dva typy běžných kamer na přesnost odhadu s reálnými naměřenými daty. Také se zde bude zkoumat vliv velikosti markru, výška položení kamery, úhel naklonění a vzdálenost položení kamery od detekované plochy. V poslední části 7 se vše shrne v závěru.

Než se dostaneme ke konečné podobě programu, který zde bude popsán, je potřeba postupovat systematicky. Prvotním problémem bylo načíst obraz z kamery a převést ho do černobílé podoby. Následujícím problémem byla kalibrace kamery, kdy jsme si nejprve opatřili šachovnici a tu jsme snímali. Po úspěšně implementované kalibraci jsem se přesunula na problém detekce markeru v obraze a v lokálním souřadném systému. Na úplný závěr jsme využili při detekci zjištěný translační a rotační vektor, ze kterých jsme určili polohu a orientaci ArUco markeru ve zkoumaných souřadných systémech.

Cílem bakalářské práce je odhadnout polohu a orientaci fiduciální značky v obraze a ve zvoleném lokálním systému. Bude se zde zkoumat horizontálně položený statický ArUco marker o třech velikostech a ze dvou knihoven ArUco markerů. Z experimentů se vyhodnotí, zda záleží na druhu knihovny ArUco markerů a velikosti markerů. Bude se diskutovat vliv typu kamery a její umístění ve výšce a úhel naklonění na odhad polohy a orientace markeru v obraze. Úhel naklonění kamery a velikost markeru bude také zkoumána v experimentu zaměřeném na lokální souřadný systém. Nakonec se vyhodnotí získané poznatky z měření.



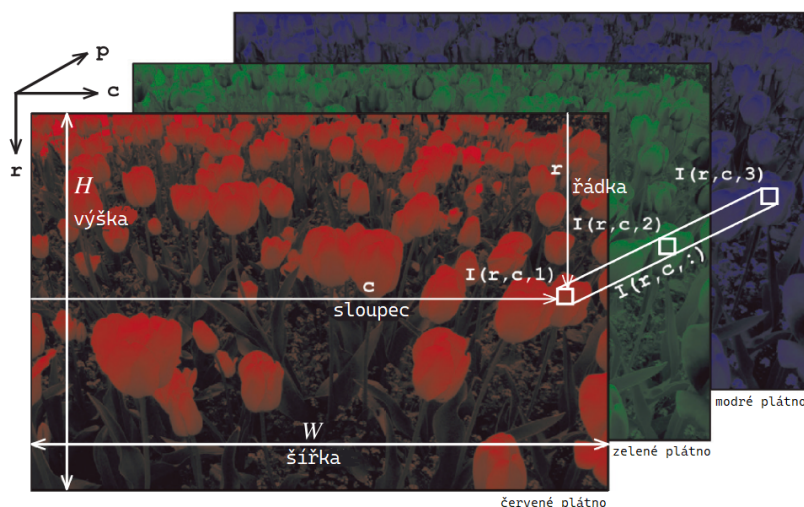
# Zpracování obrazu pro detekci objektů v prostoru

## 2

Tato kapitola se bude nejdříve zabývat samotnou interpretací obrázku v počítači. Následně zde bude popsán způsob, jakým se obraz předzpracuje. Snímaný obraz kamerou je barevný. Pro detekci ArUco markeru je potřeba mít černobílý obraz, kdy je vyšší úspěšnost detekce. Takový obraz je následně zkalibrován. V další části je popsáno tedy vnímání obrazu čočkou, možná zkreslení obrazu a kalibrace. Poslední část je věnována implementaci daných problematik (předzpracování obrazu a následná kalibrace) v jazyce C++ s pomocí knihovny OpenCV, viz dodatek A.

## 2.1 Teorie zpracování obrazu

Ze všeho nejdříve je potřeba popsat, jak se jednoduchý obrázek ve formátu JPG, PNG, apod. v počítači interpretuje. Pokud načteme jakýkoliv obrázek do určité instance, stane se z ní matice o velikosti výška  $\times$  šířka (černobílé obrázky), nebo výška  $\times$  šířka  $\times$  RGB plátina (barevné obrázky). Tyto parametry udávají šířku a výšku obrázku udávanou v pixelech. RGB plátina je vektor o třech parametrech. Popsané hodnoty jsou typu *uint8*, a tedy nabývají hodnot v intervalu  $\langle 0, 255 \rangle$ , kde 0 je nejtmaší a 255 je nejsvětější. Je však praktické hodnoty typu *uint8* převést na hodnoty typu *double*, je to z toho důvodu, že s reálnými čísly dokážeme lépe pracovat a máme přesnější výsledky. Na obr. 2.1 je znázorněna 3-dimezionální struktura barevného obrázku [3]. Pokud bychom si nechali vykreslit obrázek o velikosti (výška $\times$ šířka $\times$ 1. barva z RGB plátina), (výška $\times$ šířka $\times$ 2. barva z RGB plátina), (výška $\times$ šířka $\times$ 3. barva z RGB plátina), tak získáme černobílé obrázky.

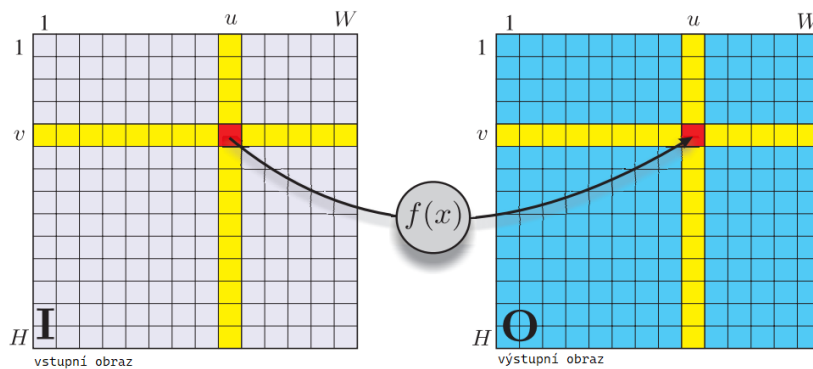


Obrázek 2.1: 3-dimenzionální struktura barevného obrázku

Dále se zaměříme na rozložení pixelů v obrázku. Ty nám poskytují informace o kvalitě obrazu a rozložení scény. Takové rozložení získáme výpočtem histogramu obrázku, který nám udá počet výskytů daných pixelů. Pokud se hodnoty histogramu budou rozprostírat od 0 do 255, jedná se o ideální stav. Pokud bude histogram více vlevo, je obraz podexponovaný. Následně opačně je obraz přeexponovaný, pokud je histogram situován vpravo - mnoho pixelů má maximální hodnotu. Pokud je histogram počítán pro barevné obrázky, vyjdou tedy tři histogramy, kdy pro každý barevný kanál je jeden samostatný histogram.

Druhy pixelů udávají vzhled histogramu (jeho vrcholy). Nízké vrcholy korespondují s tmavými pixely. Nejvyšší vrcholy odpovídají nejjasnějším barvám, např. bílá. A pak střední vrcholy jsou barvy, které jsou mezi. Je zde však problém v tom, že pokud máme černobílý obrázek, nelze jednoznačně určit kam patří nejsvětlejší pixely. Některé hodnoty mohou být tím pádem zavádějící.

Snímaný obraz můžeme pak zpracovat pomocí některé z monadických operací. Tyto operace pracují s hodnotami, které jsou "zabaleny v balíček" = v monádách. Je potřeba zmínit, že obraz v programu je reprezentován jako matice, tudíž skalární násobení a sčítání, odmocnina, či absolutní hodnota jsou monadické operace. Výsledkem těchto operací je obrázek o stejné velikosti výška  $\times$  šířka jako původní a každý nový pixel je funkcí toho původního  $\rightarrow \mathbf{O}[u, v] = f(\mathbf{I}[u, v]), \forall (u, v) \in \mathbf{I}$ . Proces je znázorněn na obr. 2.2 [3].



Obrázek 2.2: Zpracování obrazu pomocí monadické operace

Jedna třída monadických funkcí umožňuje změnu typu pixelových dat, např. převod z typu *uint8* na typ *double* a zpět. Další problém k řešení je, že snímaný obraz je barevný, jak už zde bylo zmíněno, takový obraz má tři dimenze a pixely jsou vektory. monadickou operací převedeme barevný obraz na obraz ve stupních šedi, kdy výstupní pixely jsou skaláry, které představují jas vstupního pixelu.

Prahování je také jedna ze tříd monadických operací. Metoda je jednodušší verzí segmentace, kdy rozděluje pixely do dvou tříd podle jejich intenzity. Nalezne takové hodnoty v histogramu, pro které bude platit, že hodnoty, které jsou menší než zadaný práh, jdou do první skupiny, a hodnoty vyšší, než je zadaný práh, jdou do druhé skupiny. Obrázky, které mají jen dvě barvy se nazývají binární.

Velká část monadických operací se zabývá tím, jak se mění úroveň šedi v obrazu. Může se stát, že obraz neobsahuje celý rozsah dostupných úrovní šedi, např. obraz je podexponovaný nebo přeexponovaný. Na hodnoty ve stupních šedi můžeme použít lineární mapování, které zajistí, že hodnoty pixelů pokrývají celý rozsah, který je buď  $< 0, 1 >$  nebo  $< 0, 255 >$  v závislosti na typu pixelů. Místo lineárního mapování lze použít i metodu normalizace histogramu nebo ekvalizace histogramu. Mapováním původního snímaného obrázku pomocí normalizovaného kumulativního rozdělení zajistíme, že kumulativní rozdělení výsledného obrázku je lineární (všechny hodnoty šedi se vyskytují stejně často).

Výstup kamery je zakódován pomocí gama kódování<sup>1</sup>, tudíž hodnota pixelu je nelineární funkcí  $L^\gamma$  jasu snímaného na fotomístě. Takový snímaný obraz pak může být gama dekodován pomocí nelineární monadické operace, která zvýší každý pixel na zadanou hodnotu, nebo lze RGB obraz dekodovat pomocí standartního gama dekodování. Další příkladem nelineární monadické operace jsou posterizace a páskování.

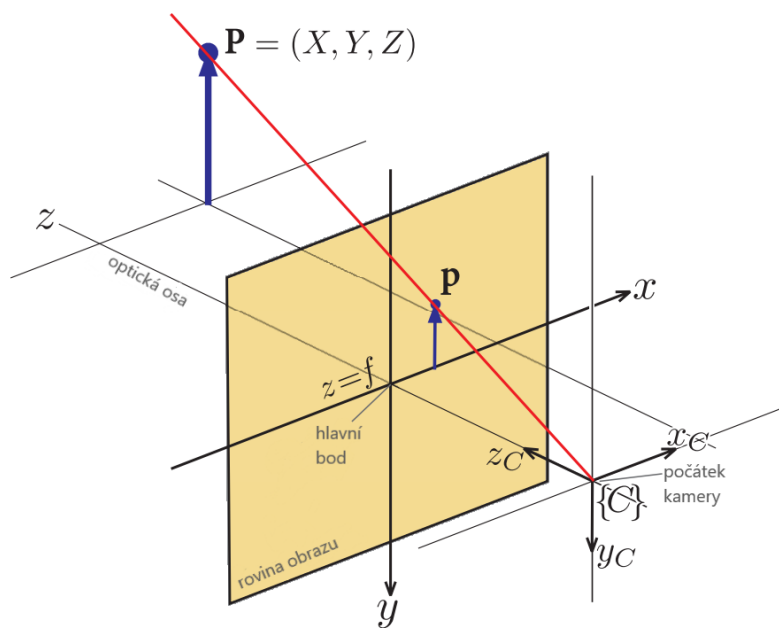
<sup>1</sup>Gama kódování a dekodování jsou často označovány jako gama komprese a gama dekomprese, protože operace kódování komprimuje rozsah signálu, zatímco dekodování jej dekomprimuje.

## 2.2 Kalibrace kamery

Na obr. 2.3 se nachází náčrt modelu středové projekce [3]. Rovina obrazu, kde se vytváří nepřevrácený obraz, se nachází ve vzdálenosti  $f$  před počátkem kamery. Souřadný systém kamery je pravotočivý. Osa  $z$  definuje střed zorného pole. Kladná osa  $z$  se nazývá optická osa. Zde se využije vztah (2.1) pro tenkou čočku [3]

$$\frac{1}{z_o} + \frac{1}{z_i} = \frac{1}{f}, \quad (2.1)$$

kde  $z_o$  je vzdálenost od objektu,  $z_i$  je vzdálenost od obrazu a  $f$  je ohnisková vzdálenost od čočky. Převrácená hodnota ohniskové vzdálenosti  $\frac{1}{f}$  je dioptrie. Pro tenké čočky položené blízko sobě je známo, že celková kombinovaná dioptrie je velmi blízká součtu individuálních dioptrií jednotlivých čoček. Pro  $z_o > f$  se v rovině obrazu vytvoří převrácený obraz ve vzdálenosti  $z_i < -f$ .



Obrázek 2.3: Model středové projekce

Ve fotoaparátu/v kameře je rovina obrazu fixována na povrchu snímacího čipu, takže zaostřovací kroužek fotoaparátu/kamery posouvá čočku podél optické osy tak, aby byla ve vzdálenosti  $z_i$  od roviny obrazu – pro objekt v nekonečnu platí  $z_i = f$ . Nevýhodou použití objektivu je nutnost ostřit. Naše vlastní oko má jednu konvexní čočku vyrobenou z průhledných krystalických proteinů a zaostření je docíleno pomocí svalů, které mění svůj tvar. Tento proces je znám jako akomodace.

Vysoce kvalitní čočka fotoaparátu je složená čočka obsahující více skleněných nebo plastových čoček.

V počítačovém vidění je běžné používat model středové projekce, viz obr. 2.3 [3]. Paprsky se protínají na počátku kamery  $\{C\}$ . Nepřevrácený obraz se pak nachází ve vzdálenosti  $z = f$  na rovině obrazu. Osa  $z$  protíná rovinu obrazu v hlavním bodě a vytváří tak počátek 2D souřadného systému. Pomocí metody podobnosti trojúhelníků můžeme ukázat, že bod v lokálních souřadnicích  $\mathbf{P} = (X, Y, Z)$  je promítán do bodu obrazu  $\mathbf{p} = (x, y)$ , což je transformace promítnutí, či specificky řečeno perspektivní projekce (2.2) [3]

$$\begin{aligned} x &= f \frac{X}{Z} \\ y &= f \frac{Y}{Z}. \end{aligned} \tag{2.2}$$

Tato transformace z 3D souřadného lokálního systému do 2D souřadného systému obrazu má svoje charakteristické vlastnosti [3]:

- Provádí mapování z 3D prostoru do 2D obrazu:  $\mathbf{R}^3 \rightarrow \mathbf{R}^2$ .
- Rovné čáry v realitě jsou promítány jako rovné čáry v obrazu.
- Rovnoběžné přímky jsou promítány do obrazu jako čáry, které se protínají v nekonečnu. V kresbě je tento jev znám jako *foreshortening*. Výjimku však mají linie, které leží v rovině rovnoběžné s rovinou obrazu. Ty zůstanou vždy paralelní.
- Kuželosečky se promítají do kuželoseček v rovině obrazu, např. kružnice se promítne jako kružnice, nebo jako elipsa.
- Velikost tvaru se nezachová a je závislá na vzdálenosti.
- Mapování není v poměru 1:1 a neexistuje žádná unikátní inverze. To znamená, že z daného obrazu  $(x, y)$  nedokážeme jednoznačně určit lokální souřadnice  $(X, Y, Z)$ . Jediné, co dokážeme říct, je to, že lokální bod  $\mathbf{P}$  leží někde na červeném promítnutém paprsku v obr. 2.3 [3].
- Transformace není konformní. To znamená, že nezachovává tvar, protože nejsou zachovány vnitřní úhly. Příklady konformní transformace jsou rotace, translace a změna měřítka.

Souřadnice bodu obrazové roviny můžeme zapsat v homogenním tvaru jako  $\tilde{\mathbf{p}} = (\tilde{x}, \tilde{y}, \tilde{z})$ , kde jsou jednotlivé souřadnice popsány ve vztahu (2.3) [3]

$$\begin{aligned}\tilde{x} &= f \\ \tilde{y} &= fY \\ \tilde{z} &= Z.\end{aligned}\tag{2.3}$$

Lze to i zapsat pomocí matice a vektoru, viz vztah (2.4) [3]

$$\tilde{\mathbf{p}} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix},\tag{2.4}$$

kde jsou nehomogenní souřadnice obrazové roviny následující, viz vztah (2.5) [3]

$$\begin{aligned}x &= \frac{\tilde{x}}{\tilde{z}} \\ y &= \frac{\tilde{y}}{\tilde{z}}.\end{aligned}\tag{2.5}$$

Tyto souřadnice se často označují jako sítnicové souřadnice obrazové roviny. Pokud nastane  $f = 1$ , jedná se o normalizované, retinální nebo kanonické souřadnice obrazové roviny.

V případě, kdy zapisujeme lokální souřadnicový systém v homogenní formě jako  ${}^C\tilde{\mathbf{P}} = (X, Y, Z)^T$  (souřadnice bodu vzhledem k rámu kamery  $\mathbf{C}$ ), je pak možné psát perspektivní projekci v lineárním tvaru (2.6) [3]

$$\tilde{\mathbf{p}} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} {}^C\tilde{\mathbf{P}} = \mathbf{C}^C\tilde{\mathbf{P}}.\tag{2.6}$$

Matice  $\mathbf{C}$  je velikosti  $3 \times 4$  a jedná se o matici kamery, která se dá vyjádřit jako součin dvou matic, kde jako první je matice z rovnice zobrazení bodu v obrazové rovině a druhá je matice projekce, viz vztah (2.7) [3].

$$\begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.\tag{2.7}$$



Na obr. 2.4 je znázorněna kamera, která má nějakou obecnou polohu  $\xi_C$  vzhledem k lokálnímu souřadnému systému [3]. Na obr. 2.4 je bod  $\mathbf{P}$ , jehož poloha vzhledem ke kameře je dána vztahem (2.8) [3]

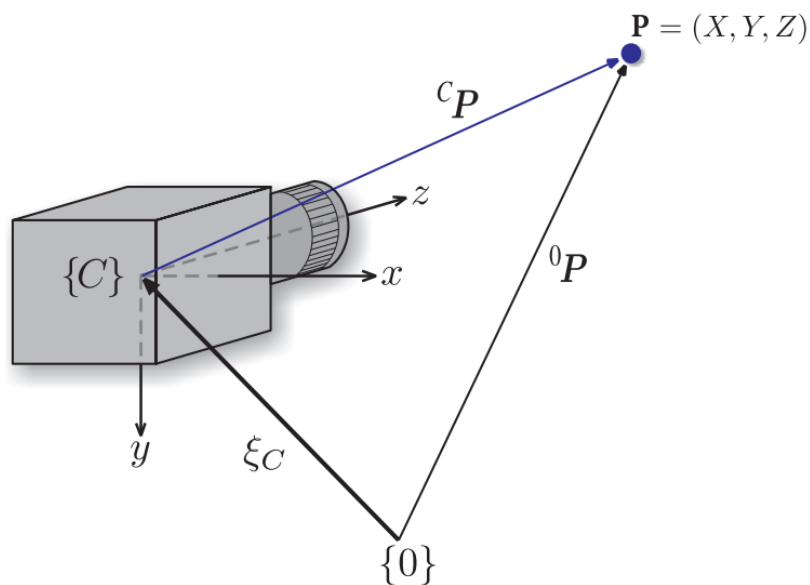
$${}^C\mathbf{P} = (\Theta\xi_C) \cdot {}^0\mathbf{P}, \quad (2.8)$$

kde  $\Theta\xi_C$  definuje zápornou hodnotu obecné polohy.

Lze využít homogenní souřadnice pro výpočet polohy bodu  $\mathbf{P}$ , díky tomu získáme vztah (2.9) [3]

$${}^C\mathbf{P} = \mathbf{T}_C^{-1} \mathbf{P}, \quad (2.9)$$

kde  $\mathbf{T}^{-1}$  je transformační matice pro převod z jednoho souřadného systému do druhého.



Obrázek 2.4: Souřadný systém kamery

Následně se zaměříme na digitální kamery. Ty mají obrazovou rovinu jako mřížku o velikosti  $W \times H$ . Mřížku tvoří na světlo citlivé částičky, které se anglicky nazývají *photosites*. Ty přímo odpovídají obrazovým elementům, nebo pixelům. Souřadnice pixelů je vektor o dvou složkách  $(u, v)$ , který obsahuje kladná celá čísla. Konvence dává, že počátek je v levém horním rohu obrazové roviny.

Pixely mají jednotnou velikost a jsou soustředěny na pravidelné mřížce, což znamená, že souřadnice pixelů souvisí se souřadnicemi obrazové roviny, jak je vidět ve vztahu (2.10) [3]

$$\begin{aligned} u &= \frac{x}{\rho_w} + u_0 \\ v &= \frac{y}{\rho_h} + v_0, \end{aligned} \quad (2.10)$$

kde  $\rho_w$  a  $\rho_h$  jsou šířka a výška jednotlivých pixelů.  $(u_0, v_0)$  je souřadnice pixelu tzv. hlavního bodu, kde optická osa protíná rovinu obrazu vzhledem k novému počátku. Rovnici projekce (2.6) lze zapsat pomocí souřadnic pixelů v matici  $\mathbf{K}$  následovným vztahem (2.11) [3]

$$\tilde{\mathbf{p}} = \underbrace{\begin{pmatrix} \frac{1}{\rho_w} & 0 & u_0 \\ 0 & \frac{1}{\rho_h} & v_0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{K}} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \mathbf{C} \tilde{\mathbf{P}}, \quad (2.11)$$

kde vektor  $\tilde{\mathbf{p}} = (\tilde{u}, \tilde{v}, \tilde{w})$  jsou homogenní souřadnice bodu  $\mathbf{P}$  zapsané pomocí pixelových souřadnic. Nehomogenní souřadnice jsou v následujícím vztahu (2.12) [3]

$$\begin{aligned} u &= \frac{\tilde{u}}{\tilde{w}} \\ v &= \frac{\tilde{v}}{\tilde{w}}. \end{aligned} \quad (2.12)$$

Spojením rovnic (2.8) a (2.11) můžeme zapsat projekci kamery v obecné formě (2.13) [3].

$$\tilde{\mathbf{p}} = \begin{pmatrix} \frac{f}{\rho_w} & 0 & u_0 \\ 0 & \frac{f}{\rho_h} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} ({}^0\mathbf{T}_C)^{-1} \tilde{\mathbf{P}} = \mathbf{K} \mathbf{P}_0 {}^0\mathbf{T}_C^{-1} \tilde{\mathbf{P}} = \mathbf{C} \tilde{\mathbf{P}}. \quad (2.13)$$

$\frac{f}{\rho_w}$  a  $\frac{f}{\rho_h}$  jsou ohniskové vzdálenosti v pixelech.  $\mathbf{C}$  je homogenní matice kamery o velikosti  $3 \times 4$ , která provádí škálování, translaci a perspektivní projekci. Díky tomuto je také matice  $\mathbf{C}$  často označována jako matice projekce, nebo jako matice kalibrace kamery.

Díky předchozímu vztahu a maticím můžeme zapsat projekci jako vztah (2.14) [3]

$$\mathbf{p} = P(\mathbf{P}, \mathbf{K}, \xi_C), \quad (2.14)$$

kde  $\mathbf{P}$  je bodový souřadnicový vektor v lokálním souřadném systému. Matice  $\mathbf{K}$  obsahuje vnitřní parametry kamery, což jsou vlastnosti kamery a snímače dané

výrobce. Jedná se o parametry  $f$ ,  $\rho_w$ ,  $\rho_h$ ,  $u_0$  a  $v_0$ .  $\xi_C$  popisuje polohu kamery. Obsahuje minimálně šest vnějších parametrů, které popisují posun kamery a její orientaci ve 3D prostoru.

Kameru lze popsat pomocí 11 nezávislých parametrů - 5 vnitřních a 6 vnějších. Matice kamery má 12 prvků, tudíž má jeden stupeň volnosti. Celkový škálovací faktor je neomezený a může být libovolně volen. V praxi parametry kamery neznáme, a proto musí být pomocí kalibrace kamery odhadnuty. Ještě než se zaměříme na kalibraci kamery je potřeba zmínit, že matice kamery  $\mathbf{C} \in R^{3 \times 4}$  má několik důležitých vlastností [3]:

- Lze matici  $\mathbf{C}$  rozdělit na  $\mathbf{C} = (\mathbf{M}|\mathbf{c}_4)$ , kde  $\mathbf{M} \in R^{3 \times 3}$  je nesingulární matice a vektor  $\mathbf{c}_4 = -\mathbf{M}\mathbf{c}$ . Vektor  $\mathbf{c}$  je lokální počátek v rovině kamery. Můžeme zpětně vyjádřit  $\mathbf{c} = -\mathbf{M}^{-1}\mathbf{c}_4$ .
- Nulový prostor matice  $\mathbf{C}$  je  $\mathbf{c}$ .
- Pixel na dané souřadnici  $p$  koresponduje s paprskem v prostoru rovnoběžném s vektorem  $\mathbf{M}^{-1}\mathbf{p}$ .
- Matice  $\mathbf{M} = \mathbf{K}\bar{\mathbf{R}}$  vychází z vnitřních vlastností matice a její inverzní orientace. Můžeme provést RQ-rozklad matice  $\mathbf{M} = \mathbf{R}\mathbf{Q}$ . Matice  $\mathbf{R}$  je horní trojúhelníková matice, odpovídá matici  $\mathbf{K}$ .  $\mathbf{Q}$  je ortogonální matice a odpovídá matici  $\bar{\mathbf{R}}$ .
- Poslední řádek matice  $\mathbf{C}$  definuje hlavní rovinu, která je rovnoběžná s rovinou obrazu a obsahuje počátek kamery.
- Jestliže jsou řádky matice  $\mathbf{M}$  vektory  $m_i$ , pak platí:
  - $\mathbf{m}_3^T$  je vektor, který je kolmý na hlavní rovinu a rovnoběžný s optickou osou.  $\mathbf{M}\mathbf{m}_3^T$  je hlavní bod v homogenní formě.
  - Pokud má kamera nulové zešíkmení, což je  $\mathbf{K}_{1,2} = 0$ , pak platí  $(m_1 \times m_3)(m_2 \times m_3) = 0$ .
  - Pokud má kamera čtvercové pixely, což je  $\rho_u = \rho_v$ , pak platí  $\|m_1 \times m_3\| = \|m_2 \times m_4\| = \frac{f}{\rho}$ .

Zorné pole kamery je funkcí její ohniskové vzdálenosti  $f$ . Ta závisí na typech objektivu kamery - širokoúhlý objektiv má malou ohniskovou vzdálenost, teleobjektiv má velkou ohniskovou vzdálenost a zoom objektiv má nastavitelnou ohniskovou vzdálenost. Zorné pole kamery je dáno geometrií. Ve vodorovném směru je poloviční úhel pohledu, viz vztah (2.15) [3]

$$\frac{\Theta}{2} = \tan^{-1} \frac{W}{2f}, \quad (2.15)$$

kde  $W$  je počet pixelů ve vodorovném směru. Můžeme tedy psát vztah (2.16) [3].

$$\begin{aligned} \Theta_h &= 2 \tan^{-1} \frac{W\rho_w}{2f} \\ \Theta_v &= 2 \tan^{-1} \frac{H\rho_h}{2f} \end{aligned} \quad (2.16)$$

Je potřeba dodat, že zorné pole kamery je též funkcí dimenzí čipu kamery, který je  $W\rho_w \times H\rho_h$ .

Dále je potřeba se zaměřit na problém zkreslení čočky kamery. Žádná čočka kamery není dokonalá. Nedokonalosti čočky mají za následek různá zkreslení, např. chromatické aberace (barevné lemování), sférické aberace, astigmatismus (kolísání ohniska během scény) či geometrická zkreslení.

Geometrické zkreslení je nejzávažnější problém, se kterým se setkáváme u robotických aplikací. Má dvě složky - radiální a tangenciální. Radiální zkreslení způsobuje, že se body obrazu posunou podél radiálních čar od hlavního bodu. Radiální error se dobře aproximuje polynomem (2.17) [3]

$$\delta r = k_1 r^3 + k_2 r^5 + k_3 r^7 + \dots, \quad (2.17)$$

kde  $r$  je vzdálenost bodu obrazu od hlavního bodu. Dále dochází k soudkovitému zkreslení, když se zvětšení snižuje se vzdáleností od hlavního bodu, což způsobuje zakřivení přímých čar poblíž okraje obrazu směrem ven. Dalším zkreslením je tzv. poduškovité. Dochází k němu, když se se zvětšením zvyšuje vzdálenost od hlavního bodu a způsobí, že se rovné čáry poblíž okraje obrazu zakříví dovnitř. Dále existuje již zmíněné tangenciální (decentralizační) zkreslení, jež se vyskytuje v pravém úhlu k poloměru. Je méně významné, než radiální zkreslení.

Ve vztahu (2.18) jsou souřadnice bodu  $(u, v)$  po zkreslení [3]

$$u^d = u + \delta_u, v^d = v + \delta_v, \quad (2.18)$$

kde posun je dán vztahem (2.19) [3]

$$\begin{pmatrix} \delta_u \\ \delta_v \end{pmatrix} = \begin{pmatrix} u(k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots) \\ v(k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots) \end{pmatrix} + \begin{pmatrix} 2p_1 uv + p_2(r^2 + 2u^2) \\ p_1(r^2 + 2v^2) + 2p_2 uv \end{pmatrix}. \quad (2.19)$$

Tento vektor je možné vykreslit pro různé hodnoty  $(u, v)$ . Vektory znázorňují posunutí, jenž je potřebné pro korekci zkreslení v různých bodech obrazu  $(-\delta_u, -\delta_v)$  a ukazují dominantní radiální zkreslení. V praxi stačí k popisu radiálního zkreslení tři koeficienty. Model zkreslení je pak parametrizován pomocí  $(k_1, k_2, k_3, p_1, p_2)$ . Tyto parametry jsou pak považovány za vnitřní parametry kamery.

Tím se dostáváme ke kalibraci kamery. Jak už zde bylo zmíněno, kalibrace kamery odhaduje parametry kamery jak ty vnitřní, tak i ty vnější, viz vztah (2.13) [3]. Obecně se hlavní bod nenachází v centru obrazové roviny. Dalším aspektem je, že ohnisková vzdálenost je přesná jen ze 4% z toho, kolik se udává. Ohnisková vzdálenost je přesná jen tehdy, když je čočka zaostřena v nekonečno. Komplikace je také v tom, že vnitřní parametry se mění, pokud je čočka odpojována a znovu připojena, nebo se nějakým způsobem upravila pro zaostřování, či clonění. Jediné dva vnitřní parametry, které lze získat předem a to napsané v manuálu od výrobce, jsou  $\rho_w$  a  $\rho_h$ . Vnější parametr, pozice kamery, vytváří problém udání, kde je střed kamery.

Metoda homogenní transformace umožňuje odhadovat parametry matice  $\mathbf{C}$ , viz vztah (2.13) [3]. Prvky této matice jsou funkcemi vnitřních a vnějších parametrů. Nastavíme  $\mathbf{p} = (u, v, 1)$ , tím rozšíříme rovnici (2.13) a dosadíme ji do rovnice (2.12) [3]. Získáme rovnice (2.20) [3].

$$\begin{aligned} C_{11}X + C_{12}Z + C_{13}Z + C_{14} - C_{31}uX - C_{32}uY - C_{33}uZ - C_{34}u &= 0 \\ C_{21}X + C_{22}Z + C_{23}Z + C_{24} - C_{31}vX - C_{32}vY - C_{33}vZ - C_{34}v &= 0 \end{aligned} \quad (2.20)$$

Zde vektor  $(u, v)$  obsahuje pixelové souřadnice odpovídající bodu v lokálním souřadném systému  $(X, Y, Z)$  a  $C_{ij}$  jsou neznámé prvky matice kamery.

Kalibrace požaduje 3D souřadný systém  $\mathbf{T}$ . Poloha středu každé značky  $(X_i, Y_i, Z_i)$ ,  $i \in \langle 1, N \rangle$ , vzhledem k souřadnému systému  $\mathbf{T}$  musí být známá. Po získání snímku jsou určeny odpovídající souřadnice obrazové roviny  $(u_i, v_i)$ . Pokud nastane  $C_{34} = 1$ , skládáme rovnice (2.20) pro každou  $N$ -tou značku do maticové rovnice (2.21) [3]

$$\begin{pmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 \\ & & & & & & & & & & \\ & & & & & & & & & & \\ X_N & Y_N & Z_N & 1 & 0 & 0 & 0 & 0 & -u_NX_N & -u_NY_N & -u_NZ_N \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -v_NX_N & -v_NY_N & -v_NZ_N \end{pmatrix} \begin{pmatrix} C_{11} \\ C_{12} \\ \vdots \\ C_{33} \end{pmatrix} = \begin{pmatrix} u_1 \\ v_1 \\ \vdots \\ u_N \\ v_N \end{pmatrix}. \quad (2.21)$$

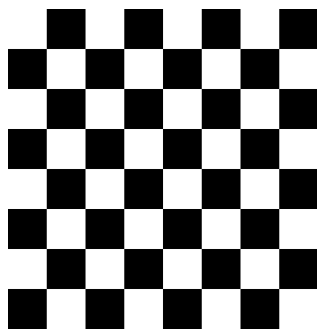
Touto rovnicí lze získat prvky matice  $C_{11} \dots C_{33}$ . Tato rovnice má 11 neznámých a pro zjištění řešení je zapotřebí, aby  $N \geq 6$ . V praxi je ale často využíváno víc než šest bodů, což vede na řešení pomocí metody nejmenších čtverců. Pokud jsou body koplanární, pak levá matice nemá dostatečnou hodnotu.

Lineární techniky, např. která je popsána výše, nedokáží odhadnout parametry zkreslení čočky. Zkreslení ovlivní prvky matice kamery, ale ve většině situací toto

## 2. Zpracování obrazu pro detekci objektů v prostoru

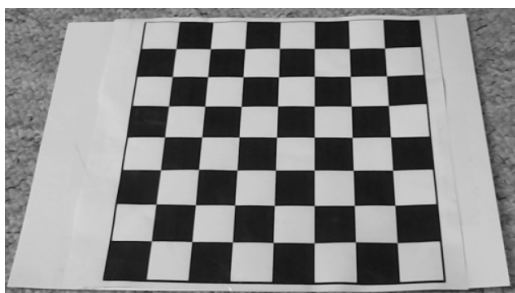
ovlivnění není nijak vysoké. Parametry zkreslení se často odhadují pomocí nelineární optimalizace přes všechny parametry, typicky 16 nebo více, přičemž lineární řešení se používá jako počáteční odhad parametru.

Zvolená metoda kalibrace byla pomocí šachovnice o velikosti  $7 \times 7$ , viz obr. 2.5.

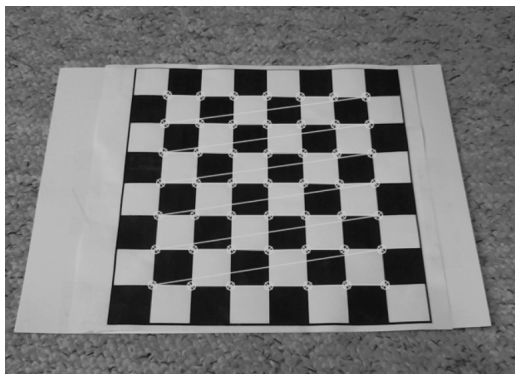


Obrázek 2.5: Šachovnice o velikosti  $7 \times 7$

Na obr. 2.6 se nachází náhled na snímání šachovnice a na obr. 2.7 jsou detekovány její vnitřní rohy.



Obrázek 2.6: Snímání šachovnice



Obrázek 2.7: Detekované vnitřní rohy šachovnice



## 2.3 Implementace v OpenCV

Zdrojový kód 2.1 popisuje kalibraci kamery v jazyce C++. Kalibrace je prováděna pomocí knihovny OpenCV, viz dodatek A.

Ze všeho nejdříve jsou definovány proměnné a jedna globální proměnná, která definuje šachovnici `int CHECKERBOARD[2][7,7]`. Šachovnice má 7 horizontálních a 7 vertikálních vnitřních vrcholů. Program následně zjistí, zda je připojena kamera. Pokud kamera připojena není, vyhodí informační hlášku a skončí. Pokud kamera připojena je, spustí se cyklus, ve kterém se obraz nejprve převede do stupnice šedi. Využila se pro tento krok metoda `cvtColor(origin_image, new_image, COLOR_BGR2GRAY)` z knihovny OpenCV. Dále se metodou `findChessboardCorners` zjistí, zda se ve zorném poli kamery nachází šachovnice [4]. Pokud je šachovnice nenalezena, program opakuje hledání šachovnice. Když program 60× zasebou neidentifikuje šachovnici, program se vypne. Avšak pokud je šachovnice detekována, ukládají se jednotlivé vnitřní rohy. Až tento proces proběhne tolikrát, kolik udává proměnná `limit`, kamera se začne kalibrovat pomocí metody `calibrateCamera`. Po kalibraci se přechází k dalšímu kroku a to k detekci markeru, které je popsána zde 3.1.

Zdrojový kód 2.1: Část programu pro kalibraci kamery

```
int CHECKERBOARD[2][7,7];

int main() {
    // kamera
    Mat image;
    bool found = false;

    //kalibrace kamery
    Mat cameraMatrix = Mat(3, 3, CV_64F);
    cameraMatrix.ptr<float>(0)[0] = 1;
    cameraMatrix.ptr<float>(1)[1] = 1;
    Mat distCoeffs, R, T;
    vector<Point2f> imageCorners;
    vector<std::vector<Point3f>> objpoints;
    vector<std::vector<Point2f>> imgpoints;
    vector<Point3f> objp;
    for (int i = 0; i < CHECKERBOARD[0]; i++) {
        for (int j = 0; j < CHECKERBOARD[1]; j++) {
            objp.push_back(Point3f(j, i, 0));
        }
    }
    int maximum = 1;
    int limit = 61;
    bool kalibrace = false;
    int stop = 0;

    namedWindow("Display window");
    VideoCapture cap(1); //0 - webkamera, 1 - externi kamera
```

## 2. Zpracování obrazu pro detekci objektů v prostoru

---

```
if (!cap.isOpened()) {
    std::cout << "Cannot open camera." << endl;
}
else {
    while (true) {
        cap >> image;
        cvtColor(image, image, COLOR_BGR2GRAY);
        found = findChessboardCorners(image,
                                     Size(CHECKERBOARD[0],
                                           CHECKERBOARD[1]),
                                     imageCorners);
        if (found && maximum < limit) {
            cout << "ANO" << endl;
            cornerSubPix(image,
                        imageCorners,
                        Size(11, 11),
                        Size(-1, -1),
                        TermCriteria(0, 30, 0.001));
            drawChessboardCorners(image,
                                  Size(CHECKERBOARD[0], CHECKERBOARD
                                       [1]),
                                  imageCorners,
                                  found);
            objpoints.push_back(objp);
            imgpoints.push_back(imageCorners);
            maximum++;
            stop = 0;
        }
        else if (found == false && maximum < limit) {
            std::cout << "NE" << endl;
            stop++;
        }
        if (stop == 50) {
            break;
        }
        else if (maximum >= limit && kalibrace == false) {
            calibrateCamera(objpoints,
                            imgpoints,
                            Size(image.rows, image.cols),
                            cameraMatrix,
                            distCoeffs,
                            R,
                            T,
                            CALIB_USE_LU);
            kalibrace = true;
        }
        if (kalibrace) {
            cout << "DETEKCE" << endl;
        }
    }
}
return 0;
}
```

# Metody detekce objektu v obraze s využitím fiduciálních značek

## 3

Fiduciální značky jsou uměle vytvořené obrazce, které se dají snadno rozpoznat a odlišit od sebe. Jedná se o 2D systém čárových kódů. Fiduciální značky obsahují informaci nenáročnou na paměť, obvykle 12 bitů a jsou navrženy tak, aby byly okamžitě detekovány a lokalizovány i přes nízké rozlišení kamery, špatné osvětlení, jiné natočení značky, či nevhodné umístění značky v prostoru [5]. Jejich detekci umožňuje jejich jedinečný vzhled. Existuje několik druhů fiduciálních značek. V této kapitole budou posány tři druhy: AprilTag markery, STag markery a ArUco markery. Bude se v nich probírat jejich vzhled, důvod, proč byly vytvořeny, a také jejich výhody, díky nimž jsou oblíbené. Zmíněné markery se hojně využívají v robotice, počítačovém vidění, či rozšířené realitě. V robotice je robot pomocí na něm umístěným markeru lokalizován a detekován.

Následně se zde bude probírat detekce ArUco markeru. Popíše se zde princip detekce, která probíhá po zkalibrování kamery. Nejdříve se popíše, jak se dojde k zjištění, že se jedná o marker. Další část bude obsahovat informace o tom, jak se zjistí ID ArUco markeru. Poslední část kapitoly je věnována samotné implementaci detekce ArUco markeru pomocí knihovny OpenCV.

## 3.1 Příklady druhů fiduciálních značek

### 3.1.1 AprilTag markery

AprilTags vychází z ARToolkit markerů, příklad na obr. 3.1 [5]. ARToolkit byl software vyvinutý v roce 1999 pro rozšířenou realitu a využíval fiduciální markery k přidávání virtuálních objektů do reálného prostředí. ARToolkit markery byly původně ve tvaru černobílých čtverců s jednoduchými vzory uvnitř. Tyto markery

### 3. Metody detekce objektu v obraze s využitím fiduciálních značek

byly náchylné k chybám při detekci a nebyly tak robustní pro různé deformace a zkreslení.



Obrázek 3.1: Příklad ARToolkit markeru

AprilTags markery byly vytvořeny jako zdokonalení původních ARToolkit markerů. Jejich design se zakládá na binárních Hammingových kódech, které poskytují vyšší robustnost a spolehlivost při detekci a identifikaci markerů. Tato nová podoba umožnila AprilTags lépe odolávat různým deformacím, rotaci a zkreslení. AprilTags byly také navrženy s ohledem na rychlost zpracování a efektivitu detekce, což umožňuje jejich použití v reálném čase [6].

Obvykle mají černobílý čtvercový vzor s vyplněným centrálním čtvercem, viz obr. 3.2 [5]. Celý vzor je rozdělen na menší čtvercové segmenty nebo čtvercové bity, které se mohou buď vyplnit (černá) nebo ponechat prázdné (bílá). Kombinace těchto bitů tvoří jedinečný identifikátor, který lze použít k identifikaci a sledování daného markeru.



Obrázek 3.2: Příklad AprilTag markeru

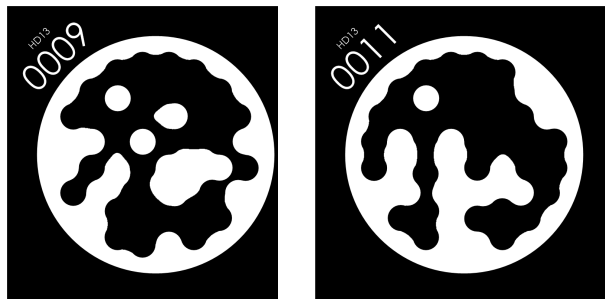
AprilTags mají několik výhod ve srovnání s jinými metodami vizuálního sledování a lokalizace [5][6]:

- AprilTags jsou relativně jednoduché a snadno rozpoznatelné vizuální markery. Jejich vzory jsou navrženy tak, aby byly snadno detekovatelné a identifikovatelné pomocí počítačového vidění.
- AprilTags jsou navrženy tak, aby byly robustní vůči různým deformacím a zkreslením. Mohou být úspěšně detekovány a identifikovány i při rotaci, škálování a perspektivní zkreslení. To zajišťuje spolehlivost a přesnost při sledování objektů.
- Každý AprilTag má jedinečný identifikátor, který lze použít k jednoznačné identifikaci a sledování daného markeru. To je užitečné při lokalizaci objektů v prostoru a při rozpoznávání jednotlivých markerů.
- Detekce a identifikace AprilTags mohou být prováděny v reálném čase, což je důležité pro aplikace, které vyžadují okamžité zpracování obrazu a rychlé rozhodování.
- AprilTags mohou být tisknuty nebo vygenerovány na různých médiích a mohou být přizpůsobeny potřebám konkrétního systému. Lze je také použít v různých prostředích a podmínkách osvětlení.

## 3.1.2 STag markery

STag markery byly opět navrženy jako vylepšení ARToolkit markerů. Jedná se o speciální vizuální markery využívané v oblasti počítačového vidění a rozšířené reality. STag markery jsou jedním z mnoha typů fiduciálních značek, které se používají k identifikaci a sledování objektů v reálném světě pomocí kamery. Jsou tvořeny čtvercovou mřížkou o pevně stanovené velikosti, která je vyplněna specifickým vzorem černých a bílých čtverečků. Tento vzor je navržen tak, aby byl snadno rozpoznatelný a dekodovatelný pomocí algoritmů počítačového vidění.

Každý marker obsahuje také integrované číslo nebo identifikátor, který slouží k jednoznačné identifikaci konkrétního markeru, viz obr. 3.3 [7]. Tato identifikace se provádí na základě rozložení černých a bílých čtverečků ve vzoru. Díky jedinečnému vzoru a identifikátoru umožňují STag markery přesné sledování polohy a orientace objektů v prostoru.



Obrázek 3.3: Ukázkové příklady STag markerů z knihovny HD13

STag markery mají několik výhod v oblasti počítačového vidění a rozšířené reality, jako jsou hry s rozšířenou realitou, virtuální navigace, vzdělávání, robotika a mnoho dalších [8]:

- Každý STag marker obsahuje unikátní identifikátor, který umožňuje jednoznačnou identifikaci konkrétního markeru. To je užitečné při sledování a identifikaci objektů v reálném světě, kde je potřeba rozlišit jednotlivé markery od sebe.
- STag markery umožňují přesné sledování polohy a orientace objektů v prostoru. Díky svému vzoru a identifikátoru umožňují algoritmům počítačového vidění přesně určit polohu a pohyb markeru v reálném čase.
- STag markery jsou navrženy tak, aby byly odolné vůči šumu a deformacím. Díky svému specifickému vzoru a optimalizovanému dekodování jsou schopny přesně fungovat i v neideálních podmínkách, jako je špatné osvětlení, částečná zakrytí nebo šum v obraze.
- STag markery mohou být tištěny na papír, nalepeny na objekty nebo integrovány přímo do designu objektu. To znamená, že mohou být použity v různých prostředích a aplikacích, od her a interaktivního umění po průmyslové aplikace a robotiku.

### 3.1.3 ArUco markery

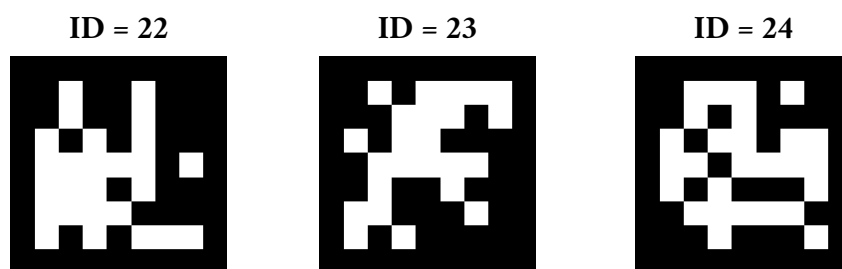
Důvod vývoje ArUco markerů byl ovlivněn několika faktory [1][2][4]:

- ArUco markery jsou navrženy specificky pro potřeby rozšířené reality. Výzkumníci a vývojáři se snažili vytvořit jednoduchý a efektivní způsob detekce a sledování objektů v reálném čase, což je klíčovým aspektem technologií s rozšířenou realitou.
- ArUco markery využívají principy počítačového vidění, které se zabývá zpracováním obrazu a rozpoznáváním vzorů. Vytvoření detekovatelného a dekodovatelného markeru vyžaduje vhodné algoritmy a metody pro extrakci a interpretaci vzoru.
- Vizualní kódy jsou speciální kódy, které mohou být detekovány a rozpoznány pomocí počítačového vidění. ArUco markery jsou jedním z typů vizuálních kódů, které se zaměřují na rozpoznávání specifických vzorů pro identifikaci a určení polohy.
- QR kódy byly jedním z hlavních zdrojů inspirace pro vývoj ArUco markerů, což jsou široce používané čárové kódy, jenž mají schopnost rychlého čtení a dekodování. Výzkumníci chtěli vytvořit podobně efektivní systém pro detekci a sledování, avšak s jednodušší strukturou a vyšší flexibilitou.

Spojením těchto faktorů a snahou o vytvoření jednoduchých, robustních a rychlých markerů pro rozšířenou realitu a počítačové vidění byly vyvinuty ArUco markery. Vývoj ArUco markerů je součástí neustálého pokroku v oblasti vizuálního zpracování a rozšířené reality.

ArUco markery jsou tedy speciální vizuální kódy používané pro detekci a sledování v oblasti rozšířené reality a počítačového vidění. Tyto markery jsou tvořeny černobílými čtvercovými vzory, které mají specifickou strukturu a kódování. Typicky se jedná o čtvercové značky, které mají široký černý okraj a vnitřní binární matici (bílé vzory). Matice/vzory mají různé tvary a konfigurace, které slouží k identifikaci markeru (určení jeho ID) a umístění a natočení v prostoru. ArUco markery mají různé velikosti a rozlišení, což závisí na konkrétních potřebách a použití. Existuje mnoho různých typů ArUco markerů, které se liší tvarem, velikostí a počtem černých čtverců.

Různé druhy ArUco markerů jsou uloženy v několika knihovnách. Za použití knihovny OpenCV jsem vygenerovala tři různé markery z knihovny ArUco markerů DICT\_7x7\_250, které jsou na obr. 3.4.



Obrázek 3.4: Ukázkové příklady ArUco markerů z knihovny DICT\_7x7\_250

ArUco markery mají několik výhod, které přispívají k jejich širokému využití v oblasti rozšířené reality a počítačového vidění [1][2][4]:

- ArUco markery jsou relativně jednoduché a levné na vtištění nebo vygenerování. Jejich jednoduchá struktura se skládá z černobílých čtvercových vzorů, což usnadňuje jejich detekci a dekodování.
- Každému ArUco markeru je přiřazen unikátní identifikátor (ID), který je zakódován v černobílém vzoru markeru. Tím umožňují jednoduchou identifikaci a rozlišení mezi různými markery.
- ArUco markery jsou navrženy tak, aby byly odolné vůči různým podmínkám osvětlení. Díky kontrastu mezi černými čtverci a bílým pozadím jsou dobře viditelné a detekovatelné i při různých úrovních osvětlení. ArUco marker musí mít pro správnou detekci bílý rám.
- ArUco markery umožňují rychlou a efektivní detekci a sledování. Díky speciálnímu kódování a struktuře markerů je jejich detekce a rozpoznání velmi rychlé, což je výhodné pro interaktivní aplikace rozšířené reality.
- ArUco markery mohou být použity ve velkém měřítku, ať už samostatně nebo v kombinaci s jinými markery. Mohou být umístěny na různých objektech, plochách nebo scénách, což umožňuje širokou škálu aplikací v oblasti rozšířené reality, robotice, počítačového vidění a dalších.

## 3.2 Detekce ArUco markeru

Nejdříve se v obraze musí detekovat čtverce. Nejvhodnější metodou je tzv. Houghova transformace. Slouží k detekci různých geometrických tvarů v obrazech, především přímk [1]. Zde je rovnice pro detekci přímek

$$\rho = x \cos(\Theta) + y \sin(\Theta), \quad (3.1)$$



kde  $\rho$  je vzdálenost od počátku souřadného systému  $(0, 0)$  k přímkce a  $\Theta$  je úhel mezi normálou přímky a osou  $x$ . Jedná se o parametrickou rovnici přímky a též ji lze nazývat jako Hesseho normální tvar. Při použití Houghovy transformace se vytváří Houghův prostor o velikosti odpovídající parametrům  $\rho$  a  $\Theta$ .

ArUco markery avšak nejsou ve tvaru přímek, ale čtverců. Proto je potřeba použít rozšířenou rovnici Houghovy transformace o to pro detekci obdélníků. Detekce probíhá následovně [1]:

1. Pro každý pixel v obraze se provede analýza hran, např. pomocí Cannyho detektoru, který je popsán v dodatku B.
2. Pro každý hranový pixel se vygenerují hypotézy čtverců tak, že se projdou všechny dvojice bodů na hraně a vyhodnotí se, zda by mohly tvořit strany čtverce.
3. Pro každou hypotézu čtverce se vyhodnotí, zda existují další body na hraně, které by se mohly nacházet na příslušných stranách čtverce.
4. Pokud jsou splněny předem stanovené podmínky pro velikost a tvar stran čtverce, je tato hypotéza považována za detekci čtverce.
5. Vytvoří se Houghův prostor se dvěma rozměry: délka stran čtverce, např.  $a$ ,  $b$  a úhel náklonu stran čtverce, např.  $\Theta$ . Každý detekovaný čtverec přispívá do Houghova prostoru svým hlasováním.
6. V Houghově prostoru se hledají maximální hlasovací body, které indikují detekované čtverce.

Je důležité poznamenat, že Houghova transformace pro detekci obdélníků je složitější než standardní Houghova transformace pro detekci přímek. Tato transformace zahrnuje kombinaci různých podmínek na strany a tvary čtverce.

Dalším způsobem pro detekci čtverců v obraze je detekce hran, např. pomocí již zmíněného Cannyho detektoru. Následně jsou detekovány rohy, jedna z metod pro detekci rohů je zmíněná v dodatku B. K tomuto účelu lze též využít Harrisův detektor rohů. Tato metoda, která je popsána v knize *Robotics, Vision and Control* [3], je založena na analýze změn intenzity v okolí každého pixelu v obraze. Harrisův rohový podíl je vypočítán pomocí strukturální matice, která se skládá z derivací intenzity v obou směrech. Pro každý pixel o souřadnicích  $(x, y)$  v obraze se vypočítá strukturální matice  $\mathbf{M}$  pomocí vztahu (3.2) [3]

$$\mathbf{M} = \begin{pmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{pmatrix}, \quad (3.2)$$

kde  $I_x$  je derivace intenzity v horizontálním směru a  $I_y$  je derivace intenzity ve vertikálním směru.

Dále se pro každý pixel o souřadnicích  $(x, y)$  vypočítá Harrisův rohový podíl, viz vztah (3.3) [3]

$$R = \det(\mathbf{M}) - k \cdot \text{trace}(\mathbf{M})^2, \quad (3.3)$$

kde  $\det(\mathbf{M})$  je determinant matice  $\mathbf{M}$ ,  $\text{trace}(\mathbf{M})$  je součet prvků na hlavní diagonále matice  $\mathbf{M}$  a  $k$  je empiricky zvolená konstanta, často volená v intervalu  $\langle 0,04; 0,06 \rangle$ .

Hodnota Harrisova rohového podílu  $R$  pro každý pixel představuje míru změny intenzity v okolí daného pixelu. Vyšší hodnota  $R$  indikuje vyšší pravděpodobnost, že pixel představuje roh. Detekce rohů se provádí nalezením maximálních hodnot Harrisova rohového podílu v obraze a jejich prahováním strukturální matice  $R$ .

Jinou možností detekce rohů je využití Shi-Tomasiho detektor rohů [3]. Jedná se o lepší verzi Harrisova detektoru rohů. Využívá minimální vlastní číslo ze spektra strukturální matice pro detekci rohů. Strukturální matice  $\mathbf{M}$  je (3.2) [3]. Poté se pro každý pixel o souřadnicích  $(x, y)$  vypočítají vlastní čísla  $\lambda_1$  a  $\lambda_2$  matice  $\mathbf{M}$ . Tyto vlastní čísla jsou řešením charakteristické rovnice

$$\det(\mathbf{M} - \lambda \mathbf{I}) = 0, \quad (3.4)$$

kde  $\mathbf{I}$  je identitní matice a  $\lambda$  je vlastní číslo.

Dále se pro každý pixel o souřadnicích  $(x, y)$  vypočítá Shi-Tomasiho rohový podíl, který je dán vztahem (3.5) [3]

$$R = \min(\lambda_1, \lambda_2). \quad (3.5)$$

Vyšší hodnota  $R$  indikuje vyšší pravděpodobnost, že pixel představuje roh. Detekce rohů se provádí nalezením maximálních hodnot Shi-Tomasiho rohového podílu v obraze a jejich prahováním.

Detekovali jsme rohy a následuje rohová lokalizace. Nejdříve se pomocí prahování (posáno v kapitole 2.1) odstraní slabé rohové body. Následuje suprese nejvyšších hodnot, která potlačuje rohové body s nejvyšší hodnotou v daném okolí. Aby se dosáhla vyšší přesnosti lokalizace rohů, tak se využívá metoda subpixelové přesnosti. Interpoluje se poloha rohu na subpixelovou úroveň. Nejběžněji používanou metodou pro subpixelovou interpolaci je např. algoritmus nejmenších čtverců.

Pro konečnou rohovou lokalizaci se provádí vyhodnocení geometrických kritérií. Jedná se o důležitý krok při lokalizaci rohů v obraze. Je potřeba zajistit, že nalezené body skutečně představují geometrické rohy a splňují požadovaná kritéria. Zde jsou jednotlivé kroky, které je potřeba provést [3]:

- Jedním z nejběžnějších kritérií je kontrola blízkosti rohů k sobě. Pokud jsou dva rohy příliš blízko sebe, mohou být považovány za jediný roh nebo falešnou detekci. V tomto případě se vybírají pouze ty rohy, které jsou dostatečně vzdálené od sebe.
- Dalším kritériem je kontrola tvaru rohů. Obvykle se očekává, že roh bude mít tvar pravého úhlu. Proto se provádí analýza úhlů mezi hranami, které se sbíhají v daném rohu. Pokud úhly odpovídají přibližně pravému úhlu, pak je roh považován za validní.
- Pro některé aplikace je důležité zjistit, zda jsou rohy stabilní a nejsou náchylné k chybám nebo fluktuacím. To se provádí analýzou změny polohy rohů v různých obrazech nebo snímcích. Pokud se poloha rohu příliš mění nebo se vyskytují výrazné fluktuace, může být považován za nestabilní nebo neplatný.
- V závislosti na konkrétní aplikaci mohou existovat další specifická kritéria, která se používají pro vyhodnocení rohů. Například pro určité geometrické tvary (např. kruh, elipsa) mohou být definovány specifické vlastnosti, které musí roh splňovat. Také se mohou použít statistické metody pro analýzu vlastností rohů ve větší souboru obrazů.

Cílem vyhodnocení geometrických kritérií je zajistit, že nalezené body představují skutečné rohy s požadovanými vlastnostmi. Tím se minimalizuje počet falešných detekcí a zajišťuje se spolehlivost a přesnost detekce rohů v obraze. Po nalezení čtvercových tvarů se provádí další analýza pro rozpoznání ArUco markerů. Máme skupinu čtverců, které představují potenciální ArUco markery. Nejdříve se segmentuje marker od okolního prostředí, pro to je využita metoda prahování, která je popsána v sekci Teorie zpracování obrazu 2.1. Dále je potřeba extrahovat vnitřní vzor markeru, který je potřeba k jeho identifikaci a následné určení jeho ID.

Zde je popsán způsob extrakce vnitřního vzoru [1][4]:

1. Nejprve je nutné definovat rohy nalezeného čtvercového tvaru, který představuje potenciální ArUco marker. To se obvykle provádí identifikací nejvzdálenějších čtyř bodů, které tvoří čtvercový tvar.
2. Pro správnou extrakci vnitřního vzoru je často potřeba provést korekci perspektivy, aby se zajistila rovnoběžnost čtverců v markeru. Tím se zajišťuje, že vnitřní vzor bude odpovídat očekávanému vzoru černých a bílých čtverců.
3. Následuje rozdělení korektně perspektivně upraveného markeru na mřížku čtverců na základě známého rozložení čtverců v ArUco markeru. Vnitřní vzor ArUco markeru je typicky čtvercová mřížka o určité velikosti.

4. Každý čtverec v rozděleném vnitřním vzoru je analyzován s cílem získat informaci o jeho intenzitě pixelů. Bílé čtverce mají obvykle vyšší intenzitu než černé čtverce. Tímto způsobem se určuje binární vzor, který představuje jedinečnou kombinaci černých a bílých čtverců v ArUco markeru.
5. Nakonec je binární vzor transformován na unikátní identifikátor (ID) ArUco markeru. Tento ID je často reprezentován celým číslem a přiřazen jednomu konkrétnímu markeru v databázi.

Extrakce vnitřního vzoru ArUco markeru je klíčovým krokem pro rozpoznání a identifikaci markeru. Správná extrakce zajišťuje, že se získá správný vzor černých a bílých čtverců, který se následně porovnává s databází ArUco markerů pro identifikaci konkrétního markeru.

Jakmile je vnitřní vzor extrahován, provádí se porovnání s předdefinovanou databází ArUco markerů. Nejprve je nutné načíst databázi vzorů, která obsahuje unikátní vzory pro všechna možná ID markery. Tato databáze může být vytvořena ručně nebo generována pomocí knihoven či nástrojů pro tvorbu ArUco markerů. Extrahovaný vzor vnitřního vzoru ArUco markeru je porovnán s každým vzorem v databázi. Existuje několik metod pro porovnání vzorů. Jedna z nich je Hammingova vzdálenost. Ta se používá k porovnání dvou binárních vzorů a vyjadřuje počet různých bitů mezi těmito vzory. Pro dva vzory  $A$  a  $B$  o stejné délce  $n$  je rovnice Hammingovy vzdálenosti (3.6) [1]

$$d(A, B) = \sum_{i=1}^n (A[i] \oplus B[i]), i \in \langle 1, n \rangle, \quad (3.6)$$

kde  $d(A, B)$  je Hammingova vzdálenost mezi vzory  $A$  a  $B$ ,  $A[i]$  a  $B[i]$  jsou jednotlivé bity na pozici  $i$  v jednotlivých vzorech.

Další metodou pro porovnání je např. korelace nebo porovnání vzájemné informace. Tyto metody kvantifikují podobnost mezi vzory a generují hodnotu, která vyjadřuje míru podobnosti. Po provedení porovnání s celou databází se vybere nejlepší výsledek, tj. vzor s nejvyšší mírou podobnosti k extrahovanému vzoru. Tento výsledek může být interpretován jako identifikační číslo (ID) ArUco markeru. Kromě přímého porovnání se často používá prahování pro rozhodnutí o shodě mezi extrahovaným vzorem a databází vzorů. Je stanoven práh nebo akceptační mez, při které je vzor považován za shodující se s databází. Pokud míra podobnosti překročí tento práh, je vzor přijat jako shodný.

Porovnání vzoru s databází je kritickým krokem pro správnou identifikaci ArUco markeru. Zajišťuje, že extrahovaný vzor je srovnáván se správnými vzory a výsledkem je přiřazení konkrétního ID markeru. Tento proces umožňuje aplikaci rozší-

řené reality nebo dalším systémům pracovat s konkrétním markerem a provádět příslušné akce.

### 3.2.1 Implementace v OpenCV

Kód 3.1 popisuje detekci ArUco markeru. Knihovna OpenCV, viz dodatek A, má aruco modul, který jsme využili. Nejdříve jsme definovali slovník a vygenerovala z něj několik ArUco markerů [4] pomocí metody `generovaniMarkeru`. Tento slovník jsme využili pak i při detekci. Ta probíhá po kalibraci kamery pomocí metody `detectMarkers` a využije se v ní zjištěná matice kamery [4]. Metodou `drawDetectedMarkers` se v obraze zvýrazní detekované markery a jejich horní levý roh a také se vypíše k nim jejich ID.

Zdrojový kód 3.1: Část programu pro generování a detekci ArUco markeru

```
void generovaniMarkeru(Ptr<aruco::Dictionary> dictionary) {
    Mat marker32, marker33, marker34, marker22, marker23, marker24;
    aruco::drawMarker(dictionary, 32, 200, marker32, 1);
    aruco::drawMarker(dictionary, 33, 200, marker33, 1);
    aruco::drawMarker(dictionary, 34, 200, marker34, 1);
    aruco::drawMarker(dictionary, 22, 200, marker22, 1);
    aruco::drawMarker(dictionary, 23, 200, marker23, 1);
    aruco::drawMarker(dictionary, 24, 200, marker24, 1);
    imshow("Marker32", marker32);
    waitKey(25);
    imshow("Marker33", marker33);
    waitKey(25);
    imshow("Marker34", marker34);
    waitKey(25);
    imshow("Marker22", marker22);
    waitKey(25);
    imshow("Marker23", marker23);
    waitKey(25);
    imshow("Marker24", marker24);
    waitKey(25);
    imwrite("marker32.png", marker32);
    imwrite("marker33.png", marker33);
    imwrite("marker34.png", marker34);
    imwrite("marker22.png", marker22);
    imwrite("marker23.png", marker23);
    imwrite("marker24.png", marker24);
}

int main() {
    //ágenerovn ArUco úmarker
    Ptr<aruco::Dictionary> dictionary =
        aruco::getPredefinedDictionary(aruco::DICT_6X6_250);
    generovaniMarkeru(dictionary);

    //detekce markeru
    Ptr<DetectorParameters> parameters = DetectorParameters::create();
    vector<vector<Point2f>> markerCorners, rejectedCandidates;
    vector<int> markerId
    Mat rvec, tvec, rmat;
    float velikost_markeru = 0.058; //metry
}
```

### 3. Metody detekce objektu v obraze s využitím fiduciálních značek

```
//0.058 //0,127 //0.05

namedWindow("Display window");
VideoCapture cap(1); //0 - webkamera, 1 - iextern kamera

if (!cap.isOpened()) {
    std::cout << "" << endl;
    std::cout << "Cannot open camera." << endl;
}
else {
    while (true) {
        cap >> image;
        cvtColor(image, image, COLOR_BGR2GRAY);
        ...
        if (kalibrace) {
            cout << "DETEKCE" << endl;
            aruco::detectMarkers(image,
                                dictionary,
                                markerCorners,
                                markerId,
                                parameters,
                                rejectedCandidates);
            if (markerId.size() > 0) {
                std::cout << "YES MARKER!" << endl;
                aruco::drawDetectedMarkers(image,
                                            markerCorners,
                                            markerId);
            }
            else {
                cout << "NO MARKER!" << endl;
            }
        }
    }
}
return 0;
}
```

# Základní transformace v souřadných systémech pro lokalizaci v obraze

## 4

V této bakalářské práci se zabýváme sledováním ArUco markeru kamerou. Marker i kamera mají svoje vlastní 3D souřadné systémy. První část kapitoly bude tedy pojednávat o transformaci dvou 3D souřadných systémů vůči sobě. Druhá část bude věnována souřadnému systému markeru.

### 4.1 Transformace 3D souřadných systémů

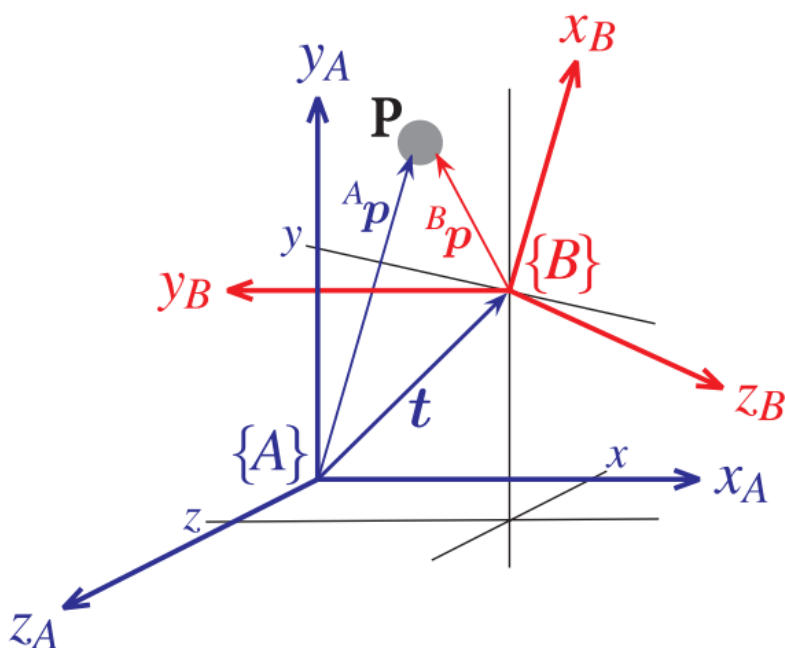
3D souřadný systém se od 2D liší tím, že je zde třetí osa  $z$ , která je ortogonální k oběma osám  $x$  a  $y$ . Orientace osy  $z$  je dána pravidlem pravé ruky, a proto hovoříme o pravo-točivém souřadném systému. Jednotkové vektory  $\hat{x}$ ,  $\hat{y}$  a  $\hat{z}$  jednotlivých os jsou ve vzájemných vztazích, viz rovnice (4.1) [3].

$$\begin{aligned}\hat{z} &= \hat{x} \times \hat{y} \\ \hat{x} &= \hat{y} \times \hat{z} \\ \hat{y} &= \hat{z} \times \hat{x}\end{aligned}\tag{4.1}$$

Bod  $P$  je reprezentován jako  $(x, y, z)$  v souřadném systému XYZ. Lze zapsat též jako vektor (4.2) [3]

$$\mathbf{p} = x\hat{x} + y\hat{y} + z\hat{z}.\tag{4.2}$$

Na obr. 4.1 se vyskytují dva souřadné systémy  $\mathbf{A}$  a  $\mathbf{B}$  [3]. Cílem je popsat souřadný systém  $\mathbf{B}$ , jeho rotaci a translaci, s ohledem na souřadný systém  $\mathbf{A}$ . Dále se na obrázku také nachází bod  $P$ , který je též potřeba popsat v závislosti na jednotlivých souřadných systémech. Vyjdou nám tedy dva vektory  ${}^A\mathbf{p}$  a  ${}^B\mathbf{p}$ . Problém s popisováním polohy bodu  $P$  rozdělíme na dvě dílčí úlohy a to na rotaci a následnou translaci.



Obrázek 4.1: 3D souřadné systémy A a B

Nejdříve se budeme zabývat rotací. Výsledný rotační vektor v souřadném systému A získáme pomocí vztahu (5.1) [3]

$$\begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} = \mathbf{R}_B^A \begin{pmatrix} B_x \\ B_y \\ B_z \end{pmatrix}, \quad (4.3)$$

kde  $\mathbf{R}_B^A$  je ortonormální rotační matice, která transformuje vektor  ${}^B\mathbf{p}$  do  ${}^A\mathbf{p}$ .

Rotační matice  $\mathbf{R}_Y^X$  ve 3D dimenzi má několik speciálních vlastností [3]:

- Matice je ortonormální, což znamená, že každý sloupec matice je jednotkový vektor a sloupce jsou vzájemně ortogonální.
- Sloupce definují rotaci souřadného systému Y vůči souřadnému systému X.
- Determinant matice je roven jedné, což má za následek to, že délka vektoru se po transformaci nemění  $\rightarrow \|{}^Y\mathbf{p}\| = \|{}^X\mathbf{p}\|, \forall \Theta$ .
- Inverzní rotační matice se rovná transponované rotační matici  $\rightarrow \mathbf{R}^{-1} = \mathbf{R}^T$ .

Ortonormální rotační matice pro úhel rotace  $\Theta$  je pro každou osu  $x$ ,  $y$  a  $z$  samostatná, jsou popsány vztahy (4.4), (4.5), (4.6) [3]. Jakákoliv rotace se pak dá vyjádřit pomocí násobení těchto tří matic.



$$\mathbf{R}_x(\Theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \Theta & -\sin \Theta \\ 0 & \sin \Theta & \cos \Theta \end{pmatrix} \quad (4.4)$$

$$\mathbf{R}_y(\Theta) = \begin{pmatrix} \cos \Theta & 0 & \sin \Theta \\ 0 & 1 & 0 \\ -\sin \Theta & 0 & \cos \Theta \end{pmatrix} \quad (4.5)$$

$$\mathbf{R}_z(\Theta) = \begin{pmatrix} \cos \Theta & -\sin \Theta & 0 \\ \sin \Theta & \cos \Theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (4.6)$$

Rotace lze rozdělit do dvou tříd: Eulerovy a Cardanovy. Eulerův typ rotací se otáčí jen kolem dvou os  $\rightarrow XYX, XZX, YXY, YZY, ZXZ$ , nebo  $ZYZ$ . Cardanův typ rotací znamená, že se objekt otáčí kolem všech os  $\rightarrow XYZ, XZY, YZX, YXZ, ZXY$ , nebo  $ZYX$ . Každá rotace kolem jednotlivé osy je dána určitým úhlem.

Eulerovy úhly je 3-dimenzionální vektor ve tvaru  $\Gamma = (\Phi, \Theta, \psi)$ . Je potřeba použití inverze, abychom našli odpovídající Eulerovy úhly k dané rotační matici. Např. pro záporný úhel  $\Theta$  nám inverzní funkce vrátí, že  $\Theta$  je kladná a také budou jiné hodnoty  $\Phi$  a  $\psi$ . Je to dáno tím, že dva odlišné vektory Eulerových úhlů mohou odpovídat jedné rotační matici. Existuje speciální případ, kdy  $\Theta = 0$ . V tu chvíli se  $\mathbf{R}_y$  rovná jednotkové matici a výsledná rotace je funkcí součtu  $\Phi + \psi$ . Inverzní funkce pak nedokáže určit součet těchto dvou úhlů, proto my zvolíme  $\Phi = 0$  a vypočte se  $\psi$ . V tuto chvíli máme dva nulové úhly a dostáváme se do problému zvaný singularita. Ta bude detailněji řešena v pozdější části.

Cardanovy úhly jsou definovány jako rolování (roll), naklonění (pitch) a stočení (yaw). Osa  $x$  jde dopředu a osa  $z$  jde nahoru, či dolů. Osa  $y$  je k oběma ortogonální. Pro rotaci objektu typu  $ZYX$  je celková rotace dána vztahem (4.7) [3]

$$\mathbf{R} = \mathbf{R}_z(\Theta_y)\mathbf{R}_y(\Theta_p)\mathbf{R}_x(\Theta_r). \quad (4.7)$$

I zde se s použitím inverzní funkce určí Cardanovy úhly k dané rotační matici. Problém nastává, pokud se úhel naklonění následující  $\Theta_p = \pm \frac{\pi}{2}$ . Opět se objevuje problém singularity.

Singularita je problémem všech možných reprezentací tří vektorů. Nastává v případě, když se rotační osa prostředního členu sekvence stane rovnoběžnou s rotační osou prvního, nebo třetího členu. Pro Eulerovy úhly  $ZYZ$  nastává singularita pro  $\Theta = k\pi$  a pro Cardanovy úhly nastává pro  $\Theta_p = \pm(2k + 1)\frac{\pi}{2}$ .

Vrátíme se zpět k rotaci jednoho souřadnicového systému vůči druhému. Můžeme určit úhel a vektor rotace pomocí vlastních čísel a vektorů matice  $\mathbf{R}$ . Existuje známý vztah (4.8) [3]

$$\mathbf{R}\mathbf{v} = \lambda\mathbf{v}, \quad (4.8)$$

kde  $\mathbf{v}$  je vlastní vektor, který je generován vlastním číslem  $\lambda$ . Pokud  $\lambda = 1$ , vznikne (4.9) [3]

$$\mathbf{R}\mathbf{v} = \mathbf{v}, \quad (4.9)$$

kde je vidět, že vlastní vektor je neměnný v závislosti na rotační matici  $\mathbf{R}$ . Takový vektor existuje pouze jeden a dochází kolem něho k rotaci. Ortonormální rotační matice má vždy jedno reálné vlastní číslo  $\lambda = 1$  a komplexně sdružená vlastní čísla  $\lambda = \cos \Theta \pm i \sin \Theta$ , kde  $\Theta$  je rotační úhel.

Pokud se řeší opačný problém, kdy je znám rotační úhel a vektor, je zapotřebí pro získání rotační matice aplikovat Rodriguesovu rotační formuli (4.10) [3]

$$\mathbf{R} = \mathbf{I}_{3 \times 3} + \sin \Theta [\hat{\mathbf{v}}]_{\times} + (1 - \cos \Theta) [\hat{\mathbf{v}}]_{\times}^2, \quad (4.10)$$

kde  $[\hat{\mathbf{v}}]_{\times}$  je šikmě-symetrická matice.

Je důležité poznamenat, že tato reprezentace jakékoliv rotace je dána čtyřmi čísly  $\rightarrow$  tři pro popis osy rotace a jedno pro rotační úhel. Je to o několik čísel méně, než u rotační matice, kde je jich potřeba devět. Směr pak může být reprezentován jednotkovým vektorem a rotační úhel může být zakomponován do délky, aby se získala tříparametrická reprezentace, např.  $\hat{\mathbf{v}}\Theta$ ,  $\hat{\mathbf{v}} \sin(\frac{\Theta}{2})$ ,  $\hat{\mathbf{v}} \tan \Theta$ , nebo  $\hat{\mathbf{v}} \tan(\frac{\Theta}{2})$ . Na první pohled by se mohlo zdát, že tyto formy jsou ideální, jelikož jsou minimální a efektivní pro ukládání dat, avšak jsou analyticky problematické a těžce definovatelné pro  $\Theta = 0$ .

Od rotační matice se přesuneme k transformační matici. Stále máme dva souřadné systémy  $\mathbf{A}$  a  $\mathbf{B}$ , viz obr. 4.1 [3]. Transformační matice vypadá následovně (4.11) [3]

$$\mathbf{T}_B^A = \begin{pmatrix} \mathbf{R}_B^A & \mathbf{t} \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix}, \quad (4.11)$$

kde  $\mathbf{t} \in R^3$  definuje posun rámu  $\mathbf{B}$  vzhledem k rámu  $\mathbf{A}$ ,  $\mathbf{R}$  je rotační matice, která popisuje orientaci os rámu  $\mathbf{B}$  s respektováním rámu  $\mathbf{A}$ . Transformační vektor bodu  $P$ , viz obr. 4.1, v souřadném systému  $\mathbf{A}$  se vypočte následujícím vzorcem (4.12) [3]

$$\begin{pmatrix} {}^A x \\ {}^A y \\ {}^A z \\ 1 \end{pmatrix} = \mathbf{T}_B^A \begin{pmatrix} {}^B x \\ {}^B y \\ {}^B z \\ 1 \end{pmatrix} \rightarrow {}^A \tilde{\mathbf{p}} = \mathbf{T}_B^{AB} \tilde{\mathbf{p}}. \quad (4.12)$$

Transformační matice  $\mathbf{T}_B^A$  je velikosti  $4 \times 4$  a má speciální strukturu, která patří do speciální Euklidovské skupiny dimenze 3  $\rightarrow \mathbf{T} \in \mathbf{SE}(3) \subset \mathbf{R}^{4 \times 4}$ . Konkrétní reprezentace relativní pozice je  $\xi \sim \mathbf{T} \in \mathbf{SE}(3)$  a  $\mathbf{T}_1 \oplus \mathbf{T}_2 \rightarrow \mathbf{T}_1 \mathbf{T}_2$ , což je klasické násobení matic, viz (4.13) [3].

$$\mathbf{T}_1 \mathbf{T}_2 = \begin{pmatrix} \mathbf{R}_1 & \mathbf{t}_1 \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_2 & \mathbf{t}_2 \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1 \mathbf{R}_2 & \mathbf{t}_1 + \mathbf{R}_1 \mathbf{t}_2 \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix} \quad (4.13)$$

Dále platí algebraické pravidlo pro pozici  $\rightarrow \xi \oplus 0 = \xi$ . Také je známo, že  $\mathbf{T} \mathbf{I} = \mathbf{T}$ , kde  $\mathbf{I}$  je jednotková matice.

Dalším algebraickým pravidlem pro pozici je  $\xi \ominus \xi = 0$ . Pro matice zase platí  $\mathbf{T} \mathbf{T}^{-1} = \mathbf{I}$ , z čehož vyplývá  $\ominus \mathbf{T} \rightarrow \mathbf{T}^{-1}$ . Inverzní transformace je pak dána vztahem (4.14) [3]

$$\mathbf{T}^{-1} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^T \mathbf{t} \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix}. \quad (4.14)$$

Homogenní transformační matice matice  $4 \times 4$  je velmi využívána v robotice, počítačové grafice a počítačovém vidění.

## 4.2 Souřadný systém ArUco markeru

Souřadný systém ArUco markeru se nachází uprostřed markeru. Souřadnice  $x$  směřuje vodorovně doprava, souřadnice  $y$  směřuje svisle nahoru a souřadnice  $z$  směřuje vzhůru kolmo k rovině značky. Nejprve je nutné provést kalibraci, viz kapitola 2.2. Získáme tím přesné transformační matice mezi kamerou a souřadným systémem markeru. Souřadný systém markeru je ve vztahu ke kameře a je závislý na konkrétní konfiguraci kamery a pozici markeru vzhledem k ní.

Pro převod mezi souřadnými systémy je proto potřeba použít transformační matice mezi kamerou a souřadným systémem markeru. Taková matice se nazývá matice kamery-markeru  $\mathbf{T}_{K \rightarrow M}$ . Matice převádí body v souřadném systému markeru na body v souřadném systému kamery. Transformační matice je o velikosti  $4 \times 4$

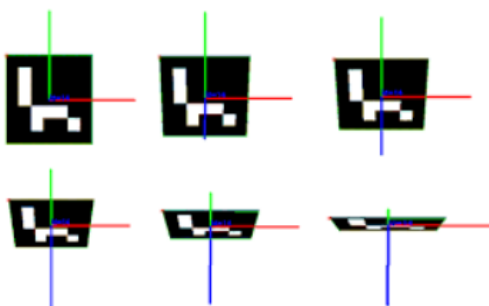
$$\mathbf{T}_{K \rightarrow M} = \begin{pmatrix} \mathbf{R}_{3 \times 3} & \mathbf{t}_{3 \times 1} \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix}. \quad (4.15)$$

Jedná se o rovnici (4.11) z předcházející části [3].  $\mathbf{R}_{3 \times 3}$  je matice rotace kamery vzhledem k souřadnému systému markeru,  $\mathbf{t}_{3 \times 1}$  je translační vektor kamery vzhledem k souřadnému systému markeru a  $\mathbf{0}_{1 \times 3}$  je vektor nul. Při použití této matice se body v souřadném systému markeru převedou na body v souřadném systému kamery pomocí následujícího vztahu

$$\begin{pmatrix} \mathbf{K} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R}_{3 \times 3} & \mathbf{t}_{3 \times 1} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{M} & 1 \end{pmatrix}, \quad (4.16)$$

kde  $\mathbf{M}$  je vektor bodu v souřadném systému markeru a  $\mathbf{K}$  je vektor bodu v souřadném systému kamery.

Na následujícím obrázku 4.2 jsou příklady souřadných systémů ArUco markeru [2].



Obrázek 4.2: Příklady orientace souřadných systémů ArUco markeru

# Metody odhadu polohy a orientace objektu na základě obrazových dat

## 5

Metod odhadu polohy a orientace z obrazových dat je nepřehledné množství. V dodatku C je popsána jedna z nich a to metoda RANSAC. Avšak tato kapitola se bude zabývat konkrétně implementovanou funkcí `estimatePoseSingleMarkers` v knihovně OpenCV, jenž má modul `aruco`. V programu neimplementujeme vlastní metodu, ale používáme tuto zabudovanou. Následně zde bude popsán program pro odhad polohy a rotace ArUco markeru v souřadném systému kamery a jejich přečet do lokálního souřadného systému.

## 5.1 Metoda `estimatePoseSingleMarkers`

Funkce má několik parametrů. Jako první povinný parametr předává rohy detekovaných markerů. Ty se zjistili pomocí metody `detectMarkers`, která se využívá v programu 3.1. Druhým parametrem je délka strany ArUco markeru. Délku lze zadat v libovolných jednotkách, např. v centimetrech, metrech, avšak standardní jednotka jsou metry, proto je lepší zadávat délku strany v metrech. Třetí parametr je matice kamery a čtvrtým parametrem je vektor koeficientů zkreslení, obě instance se získají kalibrací kamery, jenž je popsána v části 2.1. Nakonec se zadají prázdné proměnné pro rotační a translační vektor. Rotační vektor bude udáván v radiánech. Lze z tohoto vektoru vytvořit rotační matici pomocí metody `Rodrigues`. Metoda funguje i opačně, tudíž rotační matici lze převést na rotační vektor. Translační vektor je udáván v jednotkách, ve kterých se zadala délka strany markeru, pokud byla zadána v metrech, hodnoty translačního vektoru budou také v metrech. Oba vektory jsou vzhledem k souřadnému systému kamery. Metoda má ještě jeden nepovinný parametr a to matici, která obsahuje souřadnice 3D bodů v reálném světě, které odpovídají rohům ArUco markeru. Matice vypadá následovně

$$\begin{pmatrix} \frac{-\text{velikost\_markeru}}{2} & \frac{\text{velikost\_markeru}}{2} & 0 \\ \frac{\text{velikost\_markeru}}{2} & \frac{\text{velikost\_markeru}}{2} & 0 \\ \frac{\text{velikost\_markeru}}{2} & \frac{-\text{velikost\_markeru}}{2} & 0 \\ \frac{-\text{velikost\_markeru}}{2} & \frac{-\text{velikost\_markeru}}{2} & 0 \end{pmatrix}. \quad (5.1)$$

Lze tedy říct, že tato funkce na základě rohů detekovaného ArUco markeru odhadne polohu a rotaci vzhledem k souřadnému systému kamery [4]. Pokud je detekováno více ArUco markerů, funkce vytvoří pro každý marker vlastní translační a rotační vektor. Ve výchozím nastavení je určeno, že souřadný systém Aruco markeru je soustředěn uprostřed a osa z je kolmá na rovinu značky [4].

V dokumentaci [4] jsou uvedeny dva nepovinné parametry, avšak existují různé verze knihovny OpenCV. Tato bakalářská práce používá verzi z roku 2013, kde je rozhraní této metody definované, jak je popsáno výše, viz obr. 5.1.

```

/**
 * @brief Pose estimation for single markers
 *
 * @param corners vector of already detected markers corners. For each marker, its four corners
 * are provided, (e.g. std::vector<std::vector<cv::Point2f> > ). For N detected markers,
 * the dimensions of this array should be Nx4. The order of the corners should be clockwise.
 * @sa detectMarkers
 * @param markerLength the length of the markers' side. The returning translation vectors will
 * be in the same unit. Normally, unit is meters.
 * @param cameraMatrix input 3x3 floating-point camera matrix
 * \f$A = \text{vecthreethree}\{f_x\}\{c_x\}\{f_y\}\{c_y\}\{0\}\{1\}\f$
 * @param distCoeffs vector of distortion coefficients
 * \f$(k_1, k_2, p_1, p_2[, k_3[, k_4, k_5, k_6],[s_1, s_2, s_3, s_4]])\f$ of 4, 5, 8 or 12 elements
 * @param rvecs array of output rotation vectors (@sa Rodrigues) (e.g. std::vector<cv::Vec3d>).
 * Each element in rvecs corresponds to the specific marker in imgPoints.
 * @param tvecs array of output translation vectors (e.g. std::vector<cv::Vec3d>).
 * Each element in tvecs corresponds to the specific marker in imgPoints.
 * @param _objPoints array of object points of all the marker corners
 *
 * This function receives the detected markers and returns their pose estimation respect to
 * the camera individually. So for each marker, one rotation and translation vector is returned.
 * The returned transformation is the one that transforms points from each marker coordinate system
 * to the camera coordinate system.
 * The marker coordinate system is centered on the middle of the marker, with the Z axis
 * perpendicular to the marker plane.
 * The coordinates of the four corners of the marker in its own coordinate system are:
 * (-markerLength/2, markerLength/2, 0), (markerLength/2, markerLength/2, 0),
 * (markerLength/2, -markerLength/2, 0), (-markerLength/2, -markerLength/2, 0)
 */
CV_EXPORTS_W void estimatePoseSingleMarkers(InputArrayOfArrays corners, float markerLength,
                                           InputArray cameraMatrix, InputArray distCoeffs,
                                           OutputArray rvecs, OutputArray tvecs, OutputArray _objPoints = noArray());

```

Obrázek 5.1: Rozhraní metody estimatePoseSingleMarkers v knihovně OpenCV, verze z roku 2013

## 5.2 Implementace v OpenCV

Kód 5.1 popisuje odhad vzdálenosti a orientace ArUco markeru v souřadném systému kamery a následný přepočítání do lokálního souřadného systému. Nejdříve se

nadefinují potřebné proměnné. Důležitou částí je vytvoření translačního a rotačního vektoru kamery v lokálním souřadném systému. Úhly ve stupních je potřeba převést na radiány. Rotační vektor se pak pomocí metody *Rodrigues* převede na rotační matici.

V cyklu se pak používá funkce `estimatePoseSingleMarkers`, která byla popsána výše. Z translačního vektoru `tvec` se vypočte jeho norma pomocí vzorce (5.2)

$$|tvec| = \sqrt{x^2 + y^2 + z^2}, \quad (5.2)$$

což udává vzdálenost markeru od kamery.

Hodnoty rotačního vektoru `rvec` je nejdříve potřeba převést na stupně. Program si ukládá minulé hodnoty rotačního vektoru. Ty se pak odečtou od aktuálních hodnot a vyjde natočení markeru. Pro informace, kde se ArUco marker nachází v lokálním souřadném systému, je potřeba vynásobit translační vektor transformační maticí. Jelikož jazyk C++ nemá zabudovanou funkci násobení matic, celou transformační matici jsme nevytvářeli. Jen ze znalostí popsané v kapitole 4 a z pravidel násobení vektoru maticí je vytvořen postup pro získání translačního vektoru v lokálním souřadném systému.

Zdrojový kód 5.1: Část programu pro odhad polohy a orientace ArUco markeru

```
double L2norma(Mat vektor) {
    double a, b, c;
    a = vektor.at<double>(0, 0);
    b = vektor.at<double>(0, 1);
    c = vektor.at<double>(0, 2);
    return (sqrt(a * a + b * b + c * c));
}

double radiny2stupne(double radian) {
    double stupen = (radian * 180) / M_PI;
    return (stupen);
}

double stupne2radiany(double stupen) {
    double radian = (stupen * M_PI) / 180;
    return (radian);
}

int main() {
    ...
    // kamera
    ...
    //detekce markeru
    ...
    //poloha markeru
    Mat rvec, tvec;
    std::vector<cv::Point3f> objPoints;
    objPoints.push_back(cv::Point3f(-velikost_markeru / 2,
                                    velikost_markeru / 2,
                                    0));
}
```

## 5. Metody odhadu polohy a orientace objektu na základě obrazových dat

```
objPoints.push_back(cv::Point3f(velikost_markeru / 2,
                                velikost_markeru / 2,
                                0));
objPoints.push_back(cv::Point3f(velikost_markeru / 2,
                                -velikost_markeru / 2,
                                0));
objPoints.push_back(cv::Point3f(-velikost_markeru / 2,
                                -velikost_markeru / 2,
                                0));

double vzdal_od_kam;
double x_lokal = 0;
double y_lokal = 0;
double z_lokal = 0;
Mat rmat = Mat_<double>(3, 3);
Mat novy_tvec = Mat_<double>(1, 3);
Mat novy_rvec = Mat_<double>(1, 3);
Mat svet_tvec = Mat_<double>(1, 3);
Mat svet_rvec = Mat_<double>(1, 3);

\\translavni vektor kamery a rotacni matice
double z = 0.83;
Mat kamera_tvec = Mat_<double>(1, 3);
kamera_tvec.at<double>(0, 0) = 0;
kamera_tvec.at<double>(0, 1) = 0;
kamera_tvec.at<double>(0, 2) = z;

double stupne_kamery = 40-90; \\25-90;
Mat kamera_rvec = Mat_<double>(1, 3);
kamera_rvec.at<double>(0, 0) = stupne2radiany(stupne_kamery);
kamera_rvec.at<double>(0, 1) = 0;
kamera_rvec.at<double>(0, 2) = 0;
Mat kamera_rmat = Mat_<double>(3, 3);
Rodrigues(kamera_rvec, kamera_rmat);

namedWindow("Display window");
VideoCapture cap(1); // 0 - webkamera, 1 - iextern kamera

if (!cap.isOpened()) {
    std::cout << "" << endl;
    std::cout << "Cannot open camera." << endl;
}
else {
    while (true) {
        cap >> image;
        ...
        if (kalibrace) {
            cout << "DETEKCE" << endl;
            ...
            if (markerId.size() > 0) {
                std::cout << "YES MARKER!" << endl;
                ...
                aruco::estimatePoseSingleMarkers(markerCorners,
                                                velikost_markeru,
                                                cameraMatrix,
                                                distCoeffs,
                                                rvec,
                                                tvec,
                                                objPoints);

                double metr_x = tvec.at<double>(0, 0);
```



```

double metr_y = tvec.at<double>(0, 1);
double metr_z = tvec.at<double>(0, 2);

vzdal_od_kam = L2norma(tvec);

double stupen_x = radiny2stupne(rvec.at
                                <double>(0, 0));
double stupen_y = radiny2stupne(rvec.at
                                <double>(0, 1));
double stupen_z = radiny2stupne(rvec.at
                                <double>(0, 2));

double rozdil_stupen_x = stupen_x - minuly_stupen_x;
double rozdil_stupen_y = stupen_y - minuly_stupen_y;
double rozdil_stupen_z = stupen_z - minuly_stupen_z;

double minuly_stupen_x = stupen_x;
double minuly_stupen_y = stupen_y;
double minuly_stupen_z = stupen_z;

//-----TRANSFORMACE SOURADNIC-----

svet_tvec.at<double>(0, 0) =
kamera_rmat.at<double>(0, 0) * tvec.at<double>(0, 0)
+
kamera_rmat.at<double>(0, 1) * tvec.at<double>(0, 1)
+
kamera_rmat.at<double>(0, 2) * tvec.at<double>(0, 2)
+
kamera_tvec.at<double>(0, 0);

svet_tvec.at<double>(0, 1) =
kamera_rmat.at<double>(1, 0) * tvec.at<double>(0, 0)
+
kamera_rmat.at<double>(1, 1) * tvec.at<double>(0, 1)
+
kamera_rmat.at<double>(1, 2) * tvec.at<double>(0, 2)
+
kamera_tvec.at<double>(0, 1);

svet_tvec.at<double>(0, 2) =
kamera_rmat.at<double>(2, 0) * tvec.at<double>(0, 0)
+
kamera_rmat.at<double>(2, 1) * tvec.at<double>(0, 1)
+
kamera_rmat.at<double>(2, 2) * tvec.at<double>(0, 2)
+
kamera_tvec.at<double>(0, 2);
}
else {
cout << "NO MARKER!" << endl;
stop_marker++;
}
}
}
}
return 0;
}

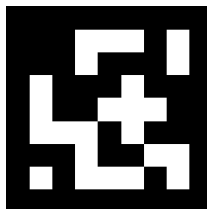
```

Tato kapitola bude věnována experimentům. Nejprve si představíme, které ArUco markery budou během měření využity. Následně si popíšeme použité kamery, jež budou dvě a budou se porovnávat hodnoty, pomocí nich naměřených. Poslední část je věnována jednotlivým experimentům.

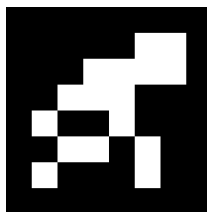
## 6.1 Použité ArUco markery

Na experimenty bude použit Aruco marker s ID = 33 z knihovny DICT\_7x7\_250 na obr. 6.1. Bude se využívat ve dvou různých velikostech, které jsou 5,8 cm a 12,7 cm.

Druhý použitý ArUco marker bude mít také ID = 33, ale bude z knihovny DICT\_6x6\_250, viz obr. 6.2. Tento marker je jen jeden a jeho velikost je 5 cm.



Obrázek 6.1: ArUco marker, ID = 33, knihovna DICT\_7x7\_250



Obrázek 6.2: ArUco marker, ID = 33, knihovna DICT\_6x6\_250

## 6.2 Použité kamery

Na testování byly použity dvě kamery. První z nich bude externí webkamera Logitech. Maximální rozlišení kamery je HD (1280 × 720) a snímkovací frekvence je 30 snímků za sekundu. Kamera také disponuje automatickým ostřením. Maximální rozlišení záznamu je 8 Mpx.

Druhá kamera je na mobilu typu Honor 9. Rozlišení kamery je 1080 × 1920. Maximálně udělá 120 snímků za sekundu. Také jak webkamera má funkci autofokusu.

## 6.3 Průběh experimentů

Tato část se již zaměřuje na samotné experimenty. Nejdříve jsme snímali marker ze dvou různých výšek. V každé výšce jsme kameru umístili do dvou různých úhlů. Bude se zkoumat, jaký vliv má poloha kamery ve výšce a naklonění na naměřené hodnoty. Snímané hodnoty budou v souřadném systému kamery.

V dalším experimentu se budeme zabývat převodem ze souřadného systému kamery do lokálního systému. Kamera bude umístěna v jedné výšce a bude nakloněna také do dvou úhlů. Bude se zkoumat, jak přesně je určena poloha ArUco markeru v rovině  $XY$  lokálního souřadného systému a jaký vliv na to má naklonění kamery.

V obou případech jsem počítala střední hodnotu  $\bar{q}$  z naměřených dat. Každou hodnotu jsme měřili 20×. Zde je vzorec pro výpočet střední hodnoty

$$\bar{q} = \frac{\sum_{k=1}^K q_k}{K}, \quad (6.1)$$

kde  $q_k$  je  $k$ -tá naměřená hodnota,  $k$  je krok, který nabývá hodnot od 1 do  $K$ , v tomto případě  $K = 20$ .

Dále jsme počítali směrodatnou odchylku  $\sigma$  pro naměřené hodnoty pomocí vzorce

$$\sigma = \sqrt{\frac{\sum_{k=1}^K (q_k - \bar{q})^2}{K}}, \quad (6.2)$$

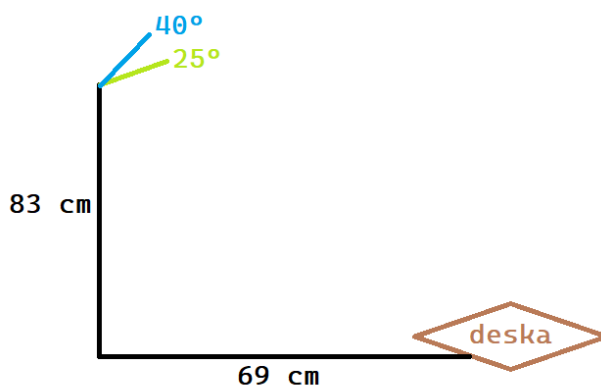
kde  $q_k$  je  $k$ -tá naměřená hodnota,  $k$  je krok, který nabývá hodnot od 1 do  $K$ , v tomto případě  $K = 20$ .

Poslední zkoumanou hodnotou je odchylka  $\tilde{q}$  průměrné hodnoty  $\bar{q}$  od reálné hodnoty  $q$ , která se vypočítá jako jejich rozdíl

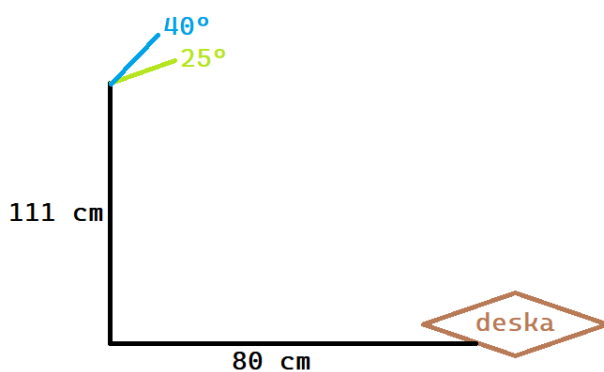
$$\tilde{q} = q - \bar{q}. \quad (6.3)$$

### 6.3.1 Poloha a orientace ArUco markeru v souřadném systému kamery

Prvním experimentem bylo zkoumání přesnosti odhadu polohy a orientace ArUco markeru v souřadném systému kamery. Připravili jsme si kartonovou desku o velikosti  $88 \times 62$  cm. Na ní jsme vyznačili šest bodů. Dále jsme určili výšky, ze kterých budeme marker snímat. Nejdříve bude kamera uchycena ve výšce 83 cm a vzdálená 69 cm od kraje desky. V této výšce bude nakloněna do dvou úhlů  $40^\circ$  a  $25^\circ$ . Schéma konstrukce je zaznamenáno na obr. 6.3. Následně bude kamera upevněna ve výšce 111 cm a vzdálená od kraje desky 80 cm. Opět bude v této výšce nakloněna do dvou úhlů  $40^\circ$  a  $25^\circ$ . Na obr. 6.4 je znázorněno schéma konstrukce pro vyšší polohu. Obraz ze snímání ArUco markeru položeného na desce externí webkamerou se nachází na obr. 6.5.

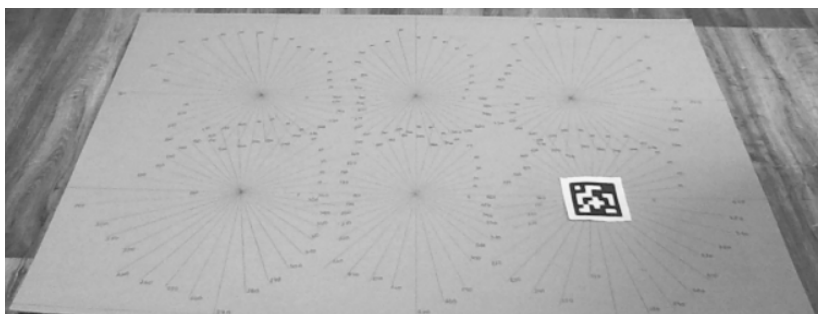


Obrázek 6.3: Schéma konstrukce pro kameru upevněnou ve výšce 83 cm



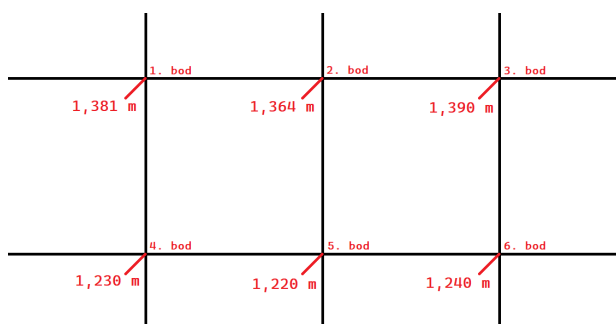
Obrázek 6.4: Schéma konstrukce pro kameru upevněnou ve výšce 111 cm

### 6.3.1. Poloha a orientace ArUco markeru v souřadném systému kamery

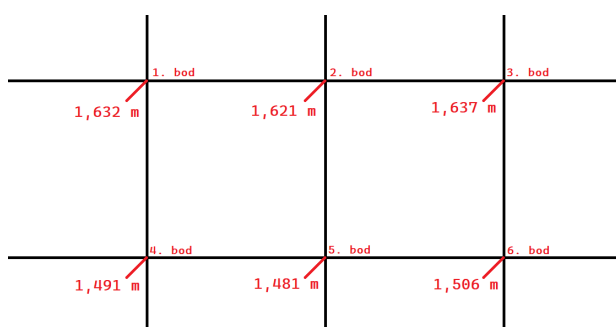


Obrázek 6.5: ArUco marker o velikosti 5,8 cm z knihovny DICT\_7x7\_250 na desce snímáný externí webkamerou

Pomocí laserového měřidla jsme zjistili reálné vzdálenosti všech šesti bodů vyznačených na desce od kamery umístěné v 83 cm a 111 cm nad zemí. Jednotlivé vzdálenosti jsou zaznamenány na obrázcích 6.6 a 6.7.



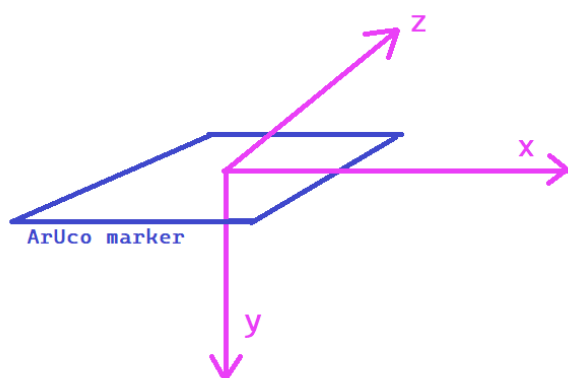
Obrázek 6.6: Reálné vzdálenosti pro šest bodů od kamery upevněné ve výšce 83 cm



Obrázek 6.7: Reálné vzdálenosti pro šest bodů od kamery upevněné ve výšce 111 cm

V tomto experimentu bude měřena nejen vzdálenost od kamery, ale také otočení markeru v obraze. ArUco markeru jsme umístili na 6. bod mřížky, viz obr. 6.5, a

v tomto bodu jsme měnili jeho úhel natočení a snímali ho. Marker je položen horizontálně. Rotaci určujeme, jak ji vnímá kamera. Metoda `estimatePoseSingleMarkers`, jež je popsána v kapitole 5, vrátí vektory `tvec` a `rvec` v souřadném systému kamery. Souřadný systém ArUco markeru vůči kamere je následovný: osa  $x$  jde vodorovně doprava, osa  $y$  je dolů kolmo k rovině markeru a osa  $z$  směřuje svisle nahoru. Proto sledovaná rotace bude kolem osy  $y$ . V programu 5.1 nás bude zajímat proměnná `rozdil_stupen_y`. Na obr. 6.8 je znázorněný souřadný systém ArUco markeru, jak je vnímán kamerou.



Obrázek 6.8: Souřadný systém ArUco markeru vůči kamere

### 6.3.1.1 Hodnoty naměřené pomocí externí webkamery

Jako první jsme měřili hodnoty pomocí externí webkamery ve výšce 83 cm a úhel naklonění kamery byl  $40^\circ$ . Dále jsme změnili výšku kamery na 111 cm a úhel naklonění zachovali. Třetí měření bylo pro kameru ve výšce 83 cm a úhel naklonění byl  $25^\circ$ . Poslední měření bylo opět ve vyšší poloze a úhel naklonění byl také  $25^\circ$ . Z naměřených hodnot  $q_k$  jsme vypočítali střední hodnotu  $\bar{q}$  pomocí vzorce (6.1). Dále jsme z dat vypočítali směrodatnou odchylku  $\sigma$  podle vzorce (6.2) a odchylku  $\tilde{q}$  střední hodnoty  $\bar{q}$  od reálné hodnoty  $q$  podle vzorce (6.3).

Reálné vzdálenosti, střední hodnoty, směrodatné odchylky a odchylky průměrné hodnoty od reálných vzdáleností jsme zanesli do tabulek 6.1, 6.3, 6.5, 6.7. Reálné úhly, střední hodnoty, směrodatné odchylky a odchylky průměrné hodnoty od reálných úhlů jsme zaznamenali do tabulek 6.2, 6.4, 6.6, 6.8. Přípustné odchylky středních hodnot vzdáleností od reálných budou v rozsahu  $< 0; 0,5 >$  pro odchylky středních hodnot úhlů od reálných budou v intervalu  $< 0; 10 >$ .

Tabulka 6.1 obsahuje vzdálenosti a tabulka 6.2 obsahuje rotaci markeru, kdy kamera byla umístěna ve výšce 83 cm pod úhlem  $40^\circ$ . Na obr. 6.6 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. Směrodatné od-

chylky naměřených hodnot vzdáleností jsou velmi minimální, což značí, že hodnoty jsou až na nějaké nižší řády konstantní. Avšak směrodatné odchytky naměřených úhlů v některých případech jsou větší než jedna, v tomto případě je praktické, že se úhel snímal 20×. Pokud se zaměříme na odchytky středních hodnot od reálných, pro vzdálenosti se hodnoty pohybují v přípustném intervalu  $< 0; 0,5 >$ . To lze říci i o odchytkách úhlů, které se pohybují v intervalu  $< 0; 10 >$ . Zde je největší odchylka  $5,9653^\circ$ , ale pořád se jedná o validní údaj, se kterým se dá dále pracovat.

Tabulka 6.1: Vzdálenosti snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve  $40^\circ$

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,381	1,3148	0,0167	0,0662
	2. bod	1,364	1,2182	0	0,1458
	3. bod	1,39	1,0503	0,0151	0,3397
	4. bod	1,23	1,2679	0,0129	0,0659
	5. bod	1,22	1,1708	0,0144	0,0492
	6. bod	1,24	1,2827	0	0,0427
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	1,381	1,3358	0,0024	0,0452
	2. bod	1,364	1,3004	0,0045	0,0636
	3. bod	1,39	1,3515	0,0035	0,0385
	4. bod	1,23	1,2915	0,0068	0,0895
	5. bod	1,22	1,157	0,0092	0,063
	6. bod	1,24	1,3451	0,0049	0,1051
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,381	1,1722	0,0082	0,2088
	2. bod	1,364	1,1897	0,0106	0,1743
	3. bod	1,39	1,3871	0,0092	0,0029
	4. bod	1,23	1,027	0	0,175
	5. bod	1,22	1,0668	0,0004	0,1532
	6. bod	1,24	1,2586	0,0293	0,0186

## 6. Experimenty

Tabulka 6.2: Rotace snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 40°

ArUco marker	reálné hodnoty $q$ [°]	střední hodnota $\bar{q}$ [°]	směrodatná odchylka $\sigma$	odchylka $\tilde{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	-0,1781	0,3641	0,1781
	90	90,755	0,2134	0,7554
	-90	-95,9653	0	5,9653
	40	39,0582	0,467	0,9418
	-40	-43,2276	0,0391	3,2276
	120	121,81	0	1,81
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	-120	-122,7624	0,181	2,7624
	0	1,6648	0,1091	1,6648
	90	91,1968	0,6984	1,1968
	-90	-91,706	1,1818	1,706
	40	44,496	0	4,496
	-40	-38,4756	0,0779	1,5244
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	120	118,8798	0,7028	1,1202
	-120	-123,0447	2,3678	3,0447
	0	1,0931	0,1718	1,0931
	90	92,0108	1,3211	2,0108
	-90	-95,6908	0,1082	5,6908
	40	38,3091	0,2586	1,6909
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	-40	-39,7927	0,0115	0,2073
	120	122,2631	1,0358	2,2631
	-120	-121,4477	0,9634	1,4477

Tabulka 6.3 obsahuje vzdálenosti a tabulka 6.4 obsahuje rotaci markeru, kdy kamera byla umístěna ve výšce 111 cm pod úhlem 40°. Na obr. 6.7 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. I v tomto měření vzdáleností jsou směrodatné odchylky minimální. Oproti předchozí tabulce rotací 6.2 jsou minimální i směrodatné odchylky naměřených úhlů. Odchylky průměrných vzdáleností od reálných se opět nacházejí v požadovaném intervalu. To samé lze říci i o tabulce úhlů.



Tabulka 6.3: Vzdálenosti snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 40°

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,632	1,6284	0,0001	0,0036
	2. bod	1,621	1,6235	0	0,0025
	3. bod	1,637	1,6869	0,005	0,0499
	4. bod	1,491	1,5242	0,0236	0,0332
	5. bod	1,481	1,4291	0,0002	0,0519
	6. bod	1,506	1,462	0,0101	0,044
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	1,632	1,6359	0,0004	0,0039
	2. bod	1,621	1,6465	0,006	0,0255
	3. bod	1,637	1,6359	0,0149	0,0011
	4. bod	1,491	1,5138	0,0152	0,0228
	5. bod	1,481	1,4512	0,2991	0,0298
	6. bod	1,506	1,5104	0,0037	0,0044
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,632	1,7278	0	0,0958
	2. bod	1,621	1,5964	0	0,0246
	3. bod	1,637	1,768	0	0,131
	4. bod	1,491	1,4702	0,0016	0,0208
	5. bod	1,481	1,471	0	0,01
	6. bod	1,506	1,4805	0,0055	0,0255

Tabulka 6.4: Rotace snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 40°

ArUco marker	reálné hodnoty $q$ [°]	střední hodnota $\bar{q}$ [°]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	0,4349	0,0683	0,4349
	90	93,8114	0,3564	3,8114
	-90	-92,9531	1,3663	2,9531
	40	37,8417	0	2,1583
	-40	-43,9458	0	3,9458
	120	123,7112	1,0833	3,7112
	-120	-117,2949	0,1069	2,7051
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	0	-1,7732	1,5054	1,7732
	90	91,625	0,1729	1,625
	-90	-89,3191	0,0713	0,6809
	40	43,2375	0,6826	3,2375
	-40	-41,2386	0	1,2386
	120	119,6141	0,2404	0,3859
	-120	-122,001	0	2,001
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	0	0,5295	0	0,5295
	90	89,872	0	0,128
	-90	-89,7542	0,4938	0,2458
	40	42,3417	0	2,3417
	-40	-43,1593	0,5848	3,1593
	120	122,4846	0,615	2,4846
	-120	-123,873	0	3,873

Tabulka 6.5 obsahuje vzdálenosti a tabulka 6.6 obsahuje rotaci markeru, kdy kamera byla umístěna ve výšce 83 cm pod úhlem 25°. Na obr. 6.6 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. Směrodaté odchylky vzdáleností, ale i úhlů jsou i v tomto měření malé. Změna úhlu naklonění kamery ovlivnila snímání vzdáleností. Je zde větší rozdíl odchylek průměrných vzdáleností od reálných, ačkoliv i v tomto případě se hodnoty pohybují v přípustném intervalu.

## 6. Experimenty

Na snímání rotace markeru změna naklonění neukázala nějaké viditelné ovlivnění výsledků.

Tabulka 6.5: Vzdálenosti snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 25°

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,381	1,5495	0,0003	0,1685
	2. bod	1,364	1,6115	0	0,2475
	3. bod	1,39	1,6797	0,0051	0,2897
	4. bod	1,23	1,3134	0,0126	0,1114
	5. bod	1,22	1,4037	0,0073	0,1837
	6. bod	1,24	1,4674	0	0,2274
ArUco marker o velikosti 17,7 cm z knihovny DICT_7x7_250	1. bod	1,381	1,4042	0,0002	0,0232
	2. bod	1,364	1,3908	0,0051	0,0268
	3. bod	1,39	1,4744	0,0042	0,0844
	4. bod	1,23	1,254	0,0044	0,052
	5. bod	1,22	1,414	0,0913	0,194
	6. bod	1,24	1,4789	0	0,2389
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,381	1,2805	0	0,1005
	2. bod	1,364	1,3324	0,009	0,0316
	3. bod	1,39	1,4211	0,0124	0,0311
	4. bod	1,23	1,0691	0	0,1329
	5. bod	1,22	1,1522	0,0096	0,0678
	6. bod	1,24	1,2233	0,0055	0,0167

Tabulka 6.6: Rotace snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 25°

ArUco marker	reálné hodnoty $q$ [°]	střední hodnota $\bar{q}$ [°]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	3,2403	0	3,2403
	90	89,0368	0,452	0,9632
	-90	-90,2265	0,7641	0,2265
	40	41,9667	0,1541	1,9667
	-40	-39,5953	0,6037	0,4047
	120	118,607	0,275	1,393
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	-120	-120,729	0,1549	0,729
	0	-0,7491	0,0142	0,7491
	90	89,9578	0,448	0,0422
	-90	-89,2959	0	0,7041
	40	42,7095	0,068	2,7095
	-40	-43,5788	0,4672	3,5788
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	120	120,916	0,5789	0,916
	-120	-119,2526	0,6416	0,7474
	0	1,1778	0,6731	1,1778
	90	90,439	0	0,439
	-90	-93,592	0	3,592
	40	40,3519	1,2375	0,3519
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	-40	-43,7032	0	3,7032
	120	118,8147	0,3592	1,1853
	-120	-121,2362	0,254	1,2362

Tabulka 6.7 obsahuje vzdálenosti a tabulka 6.8 obsahuje rotaci markeru, kdy kamera byla umístěna ve výšce 111 cm pod úhlem 25°. Na obr. 6.7 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. I ve vyšší poloze ovlivnilo nižší naklonění kamery nasnímaná data, směrodatné odchylky jsou menší,

než v tabulkách 6.3 a 6.4. Odchytky středních hodnot vzdáleností od reálných jsou všechny v intervalu pro určení validity dat. To samé platí i pro hodnoty úhlů.

Tabulka 6.7: Vzdálenosti snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 25°

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchytka $\sigma$	odchytky $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,632	1,6818	0	0,0498
	2. bod	1,621	1,6105	0	0,0105
	3. bod	1,637	1,557	0,0204	0,08
	4. bod	1,491	1,4475	0,0012	0,0435
	5. bod	1,481	1,4523	0,1113	0,0287
	6. bod	1,506	1,4912	0,001	0,0148
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	1,632	1,5882	0,0028	0,0438
	2. bod	1,621	1,5651	0,0062	0,0559
	3. bod	1,637	1,5345	0,0013	0,1025
	4. bod	1,491	1,4405	0,0459	0,0505
	5. bod	1,481	1,4321	0,0015	0,0489
	6. bod	1,506	1,5397	0,0029	0,0337
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,632	1,6346	0,0839	0,0026
	2. bod	1,621	1,5924	0,0008	0,0286
	3. bod	1,637	1,5826	0,0008	0,0544
	4. bod	1,491	1,5811	0	0,0901
	5. bod	1,481	1,4813	0,0008	0,0003
	6. bod	1,506	1,5132	0,0162	0,0072

Tabulka 6.8: Rotace snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 25°

ArUco marker	reálné hodnoty $q$ [°]	střední hodnota $\bar{q}$ [°]	směrodatná odchytky $\sigma$	odchytky $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	1,4775	0,1914	1,4775
	90	91,3703	0,6503	1,3703
	-90	-93,7737	0,2971	3,7737
	40	41,546	0,0684	1,546
	-40	-42,6836	0	2,6836
	120	121,368	0	1,368
	-120	-123,1464	0,0399	3,1464
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	0	1,5896	1,991	1,5896
	90	93,8069	1,0083	3,8069
	-90	-90,8731	0,5689	0,8731
	40	39,2589	1,0739	0,7411
	-40	-43,1408	0,1496	3,1408
	120	120,4722	0	0,4722
	-120	-119,7947	0,4868	0,2053
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	0	-1,1681	0,2706	1,1681
	90	89,6652	0	0,3348
	-90	-91,2594	1,3389	1,2594
	40	39,3482	0,9352	0,6518
	-40	-42,9364	0	2,9364
	120	122,1391	0,3236	2,1391
	-120	-121,6673	1,3008	1,6673

### 6.3.1.2 Hodnoty naměřené pomocí kamery v mobilu

Podruhé jsme měřili hodnoty pomocí kamery v mobilu ve výšce 83 cm a 111 cm a úhel naklonění kamery byl opět  $40^\circ$  a  $25^\circ$ . Z naměřených hodnot  $q_k$  jsme vypočítali střední hodnotu  $\bar{q}$  pomocí vzorce (6.1), směrodatnou odchylku  $\sigma$  podle vztahu (6.2) a odchylku  $\tilde{q}$  střední hodnoty  $\bar{q}$  od reálné hodnoty  $q$  podle rovnice (6.3).

Reálné vzdálenosti, střední hodnoty, směrodatné odchylky a odchylky průměrné hodnoty od reálných vzdáleností jsme zanesli do tabulek 6.9, 6.11, 6.13, 6.15. Reálné úhly, střední hodnoty, směrodatné odchylky a odchylky průměrné hodnoty od reálných úhlů jsme zaznamenali do tabulek 6.10, 6.12, 6.14, 6.16.

Tabulka 6.9 obsahuje vzdálenosti a tabulka 6.10 obsahuje rotaci markeru, kdy kamera byla umístěna ve výšce 83 cm pod úhlem  $40^\circ$ . Na obr. 6.6 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. Směrodatné odchylky naměřených vzdáleností nasvědčují tomu, že až na dvě výjimky se hodnoty nijak nevychylovaly od průměrné hodnoty. Oproti tomu naměřené hodnoty úhlů se více odchýlovaly od průměrných hodnot až na ArUco marker o velikosti 12,5 cm, jehož úhly  $\pm 40^\circ$  a  $\pm 25^\circ$  mají nulovou směrodatnou odchylku. Odchylky průměrných vzdáleností od reálných se vyskytují v přípustném intervalu, který je  $< 0; 0,5 >$ . To samé lze říci i o odchylkách úhlů, kde největší odchylka je  $3,961^\circ$ . Hodnoty z obou tabulek lze pro to považovat za validní.

Tabulka 6.9: Vzdálenosti snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve  $40^\circ$

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\tilde{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,381	1.4442	0.0063	0.0632
	2. bod	1,364	1.4297	0.015	0.0657
	3. bod	1,39	1.4267	0.0634	0.0367
	4. bod	1,23	1.3798	0.0032	0.1778
	5. bod	1,22	1.3299	0.0017	0.1099
	6. bod	1,24	1.3742	0.1523	0.1342
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	1,381	1.4652	0.0041	0.0842
	2. bod	1,364	1.4439	0.0276	0.0799
	3. bod	1,39	1.3324	0.3348	0.0576
	4. bod	1,23	1.3732	0.0429	0.1712
	5. bod	1,22	1.3251	0.0018	0.1051
	6. bod	1,24	1.3274	0	0.0874
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,381	1.4597	0.0252	0.0787
	2. bod	1,364	1.4791	0.0092	0.1151
	3. bod	1,39	1.4798	0.0012	0.0898
	4. bod	1,23	1.3065	0.0096	0.1045
	5. bod	1,22	1.3077	0.0941	0.0877
	6. bod	1,24	1.3219	0.0013	0.0819

Tabulka 6.10: Rotace snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve 40°

ArUco marker	reálné hodnoty $q$ [°]	střední hodnota $\bar{q}$ [°]	směrodatná odchylka $\sigma$	odchylka $\tilde{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	3.3271	0.4477	3.3271
	90	88.9335	2.0646	1.0665
	-90	-88.6992	0.8615	1.3008
	40	41.0006	0.3693	1.0006
	-40	-40.2104	0.7331	0.2104
	120	121.3046	1.2802	1.3046
	-120	-121.2544	1.7263	1.2544
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	0	3.5961	0.2655	3.5961
	90	91.5293	1.1558	1.5293
	-90	-91.0784	1.3774	1.0784
	40	41.7624	0	1.7624
	-40	-41.1086	0	1.1086
	120	121.025	0	1.025
	-120	-119.656	0	0.344
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	0	-0.1883	2.6198	0.1883
	90	92.5605	1.2917	2.5605
	-90	-91.0205	0.6984	1.0205
	40	42.4578	0.0882	2.4578
	-40	-41.6847	0.7736	1.6847
	120	123.6243	1.1112	3.6243
	-120	-123.4566	0.5818	3.4566

Tabulka 6.11 obsahuje vzdálenosti a tabulka 6.12 obsahuje rotaci markeru, když kamera byla umístěna ve výšce 111 cm pod úhlem 40°. Na obr. 6.7 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. Jako první se opět zaměříme na směrodatné odchylky vzdáleností, kdy i v tomto případě se blíže spíše víc k nule. U úhlů jsou opět patrné odchylky od průměrné hodnoty, ale nijak velká čísla, která by narušovala důvěryhodnost měření, avšak v tomto případě je vhodné určovat střední hodnotu pro zjištění orientace markeru v obraze. U odchylek průměrných vzdáleností od reálných vidíme, že na vyšší poloha kamery neměla na měření nijak zásadní vliv. Hodnoty jsou stále přesné. Avšak hodnoty úhlů se od reálných liší mnohem více, než tomu bylo u kamery ve výšce 83 cm.

## 6. Experimenty

Tabulka 6.11: Vzdálenosti snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve 40°

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,632	1,5543	0,153	0,0777
	2. bod	1,621	1,6015	0,0271	0,0195
	3. bod	1,637	1,7045	0,3052	0,0675
	4. bod	1,491	1,4804	0,1337	0,0106
	5. bod	1,481	1,4377	0,1303	0,0433
	6. bod	1,506	1,4461	0,0274	0,0599
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	1,632	1,498	0,0673	0,134
	2. bod	1,621	1,6264	0,0178	0,0054
	3. bod	1,637	1,665	0,0301	0,028
	4. bod	1,491	1,4571	0,0018	0,0339
	5. bod	1,481	1,5186	0,0221	0,0376
	6. bod	1,506	1,5255	0,0301	0,0195
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,632	1,6793	0,0655	0,0473
	2. bod	1,621	1,6489	0,0535	0,0279
	3. bod	1,637	1,698	0,1059	0,061
	4. bod	1,491	1,5494	0,0335	0,0584
	5. bod	1,481	1,5354	0,0255	0,0544
	6. bod	1,506	1,5646	0,0263	0,0586

Tabulka 6.12: Rotace snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve 40°

ArUco marker	reálné hodnoty $q$ [°]	střední hodnota $\bar{q}$ [°]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	0,6349	1,1055	0,6349
	90	90,4417	0,991	0,4417
	-90	-91,509	3,4579	1,509
	40	43,0281	0,9744	3,0281
	-40	-43,321	1,6492	3,321
	120	118,3454	0,9666	1,6546
	-120	-117,723	0,9403	2,277
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	0	0,6283	1,4187	0,6283
	90	89,4063	2,1484	0,5937
	-90	-88,2666	0,6649	1,7334
	40	37,6178	0	2,3822
	-40	-39,5292	2,1873	0,4708
	120	117,3312	1,4405	2,6688
	-120	-120,4572	1,2502	0,4572
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	0	-0,3929	0,7563	0,3929
	90	94,2356	1,0672	4,2356
	-90	-92,3144	1,4969	2,3144
	40	41,8001	2,6641	1,8001
	-40	-37,5202	1,4869	2,4798
	120	118,7728	2,678	1,2272
	-120	-123,6177	1,9341	3,6177

Tabulka 6.13 obsahuje vzdálenosti a tabulka 6.14 obsahuje rotaci markeru, když kamera byla umístěna ve výšce 83 cm pod úhlem 25°. Na obr. 6.6 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. Směrodatné odchylky vzdáleností pro tuto polohu kamery jsou velmi blízké nule. Směrodatné odchylky úhlů jsou podobné jako v předešlých tabulkách. Hodnoty se tedy během měření měnily (pohybovaly se kolem střední hodnoty). Odchylky od reálných vzdáleností

jsou velmi malé. Je to dáno změnou naklonění kamery. Ochytky středních hodnot úhlů od reálných jsou opět v přípustném intervalu.

Tabulka 6.13: Vzdálenosti snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve 25°

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,381	1.3426	0.0338	0.0384
	2. bod	1,364	1.412	0.0224	0.048
	3. bod	1,39	1.3774	0.0461	0.0126
	4. bod	1,23	1.2246	0.0265	0.0226
	5. bod	1,22	1.2509	0.1034	0.0309
	6. bod	1,24	1.2491	0.0229	0.0091
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	1,381	1.3683	0.0197	0.0127
	2. bod	1,364	1.3411	0.0165	0.0229
	3. bod	1,39	1.4866	0.2224	0.0966
	4. bod	1,23	1.2116	0.1573	0.0096
	5. bod	1,22	1.2311	0.014	0.0111
	6. bod	1,24	1.2277	0.0244	0.0123
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,381	1.3801	0.0133	0.0009
	2. bod	1,364	1.3827	0.0106	0.0187
	3. bod	1,39	1.3926	0.0211	0.0026
	4. bod	1,23	1.2398	0.0283	0.0378
	5. bod	1,22	1.2473	0.0384	0.0273
	6. bod	1,24	1.267	0.0263	0.027

Tabulka 6.14: Rotace snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve 25°

ArUco marker	reálné hodnoty $q$ [°]	střední hodnota $\bar{q}$ [°]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	-0.1085	0.7877	0.1085
	90	93.0828	2.2772	3.0828
	-90	-93.4083	1.1414	3.4083
	40	41.5852	2.0481	1.5852
	-40	-40.2227	1.6975	0.2227
	120	123.339	1.5515	3.339
	-120	-119.5436	1.1327	0.4564
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	0	-1.0589	2.2021	1.0589
	90	90.8575	1.4409	0.8575
	-90	-93.4312	0.7303	3.4312
	40	37.6624	0.7429	2.3376
	-40	-40.8739	2.4554	0.8739
	120	121.2745	3.0734	1.2745
	-120	-121.3474	2.5311	1.3474
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	0	-0.4724	1.6263	0.4724
	90	93.0734	2.3555	3.0734
	-90	-92.0715	2.2895	2.0715
	40	42.2893	1.0683	2.2893
	-40	-41.0964	2.6824	1.0964
	120	122.8137	1.4383	2.8137
	-120	-122.6775	1.4188	2.6775

Tabulka 6.15 obsahuje vzdálenosti a tabulka 6.16 obsahuje rotaci markeru, když kamera byla umístěna ve výšce 111 cm pod úhlem 25°. Na obr. 6.7 je vidět, jak daleko jsou jednotlivé body od kamery a kde se v mřížce nacházejí. Směrodatné odchylky

## 6. Experimenty

vzdáleností jsou opět velmi blízké nule. Pro úhly tyto hodnoty opět vykazují, že hodnoty se během měření střídaly, neměřila se jedna ustálená hodnota. I když je kamera položena výše díky menšímu úhlu naklonění jsou minimální rozdíly naměřených hodnot od reálných. Odchytky úhlů se nacházejí v povoleném intervalu  $< 0; 10 >$ . Hodnoty jsou tedy dostatečně validní pro učení orientace ArUco markeru v obraze.

Tabulka 6.15: Vzdálenosti snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve  $25^\circ$

ArUco marker	body	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	1,632	1.5849	0.0342	0.0471
	2. bod	1,621	1.5832	0.0234	0.0378
	3. bod	1,637	1.5453	0.0508	0.0917
	4. bod	1,491	1.5219	0.0305	0.0309
	5. bod	1,481	1.4484	0.0211	0.0326
	6. bod	1,506	1.4554	0.0248	0.0506
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	1,632	1.5718	0.1177	0.0602
	2. bod	1,621	1.665	0.0223	0.044
	3. bod	1,637	1.6612	0.0193	0.0242
	4. bod	1,491	1.4537	0.0269	0.0373
	5. bod	1,481	1.4221	0.0726	0.0589
	6. bod	1,506	1.4502	0.0305	0.0558
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	1,632	1.5604	0.1321	0.0716
	2. bod	1,621	1.6829	0.0257	0.0619
	3. bod	1,637	1.6516	0.0219	0.0146
	4. bod	1,491	1.5285	0.0399	0.0375
	5. bod	1,481	1.5094	0.0042	0.0284
	6. bod	1,506	1.5114	0.0057	0.0054

Tabulka 6.16: Rotace snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve  $25^\circ$

ArUco marker	reálné hodnoty $q$ [ $^\circ$ ]	střední hodnota $\bar{q}$ [ $^\circ$ ]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$ průměrné hodnoty od reálných hodnot
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	0	-0.0397	2.677	0.0397
	90	91.2658	1.1289	1.2658
	-90	-92.5538	0.9358	2.5538
	40	41.8118	2.284	1.8118
	-40	-41.1645	2.7902	1.1645
	120	124.0055	0.6285	4.0055
	-120	-121.6996	3.0619	1.6996
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	0	-0.5243	2.8064	0.5243
	90	89.994	1.9926	0.006
	-90	-90.8225	3.0049	0.8225
	40	41.7631	2.0252	1.7631
	-40	-38.8685	2.042	1.1315
	120	121.0708	1.6808	1.0708
	-120	-123.1909	2.0852	3.1909
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	0	0.902	1.9989	0.902
	90	91.8633	1.4463	1.8633
	-90	-92.1845	1.2648	2.1845
	40	39.0196	0.7828	0.9804
	-40	-38.5072	1.0791	1.4928
	120	123.0386	0.9196	3.0386
	-120	-122.3812	1.4007	2.3812



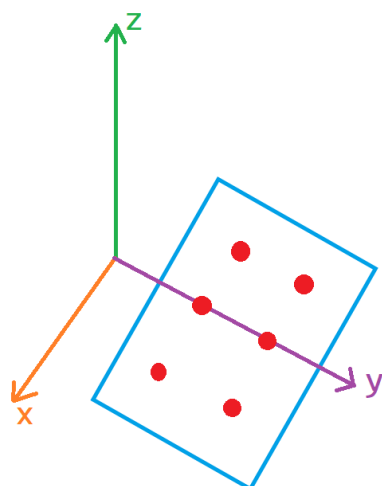
### 6.3.1.3 Vyhodnocení výsledků

V těchto experimentech jsme se zaměřili, jaký vliv má velikost ArUco markeru a poloha kamery na získané výsledky. Měření ukázalo, že druh knihovny ArUco markerů nemá vliv na detekci a učení vzdálenosti od kamery a rotaci markeru. Druhým zkoumaným parametrem byla velikost ArUco markerů. Byly zkoumány tři různé markery: ArUco marker o velikosti 5,8 cm a 12,7 cm z knihovny DICT\_7x7\_250 a ArUco marker o velikosti 5 cm z knihovny DICT\_6x6\_250. Marker o velikosti 12,7 cm měl větší odchylky průměrných hodnot úhlů, než menší markery. Hodnoty zbývajících dvou markeru jsou téměř shodné. Lze proto usuzovat, že vhodná velikost ArUco markeru je v rozsahu od 5 cm do 6 cm. Třetím zkoumaným parametrem byl typ kamery. Po prozkoumání výsledků kamera mobilu měla častější směrodatné odchylky naměřených úhlů a také odchylky průměrných hodnot od reálných. Tento jev v budoucích měřeních nemusí být žádoucí. Výška umístění neměla nijak zásadní vliv na přesnost polohy a orientace markeru. Naklonění kamery v různých výškách snímání markeru buď zpřesnilo, nebo zhoršilo.

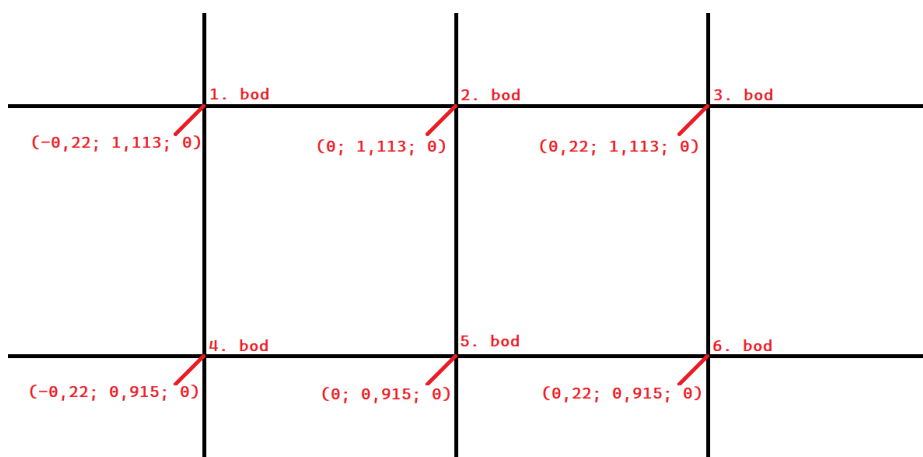
## 6.4 Poloha ArUco markeru v lokálním souřadném systému

Druhým experimentem bylo testování odhadu polohy ArUco markeru v lokálním souřadném systému. Opět jsme si připravili kartonovou desku o velikosti  $88 \times 62$  cm s vyznačenými šesti body jako v předchozím experimentu. Kamera bude uchycena ve výšce 83 cm a vzdálená 69 cm od kraje desky. V této výšce bude také nakloněna do dvou úhlů  $40^\circ$  a  $25^\circ$ . Schéma konstrukce je zaznamenáno na obr. 6.3.

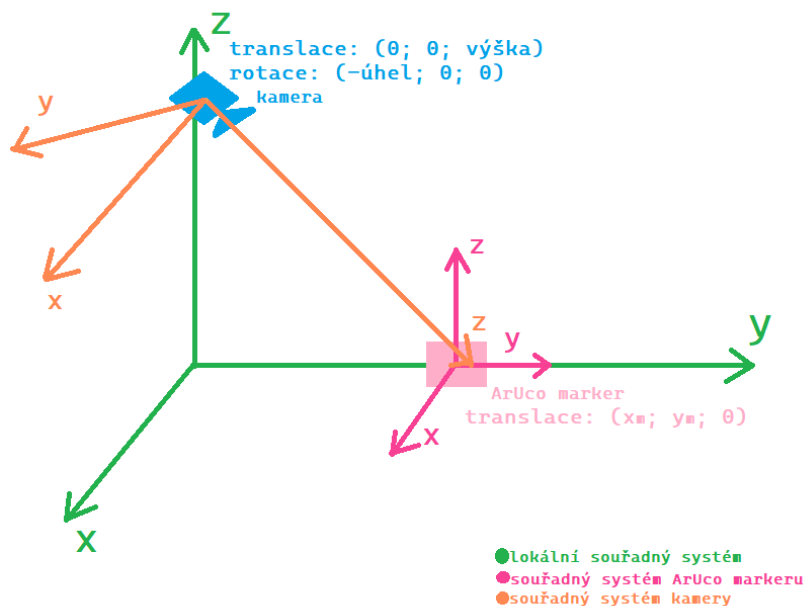
Zavedli jsem lokální souřadný systém. Osa  $z$  míří vzhůru, osa  $y$  protíná desku a osa  $x$  je rovnoběžná s delší hranou desky. Na obr. 6.9 se nachází schéma lokálního souřadného systému a poloha desky. Šesti bodům desky jsme určili jejich souřadnice v rovině  $XY$ , které jsou znázorněny na obr. 6.10. Hodnoty jsou udávány v metrech. Ještě je potřeba definovat souřadný systém nakloněné kamery, který je znázorněn již v sekci Kalibrace kamery 2.2 na obr. 2.4, a ArUco markeru, který je popsán v sekci 4.2. Na obr. 6.11 jsou všechny souřadné systémy viditelné.



Obrázek 6.9: Schéma lokální souřadného systému s deskou o šesti bodech



Obrázek 6.10: Souřadnice pro šest bodů na desce



Obrázek 6.11: Schéma souřadných systémů - lokální souřadný systém, souřadný systém ArUco markeru a kamery

Nejdříve je potřeba vytvořit transformační matici, aby bylo možné převést vektory ze souřadného systému kamery do lokálního souřadného systému. Pomocí známé výšky lze určit translační vektor kamery  $\mathbf{t}_K = (0; 0; -0,83)$ . Rotační vektory budou dva pro každé naklonění kamery  $\mathbf{r}_K = (40^\circ - 90^\circ; 0; 0)$  a  $\mathbf{r}_K = (25^\circ - 90^\circ; 0; 0)$ . Kamera je nakloněna o daný úhel do země a je ji potřeba vrátit do kolmé polohy, proto volíme zmíněné úhly. Rotace probíhá kolem osy  $x$ . Pro vytvoření rotační matice kamery se bude využívat vztah (4.4) z kapitoly 4. V tomto případě bude úhel  $\Theta = 40^\circ - 90^\circ$ , nebo  $\Theta = 25^\circ - 90^\circ$  v závislosti na úhlu naklonění kamery. Ze znalosti transformačního vektoru kamery  $\mathbf{t}_K$  a vypočítané rotační matice můžeme sestavit transformační matici podle vztahu (4.11). Nasnímané vektory translace  $\mathbf{t}_{\text{vec}}$  z programu 5.1 se rozšíří o jedničku a přepočtou se transformační maticí do lokálního souřadného systému.

Každá souřadnice ArUco markeru je snímána  $20\times$ . V tabulkách níže budou vypočítány střední hodnoty  $\bar{q}$  podle vztahu (6.1), směrodatné odchytky  $\sigma$  pomocí vztahu (6.2) a odchytky  $\tilde{q}$  střední hodnoty  $\bar{q}$  od reálné hodnoty  $q$ , které jsou dány vztahem (6.3).

V tabulce 6.17 a 6.18 jsou zapsány souřadnice  $x$ ,  $y$  a  $z$  tří testovaných ArUco markerů. Směrodatné odchytky všech naměřených hodnot jsou minimální a v některých případech jsou rovny nule. Hodnoty jsou tedy snímány konstantně.

Nejdříve se zaměříme na tabulku, kdy kamera byla nakloněna o  $40^\circ$ . Pro horní

první dva body mřížky na desce vyšly souřadnice z blízké nule pro ArUco marker o velikosti 5,8 cm. U pravého horního rohu je už detekovaná větší odchylka naměřených hodnot od reálných. Souřadnice z pro tři spodní body mají též větší odchylky naměřených hodnot od reálných. Je to dáno právě úhlem naklonění kamery. Pokud je kamera ve výšce 83 a střed (0; 0; 0) lokálního souřadného systému je vzdálen od hrany desky 69 cm, viz obr. 6.3, dalo by se říct, že "kamera se přímo dívá na horní tři body". U markeru o velikosti 12,7 cm pozorujeme větší odchylky od reálných hodnot. Potvrzuje nám to tvrzení, které vyplynulo z předešlého experimentu a to, že je vhodné volit ArUco marker o velikosti, jenž se pohybuje mezi 5 cm a 6 cm.

V druhé tabulce, kdy kamera byla nakloněna o  $25^\circ$ , pozorujeme opačný jev. Kamera je teď spíše zaměřena na tři spodní body. Souřadnice z pro markery položené v horních třech bodech mají viditelnou větší odchylku  $\tilde{q}$ . Avšak pokud se zaměříme na marker o velikosti 12,7 cm, vidíme malé odchylky hodnot pro horní body.

V tomto experimentu jsme se více zaměřili na naklonění kamery a velikost markeru. Zde byl patrný vliv naklonění kamery i velikost markeru. Pokud jsme měli větší úhel, kamera mířila na horní řadu bodů, ve kterých byly markery snímány. Měření ukázalo, že byl vhodnější menší marker. Oproti tomu menší úhel naklonění se zaměřil spíše na dolní řadu bodů, v této poloze byly opět dobře detekovány menší markery. Experiment dále ukázal, že v tomto naklonění kamery byl i dobře detekován větší marker o velikosti 12,7 cm a to v horních třech bodech.

#### 6.4. Poloha ArUco markeru v lokálním souřadném systému

Tabulka 6.17: Souřadnice translace ArUco markeru snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 40°

ArUco marker	body	souřadnice ArUco markeru	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	x	-0,22	-0,1077	0,0015	0,1123
		y	1,113	1,0272	0,013	0,0858
		z	0	-0,0163	0,0105	0,0163
	2. bod	x	0	0,0519	0	0,0519
		y	1,113	0,9576	0	0,1554
		z	0	-0,0787	0	0,0787
	3. bod	x	0,22	0,2366	0,0033	0,0166
		y	1,113	0,8071	0,0119	0,3059
		z	0	-0,201	0,0087	0,201
	4. bod	x	-0,22	-0,0989	0,0007	0,1211
		y	0,915	1,1068	0,0111	0,1918
		z	0	-0,2196	0,0065	0,2196
5. bod	x	0	0,0694	0,0006	0,0694	
	y	0,915	1,0355	0,0129	0,1205	
	z	0	-0,2196	0,0064	0,288	
6. bod	x	0,22	0,2837	0	0,0637	
	y	0,915	1,1144	0	0,1994	
	z	0	-0,2617	0	0,2617	
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	x	-0,22	-0,2207	0,0005	0,0007
		y	1,113	1,0995	0,002	0,0135
		z	0	-0,1043	0,0012	0,1043
	2. bod	x	0	0,011	0,0003	0,011
		y	1,113	1,0845	0,0038	0,0285
		z	0	-0,1126	0,0025	0,1126
	3. bod	x	0,22	0,2401	0,0005	0,0201
		y	1,113	1,1151	0,003	0,0021
		z	0	-0,1051	0,0017	0,1051
	4. bod	x	-0,22	-0,2455	0,004	0,0255
		y	0,915	1,1281	0,0056	0,2131
		z	0	-0,2511	0,0026	0,2511
5. bod	x	0	0,0091	0,0014	0,0091	
	y	0,915	1,0333	0,0054	0,1183	
	z	0	-0,3096	0,0107	0,3096	
6. bod	x	0,22	0,2663	0,0013	0,0463	
	y	0,915	1,1801	0,0055	0,2651	
	z	0	-0,2419	0,0041	0,2419	
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	x	-0,22	-0,1965	0,0014	0,0235
		y	1,113	0,8786	0,0061	0,2344
		z	0	-0,0794	0,0052	0,0794
	2. bod	x	0	0,0198	0,0001	0,0198
		y	1,113	0,9087	0,0081	0,2043
		z	0	-0,0623	0,0069	0,0623
	3. bod	x	0,22	0,2589	0,0015	0,0389
		y	1,113	1,0408	0,0069	0,0722
		z	0	0,0497	0,0059	0,0497
	4. bod	x	-0,22	-0,196	0	0,024
		y	0,915	0,8409	0	0,0741
		z	0	-0,2738	0	0,2738
5. bod	x	0	0,0218	0	0,0218	
	y	0,915	0,8938	0,0002	0,0212	
	z	0	-0,2479	0,0003	0,2479	
6. bod	x	0,22	0,2555	0,0058	0,0355	
	y	0,915	1,0268	0,024	0,1118	
	z	0	-0,1485	0,0159	0,1485	

## 6. Experimenty

Tabulka 6.18: Souřadnice translace ArUco markeru snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 25°

ArUco marker	body	souřadnice ArUco markeru	reálné hodnoty $q$ [m]	střední hodnota $\bar{q}$ [m]	směrodatná odchylka $\sigma$	odchylka $\bar{q}$
ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250	1. bod	x	-0,22	-0,0825	0	0,1375
		y	1,113	1,1437	0,0003	0,0307
		z	0	0,2121	0,0004	0,2121
	2. bod	x	0	0,1485	0	0,1485
		y	1,113	1,1933	0	0,0803
		z	0	0,2428	0	0,2428
	3. bod	x	0,22	0,3826	0,001	0,1626
		y	1,113	1,2292	0,0037	0,1162
		z	0	0,2489	0,0034	0,2489
	4. bod	x	-0,22	-0,0883	0,0008	0,1317
		y	0,915	1,0425	0,0101	0,1275
		z	0	-0,0361	0,0075	0,0361
5. bod	x	0	0	0,0009	0,1367	
	y	0,915	1,1185	0,0058	0,2035	
	z	0	0,0071	0,0043	0,0071	
6. bod	x	0,22	0,3689	0	0,1489	
	y	0,915	1,1467	0	0,2317	
	z	0	0,0081	0	0,0081	
ArUco marker o velikosti 12,7 cm z knihovny DICT_7x7_250	1. bod	x	-0,22	-0,1477	0	0,0723
		y	1,113	1,1305	0,0002	0,0175
		z	0	-0,0103	0	0,0103
	2. bod	x	0	0,0783	0,0005	0,0783
		y	1,113	1,124	0,0041	0,011
		z	0	-0,0146	0,0029	0,0146
	3. bod	x	0,22	0,3192	0,0011	0,0992
		y	1,113	1,1724	0,0033	0,0594
		z	0	0,0051	0,0024	0,0051
	4. bod	x	-0,22	-0,165	0,002	0,055
		y	0,915	1,0722	0,0041	0,1572
		z	0	-0,201	0,0022	0,201
5. bod	x	0	0,2448	0,1067	0,2448	
	y	0,915	1,1436	0,0407	0,2286	
	z	0	-0,0437	0,0974	0,0437	
6. bod	x	0,22	0,3231	0	0,1031	
	y	0,915	1,1732	0	0,2582	
	z	0	0,0104	0	0,0104	
ArUco marker o velikosti 5 cm z knihovny DICT_6x6_250	1. bod	x	-0,22	-0,0938	0	0,1262
		y	1,113	0,9084	0	0,2046
		z	0	0,0676	0	0,0676
	2. bod	x	0	0,1259	0,001	0,1259
		y	1,113	0,9511	0,0063	0,1619
		z	0	0,0945	0,0063	0,0945
	3. bod	x	0,22	0,3551	0,0031	0,1351
		y	1,113	1,0021	0,0087	0,1109
		z	0	0,113	0,0083	0,113
	4. bod	x	-0,22	-0,0953	0	0,1247
		y	0,915	0,8217	0	0,0933
		z	0	-0,1528	0	0,1528
5. bod	x	0	0,1187	0,001	0,1187	
	y	0,915	0,8931	0,0074	0,0219	
	z	0	-0,1117	0,0062	0,1117	
6. bod	x	0,22	0,3406	0,0015	0,1206	
	y	0,915	0,9289	0,0042	0,0139	
	z	0	-0,1105	0,0033	0,1105	

Na začátku této bakalářské práce jsme se podívali na zpracování obrazu a kalibraci kamery. Vyřešili jsme problematiku převedení RGB obrázku na obrázek ve stupnici šedi za použití jedné z monadických funkcí. Abychom mohli pokračovat s detekcí markeru, je potřeba kameru kalibrovat. Díky tomu získáme parametry kamery, které v další části využijeme. Na konci kapitoly byl přidán program napsaný v jazyce C++ s využitím knihovny OpenCV. Ten umožňuje zobrazit snímání obrazu kamery, který převede do stupnic šedi. Po tomto kroku běží algoritmus kalibrace. Kalibrace se provádí pomocí snímání šachovnice.

Od kalibrace jsme se přesunuli k samotným fiduciálním značkám. V kapitole byly představeny tři druhy: AprilTag, STag a ArUco marker. Pro bakalářskou práci byly vybrány ArUco markery, jejichž detekci je ve stejné kapitole také popsána. Na závěr sekce jsme opět přidala kód psaný v jazyce C++ s použitím knihovny OpenCV, který umí detekovat ArUco marker.

Dále bylo potřeba si objasnit problematiku transformací 3D souřadných systémů. V této práci se setkáváme se dvěma souřadnými systémy: kamery a ArUco markeru. V kapitole věnované souřadným systémům jsme si je definovali. Byly zde zmíněny translační vektory, rotační matice a transformační matice, pomocí kterých lze převádět souřadnice bodů mezi dvěma souřadnými systémy.

Cílem bakalářské práce je odhad polohy a orientace v obraze a prostoru fiduciální značky. Používaná knihovna OpenCV obsahuje funkce, který tento problém řeší. Byla zde popsána metoda `estimatePoseSingleMarkers`, která je součástí knihovny OpenCV a vrací translační a rotační vektor Aruco markeru v souřadném systému kamery. Druhá část kapitoly se zabývala programem napsaným v jazyce C++, který odhadoval parametry jak v souřadném systému kamery, tak i v lokálním souřadném systému.

Poslední část celé práce je věnována experimentům. Nejdříve se definovaly použité ArUco markery a kamery. První experiment byl věnován odhadu polohy a orientace ArUco markeru v souřadném systému kamery. Zde se do detailu popsalo, jakým způsobem se měřilo. Definovalo se, v jaké výšce a naklonění byla umístěná kamera. ArUco marker byl položen na desce, na které bylo vyznačeno šest bodů.

Zkoumali jsme statický marker, jenž jsme v jedné poloze snímali 20×. Nakonec jsme vyhodnotili výsledky svého měření. Bylo zjištěno, že druh knihovny ArUco markerů nemá vliv na detekci markeru a jeho určení polohy a natočení v obraze. Dále jsem definovala ideální velikost pro sledování ArUco markeru a typ kamery, který by se pro tuto úlohu měl použít. Druhý experiment byl věnován odhadu polohy ArUco markeru v lokálním souřadném systému. Kamera byla umístěna v jedné výšce a zkoumaly se dva úhly jejího natočení. Oba experimenty ukázaly, že velmi záleží na volbě naklonění kamery vůči velikosti markeru. Pokud bychom měli menší úhel naklonění kamery a náš ArUco marker by se měl pohybovat na okraji zorného pole kamery, je vhodnější volit větší marker. V opačném případě, kdy kamera přímo míří na marker, je vhodné volit ArUco marker menších velikostí. Tyto dva aspekty závisí na typu experimentu.

Tato bakalářská práce by se pak dala rozšířit několika způsoby. Přesnost měření by se dala zvýšit přidáním jedné, či více kamer. ArUco marker by tam byl snímán z několika poloh. Dále by se dal program upravit tak, aby místo statického markeru sledoval dynamický, a také aby sledoval více markerů, než jen jeden. V rozšířených úlohách by se pak mohl aplikovat Kalmanův filtr.



# Knihovna OpenCV



Celý program je psán v jazyce C++, ve kterém se používala knihovna OpenCV. Ta je určena pro počítačové vidění a strojové učení. Má velkou škálu funkcí pro počítačové vidění a zpracování obrazu, včetně detekce a rozpoznávání objektů, sledování pohybu, kalibrace kamer, segmentace obrazu, stereo vidění, rozpoznávání tváří, extrakce rysů, manipulaci s obrazem a mnoho dalšího. Knihovna je velmi dobrá pro práci s obrazem, snímaném v reálném čase [9]. Aby náš program byl schopný mít tyto všelijaké funkce, potřebujeme moduly, jenž obsahují funkce pro danou oblast. Zde je seznam několika modulů [4].:

- **core** - Tento modul obsahuje základní datové struktury, funkce a algoritmy pro zpracování obrazu, včetně načítání a ukládání obrazů, manipulace s maticemi, operace s barevnými prostory a základní aritmetiku obrazů.
- **imgproc** - Tento modul poskytuje širokou škálu funkcí pro zpracování obrazu, jako je vyhlazování, filtraci, morfologické operace, detekce hran, prahování, transformace obrazu a další.
- **highgui** - Tento modul pracuje s uživatelským rozhraním, jako je načítání a zobrazování obrazů, vytváření oken, zachytávání videa a práce s klávesnicí a myší.
- **video** - Tento modul obsahuje funkce pro analýzu a zpracování videa, včetně sledování pohybu objektů, odstranění šumu, odhad pohybu, segmentace videa a dalších technik zpracování videa
- **videoio** - Tento modul poskytuje funkce pro práci se vstupním a výstupním souborem videa. Lze díky němu načítat videa ze souborů, zařízení nebo streamů a také zapisovat videa do souborů.
- **features2D** - Tento modul obsahuje algoritmy pro detekci a popis rysů v obraze, jako je SIFT (Scale-Invariant Feature Transform), SURF (Speeded-Up Robust Features) a ORB (Oriented FAST and Rotated BRIEF). Tyto algoritmy jsou často používány pro rozpoznávání objektů a sledování funkcí.

- **objdetect** - Tento modul je zaměřen na detekci objektů v obraze, jako jsou obličeje, oči, ústa, lidé, auta atd. Obsahuje také trénované klasifikátory, jako jsou Haarovské kaskády, které mohou být použity pro detekci specifických objektů.
- **calib3d** - Tento modul je určen pro kalibraci kamer, rekonstrukci 3D scény z páru stereo obrazů, estimaci pohybu kamery a další geometrické operace.
- **tracking** - Tento modul je určen pro sledování objektů v sekvencích videa. Obsahuje implementace různých algoritmů pro sledování, jako je MOSSE (Minimum Output Sum of Squared Error), KCF (Kernelized Correlation Filters) a další.
- **cuda** - Jedná se o modul pro paralelní zpracování na grafických kartách NVIDIA pomocí technologie CUDA. Poskytuje optimalizované implementace algoritmů z OpenCV, které využívají výhody paralelního zpracování na GPU.
- **aruco** - Tento modul je zaměřen na detekci a sledování ArUco markerů, které jsou používány v rozšířené realitě a počítačovém vidění pro značkování a identifikaci objektů.
- **xfeatures2D** - Tento modul rozšiřuje modul Features2D a poskytuje pokročilejší algoritmy pro detekci a popis rysů, jako je SIFT, SURF, STAR (Scale and Rotation Invariant Feature Transform) a další.

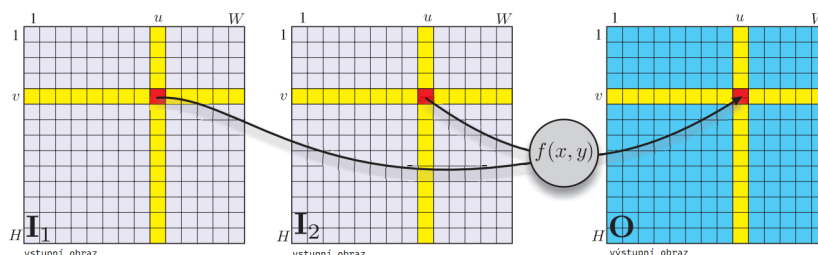
OpenCV má rozhraní pro C++, MATLAB, Python a Java. Podporuje Windows, Linux, Android a Mac OS. Knihovna je nativně psána v C++ a má šablonované rozhraní [9].

# Diadické operace pro zpracování obrazu

## B

Tato kapitola bude pojednávat o diadických operacích pro zpracování obrazu. Budou zde zmíněny různé druhy těchto operací. V této bakalářské práci se využívají funkce z knihovny OpenCV, které obsahují metody pro detekci rohů a hran.

Diadické operace fungují následovně. Existují dva vstupní obrazy o stejné velikosti a výstupem je jeden obraz o stejné velikosti. Výstupní pixely jsou funkcí pixelů ve vstupních obrazech  $\rightarrow \mathbf{O}[u, v] = f(\mathbf{I}[u, v]), \forall (u, v) \in \mathbf{I}$ , viz obr. B.1 [3]. Příkladem diadických operací můžou být binární aritmetické operace (sčítání, odčítání, násobení po prvcích, apod.).



Obrázek B.1: Zpracování obrazu pomocí diadické operace

Jedna z metod třídy diadických operací je barevné klíčování, kdy překrýváme obrázky přes sebe. Metoda se hojně využívá v televizi, např. moderátor počasí stojí před mapou ukazující počasí. Objekt je nasnímán před zeleným, nebo modrým pozadím. Díky tomu jsou jasně rozlišitelné pixely, co je objekt a co je pozadí. Snímek se načte, použije se gama dekódování, abychom získali lineární zobrazení. Pak se vypočtou jeho barevné souřadnice. Histogramem určíme hranice pixelů, kde se nachází objekt a kde pozadí. Díky tomu vytvoříme masku obrazu. Ta obsahuje jen dvě hodnoty true a false, kdy hodnoty true jsou znázorněny jako bílé a značí, že se daný pixel nachází v objektu, hodnoty false jsou černé a značí pixely, které jsou součástí pozadí. Maska se aplikuje na všechny tři barevné roviny obrazu. Teď je potřeba vybrat a upravit požadující pozadí. To se načte a upraví na stejnou velikost jako je obraz objektu.

Nakonec tyto dva obrazy spojíme  $\rightarrow$  *matice objektu*  $\cdot$  *matice masek* + *matice pozadí*  $\cdot$  (1-*matice masek*). Tento postup není úplně ideální, pokud objekt obsahuje barvy, které odpovídají barvě pozadí. Tento problém se dá vyřešit pomocí metody přepínání pixelů, kde se porovnávají pixely obrázku s objektem a obrázku s pozadím s pixely masky.

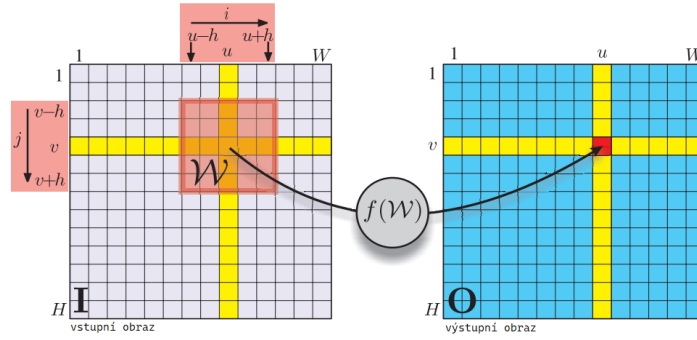
Rozpoznávání objektů od pozadí je velmi důležitý problém k řešení v robotickém vidění. Je náročné říct, co přesně je popředí a co pozadí obrazu. V robotice nelze využít metody barevného klíčování. Jedna z možností je nasnímat pozadí bez objektu, ale vyžaduje to znalost navíc v tom, že potřebujeme informaci o tom, kdy se objekt objeví na scéně a následně zmizí ze scény. Taky v tomto případě uvažujeme neměnnost pozadí. V reálném světě je problém, že se mění osvětlení a stíny v krátkých intervalech, scéna se tedy mění. V tomto případě se bude zpracovávat sekvence scén a bude se odhadovat pozadí, i když se ve scéně bude objevovat několik objektů. Jedná se o rekurzivní algoritmus, který bude aktualizovat odhadované pozadí obrázku  $\widehat{\mathbf{B}}$ . Odhad bude probíhat v každém časovém okamžiku na základě předchozího odhadu a aktuálního obrazu scény (B.1) [3]

$$\widehat{\mathbf{B}} \langle k+1 \rangle \leftarrow \widehat{\mathbf{B}} \langle k \rangle + c(\mathbf{I} \langle k \rangle - \widehat{\mathbf{B}} \langle k \rangle), \quad (\text{B.1})$$

kde  $k$  je časový krok a  $c()$  je monadická funkce saturace obrazu (B.2) [3]

$$c(x) = \begin{cases} \sigma, & \text{pro } x > \sigma \\ x, & \text{pro } -\sigma \leq x \leq \sigma \\ -\sigma, & \text{pro } x < -\sigma \end{cases} \quad (\text{B.2})$$

Zpracování obrazů může probíhat také pomocí sady prostorových operací. Pixely výstupního obrázku jsou funkcí všech pixelů v oblasti obklopující odpovídající pixel ve vstupním obrázku  $\rightarrow \mathbf{O}[u, v] = f(\mathbf{I}[u+i, v+j]), \forall (i, j) \in W, \forall (u, v) \in \mathbf{I}$ .  $W$  je okénko ve tvaru čtverce o velikosti  $w \times w$ .  $w = 2h+1$ , kde  $h \in \mathbb{Z}^+$  je poloviční šířka, viz obr. B.2 [3]. Funkce  $f()$  je velmi rozmanitá. Lze použít lineární, či nelineární.



Obrázek B.2: Zpracování obrazu pomocí prostorové operace

Nejprve se budeme zabývat lineární prostorovou filtrací (B.3), jedná se o korelaci [3]

$$\mathbf{O}[u, v] = \sum_{(i,j) \in \mathcal{W}} \mathbf{I}[u + i, v + j] \mathbf{K}[i, j], \forall (u, v) \in \mathbf{I}, \quad (\text{B.3})$$

kde  $\mathbf{K}$  je jádro a prvky se označují jako koeficienty filtru. Pro každý výstupní pixel se odpovídající okno pixelů násobí elementárně jádrem  $\mathbf{K}$ . Střed okna a jádra má souřadnice  $(0, 0)$  a  $i, j \in \langle -h, h \rangle \subset \mathbb{Z} \times \mathbb{Z}$ . Považujeme to za vážený součet pixelů v okně. Váhy jsou definovány jádrem  $\mathbf{K}$ . Korelace se často zapisuje jako vztah (B.4) [3]

$$\mathbf{O} = \mathbf{K} \otimes \mathbf{I} \quad (\text{B.4})$$

a konvoluce je dána vztahem (B.5) [3]

$$\mathbf{O}[u, v] = \sum_{(i,j) \in \mathcal{W}} \mathbf{I}[u - i, v - j] \mathbf{K}[i, j], \forall (u, v) \in \mathbf{I}, \quad (\text{B.5})$$

kde  $\mathbf{K} \in \mathbb{R}^{w \times w}$  je jádro konvoluce. Znaménko u indexů  $i$  a  $j$  se změnilo. Konvoluci tedy lze psát jako vztah (B.6) [3]

$$\mathbf{O} = \mathbf{K} * \mathbf{I}. \quad (\text{B.6})$$

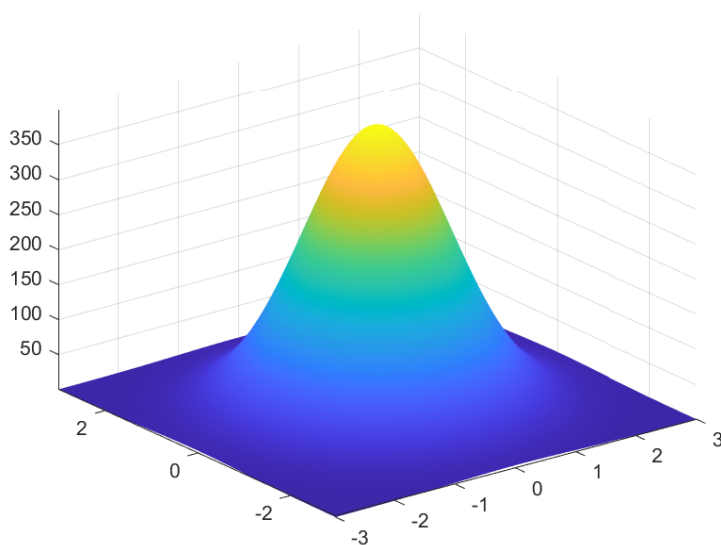
Konvoluce je velmi důležitá při zpracování obrazu a jádro  $\mathbf{K}$  může být voleno v závislosti na volbě na různých funkcích, které se mají provádět - vyhlazování, výpočet gradientu, detekce hran. Konvoluce je výpočetně náročná pro vstupní obraz o velikosti  $N \times N$  a jádrem  $w \times w$  je potřeba  $w^2 N^2$  násobení a sčítání. Pokud má

vstupní obraz více barevných rovin, tak i výstupní obraz má více rovin a každá rovina je konvolucí té vstupní a jádrem  $\mathbf{K}$ .

Zaměříme se na vyhlazování. Máme čtvercové jádro  $\mathbf{K}$  o velikosti  $N \times N$  a o jednotkovém objemu. To znamená, že prvky jádra jsou v součtu rovny jedné. Výsledkem konvoluce vstupního obrazu s tímto jádrem je obraz, kde každý výstupní pixel je průměrem pixelů v odpovídajícím  $N \times N$  okolí vstupního obrázku. Průměrování vede k vyhlazení, rozmazání, nebo rozostření výstupního obrazu. Pokud použijeme takovéto čtvercové jádro může dojít k efektu tzv. zvonění. Ve výstupním obrazu mohou být slabě viditelné svíslé a vodorovné čáry. Aby se k takovému jevu předešlo, je vhodné volit jádro pro vyhlazování jako dvoudimenzionální Gaussovu funkci (B.7) [3]

$$\mathbf{G}(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}, \quad (\text{B.7})$$

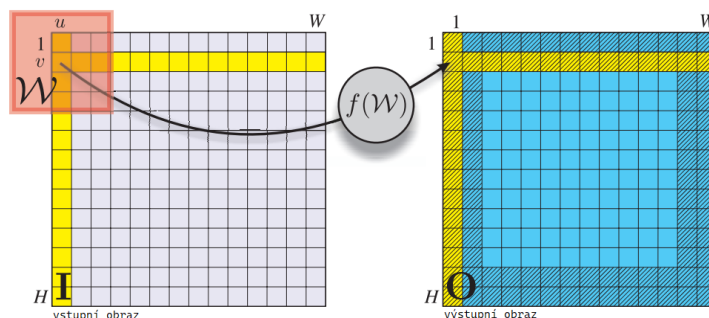
která je symetrická podle počátku a objem pod křivkou je jednotkový, jak lze vidět na obr. B.3. Šířka Gaussovy funkce je dána parametrem směrodatné odchylky  $\sigma$ .



Obrázek B.3: Gaussova funkce

Teď jsme se bavili o tom, kdy okno kolem určitého pixelu je uprostřed obrazu. Problém nastává, když se objeví blízko okraje vstupního obrázku. Pak jsou výstupní pixely funkcí okna, které obsahuje pixely za okrajem vstupního obrázku – tyto pixely nemají žádnou definovanou hodnotu, viz obr. B.4 [3]. Tento problém s neznalostí hodnot pixelů za hranicí obrázku lze řešit několika možnostmi. Za prvé, můžeme předpokládat, že pixely za obrázkem mají určitou hodnotu. Běžná volba je nula.

Okraje vyhlazeného obrázku jsou tmavé kvůli vlivu těchto nul. Další možností je udat, že výsledek je neplatný, pokud okno překročí hranici obrázku. Výsledkem je výstupní obraz o velikosti  $(W - 2h) \times (H - 2h)$ . Je o něco menší než vstupní obraz. Na obr. B.4 jsou vyšrafované pixely ty, co neplatí [3].



Obrázek B.4: Metoda detekce hran

Problém detekce hran je velmi častý. Je praktické si zjistit hodnoty pixelů podél jednorozměrného průřezu v obraze. Derivace prvního řádu podél tohoto průřezu je dána vzorcem (B.8) [3]

$$p'[v] = p[v] - p[v - 1], \quad (\text{B.8})$$

kde  $p$  je vektor hodnot pixelů tohoto průřezu a  $v$  je daný bod, kde derivaci zjišťujeme. Takovou derivaci v daném bodě lze také zapsat jiným vzorcem (B.9) a to jako symetrický podíl prvního řádu [3]

$$p'[v] = \frac{1}{2}(p[v + 1] - p[v - 1]), \quad (\text{B.9})$$

což je ekvivalent ke konvoluci jednorozměrného jádra (B.10) [3]

$$\mathbf{K} = \begin{pmatrix} \frac{1}{2} & 0 & -\frac{1}{2} \end{pmatrix}. \quad (\text{B.10})$$

Při konvoluci jádra s obrázkem jsou jasně vidět svislé okraje, vysoké horizontální gradienty. Pro výpočet horizontálního gradientu se vytvořilo několik konvolučních jader. Jedno z jader je např. jádro Sobel (B.11) [3].

$$\text{Sobel} = \begin{pmatrix} 0,125 & 0 & -0,125 \\ 0,250 & 0 & -0,250 \\ 0,125 & 0 & -0,125 \end{pmatrix}. \quad (\text{B.11})$$

Je vidět, že každá řádka jádra Sobel je zmenšenou verzí jádra popsaného v rovnici (B.10) [3]. Celkovým výsledkem je vážený součet horizontálního gradientu pro aktuální řádek a řádky nad a pod.

Značení gradientů se v literatuře různě mění. Nejčastěji se horizontální a vertikální gradient značí jako  $\rightarrow \frac{\partial I}{\partial u}, \frac{\partial I}{\partial v}, \nabla_u I, \nabla_v I, I_u, I_v$ . Gradienty zapíšeme pomocí operátorů jako (B.12) [3]

$$\begin{aligned} I_u &= \mathbf{D} * \mathbf{I} \\ I_v &= \mathbf{D}^T * \mathbf{I}. \end{aligned} \quad (\text{B.12})$$

$\mathbf{D}$  je zde derivace jádra.

Pokud se použije derivace signálu, tak se zvýrazní vysokofrekvenční šum. Všechny obrázky mají šum. Na úrovni pixelů je stacionární náhodný šum. Hodnoty mezi pixely nekorelují, ale hrany mají korelované změny v hodnotě pixelu ve větším prostorovém měřítku. Vliv šumu můžeme snížit vyhlazením obrazu, dáno vztahem (B.13), před použitím derivace [3].

$$I_u = \mathbf{D} * (\mathbf{G}(\sigma) * \mathbf{I}). \quad (\text{B.13})$$

Místo konvoluce obrazu pomocí Gaussovy funkce a následné derivace se využívá asociativní vlastnost konvoluce k zápisu, proto lze zapsat předchozí vztah jako (B.14) [3]

$$I_u = \mathbf{D} * (\mathbf{G}(\sigma) * \mathbf{I}) = \underbrace{(\mathbf{D} * \mathbf{G}(\sigma))}_{DG} * \mathbf{I}. \quad (\text{B.14})$$

$DG$  je derivace Gaussovy funkce. Ta jde do konvoluce s obrázkem. Analyticky se derivace Gaussovy funkce ve směru  $u$  zapíše jako (B.15) [3]

$$\mathbf{G}_u(u, v) = -\frac{u}{2\pi\sigma^4} e^{-\frac{u^2+v^2}{2\sigma^2}}. \quad (\text{B.15})$$

Směrodatná odchylka  $\sigma$  ovlivňuje měřítko hran, které jsou detekovány. Pro vysoké  $\sigma$  je vyšší vyhlazení. Hrany díky tomuto vyhlazení budou zeslabeny a zůstanou viditelné jen hrany velkých prvků. Tato funkce pro nalézání hran v různém prostorovém měřítku je důležitá a je základem konceptu měřítkového prostoru. Další interpretací tohoto operátoru může být, že je brán jako prostorový pásmový filtr. Jedná se o kaskádu dolní propusti (vyhlazení) s horní propustí (rozlišení).

Velmi účinným je detektor hran Canny. Používá velikost a směr hrany a pracuje ve dvou krocích. Prvním krokem je potlačení nelokálního maxima. Druhý



krok je prahování hystereze, což znamená, že pro každý nenulový pixel, který překročí horní práh, se vytvoří řetězec sousedních pixelů, které překročí spodní práh. Všechny ostatní pixely jsou nastaveny na nulu.

Doteď jsme uvažovali, že hrany jsou body s vysokým gradientem a pro vyhledávání maxim v blízkém okolí byla využita metoda potlačení nelokálních maxim. Jiným způsobem, jak najít bod maximálního gradientu, je vypočítat druhou derivaci a určit, kde tato nula je. Využijeme Laplaceův operátor (B.16) [3]

$$\nabla^2 \mathbf{I} = \frac{\partial^2 \mathbf{I}}{\partial u^2} + \frac{\partial^2 \mathbf{I}}{\partial v^2} = \mathbf{I}_{uu} + \mathbf{I}_{vv}. \quad (\text{B.16})$$

Jedná se o součet druhých derivací v horizontálním a vertikálním směru. Pro diskrétní obrázky lze vypočítat konvoluci s Laplaceovým jádrem, viz (B.17) [3]

$$\text{Laplace} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}. \quad (\text{B.17})$$

Jedná se o izotropní jádro - na hrany jakéhokoliv směru reaguje stejně. Druhá derivace je ještě víc citlivá na šum než první a opět se běžně používá ve spojení s obrazem vyhlazeným pomocí Gaussovské funkce, která je dána rovnicí (B.18) [3]

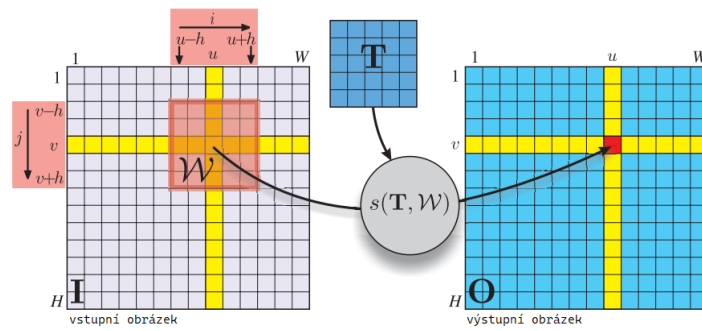
$$\nabla^2 \mathbf{I} = \mathbf{L} * (\mathbf{G}(\sigma) * \mathbf{I}) = \underbrace{(\mathbf{L} * \mathbf{G}(\sigma))}_{\text{LGJ}} * \mathbf{I}, \quad (\text{B.18})$$

kde  $\text{LGJ}$  znamená: Laplaceovo jádro a  $\mathbf{L}$  je Laplaceovo jádro (B.17). Lze to přepsat do analytické podoby (B.19) [3]

$$\text{LGJ}(u, v) = \frac{\partial^2 \mathbf{G}}{\partial u^2} + \frac{\partial^2 \mathbf{G}}{\partial v^2} = \frac{1}{\pi \sigma^4} \left( \frac{u^2 + v^2}{2\sigma^2} - 1 \right) e^{-\frac{u^2 + v^2}{2\sigma^2}}. \quad (\text{B.19})$$

Tato rovnice je známá jako Marrovo-Hildrethův operátor, nebo jako jádro waveletu Mexican hat.

Dále se zaměříme na jiné typy jader. Budeme uvažovat, že jsou to obrázky, či části obrázků, a budeme je nazívat šablonou. Při porovnávání šablon chceme zjistit, které části vstupního obrázku jsou nejvíce podobné šabloně. Lze postup porovnání vidět na obr. B.5 [3].



Obrázek B.5: Princip porovnání šablon

Každý výstupní pixel je dán vztahem (B.20) [3]

$$\mathbf{O}[u, v] = s(\mathbf{T}, \mathcal{W}), \forall (u, v) \in \mathbf{I}, \quad (\text{B.20})$$

kde  $\mathbf{T}$  je šablona velikosti  $w \times w$  a délka hrany je  $w = 2h + 1$ .  $\mathcal{W}$  je okénko o velikosti také  $w \times w$  a se středem v bodě  $(u, v)$  ve vstupním obrázku. Funkce  $s(\mathbf{I}_1, \mathbf{I}_2)$  je skalární míra, která popisuje podobnost dvou stejně velkých obrázků  $\mathbf{I}_1$  a  $\mathbf{I}_2$ . K určení nejlepší shody lze použít hodnocení míry podobnosti.

# Metoda RANSAC pro odhad polohy a orientace objektu na základě obrazových dat



V této kapitole bude popsána metoda RANSAC, která se využívá pro odhad polohy a orientace objektu na základě obrazových dat. Jedná se o statickou metodu. Jejím hlavním cílem je najít nejlepší sadu bodů, které nejlépe odpovídají modelu objektu, přičemž je robustní vůči odlehlým hodnotám a šumu v datech.

Metoda funguje následovně a je popsána v knize OpenCV 3 Computer Vision Application Programming Cookbook [10]. Nejprve se vybere minimální počet bodů, které jsou potřebné pro odhad modelu. Tato hodnota se nazývá inliner. Sada bodů (inlinerů) se vybere náhodně z celého datasetu. Na základě vybraných bodů se vytvoří model objektu. To může být například přímka, rovina, elipsa nebo jiná geometrická entita, která představuje objekt. Všechny body v datasetu jsou následně testovány, zda spadají do přijatelného rozpětí vzhledem k modelu. Kritériem pro určení, zda bod patří do rozpětí, může být například minimální vzdálenost bodu od modelu. Body, které splňují kritérium vzdálenosti tvoří podpůrnou sadu. Čím větší je podpůrná sada, tím vyšší je pravděpodobnost, že vypočtený model je správný. Pokud je však podpůrná sada velmi malá, pak je model ve výsledku nesprávný. S nově vybranými sadami se opět opakují kroky od vytvoření modulu objektu až k bodům, které jsou vybrány jako kandidáti pro odhad polohy a orientace objektu. Cílem je tedy najít sadu bodů, která má největší podpůrnou sadu a nejpřesněji odhaduje polohu a orientaci objektu. Toho se docílí několikrátým opakováním popsanych kroků.

V závislosti na poměru nesprávných shod v celé datové sadě se pravděpodobnost výběru sady správných shod bude lišit. Víme však, že čím více výběrů provedeme, tím větší bude naše důvěra, že mezi těmito výběry máme alespoň jednu dobrou sadu shod. Předpokládáme, že množina shod obsahuje  $w$  inlinerů. Následná prav-

děpodobnost, že vybereme  $N$  shod je  $w^N$ . Pravděpodobnost, že ve vybrané sadě je alespoň jedna nesprávná shoda, je doplňkem k předešlé  $1 - w^N$ . Pokud provedeme určitý počet výběrů  $k$ , pak pravděpodobnost, že budeme mít alespoň jeden náhodný soubor obsahující pouze dobré shody, je dána vztahem (C.1) [10]

$$c = 1 - k(1 - w^N). \quad (C.1)$$

Jedná se o pravděpodobnost důvěry a chceme, aby tato pravděpodobnost byla co nejvyšší, protože potřebujeme alespoň jednu dobrou sadu shod k získání správného modelu. Při použití algoritmu RANSAC je proto třeba určit číslo  $k$  pro počet výběrů, které je třeba provést, abychom dosáhli určité úrovně důvěry.

# Seznam obrázků

2.1	3-dimenzionální struktura barevného obrázku . . . . .	6
2.2	Zpracování obrazu pomocí monadické operace . . . . .	7
2.3	Model středové projekce . . . . .	8
2.4	Souřadný systém kamery . . . . .	11
2.5	Šachovnice o velikosti $7 \times 7$ . . . . .	16
2.6	Snímání šachovnice . . . . .	16
2.7	Detekované vnitřní rohy šachovnice . . . . .	16
3.1	Příklad ARToolkit markeru . . . . .	20
3.2	Příklad AprilTag markeru . . . . .	20
3.3	Ukázkové příklady STag markerů z knihovny HD13 . . . . .	22
3.4	Ukázkové příklady ArUco markerů z knihovny DICT_7x7_250 . . . . .	24
4.1	3D souřadné systémy <b>A</b> a <b>B</b> . . . . .	32
4.2	Příklady orientace souřadných systémů ArUco markeru . . . . .	36
5.1	Rozhraní metody <code>estimatePoseSingleMarkers</code> v knihovně OpenCV, verze z roku 2013 . . . . .	38
6.1	ArUco marker, ID = 33, knihovna DICT_7x7_250 . . . . .	42
6.2	ArUco marker, ID = 33, knihovna DICT_6x6_250 . . . . .	42
6.3	Schéma konstrukce pro kameru upevněnou ve výšce 83 cm . . . . .	44
6.4	Schéma konstrukce pro kameru upevněnou ve výšce 111 cm . . . . .	44
6.5	ArUco marker o velikosti 5,8 cm z knihovny DICT_7x7_250 na desce snímaný externí webkamerou . . . . .	45
6.6	Reálné vzdálenosti pro šest bodů od kamery upevněné ve výšce 83 cm	45
6.7	Reálné vzdálenosti pro šest bodů od kamery upevněné ve výšce 111 cm	45
6.8	Souřadný systém ArUco markeru vůči kameře . . . . .	46
6.9	Schéma lokální souřadného systému s deskou o šesti bodech . . . . .	58
6.10	Souřadnice pro šest bodů na desce . . . . .	58
6.11	Schéma souřadných systémů - lokální souřadný systém, souřadný sys- tém ArUco markeru a kamery . . . . .	59

B.1	Zpracování obrazu pomocí diadické operace . . . . .	67
B.2	Zpracování obrazu pomocí prostorové operace . . . . .	69
B.3	Gaussova funkce . . . . .	70
B.4	Metoda detekce hran . . . . .	71
B.5	Princip porovnání šablon . . . . .	74

# Seznam tabulek

6.1	Vzdálenosti snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 40° . . . . .	47
6.2	Rotace snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 40° . . . . .	48
6.3	Vzdálenosti snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 40° . . . . .	49
6.4	Rotace snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 40° . . . . .	49
6.5	Vzdálenosti snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 25° . . . . .	50
6.6	Rotace snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 25° . . . . .	50
6.7	Vzdálenosti snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 25° . . . . .	51
6.8	Rotace snímané externí webkamerou položenou ve výšce 111 cm a nakloněnou ve 25° . . . . .	51
6.9	Vzdálenosti snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve 40° . . . . .	52
6.10	Rotace snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve 40° . . . . .	53
6.11	Vzdálenosti snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve 40° . . . . .	54
6.12	Rotace snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve 40° . . . . .	54
6.13	Vzdálenosti snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve 25° . . . . .	55
6.14	Rotace snímané kamerou v mobilu položenou ve výšce 83 cm a nakloněnou ve 25° . . . . .	55
6.15	Vzdálenosti snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve 25° . . . . .	56

*Seznam tabulek*

---

6.16	Rotace snímané kamerou v mobilu položenou ve výšce 111 cm a nakloněnou ve 25° . . . . .	56
6.17	Souřadnice translace ArUco markeru snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 40° . . . . .	61
6.18	Souřadnice translace ArUco markeru snímané externí webkamerou položenou ve výšce 83 cm a nakloněnou ve 25° . . . . .	62



# Seznam výpisů

2.1	Část programu pro kalibraci kamery . . . . .	17
3.1	Část programu pro generování a detekci ArUco markeru . . . . .	29
5.1	Část programu pro odhad polohy a orientace ArUco markeru . . . . .	39

# Bibliografie

1. SARMADI, Hamid; MUNOZ-SALINAS, Rafael; OLIVARES-MENDEZ, Miguel A.; MEDINA-CARNICER, Rafael. Detection of Binary Square Fiducial Markers Using an Event Camera. *IEEE Access*. 2021, roč. 9, s. 27813–27826. Dostupné z DOI: 10.1109/access.2021.3058423.
2. SAMPATHKRISHNA, Abhijith. *ArUco Maker based localization and Node graph approach to mapping*. 2022. Dostupné z arXiv: 2208.09355 [cs.RO].
3. CORKE, Peter. *Robotics, Vision and Control*. Brisbane: Springer-Verlag, 2011. ISBN 978-3-319-54412-0.
4. *OpenCV - dokumentace*. Dostupné také z: <https://docs.opencv.org/4.x/>.
5. WANG, John; OLSON, Edwin. AprilTag 2: Efficient and robust fiducial detection. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016.
6. OLSON, Edwin. AprilTag: A robust and flexible visual fiducial system. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2011, s. 3400–3407.
7. *S-Tag markery - databáze obrázků*. Dostupné také z: <https://drive.google.com/drive/folders/0ByNTNYCAhWbIV1RqdU9vRnd2Vnc?%20%5C%5C%20resourcekey=0-9ipvecbezW8EWUva5GBQTQ>.
8. BENLIGIRAY, Burak; TOPAL, Cihan; AKINLAR, Cuneyt. *S-Tag: A Stable Fiducial Marker System*. 2019. Dostupné z arXiv: 1707.06292 [cs.CV].
9. *OpenCV - oficiální stránka*. Dostupné také z: <https://opencv.org/>.
10. LAGANIERE, Robert. *OpenCV 3 Computer Vision Application Programming Cookbook*. Birmingham: Packt Publishing Ltd., 2017. ISBN 978-1-78646-971-7.

