VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

MĚŘENÍ ROZMĚRŮ ROVINNÝCH OBJEKTŮ V OBRAZE

PLANAR OBJECT MEASUREMENT IN IMAGE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PŘEMYSL MLÝNEK**
AUTHOR

VEDOUCÍ PRÁCE **Ing. VÍTĚZSLAV BERAN, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Mlýnek Přemysl**
Program: Informační technologie
Název: **Měření rozměrů rovinných objektů v obraze**
Planar Object Measurement in Image
Kategorie: Zpracování obrazu

Zadání:

1. Prostudujte metody detekce parametrických objektů v obraze. Seznamte se s postupy měření rozměrů rovinných objektů z fotografie.
2. Navrhněte sadu algoritmů, které umožní uživateli provádět měření ve fotografii rovinného objektu.
3. Vytvořte anotovanou sadu dat.
4. Pomocí vhodných nástrojů navržené funkce implementujte a vytvořte demonstrační uživatelskou aplikaci.
5. Proveďte experimenty na efektivitu a použitelnost řešení.
6. Vytvořte plakát a krátké video prezentující klíčové výsledky vašeho řešení.

Literatura:

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

Pro udělení zápočtu za první semestr je požadováno:

- Body 1, 2 a částečně body 3 a 4.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Beran Vítězslav, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 1. listopadu 2018

Abstrakt

Cílem této práce je měření nábytkových dvířek pomocí zpracování obrazu a výpočet ceny nových dvířek z naměřených rozměrů. Práce je řešena pomocí knihovny OpenCV a programovacího jazyka Python. Jádro práce je založeno na algoritmu FloodFill a Houghových transformacích. Grafické uživatelské rozhraní je řešeno pomocí knihovny PyQt. Při získávání rozměrů nábytkových dvířek byl použit následující postup – pořízení datové sady, předzpracování obrazu, segmentace objektů, klasifikace objektů, měření jednotlivých objektů, vypsání výsledků měření na výstup a výpočet výsledné ceny. Vytvořil jsem řešení ve formě desktopové aplikace, která na vstupu přijímá obrázek a na výstup dává naměřené rozměry spolu s cenou nových dvířek. Při měření bylo dosaženo průměrné odchylky 6 mm od reálných rozměrů. Tato práce mi pomohla pochopit základy zpracování obrazu. Běžnému uživateli aplikace přinese možnost odhadnout cenu nábytkových dvířek pouze na základě fotografie.

Abstract

The aim of this work is to measure furniture doors using image processing and calculating the price of new doors from measured dimensions. The work is solved using OpenCV library and Python programming language. The core of the work is based on the FloodFill algorithm and Hough transforms. The graphical user interface is solved using the PyQt library. When obtaining the measurements of the furniture door, the following procedure is used - acquisition of a data set, imagine preprocessing, object segmentation, object classification, measurement of individual objects, output of measurement results and output price calculation. I have created a solution in the form of a desktop application that accepts an image at the input and outputs the measured dimensions along with the price of the new door. I managed to achieve an average deviation of 6 mm from real dimensions. This work has helped me understand the basics of image processing. An ordinary user of the application will be able to estimate the price of new furniture doors just based on a photo.

Klíčová slova

měření objektů, zpracování obrazu, OpenCV, Python, PyQt

Keywords

objects measurement, image processing, OpenCV, Python, PyQt

Citace

MLÝNEK, Přemysl. *MĚŘENÍ ROZMĚRŮ ROVINNÝCH OBJEKTŮ V OBRAZE*. Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vítězslav Beran, Ph.D.

MĚŘENÍ ROZMĚRŮ ROVINNÝCH OBJEKTŮ V OBRAZE

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vítězslava Berana Ph.D. Další informace mi poskytly literární zdroje. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Přemysl Mlýnek
15. května 2019

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce Ing. Vítězslavovi Beranovi, Ph.D., za odborné rady a vedení při vytváření této bakalářské práce.

Obsah

Kapitola 1

Úvod

Zrak je pro člověka jedním z nejdůležitějších smyslů a v dnešním světě je jím vnímáno kolem 80 % informací. Lidstvo tedy v 60. letech 20. století napadlo naučit vidět i stroje. Tato myšlenka se zachovala až dodnes a v dnešní době je známa jako zpracování obrazu.

Zpracování obrazu pomáhá v různých odvětvích – průmyslu, lékařství, armádě či v uživatelských aplikacích, které pomáhají v každodenním životě. Tato práce se zabývá posledním zmíněným, a to implementací aplikace pro získání rozměrů nábytkových dvířek pomocí fotografie a následným výpočtem ceny nových nábytkových dvířek z naměřených rozměrů.

V posledních letech se zpracování obrazu rozrůstá a již existuje spousta aplikací pro měření rozměrů pomocí fotoaparátu. Nenašel jsem však žádnou aplikaci, která by měřila nábytková dvířka. Proto jsem se rozhodl nahlédnout do odvětví zpracování obrazu, pochopit základy a navrhnout aplikaci zabývající se tímto problémem.

Cílem práce je tedy aplikace, která změří rozměry nábytkových dvířek z fotografie a vypočítá cenu dvířek nových. Práce se dále zabývá zkoumáním jakých přesností měření lze za daných podmínek dosáhnout. Dále návrhem metody pro detekci a měření nábytkových dvířek. A v poslední řadě implementací aplikace, která ulehčí práci při zjišťování rozměrů nábytkových dvířek.

V druhé kapitole práce je zmíněna základní teorie, bez které by vytvoření návrhu nebylo možné. Třetí kapitola se zabývá návrhem řešení. Tedy návrh metod pro získání dvířek z fotografie a následné změření rozměrů získaného dvířek. Dále se zde řeší podrobný návrh GUI. V třetí kapitole je taky zmíněno, co všechno za akce je uživatel schopen/nucen udělat, aby bylo dosaženo požadovaného výsledku. Ve čtvrté kapitole je již samotná implementace výsledné aplikace. Jsou zde rozebrány metody zpracování obrazu, které jsou použity. Dále jsou zde zmíněny prvky GUI, kterých je využito pro uživatelské rozhraní. Poslední kapitola se zabývá testováním přesnosti měření a úspěšnosti při klasifikaci dvířek z fotografie.

Kapitola 2

Základní teorie zpracování obrazu

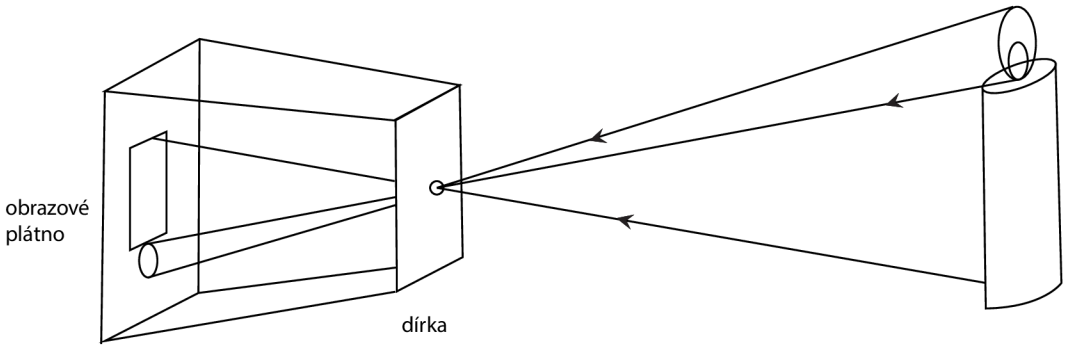
V této kapitole jsou rozebrány základní teoretické pojmy, bez kterých se v dalších částech nelze obejít. Níže je vysvětleno, co je to model dírkové kamery a jak se oproti tomu liší reálné čočky (objektivy). Dále také teorie jednotlivých podúloh předzpracování (preprocessingu) obrazu. Následně je vysvětleno, jak funguje segmentace a klasifikace při zpracování obrazu. V poslední části se nachází informace o měření rozměrů objektů v obraze.

2.1 Model dírkové kamery

Tato část je převzata z [3]. V této části jsou probrány základy modelu dírkové kamery (pinhole camera) a taky na jakém principu funguje snímaní obrazu. Je zde ukázáno, jak je světlo ze scény odráženo či vyzařováno, projde dírkou a je promítnuto na obrazovou rovinu.

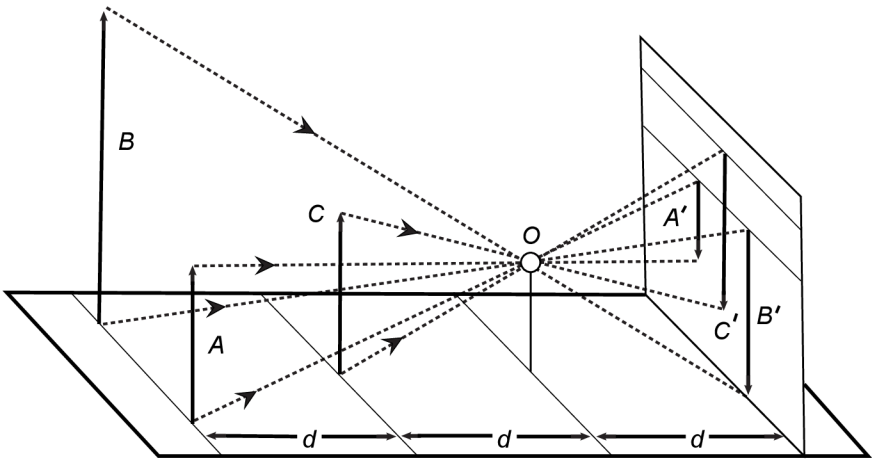
Definice funkčnosti dírkové kamery (pinhole camera)

Dírkovou kameru si můžeme představit jako krabici, do které je na jedné straně vyřezán kruhový otvor a na protější straně umístěno průhledné plátno. Toto plátno je nazýváno obrazové plátno (image plane). Pokud bychom tuto krabici umístili v zatemněné místnosti před sebe a směrem k dírce bychom postavili nějaký objekt, který vydává světlo např. svíčku, viděli bychom obraz objektu na obrazovém plátně obrácený (invertovaný) oproti reálnému objektu 2.1.



Obrázek 2.1: Model dírkové kamery. [3]

Tento obraz je vytvořen ze světelných paprsků odrážejících či přímo vycházejících z reálného objektu. Tyto paprsky prochází dírkou a promítají se na obrazové plátno. Na obrazovém plátně se poté vytvoří snímaný objekt, který je oproti reálnému objektu obrácený. Celý tento model funguje na principu perspektivní projekce. Perspektivní projekce udává, že velikost objektů na plátně je závislá na vzdálenosti od dírkové kamery a jejich velikosti v reálném světě 2.2.



Obrázek 2.2: Perspektivní projekce. [3]

Na obrázku výše jsou reálné objekty A, B, C. Tyto objekty jsou zobrazeny na obrazové plátno jako A', B', C'. Obraz objektů B a C je stejně velký, avšak obraz objektu A je menší. Objekty A a C jsou v reálném světě stejně velké. Menší zobrazení objektu A je zapříčiněno vzdáleností objektu A od objektivu. Tento obrázek popisuje jednu z vlastností perspektivní projekce.

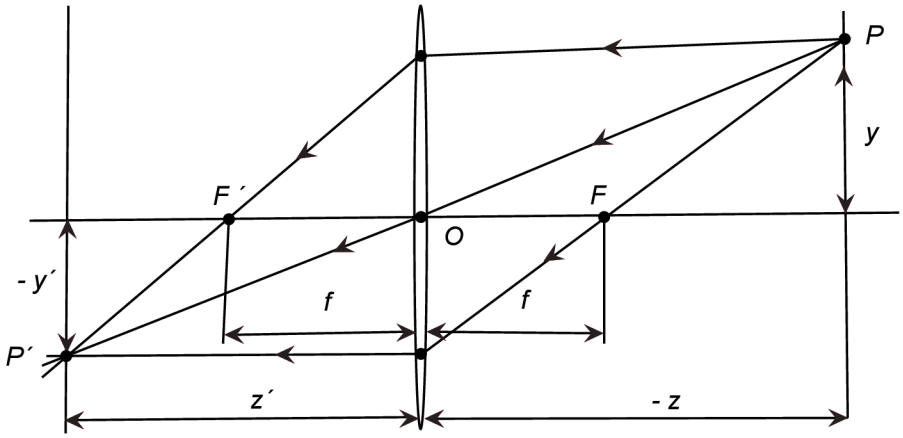
Mezi další vlastnost patří to, že dvě rovnoběžné přímky, které si lze představit jako fotku cesty linoucí se do dálky, se protnou v obrázku v určitém horizontu. Model dírkové kamery je považován za počátek dnešní fotografie a v další kapitole je uvedeno, jak fungují dnešní objektivy a v čem se liší oproti modelu dírkové kamery.

2.2 Snímání reálného světa pomocí fotoaparátu

Tato část je převzata z [3]. V této části je probráno, jak funguje snímání pomocí fotoaparátu, neboli pomocí optických čoček. V dnešním světe je většina fotoaparátů vybavená čočkou. Hlavní důvod proč čočky používat je, že k tomu, aby byl získán kvalitní obraz je zapotřebí hodně světla. U dírkové kamery je možné to reguloval šířkou dírky. Avšak čím větší je dírka, tím víc je obrázek rozmazaný. To by vyřešila malá dírka, ale u ní zase dochází k difrakci. Čočka zajistí dostatečnou velikost pro průchod světla i ostrost předmětů. Na následujících řádcích je uvažováno o lomu světla přecházejícího z jednoho prostředí do druhého.

Tenká čočka (Thin lens)

Jedná se o dvě kulové prostředí, které mají určitý rádius a index udávající lom světla. Představte si, že jste ve vakuu a lom světla je roven 1. Tento lom značí to, že je tloušťka čočky nulová. Z těchto vlastností je možné vyvodit, že paprsek odrazený/vycházející z reálného bodu ve scéně (značen P) procházející středem čočky O se podle Snellových zákonů nezlomí a promítne se na obrazové plátno do bodu P'. Další paprsky světla dopadají na čočku a lámou se do bodu P' tam, kde dopadl předchozí paprsek. Paprsky rovnoběžné s optickou osou prochází ohniskem F'. Ohnisková vzdálenost je značena písmenem f 2.3.



Obrázek 2.3: Obrázek tenké čočky (thin lens). [3].

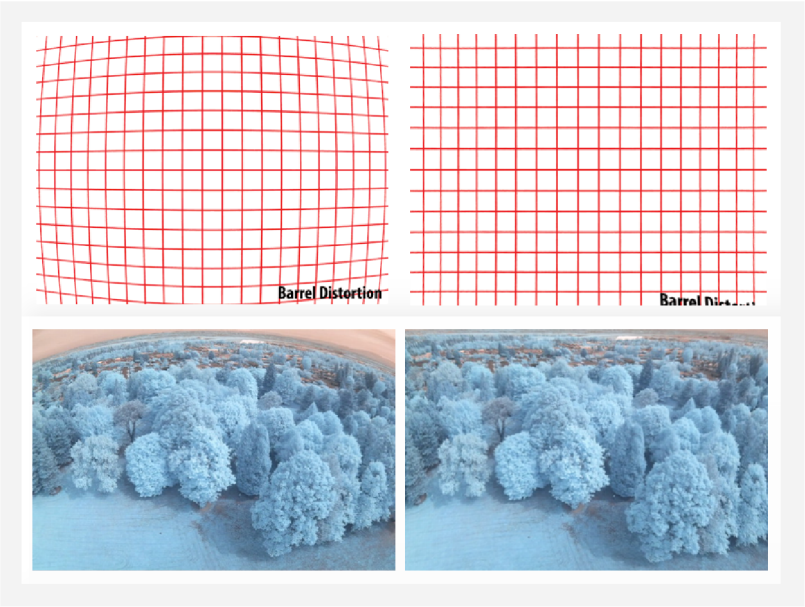
Reálná čočka (Real lens)

V textu výše je probrána jen teorie, z které plyne, že tenkou čočku není možné v praxi sestrojit – její tloušťka nikdy nebude nulová. Dále jsou zmíněny informace o reálné čočce používané v praxi. Pracuje na podobném principu jako tenká čočka, avšak v praxi dochází při snímání k určitým odchylkám. Tyto odchylky nastávají z toho důvodu, že obraz je pouze aproximován (odhadován), fotoaparát neví, kde přesně bod v reálném prostředí leží. Vychází z toho, v jakém úhlu k optické ose paprsek dopadá na čočku a taky z toho, jak je objekt nasvícený. Může dojít k různým odchylkám jako například zkreslení. Zkreslení mění tvar reálných objektů, pod čímž si můžeme představit určitou deformaci 2.4. Jelikož dochází k tomuto jevu, je potřeba ho vhodným způsobem odstranit. Tento problém je řešen v části Homografie.

2.3 Homografie

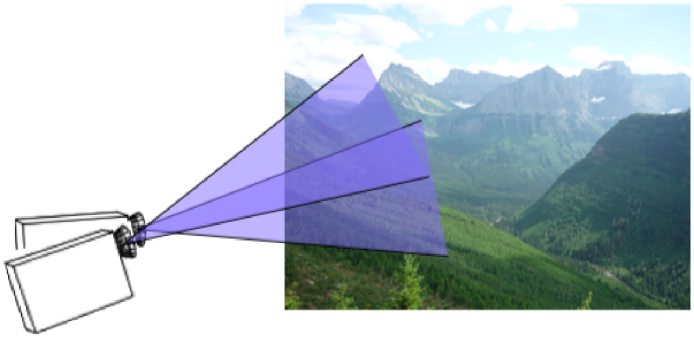
Tato část je převzata z [6] a [2]

Při zachycení snímku fotoaparátem dochází k jeho zkreslení. Pro odstranění deformace je možné v 2D prostoru použít tzv. homografii. Homografie je mapování bodů mezi dvěma



Obrázek 2.4: Na levém obrázku je vidět zkreslení způsobené čočkou. Na pravé straně obrázek po odstranění zkreslení. Zdroj: https://publiclab.org/system/images/photos/000/021/423/large/PublicLab_copy.png

projekčními snímky tak, že fotoaparát má u obou snímků stejný střed projekce. Dva snímky jsou nasnímány stejnou kamerou jen pod jiným úhlem. Tzv. je použita jen rotace nikoliv translace neboli posun 2.5.



Obrázek 2.5: Obrázek znázorňující rotaci kamery/fotoaparátu. [6]

.

Homografie je v homogenních souřadnicích reprezentována jako matice 3x3 a značena písmenem H.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{2.1}$$

Nový bod p' je vypočítán pomocí násobení původního bodu p a matice H.

$$p' = H * p \tag{2.2}$$

Homografie hraje důležitou roli při perspektivní transformaci deformovaného snímku. Díky matici homografie je možné odstranit perspektivní zkreslení a získat rekonstruovaný snímek. V další části jsou probrány metody sloužící k předzpracování obrazu.

2.4 Předzpracování obrazu

Vstupní obraz může obsahovat určité vlastnosti, které je potřeba potlačit či zvýraznit. Aby bylo dosaženo očekávaného výsledku, musí se provést tzv. předzpracování obrazu. Níže naleznete činnosti, které jsou do předzpracování obrazu zařazeny. [5]

Převod na odstíny šedi

Jelikož barevná informace není vždy potřeba, je obrázek převeden do odstínů šedi. Monochromatický obraz (černobílý) je oproti multispektrálnímu (barevnému) jednodušší na zpracování. Barevný obrázek obsahuje navíc oproti šedotónovému jednotlivé barevné složky obrazu. [5]

Filtrace obrazu

Tato podkapitola je převzata z [5]. Filtrace funguje na základě výpočtu jasu výstupního bodu pomocí jeho lokálního okolí O, kdy je postupně procházen celý obraz. Cílem filtrace je potlačení či zvýraznění určitých vlastností obrazu. Je možné ji rozdělit do dvou skupin – vyhlazování a gradientní operace (detekce hran).

Vyhlazování slouží k potlačení šumu. Problémem je, že dochází k zániku ostrých čar a hran, které nesou velmi důležitou informaci. Gradientní operace vedou k ostření obrazu a tedy ke zvýraznění hran, což poskytne významnou informaci v obraze. Problémem je, že při gradientních operacích dochází ke zvýraznění šumu. U obou metod se vyskytuje problém, proto vznikly algoritmy, které kombinují výše zmíněné metody, a dojte tak k vyhlazení obrazu i zvýraznění potřebných hran. Níže jsou uvedeny některé metody sloužící k filtraci.

Jednou z používaných metod je filtrace šumu pomocí průměrování. Hodnota jasu výstupního bodu vzniká pomocí průměrování (aritmetický průměr) jasů jeho okolních bodů. Pro okolí 3x3 je zvolena následující konvoluční maska (2.3).

$$h = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.3}$$

U předcházející metody dochází k rozmazání hran. Proto existuje metoda tzv. rotující masky. Tato metoda se snaží na základě homogenity jasu zařadit význačný bod do okolí, kterému patří. Pro výpočet jasu význačného bodu (např. průměrováním) je použita jen ta část, která je považována za homogenní okolí.

Gradientní operace – detekce hran

Tato podkapitola je převzata z [3], [4] a [5]. Při hledání hran jsou sledovány v obraze náhlé změny jasu. Tyto změny většinou nastávají na hranicích objektů. Světlý objekt může ležet

na tmavém pozadí a díky tomu je možné rozpoznat hranu. Body v obraze, u kterých dochází k rychlým změnám jasové funkce jsou nazývány hranou.

Není to však tak jednoduché, jak se může zdát. Praktická hrana se liší od teoretické tím, že je zašumělá. Z toho plyne, že obrázek může obsahovat šum, který rozpoznání hran ztěžuje. Tento problém je možné řešit pomocí filtrace viz. výše.

Hrana je tedy v obraze detekovatelná pomocí skokové změny jasu. Funkce obrazu má dvě proměnné. Pro zkoumání změny funkce dvou proměnných slouží tzv. gradient. Gradient je vektor (Δ), který udává směr růstu obrazové funkce a její rychlost/velikost (modul gradientu). Velikost gradientu se získává pomocí parciálních derivací a je dána následujícím vztahem (2.4).

$$|\nabla g(x,y)| = \sqrt{\left(\frac{\partial g}{\partial x}\right)^2 + \left(\frac{\partial g}{\partial y}\right)^2} \tag{2.4}$$

.

Gradientní operátory se dají rozdělit do dvou skupin. První skupina operátorů používá aproximace první derivace obrazové funkce $f(x,y)$ pomocí diferencí. Směr gradientu určuje maska, která odpovídá největší velikosti gradientu. Detekce hran pomocí druhé skupiny operátorů je postavena na principu druhé derivace obrazové funkce $f(x,y)$ procházející nulou.

Na následujících řádcích jsou probrány operátory první skupiny. Tyto operátory jsou založeny na diskrétní konvoluci, a proto budou uváděny u každého příkladu konvoluční masky. Oproti Laplaceově operátoru je zde zjištěn i směr gradientu.

Robertsův operátor se vyznačuje tím, že je nejstarší a velmi jednoduchý. Používá okolí 2x2 obrazového bodu. Má dvě konvoluční masky (2.5). Mínusem je, že je hodně citlivý na šum vzhledem k jeho úzké konvoluční masce.

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{2.5}$$

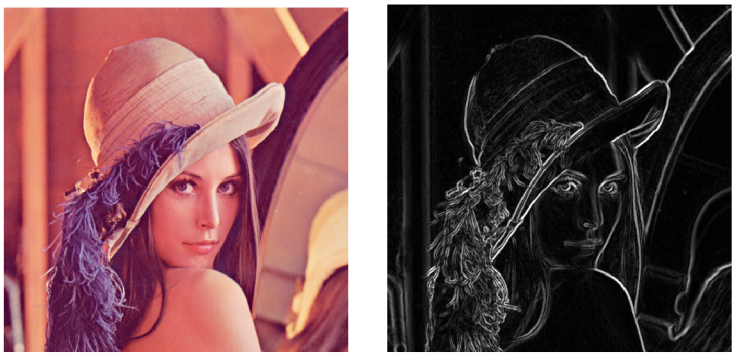
Sobelův operátor je možné použít s různě velkými rozměry masek. Jedná se tedy o lepší aproximaci gradientu obrazové funkce. Níže je uvedena 3x3 konvoluční maska pro dva z osmi směrů (2.6). Níže se nachází obrázek 2.6, který ukazuje co způsobí aplikace Sobelova operátoru. Často se využívá pro detekci vodorovných a svislých hran. Proto se nejčastěji využívají konvoluční masky h_1 a h_2.

$$h_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad h_2 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.6}$$

Druhá skupina gradientních operátorů vyhledává v obrazové funkci hrany v místě, kde druhá derivace obrazové funkce $f(x,y)$ prochází nulou (tzv. zero crossing). Na obrázku 2.7 je možné vidět ilustraci obrazové funkce procházející nulou. Jestliže je nutné znát velikost gradientu, který označuje hranové body, je možné použít Laplaceův operátor (2.7). Tento operátor používá aproximaci pomocí druhé derivace a slouží pouze k určení velikosti gradientu, nikoliv jeho směru tzv. je necitlivý vůči otočení.

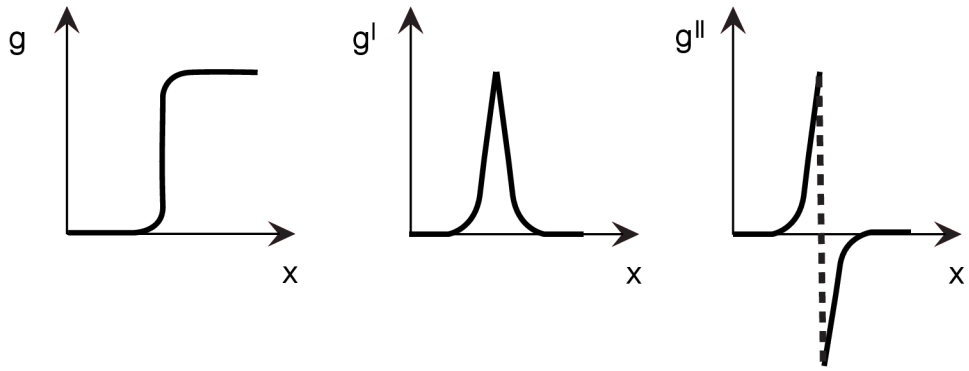
$$\nabla^2(x,y) = \frac{\partial^2 g(x,y)}{\partial x^2} + \frac{\partial^2 g(x,y)}{\partial y^2} \tag{2.7}$$

.



Obrázek 2.6: **Lena – Sobelův operátor.** Obrázek před a po průchodu Sobelovým operátorem. Zdroj: http://optipng.sourceforge.net/pngtech/img/lena.html

Závěrem je důležité vzpomenout, že gradientní operátory neurčují oblasti objektů. Určení oblastí se uskuteční zřetězením hran získaných pomocí gradientních operátorů, které bude probráno v segmentační části. Není jednoduché určit, který operátor je nejlepší. Proto se většinou experimentuje a je vybrán ten, který dává nejlepší výsledky.



Obrázek 2.7: Ilustrační obrázek znázorňující průchod nulou. [5]

2.5 Segmentace obrazu

Aby bylo možné najít v obraze požadované objekty, je nutné provést segmentaci. Segmentace rozčleňuje obraz do částí, které mají určitou spojitost s hledanými objekty neboli objekty reálného světa. Pokud se podaří získat oblasti, které jednoznačně korespondují s hledanými objekty, jedná se o kompletní segmentaci. Když oblasti neúplně korespondují s objekty, je to nazýváno částečnou segmentací. Ne vždy je tato vrstva k nalezení objektů postačující. Z toho plyne, že ke kompletní segmentaci je ještě nutné použít vyšší vrstvy, které nějakým způsobem popisují daný objekt. Jedná se o klasifikaci, která bude popsána v další kapitole. [5]

Segmentace pomocí prahování

Tato podkapitola je převzata z [5]. Prahování je postaveno na principu, že určité objekty mají kontrastní odrazivost či pohltivost světla. Objekt má tedy určitý interval jasů, které obsahuje. Je tedy možné určit práh, který body pod prahovou úrovní požaduje za pozadí a body nad prahem za hledané objekty 2.8. Při prahování dochází k transformaci vstupního obrazu f na výstupní (segmentovaný) binární obraz g podle pravidla :

$$g(i,j) \quad = \quad \begin{array}{ll} 1 & pro\, f(i,j) \geq T \\ 0 & pro\, f(i,j) < T \end{array} \tag{2.8}$$

kde T je zvolený práh. Dále $g(i,j) = 1$ pro části obrazu náležící po segmentaci objektům a $g(i,j) = 0$ pro pozadí (může být i naopak).

Obrázek 2.8: Výsledky prahování. [5]

Segmentace pomocí Houghových transformací

Tato podkapitola je převzata z [8]. Houghovy transformace slouží k detekci čar v obraze. Před detekcí čar je nutné provést hranovou detekci obrazu. Čára může být v obraze reprezentována pomocí dvou soustav:

- Kartézská soustava souřadnic
- Polární soustava souřadnic

Při použití Houghových transformací se používá polární souřadnicový systém a rovnice pro čáru je ve tvaru (2.9).

$$y \quad = \quad \frac{-(\cos\theta)}{\sin\theta}x + \frac{\rho}{\sin\theta} \tag{2.9}$$

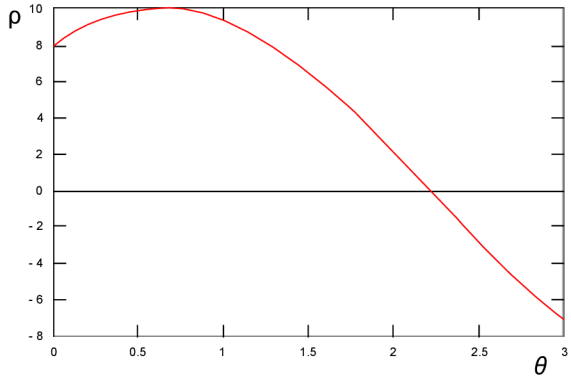
Která po úpravě vypadá následovně (2.10).

$$\rho \quad = \quad x\cos\theta + y\sin\theta \tag{2.10}$$

Pro každý bod (x_0, y_0) existuje skupina čar, které jím prochází a je definována jako:

$$\rho_\theta = x_0 * cos\theta + y_0 * \sin\theta \tag{2.11}$$

Tedy dvojice (ρ, θ) reprezentuje přímku procházející bodem (x_0, y_0). Pro každý bod je možné v polární soustavě vykreslit skupinu přímek, které jím prochází. Vykreslením vznikne sinusoida. Pro $x_0 = 8$ a $y_0 = 6$ vypadá graf v polárním souřadnicovém systému následovně 2.9.

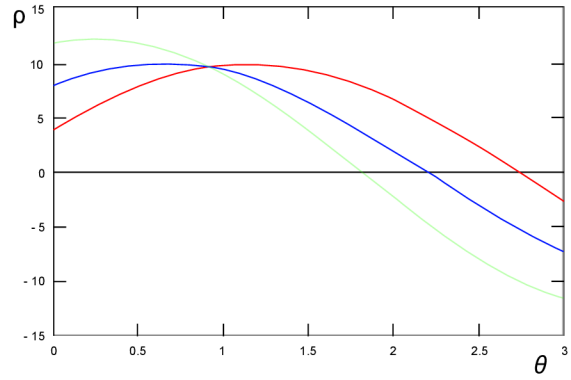


Obrázek 2.9: Ukázka vykreslené sinusoidy v polárním systému pro jeden bod. [8]

Takto je možné do polární soustavy vykreslit všechny body v obraze. Jestliže se sinusoidy dvou různých bodů protínají v polárním systému, znamená to, že oba body jsou v jedné přímce. Příklad grafu 2.10 pro body:

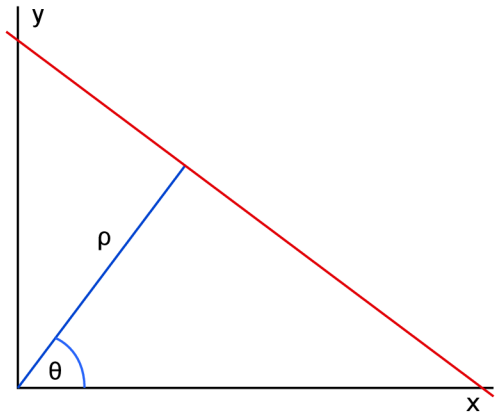
- $x_0 = 8$ a $y_0 = 6$
- $x_1 = 4$ a $y_1 = 9$
- $x2 = 12$ a $y_2 = 3$

Pro více bodů v obraze můžou křivky v polárním systému vytvářet průnik. Tento průnik reprezentuje pomocí ρ a θ rovnici přímky, která je v obraze hledaná. Znamená to tedy, že pokud se v polárním systému protíná víc křivek, tak tento průsečík reprezentuje přímku. Přímka je reprezentována pomocí parametrů ρ a θ. Pomocí těchto parametrů je možné převést přímku do kartézské soustavy. ρ značí délku normály od přímky a θ je úhel mezi normálou a osou x.



Obrázek 2.10: Ukázka vykreslených sinusoid v polárním systému pro tři body. Je zde vidět i zmíněný průnik. [8]

Čáru je tedy možné detekovat nalezením průsečíku mezi křivkami v polárním systému. Čím více křivek se protíná, tím více má detekovaná čára bodů. Je možné určit práh, kolik průsečíků je potřebných pro to, aby byla čára přijata. Pomocí parametrů ρ a θ je možné sestavit rovnici přímky. ρ značí délku normály od přímky a θ je úhel mezi normálou a osou x 2.11.



Obrázek 2.11: Ukázka parametrů ρ a θ na ilustračním obrázku. [8]

To co je popsáno výše je klasická Houghova transformace. Existuje však ještě probabilistická Houghova transformace. Ta pracuje efektivněji než klasická Houghova transformace a poskytuje na výstup krajní body přímky (x_0, y_0, x_1, y_1), teď už detekované úsečky.

Metoda FloodFill

Tato podkapitola je převzata z [9]. Segmentaci je možné provést i pomocí rozlévání, které se řídí podle určitých pravidel. Metoda zaplní body, které jsou označeny jako připojené, barvou. Plnění probíhá od výchozího bodu (seed). Připojení bodů k výchozímu bodu probíhá na základě barvy/jasu bodů s ním sousedících. Pixel (x, y) patří do zaplavované skupiny pokud platí (2.12).

$$src(x', y') - loDiff \leq src(x, y) \leq src(x', y') + upDiff \tag{2.12}$$

kde:

- $src(x', y')$ – Označuje jasovou hodnotu sousedního pixelu.
- $src(x, y)$ – Jasová hodnota pixelu, který již patří do skupiny připojených.
- $loDiff$ – Dolní hranice vychýlení jasu sousedního pixelu od jasové hodnoty aktuálního pixelu.
- $upDiff$ – Horní hranice vychýlení jasu sousedního pixelu od jasové hodnoty aktuálního pixelu.

Pomocí výše uvedeného postupu je možné vytvořit masku hledaného objektu, a poté pomocí metody pro nalezení obrysů najít jeho hranice, zkopírovat vybranou oblast na jiný obrázek, atd.

V této části jsou probrány segmentační metody. Je zde vysvětleno prahování, na jakém principu pracují Houghovy transformace a jak funguje metoda FloodFill. V další části je řeč o rozpoznávání objektů neboli klasifikaci.

2.6 Rozpoznání objektů (klasifikace)

Tato část je převzata z [17] a [5]. Bez rozpoznání by nebylo možné pomocí počítačového vidění splnit ani nejjednodušší úkoly. Pokud je prováděno zpracování obrazu zdola-nahoru rozpoznání je poslední operací. Rozpoznávání funguje na základě určitých vstupních informací.

Je možné si představit, že jsou dva večírky konající se v jednom hotelu. Jeden večírek mají basketbalisté, druhý žokejové. Recepční posílá hosty do správných sálů. Recepční se na začátku večera zeptal prvotních hostů, na kterou oslavu patří. Z jejich odpovědí získal vstupní informace, na jejichž základě bude vyhodnocovat příchozí hosty bez dalších otázek. Recepční si vytvořil „klasifikátor“, který rozpozná basketbalisty podle toho, že jsou vysocí a svalnatí. Žokejové jsou pravý opak. Zdá se, že tento úkol je jednoduchý. Je tomu tak, avšak v praxi se vyskytují mnohem složitější úlohy, kdy je potřeba rozpoznat určité reálné objekty pomocí počítačového vidění. Níže na obrázku 2.12 je možné vidět graf znázorňující třídu žokejů a třídu basketbalových hráčů. Osa x znázorňuje výšku. Osa y značí váhu.

Rozpoznávání je možné rozdělit do dvou kategorií – rozpoznání na základě příznakově popsaných předmětů, rozpoznávání syntakticky popsaných předmětů. Klasifikace je rozsáhlé téma a byl zde uveden jen základ z důvodu omezeného rozsahu práce a okrajového využití v této práci.

2.7 Měření objektů v obraze

Tato část je převzata z [17]. V této části je probráno hlavní téma této práce a tím je měření objektů. Samotné měření probíhá až po provedení operací v předchozích sekcích. Tzv. obraz je již rozdělen na objekty, které jsou hledány a na pozadí. Získané objekty je možné měřit.

Vzdálenost mezi dvěma body se souřadnicemi (x,y) a (o,p) lze získat několika způsoby. Je probrána pouze Euklidovská vzdálenost. Dále existují ještě „city block“ a „chessboard distance“.

Euklidovská vzdálenost

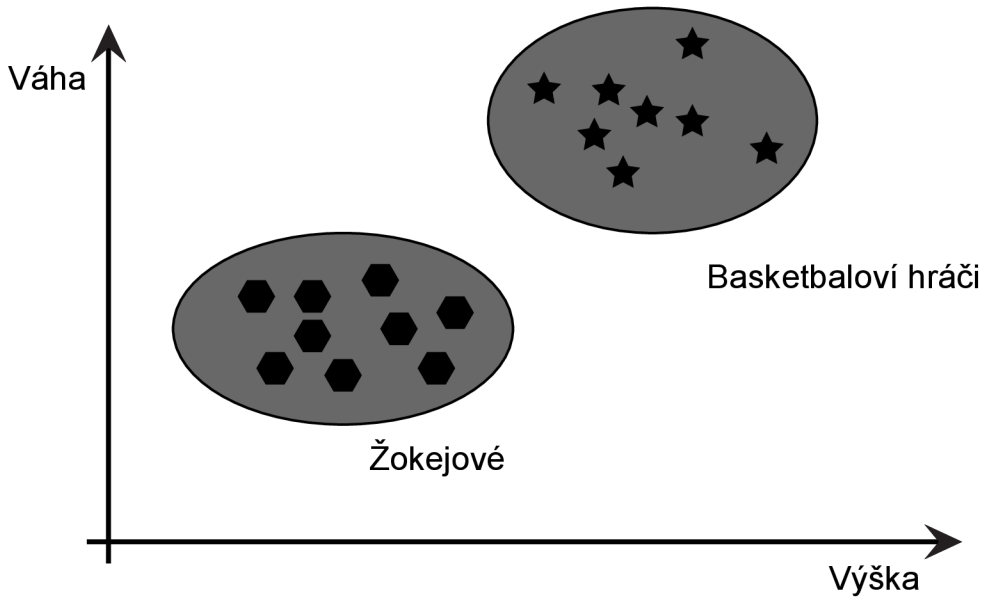
Euklidovská vzdálenost D_E je známa z geometrie a je matematicky definována jako (2.13).

$$D_E = \sqrt{(i-h)^2 + (j-k)^2} \tag{2.13}$$

kde (i, j), (h, k) jsou souřadnice obrazových bodů mezi, kterými se počítá vzdálenost.

Výhodou je, že je vcelku intuitivní a dobře pochopitelná. Nevýhodou je drahý výpočet z toho důvodu, že má v sobě druhou odmocninu.

V této části je vysvětleno, jak měřit objekty a jaké jsou metody pro měření. Pro hlubší prozkoumání odkazuji na [17].



Obrázek 2.12: Ilustrační obrázek znázorňující klasifikaci. [17]

2.8 Existující řešení

Pro měření rozměrů rovinných objektů existuje spousta aplikací. Převážně v posledních letech se s nástupem technologie ARKit [1] od firmy Apple vynořilo na AppStoru několik aplikací sloužící pro měření pomocí virtuální reality. Aplikace požádá o přístup k fotoaparátu a poté je uživatel schopen zadáním několika bodů na displeji měřit vzdálenost. Jsou to aplikace – Měření (Apple), AirMeasure – AR Tape & Ruler (Laan Labs) a spoustu dalších. Tyto aplikace bych zařadil jako podobné avšak ne stejné.

Tato práce se zabývá měřením rozměrů nábytkových dvířek pomocí fotografie a výpočtem ceny nových dvířek z naměřených rozměrů. Tzv. každý člověk co má fotoaparát si může vyfotografovat kuchyň a provést měření pomocí této aplikace. Aplikaci zabývající se touto problematikou se mi nepodařilo najít. Proto vznikla tato bakalářská práce.

Závěr

V této kapitole je probrána základní teorie, z které je vycházeno v dalších kapitolách. Nachází se zde informace o modelu dírkové kamery, jaké jsou čočky, co je to homografie, metody předzpracování obrazu, segmentace obrazu, klasifikace objektů, měření objektů a v poslední řadě existující řešení. Další kapitola pojednává o návrhu řešení.

Kapitola 3

Návrh řešení

V této kapitole bude probrán návrh řešení tj. návod, jak vyřešit tuto úlohu. Bude zde detailně rozebráno zadání práce, jaké jsou cíle a jak jich je možné dosáhnout.

Ještě před samotným návrhem bylo nutné porozumět základním principům, které jsou uvedeny v kapitole 2.

3.1 Hlavní cíl

Hlavním cílem je vytvořit aplikaci, která přijímá na vstupu fotografii (nábytku, kuchyně), provede měření nábytkových dvířek a na výstup dá naměřené rozměry. Z této míry je poté vypočítána orientační cena nových dvířek. Tato aplikace vznikla za účelem přinést běžným lidem informaci o ceně nábytkových dvířek bez příplatku za instalaci odborníky z nábytkových studií. Běžní uživatelé si budou moct odhadnout cenu nových nábytkových dvířek a poté zvážit objednávku a instalaci svépomocí. Jsou tedy schopni porovnat cenu od stolaře/studia s cenou od výrobce nábytkových dvířek.

Kdyby bylo možné dosáhnout přesných rozměrů nábytkových dvířek, pomocí běžného fotoaparátu s odchylkou $\pm$ 1 mm, přineslo by to určitou revoluci v objednávání nábytkových dvířek. Aby bylo možné dosáhnout této přesnosti, jednalo by se o velmi komplexní a náročnou úlohu. Proto je uvažována přesnost v řádech cm a výsledné míry slouží pouze k odhadu ceny a ne přímo k objednání dvířek s naměřenými rozměry.

Spousta lidí má doma starší kuchyň či nábytek a zvažuje koupi nového. Investice do komplet nové kuchyně, stoupá do vysokých peněžních částek. Možností je vyměnit pouze kuchyňská dvířka, která mohou rapidně ovlivnit vzhled staré kuchyně.

Zákazník má možnost zajít za odborníky, kteří mu nabídnou cenu za výměnu kuchyňských dvířek. V této ceně je ovšem i přirážka za montáž, manipulaci, atd. Zákazník má tedy možnost namontovat a objednat si dvířka svépomocí.

Aby bylo možné odhadnout cenu nových dvířek, je potřeba znát jejich rozměry. Výměra dvířek na celé kuchyni však zabere velké množství času. Tato práce umožní zákazníkovi vyfotit kuchyňskou linku a po provedením pár kliků na PC získá rychle informace o ceně nových kuchyňských dvířek. Zákazník se poté může rozhodnout zda zrealizuje investici od profesionálů, či svépomocí nebo se spokojí se starou kuchyní.

Aby bylo možné dosáhnout cíle této práce, je potřeba pochopit, jak funguje výměna nábytkových dvířek v reálném prostředí. Níže je popsán postup, co všechno je nutné udělat k tomu, aby si obyčejný člověk mohl vyměnit nábytková či kuchyňská dvířka.

Člověk který se rozhodne pro výměnu má dvě možnosti:

1. Kontaktuje profesionály a ti vše udělají za něj.

2. Rozhodne se pro výměnu svépomocí.

Při rozhodování hrajou roli dva faktory – **cena** a **manuální zručnost**. U nábytkových studií či stolařů je cena vysoká. Příplatky za značku a práci jsou vysoké a spoustu lidí to odradí. Raději si tedy řeknou že si nechají starou kuchyň a nic se nezmění. U manuální zručnosti je kladena otázka, zda je daná osoba schopna výměnu nábytkových dvířek uskutečnit sama, zda si na to troufá či nikoliv. K výměně nábytkových dvířek svépomocí vede několik kroků:

1. Výměra rozměrů jednotlivých dvířek – tato operace je časově náročná, kuchyň či nábytek obsahují několik položek.

2. Výměra roztečí pro uchycení dvířek.

3. Zjištění ceny – možnost zadat rozměry do on-line formulářů či odeslat na email stolařům či nábytkovým studiím.

4. Objednávka dvířek.

5. Odstranění starých dvířek a instalace nových.

Výsledná aplikace se zabývá krokem 1. a 3. tj. měřením nábytkových dvířek pomocí fotoaparátu a následným spočítáním ceny dvířek nových. Uživatel tedy získá rychlou informaci o tom, zda se mu vyplatí realizovat svépomocí či požádat o pomoc profesionály.

3.2 Grafické uživatelské rozhraní

Výslednou aplikaci je možné implementovat jak na stolní počítače tak na mobilní zařízení. Pro dosažení větší intuitivnosti a rychlosti jsou lepší mobilní zařízení, mínusem je však malá velikost displeje. Převážně z důvodu velikosti displeje jsem se rozhodl pro implementaci na desktop 4.

Obrázek zde hraje důležitou roli a je nutné, aby v GUI zabírat největší část. Uživatelské rozhraní se skládá ze čtyř částí:

1. **Welcome okno** 3.1

2. **Homography okno** 3.2

3. **Doors okno** 3.3

4. **Results okno** 3.4

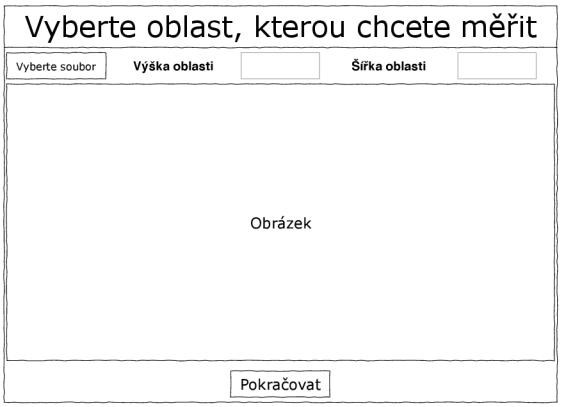
Welcome okno

Hlavička zabírá zhruba 10 % celkového prostoru a nachází se zde nadpis „Vítejte“. Středová část obsahuje název aplikace, jméno autora, a tlačítko pro vstup do aplikace. Tyto prvky jsou horizontálně centrovány.

Obrázek 3.1: Prototyp úvodní obrazovky aplikace.

Homography okno

Hlavička zabírá stejně prostoru jako v předcházejícím okně. Část pod hlavičkou zabírá cca 10 % celkového prostoru. Nachází se zde dvě vstupní pole pro zadání výšky a šířky oblasti. Do těchto vstupních polí uživatel zadává reálné rozměry měřené oblasti. Dále se zde objevuje tlačítko pro výběr fotografie. Po kliknutí na tlačítko „Vyberte soubor“ vyskočí okno v kterém je možné procházet soubory v počítači. Uživatel má možnost vybrat fotografii a výběr potvrdit. Střední část cca 70 procent celkového prostoru náleží oblasti k zobrazení fotografie. Po potvrzení výběru souboru se do této oblasti promítne vybraná fotografie. Uživatel provádí v oblasti fotky určité interakce (kliknutí, pohyb). Více podrobností v části níže 3.3. Dolní část obsahuje tlačítko pro přesun o stranu vpřed. Po kliknutí na tlačítko je aplikace přepnuta na nadcházející okno Doors.

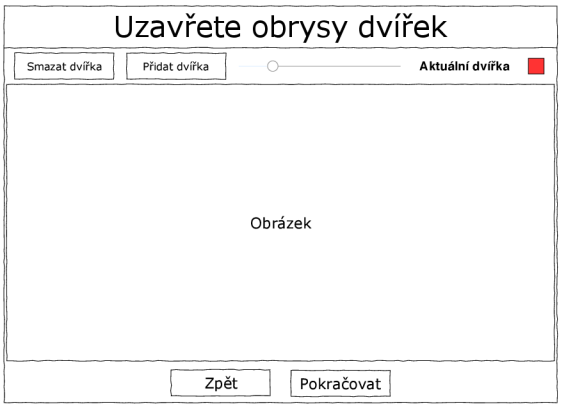


Obrázek 3.2: Prototyp okna homografie.

Doors okno

Část pod hlavičkou zabírá zhruba 10 % celkového prostoru. Nachází se zde posuvník, dvě tlačítka pro smazání a přidání bodů dvířek do obrázku a barva dvířka, které je aktuálně

označeno. Posuvník má rozsah od 50 do 500 a nastavuje se pomocí něj parametr prahování v hranové detekci. Tlačítko „Smazat dvířka“ slouží k odstranění aktuálně označeného dvířka. Střední část zabírá stejné množství prostoru jako v předešlém okně. Slouží k zobrazení zpracovávané fotografie. Dolní část je stejná jako v předchozím okně.



Obrázek 3.3: Prototyp okna s výběrem dvířek.

Results okno

Hlavička cca 10 %. V části pod hlavičkou se nachází tabulka s výsledky měření nábytkových dvířek popřípadě s odchylkou (testovací mód), která udává rozdíl reálných a naměřených rozměrů cca 60 %. Dále je zde umístěna část s tlačítky pro nacenění naměřených dvířek cca 10 % prostoru. Pod touto částí se nachází výsledná cena nábytkových dvířek 10 %. Poslední část obsahuje tlačítko sloužící pro přesun na předešlou stránku (10 %).

Obrázek 3.4: Prototyp okna s výsledky.

3.3 Akce uživatele

Aplikace není plně automatická, takže aby fungovala a na výstup dávala požadovaný výsledek, je potřebné zapojení uživatele. V této sekci je popsáno, co za uživatelské akce je možné provést a nutné implementovat.

Welcome okno

První akci, kterou je nucen uživatel provést je vyfotografovat nábytek či kuchyň, kterou chce měřit. Je taky potřeba vyměřit reálné rozměry oblasti, kterou bude chtít v aplikaci změřit.

Homography okno

Druhou akcí je nahrání fotografie do aplikace. To se provede pomocí tlačítka „Vybrat soubor“ v části 3.2 uživatelského rozhraní. Kliknutím myši do oblasti s obrázkem přidá uživatel 4 body, která vytyčují hranici oblasti, na kterou se aplikuje perspektivní transformace. Pro tuto oblast je uživatel povinen zadat reálné rozměry do vstupních polí nacházející se v horní části GUI. Uživatel má možnost s přidanými body pohybovat v libovolném směru, či vytvořit novou oblast s tím, že stará zanikne. V dolní části má možnost kliknout na tlačítko pokračovat. Tímto je vyvolána metoda, která vypočítá matici homografie a provede perspektivní transformaci 3.5.

Doors okno

Ve třetím okně má uživatel za úkol uzavřít obrysy nábytkových dvířek. Obrysy je možné regulovat pomocí posuvníku v horní části. Výsledek se poté projevuje na binárním obrázku uprostřed. Binární obrázek uprostřed je získán jako výstup z okna 3.2. Při posunutí posuvníku je volána hranová detekce a aktualizován obrázek. Po uzavření obrysů dvířek posuvníkem, je možné vybírat dvířka z obrázku, které chce uživatel měřit, kliknutím do středu dvířek. Po kliknutí je volán algoritmus pro segmentaci a klasifikaci a obrázek je překreslen RGB obrázkem s rohovými body vybraných dvířek. Tyto body je možné pohybem myši přesouvat, tak aby byl uživatel s ohraničením spokojen. Dále je možné vybrané body dvířka smazat a to jedním kliknutím na tlačítko v horní části „Smazat dvířka“. Po kliknutí na určitý bod uživatelem, je označeno dvířko jako aktuální. Uživatel má možnost vybrat kolik dvířek chce měřit a jakmile je s výběrem spokojen, kliknutím na tlačítko „Pokračovat“ pokračuje na Results okno.

Results okno

V poslední části uživatel vidí změřené rozměry nábytkových dvířek. Má zde možnost vybrat si, jaký typ nábytkových dvířek chce (lesklé, matné, dřevodekor). Po kliknutí na tlačítko je volána metoda, která přepočítá naměřené výsledky na metry čtvereční a vynásobí je cenou za m^2. Tato cena se poté zobrazí do textového návěští. Uživatel již teď zná cenu a ví zda se mu vyplatí si dvířka udělat sám nebo se spolehne na odborníky. Dále je zde ještě možnost vrátit se tlačítkem zpět na předchozí okno a vypočítat si cenovou relaci pro jiný kus nábytku.

Pro testování je do tabulky přidána přesnost měření. Takže při zadání reálných rozměrů všech dvířek a zapnutí testovacího módu je možné vidět přesnost měření. Reálné rozměry sloužící pro porovnání jsou zadány přímo v kódu. Testování je zmíněno v části 5.

3.4 Datové struktury

Nyní, když je znám cíl a podrobný návrh aplikace, je možné navrhnout datové struktury, do kterých budou data ukládána. Na následujících řádcích je popsáno co je nutné ukládat a jaké datové struktury jsou použity.

- Obrázek – je uložen do matice MxN prvků, kdy každý prvek nabývá určité jasové hodnoty. Pokud se jedná o barevný obrázek jsou to hodnoty od 0 po 255 na třech jasových složkách. V aplikaci je využito RGB obrázků tak i binárních obrázků.
- Informace od uživatele – Reálná výška a šířka od uživatele je uložena do třídních proměnných. Rozměr je zadáván v celých číslech, takže proměnné jsou zastoupeny datovým typem `Int`. Tyto proměnné jsou poté použity dle potřeby.
- Informace o dvířkách – Pro ukládání informací o dvířkách je vytvořena třída. Třída obsahuje atributy které nesou informace o velikosti dvířek a dále souřadnice vrcholů. Třída také obsahuje třídní metody, které slouží k nastavování a získávání všech atributů této třídy. Instanciací této třídy nám vznikne objekt, který ponese všechny výše zmíněné informace a bude možné volat jeho metody.
- Objekty dvířek – Jelikož je v obraze vyhledáváno více než jedno dvířko, je potřebné tyto objekty někam ukládat. Pro ukládání objektů je vytvořeno pole, do kterého jsou objekty přidávány, editovány, mazány.

3.5 Metody prováděny na pozadí

Datová struktura je navržena výše a je tedy možné přistoupit k logice aplikace. Logika aplikace je rozdělena do 6 samostatných celků. Každé UI má vlastní soubor s logikou k tomuto UI. Logika aplikace je prováděna na pozadí a uživatel ji „nevidí“ (vidí jen výsledek), avšak má možnost tento výsledek ovlivnit pomocí uživatelského rozhraní.

Patří sem metody zpracování obrazu, metody sloužící pro různé výpočty, inicializace proměnných, nastavování slotů pro signály tlačítek, atd. Tyto metody pracují se vstupním obrazem a umožňují uživateli proces měření. Níže je popsáno co vše se musí na pozadí provést, aby bylo dosaženo požadované funkcionality.

Okno Welcome

Zde obrázek ještě není načtený, takže žádné metody pro zpracování obrazu zde nejsou využity. Je zde však nutné provést propojení tlačítka s příslušnými sloty (metodami), které při stisku tlačítka uživatelem vyvolají potřebnou akci. Jakmile přijde signál, že je tlačítko stisknuto volá se metoda, kterou jsme tlačítku přiřadili. Pokud se jedná o tlačítko „Začít měřit“, metoda provede přepnutí z úvodní obrazovky (okno Welcome) na druhé okno (Homography).

Okno Homography

V tomto okně proběhne inicializace potřebných proměnných (pro uložení obrázku, pro pole bodů, ...). Dále se nastaví slot pro tlačítko „Vyberte soubor“ a obrázkovému návěští se též nastaví slot reagující na kliknutí, pohyb a uvolnění myši. Poté se čeká na akci uživatele 3.3.

V následující části najdete seznam, v kterém tučný text na začátku značí akci uživatele a za ním následuje popis chování při této akci tj. to co se děje v programu.

- **Výběr obrázku ze souboru** – Uživatel provedl akci výběr obrázku. Metoda, kterou vyvolal signál po stisku tlačítka má za úkol načíst obrázek z uživatelova úložiště a nastavit ho středovému prvku UI.
- **Vyplnění textových polí** – Probíhá pouze kontrola, zda nejsou textová pole prázdná. Pokud ano, uživatel je na tuto skutečnost upozorněn.
- **Kliknutí do obrázku** – Kliknutím uživatel do obrázku vyvolá funkci starající se o vytvoření bodu v místě kliku uživatele. Tento bod je uložen do pole s body a poté vykreslen na obrázek. Maximální počet bodů jsou 4 body. Existují-li 4 body a přijde signál, body jsou vymazány a nastavování čtyř bodů probíhá znovu.
- **Pohyb s bodem** – Při této akci se v první fázi ověří, zda se povedlo najít bod, s kterým chce uživatel pohnout. Pokud ano, jedná se o pohyb s bodem. Pokud by bod nebyl nalezen, vytvoří se nový bod. Po uvolnění tlačítka se bodu nastaví nové souřadnice místa, kde došlo k uvolnění a je vykreslen do obrázku.
- **Další okno (tlačítko „Pokračovat“)** – Při této akci dojde k perspektivní transformaci obrazu a výstup této operace je odeslán nadcházejícímu oknu. Aktuální okno je zavřeno a řízení předáno oknu Doors.

Výše je popsána logika, která se provede při různých akcích uživatele. Vyskytla se zde funkce zabývající se zpracováním obrazu – perspektivní transformace. Tato funkce je pro větší přehlednost oddělena a popsána níže.

Perspektivní transformace

Proč provádět perspektivní transformaci je popsáno v teoretické části 2.3. Zde je uvedeno co se děje v programu.

Po vyvolání odpovídající metody tlačítkem, dojde k předání čtyř bodů zadané uživatelem. Body můžou být přeházené a tak je nutné provést řazení. Řadí se od levého horního po levý dolní roh po směru hodinových ručiček. Jakmile jsou body seřazeny, a je známa reálná šířka a výška je vypočítána matice Homografie. Matice je poté použita k odstranění perspektivního zkreslení a výsledný obrázek uložen a odeslán dalšímu oknu ke zpracování.

Okno Doors

V následující části je již odstraněno perspektivní zkreslení. Přistoupí se tedy k předzpracování obrazu pomocí metod k tomu určených:

- **Převod barevného obrazu na stupně šedi** – Obraz je převeden z barevného prostoru na stupně šedi.
- **Filtrování obrazu** – Obraz v odstínech šedi je poslán na vstup metody, která provede filtrování. Pro filtrování je vybrána funkce, která sloužící k ostření hran a odstranění šumu (Gaussův filtr).

- **Hranová detekce** – K rozpoznání potřebných objektů (nábytkových dvířek), je třeba provést hranovou detekci. Definici hrany naleznete v 2.4. K hranové detekci je použit Cannyho detektor, který je založen na průchodu druhé derivace obrazové funkce nulou 2.4 a prahování 2.5. Pomocí tohoto kroku jsou získány hrany potencionálních objektů. Můžou se zde ovšem objevit i další hrany, které nemají s objekty nic společného. Tento problém je řešen pomocí uživatele (vybere jen ty objekty, které chce měřit) a klasifikátoru. Výstupem Cannyho hranového detektoru je binární obraz.

- **Dilatace a eroze** – Obraz po hranové detekci může mít některé hrany rozpojené. Tyto dvě operace zajistí spojení rozpojených hran.

Po provedení metod výše je binární obraz nastaven obrázkovému návěští. Uživatel nyní může provádět akce, které jsou popsány ve stejném formátu jako výše. Těmi akcemi jsou:

- **Uzavření obrysů dvířek** – Pro správnou segmentaci objektů je důležité, aby uživatel co nejlíp uzavřel obrysy nábytkových dvířek. Při pohybu posuvníkem je volána metoda pro úpravu obrazu pomocí Cannyho hranového detektoru. Do této metody je poslána horní hodnota prahování, která je získána z posuvníku. Pokud je gradient pixelu vyšší než prahová hodnota je prohlášen za hranu. Pokud je nižší než dolní prahová hodnota není prohlášen za hranu. Pokud je mezi těmito dvěma hranicemi je prohlášen za hranu jen tehdy pokud je připojen k pixelu, který překročil horní hranici a je prohlášen za hranu. [7]

- **Výběr dvířek** – Kliknutím do obrázku uživatelem je proveden výběr dvířek. Po kliknutí jsou volány metody slouží k rozpoznání dvířek. Výstupem jsou nalezená dvířka, která jsou vykresleny do obrázku.

- **Úprava rohových bodů** – Na obrázkové návěští jsou nastaveny sloty, které pracují podobně jako v předešlém okně. Kliknutím uživatele je nalezen požadovaný bod, který je možné pohybem přesunout a uvolněním umístit. Při kliknutí na bod dochází k označení dvířka, kterému bod náleží. Toto označení je pomocí barvy nastaveno do hlavičky okna.

- **Výpočet (tlačítko „Pokračovat“)** – Po příchodu signálu, je předáno pole nesoucí objekty dvířek do následujícího okna (okno Results). Dále je tomuto oknu předáno řízení a aktuální okno je uzavřeno.

- **Přechod na předchozí okno** – Po stisku tlačítka zpět dojde k předání řízení předešlému oknu (okno Homography).

Metody sloužící k rozpoznání dvířek

Při výběru dvířek jsou jako první získány souřadnice kliku uživatele. Tyto souřadnice se nastaví do tzv. semínka (seed). Seed je poslán do metody FloodFill, což je tzv. rozlévání po určitou hranici. Zde jsou hranicí bílé pixely v binárním obraze. Výstupem rozlévání je maska 3.5 potencionálního objektu. U této masky jsou nalezeny pomocí houghových transformací existující úsečky. Tyto úsečky jsou vytříděny na nejlevější, nejpravější, nejvyšší a nejnižší. Zde proběhne klasifikace, tj. kontrola zda jsou úsečky čtyři a zda mají správný úhel. Pokud klasifikace proběhla v pořádku je objekt prohlášen za přijatý. Jelikož výstupem je jen potencionální hledaný objekt, jsou tyto úsečky hranicí hledaného objektu. Výpočtem průsečíků se získají rohové body dvířka, které jsou nastaveny objektu odpovídající třídě

nesoucí informace o nábytkových dvířkách. Tento objekt je uložen do pole objektů nábytkových dvířek. Pole se prochází a dvířka jsou vykreslena do obrázku pomocí rohových bodů 3.5. Toto pole je poté odesláno do okna Result kde dochází k měření dvířek v pixelech, přepočítání do reálných rozměrů a následné zobrazení do tabulky.

Obrázek 3.5: **Nalevo maska dvířek napravo nalezená dvířka v obraze.**

Okno Results

V posledním okně dochází k měření dvířek a vypsání výsledku do tabulky. Vstupem je pole objektů dvířek předané z předešlého okna. Pokud má uživatel více fotek, u kterých chce provést měření může se vrátit tlačítkem „Zpět“. Jakmile nastane tato skutečnost je předáno řízení předešlému oknu Doors a aktuální okno uzavřeno.

Postup při měření dvířek

Každý objekt nesoucí informace o dvířkách má podstatné čtyři atributy pro měření. Těmito čtyřmi atributy jsou levý horní, pravý horní, pravý dolní a levý dolní roh. K tomu abychom získali rozměry v pixelech musíme vypočítat středy těchto čtyř bodů. Takže vzniknout 4 středové body mezi levým horním – pravým horním, pravým horním – pravým dolním, ... rohem. Tyto středové body nám slouží k výpočtu šířky a výšky dvířka v pixelech. Vznikly tedy dva horizontální středové body horní a dolní a dva vertikální levý a pravý 3.6. Nyní je mezi horizontálními a vertikálními body vypočítána vzdálenost. Vzdálenost je vypočítána pomocí euklidovské vzdálenosti dvou bodů 2.7. Vzdálenost mezi horizontálními středovými body nám udává výšku v pixelech objektu. Šířka je udána vzdáleností mezi vertikálními body. Pro přepočet rozměrů v pixelech do reálných rozměrů je nutné vypočítat poměr. Poměr je počítán jen jednou a to při prvním průchodu. Šířka v pixelech dělena šířkou obrázku (vycházející z reálné šířky zadané uživatelem) dá výsledný poměr. Každý rozměr v pixelech je dělen tímto poměrem a tím se provede převod na reálné míry. Tyto míry jsou nastaveny objektu (dvířkám). Po výpočtu všech reálných rozměrů objektů v poli je pole procházeno a naměřené hodnoty vypisovány do tabulky.

Obrázek 3.6: **Ilustrační obrázek bodů sloužících k výpočtu velikosti**. Červeně jsou znázorněny známé rohové body. Zeleně horizontální středové body. Modře vertikální středové body.

Výpočet ceny

Uživatel má možnost výběrem ze tří typů dvířek dozvědět výslednou cenu nových. Po kliknutí na jedno z tlačítek se naměřené hodnoty přepočítají na m^2. Cena se vypočítá násobením ceny za m^2 spolu s naměřenými m^2. Výsledná cena je nastavena textovému návěští.

Závěr

V této kapitole je vysvětlen hlavní cíl práce a provedena analýza reálného prostředí. Dále jsou zmíněny akce, které je uživatel nucen provádět, a programátor nucen implementovat. Důležité je také zmínit, že byl probrán detailní návrh metod sloužící pro zpracování obrazu a měření objektů. Je důležité tuto část pochopit, jelikož je základním stavebním kamenem pro kapitolu 4.

Kapitola 4

Aplikace pro měření nábytkových dvířek

Tato kapitola se zabývá implementací návrhu řešení 3. Aplikace je rozdělena na dva logické celky – grafické uživatelské rozhraní a logika aplikace. Tyto dva celky jsou rozděleny do 6 logických a 4 grafických souborů. Pro každé uživatelské okno je přítomna logika okna. Metody z knihovny OpenCV jsou značeny značkou cv. Tato kapitola bude rozdělena na 2 části:

- Implementace uživatelského rozhraní.
- Implementace logiky aplikace.

Aplikace je implementována v programovacím jazyce Python a pro GUI je využita PyQt knihovna [13]. Aplikace je vyvíjena na operačním systému macOS Mojave. Pro úpravy kódu je využit textový editor Sublime Text [14]. Pro tvorbu GUI program Qt Designer [15]. Pro metody sloužící ke zpracování obrazu je použita OpenCV knihovna [11].

4.1 Datová sada

Prvním a základním podcílem této práce je získat data, které poté budou sloužit k testování vytvořeného měřícího algoritmu. Datovou sadou pro tuto práci se rozumí několik fotek nábytku či kuchyňské linky 4.1. Tyto fotky slouží jako vstup aplikace. Ke každé fotce je nutné znát i reálné rozměry nábytku/kuchyně, aby bylo možné provést porovnání přesnosti měření s naměřenými hodnotami aplikace.

Velikost datové sady by měla být kolem 4 fotek různých nábytků. Je to z toho důvodu, že každé dvířka mají jiný tvar a můžou být na nábytku jinak poskládána. Abychom tedy otestovali navržený algoritmus sloužící k rozpoznání dvířek, je potřebné použít 4 a více fotek. Fotky by měly být pořízeny kvalitnějším fotoaparátem, aby při dalších operacích (např. detekci hran) nedocházelo k selhání algoritmu z důvodu nízké kvality fotek.

4.2 Grafické uživatelské rozhraní

Způsobů jak v Pythonu udělat grafické uživatelské rozhraní je několik. Je možné použít integrovaný Tkinter [12] či některý z dostupných frameworků. Pro implementaci této části je využito knihovny PyQt. [16]


Obrázek 4.1: Ukázka obrázků z testovací sady.

PyQt knihovna

Tato knihovna je hodně komplexní, tedy má zabudováno spoustu významných uživatelských prvků. Je multiplatformní a její instalace je jednoduchá. Další výhodou je možnost použít pro tvorbu uživatelského rozhraní program jménem Qt Designer. Qt Designer má již předpřipravené prvky uživatelského rozhraní. Takže pohybem myši je možné je vzít a přetáhnout do tvořené aplikace. U každého prvku je možné nastavovat jeho vlastnosti (odsazení, velikost, font písma). [16]

Výsledná aplikace se skládá ze čtyř oken:

1. Welcome
2. Homography
3. Doors
4. Results

Mezi těmito okny je možné libovolně přepínat tam i zpět. V následující části je popsáno, jaké prvky jsou v GUI použity a jak to celé funguje.

Event loop

Event loop česky smyčka událostí, probíhá v metodě QApplication.exec. Jedná se o nekonečnou smyčku, která čeká na externí událost(kliknutí myši, pohyb myši, žádost OS o vykreslení okna, ...). [16]

Signály a sloty

Aby bylo možné vykonat nějakou akci uživatelem, je zde použit mechanizmus signálů a slotů. Signál se vyšle při akci uživatele (klik na tlačítko). K tomuto signálu je připojeno různé množství slotů (funkcí). Tyto funkce jsou po vyslání signálu zavolány a provedeno jejich tělo. [16]

Vytváření GUI funguje na principu skládání prvků do stromové hierarchie. Existuje tedy nejvyšší rodičovský prvek a do něj jsou přidáni potomci. Prvky jsou v PyQt též nazývány

widgety. [16] Níže jsou popsány prvky GUI, které jsou v této práci použity. Popis těchto prvků vychází z dokumentace [13].

- **QObject** – Jedná se o základní třídu všech Qt objektů. Klíčovou vlastností jsou signál a sloty, které jsou zmíněny výše.
- **QWidget** – Je potomkem QObject a základní třídou pro všechny prvky uživatelského rozhraní.
- **QMainWindow** – Hlavní okno aplikace. Poskytuje prostředí pro vytvoření prvků uživatelského rozhraní.
- **QVBoxLayout** – Oblast v které jsou prvky řazeny pod sebe (vertikálně).
- **QHBoxLayout** – Oblast v které jsou prvky řazeny vedle sebe (horizontálně).
- **QLabel** – Oblast která slouží k zobrazení textu či obrázku.
- **QPushButton** – Prvek reprezentující tlačítko, které se dá stisknout.
- **QTableWidget** – Prvek poskytující prostředí pro zobrazení tabulky.
- **QTableWidgetItem** – Buňka tabulky nesoucí určitý obsah.

Implementace GUI níže je popsána od shora dolů, tedy od nejvyššího prvku (rodiče) po nejnižší (potomky). Každému prvku je možné nastavit určitý typ vlastností, které ovlivní vzhled výsledné aplikace. Při nastavování vlastností jsem se řídil navrhnutým prototypem GUI, v části návrh řešení 3, i tak ale došlo k drobným změnám oproti návrhu.

Welcome okno

První okno, které uživatel uvidí je vytvořeno pomocí prvku `QMainWindow` (hlavní okno). Do hlavního okna je vložen `QWidget` (centralwidget), který je určený pro obsah aplikace. Centrální widget je rozdělen na tři části – hlavičku, tělo a patičku. Hlavička a patička jsou `QWidgety`, kterým je nastaveno horizontální rozležení prvků. Střední část je `QHBoxLayout` a nachází se zde název, typ a autor práce. V patičce se náchází `QPushButton` tlačítko pro vstup na další okno aplikace.

Další okna GUI jsou implementována podobně jako Welcome okno. Dochází zde pouze k obměně prvků uživatelského rozhraní.

4.3 Implementace logické části aplikace

Tato část je zaměřena na implementaci logické části aplikace, převážně na funkce sloužící ke zpracování obrazu. Pro účely zpracování obrazu je využito knihovny OpenCV [11].

Třída Main

Spouštěcí třída která vytvoří instanci třídy `QApplication`. Tato instance se stará o nastavení a řízení GUI. Poté je vytvořena instance okna Welcome a okno je vykresleno na obrazovku.

Třída Welcome

V této třídě nejsou použity žádné metody sloužící ke zpracování obrazu. Uživatel v této fázi zatím nenahrál obrázek. V konstruktoru třídy je volán příkaz pro načtení GUI ze souboru a k signálu tlačítka připojen odpovídající slot. Poslední příkaz `self.show()` vykreslí GUI na obrazovku.

```
class Welcome(QtWidgets.QMainWindow):
    def __init__(self):
      super(Welcome, self).__init__()
      uic.loadUi('Welcome.ui', self)
      self.measureButton.clicked.connect(self.switchToHomographyWindow)
      self.headerShadow = QGraphicsDropShadowEffect(self)
      self.headerShadow.setOffset(0,5)
      self.headerShadow.setBlurRadius(8)
      self.headerWidget.setGraphicsEffect(self.headerShadow)
      self.show()

    def switchToHomographyWindow(self):
      self.homographyWindow = Homography.Homography()
      self.homographyWindow.show()
      self.close()
```

Třída Homography

V konstruktoru je vytvořena instance třídy Vision. Její vytvoření je z důvodu uložení šířky zadané uživatelem a šířky měřené oblasti v pixelech. Tyto dvě proměnné budou v třídě Results sloužit k přepočtu naměřených rozměrů na reálné rozměry.

Signál tlačítka „Vybrat fotku“, vyvolá `QFileDialog`. Ten vrací cestu k vybranému souboru. Obrázek je pomocí `cv.imread` načten. Pokud je obrázek větší než velikost `QLabelu`, je provedena úprava velikosti na velikost `QLabelu` se zachováním poměru stran obrázku. O úpravu velikosti se stará metoda `changeImageSize`. Po úpravě velikosti je obrázek převeden z OpenCV reprezentace na PyQt reprezentaci. Jelikož probíhá komunikace mezi dvěma knihovnami OpenCV a PyQt. Je třeba provádět převod obrázku mezi těmito dvěma knihovnami. To zajišťují funkce `cvImg2QImageRGB` a `cvImg2QImage`. Po převodu je obrázek vykreslen do `QLabelu`. O vykreslení se stará třída Paint 4.3. Provede se instanciace třídy Paint a zobrazí se příkazem `show`.

Aby bylo možné vypočítat matici homografie musí uživatel zadat výšku a šířku měřené oblasti. Šířka zadaná uživatelm je uložena do instace třídy vision pomocí `setWidthFromUser`. Dále tuto oblast musí vybrat na zvoleném obrázku. Pro vybranou oblast je vypočítána její šířka v pixelech a uložena instanci třídy vision pomocí `setWidthInPixels`. Body nesou souřadnice kliku uživatele tedy (x, y). Tyto souřadnice je nutné posunout. Je to z toho důvodu že OpenCV funkce pracují čistě s obrázkem a ne s velikostí celého `QLabelu`. Posun zajišťují proměnné `shiftY` a `shiftX`. Upravené body jsou uloženy do pole a spolu s rozměry jsou odeslány metodě `perspectiveTransform`. Tato metoda volá pomocnou metodu `orderPoints`. Pomocná metoda seřadí vstupní body od levého horního po levý dolní. Je zde využito tzv. numpy array. Levý horní roh má nejmenší součet, pravý dolní roh má největší součet. Jednoduše pomocí `np.argmin` a `np.argmax` jsou získány zmíněné body. Podobně získáme další dva body. Získané body jsou uložen do pole dle výše zmíněného pořadí.

Dále jsou pomocí reálných rozměrů vytvořeny cílové body a též uloženy do pole. Níže je ukázka, jak je v programu pracováno se zdrojovými a cílovými body.

```
suma = points.sum(axis = 1)
difference = np.diff(points, axis = 1)

topLeftCorner = points[np.argmin(suma)]
bottomRightCorner = points[np.argmax(suma)]
topRightCorner = points[np.argmin(difference)]
bottomLeft = points[np.argmax(difference)]

sourcePoints[0] = topLeftCorner
sourcePoints[1] = topRightCorner
sourcePoints[2] = bottomRightCorner
sourcePoints[3] = bottomLeft
.
.
.
width = int(self.realWidth)
height = int(self.realHeight)

destinationPoints = np.array([[0, 0], [width - 1, 0],
[width - 1, height - 1], [0, height - 1]], dtype = "float32")
```

Aby bylo možné zavolat funkci `cv.getPerspectiveTransform` je potřeba znát zdrojové body a cílové body. Ty jsou již nadefinovány. Nyní je tedy volána funkce `cv.getPerspectiveTransform`, která vrátí matici homografie.

Pomocí této matice H je možné částečně odstranit perspektivní zkreslení nahraného obrázku. Perspektivní zkreslení je odstraněno pomocí funkce `cv.warpPerspective`. Jako argumenty jsou funkci předány obrázek, matice homografie, a velikost výsledného obrázku. Obrázek je vrácen a je předáno řízení oknu Doors.

Paint

Třída `Paint` dědí z třídy `QLabel`, instanciací této třídy vznikne objekt nesoucí obrázek zvolený uživatelem. Tato třída se stará krom obrázku o vytváření bodů a jejich pohyb.

Třída Doors

Vstupem této třídy je obrázek po perspektivní transformaci. Jejím úkolem je segmentace a klasifikace dvířek. Tedy nalézt rohové body hledaných objektů, nastavit je příslušné instanci třídy Door a vykreslit je do obrázku.

O vykreslení obrázku a správu bodů se stejně jako v třídě Homography stará třída Paint, s menšími obměnami. V konstruktoru je jako třídní atribut natavena instance třídy Vision získaná z předcházející třídy Homography. Pomocí vytvořené instance třídy Vision, je poté volána třídní metoda `callPreprocessorMethods`, která zajistí předzpracování obrazu. Obrázek je vrácen a vykreslen pomocí třídy Paint do středového `QLabelu`. Uživatel tedy vidí binární obrázek s obrysy objektů v obraze. Pohybem posuvníku je volána metoda `updateCannyImg`, která provede znovu hranovou detekci pomocí `cv.Canny` a nastaví jí jako hodnotu horního prahu hodnotu z posuvníku. Poté je provedena dilatace `cv.dilate` a

eroze `cv.erode` k uzavření obrysů. Aktualizovaný obrázek je převeden z OpenCV do PyQT a vykreslen.

Kliknutím do středu objektu, který chce uživatel měřit. Je volána třídní metoda `floodFill` třídy Vision.

Třída Vision

Popis OpenCV metod vychází z [10]. Tato třída v sobě nese podstatné metody zabývající se zpracováním obrazu. Metody vyobrazeny v seznamu níže jsou z knihovny OpenCV a jsou řazeny tak, jak po sobě následují v programu. Výstup předešlé funkce je vždy vstupem aktuální. Třídní metoda `callPreprocessorMethods` ve svém těle obsahuje:

- `cv.cvtColor` – Provede převod obrázku z barevného na stupně šedi.
- `cv.GaussianBlur` – Zvýraznění hran a potlačení šumu. Argumenty jsou obraz, konvoluční jádro a odchylka jádra ve směru X.
- `cv.Canny` – Provede na obraze hranovou detekci. Hrany jsou značeny bíle, zbytek černě. Výstupem je tedy binární obraz. Na vstup metody je poslán obraz, dolní práh a horní práh.
- `cv.dilate a cv.erode` - Slouží k uzavření otevřených obrysů. Argumenty jsou obraz, jádro ve tvaru matice (5x5) a počet kolikrát se dilatace/eroze aplikuje.

Další podstatnou metodou je `floodFill`. Metoda přijímá předzpracovaný obrázek a provádí segmentaci a klasifikaci objektů. Obsahuje:

- `cv.floodFill` – Tato metoda je jádrem celé aplikace. Přijímá tzv. seed, což jsou upravené souřadnice (x, y) kliku myši uživatele. Z tohoto bodu se poté rozlévá až narazí na bílou hranici (hrany). Vznikne maska oblasti, která pokrývá potencionální objekt. Argumenty tedy jsou předzpracovaný obraz, maska, semínko (seed) a barva pixelů 4.2.
- `cv.Canny` – Na masku je aplikována hranová detekce, kvůli nalezení hranice objektů pomocí Houghových transformací.
- `cv.HoughLinesP` – Slouží k detekci přímek v obraze v původní verzi. Tato upravené verze pracuje rychleji a vrací souřadnice dvou krajních bodů - tedy detekuje úsečky. Konkrétně hranice hledaného objektu. Vstupem je maska objektu po hranové detekci. Dalšími argumenty jsou:
  - ρ značí velikost kroku v pixelech při průchodu algoritmem
  - θ značí velikost úhlu při průchodu algoritmem.
  - Threshold minimální množství průsečíků pro schválení čáry.
- `classifyLines` – Vstupem jsou nalezené úsečky. V těle této metody jsou procházeny v cyklu a tříděny na horizontální a vertikální. Jsou vybrány pouze ty, které odpovídají určité odchylce od nuly (horizontální úsečky) a devadesáti stupňů (vertikální úsečky).
- `findLinesExtremes` – Tato metoda najde nejlevější, nejpravější, nejvyšší a nejnižší úsečku. Což jsou hranice hledaného objektu. Proběhne zde kontrola zda jsou nalezeny 4 úsečky.

- `computeIntersection` – Vypočítá průsečíky pomocí čtyř úseček a nastaví je objektu nesoucí informace o nábytkových dvířkách. Tyto průsečíky jsou rohy objektu. Objekt byl úspěšně segmentován a klasifikován.

- `computeDoorsSize` – Prochází se pole objektů (dvířek). A počítají se středové body za pomoci známých souřadnic rohů. Pomocí getterů jsou souřadnice rohů z objektu získány a odeslány metodě `computeMiddlePoint`. Ta vypočítá střed mezi dvěma rohy. Jakmile jsou vypočítány dva horizontální (horní, dolní) a dva vertikální (pravý, levý) středy, odešlou se do funkce `distanceBetweenTwoPoints`. Vzdálenost mezi levým středovým bodem a pravým středovým bodem je šířka v pixelech. Vzdálenost mezi horním středovým bodem a dolním středovým bodem je výška objektu v pixelech.

- `computeRealDimension` – Metoda převede rozměry v pixelech na reálné rozměry. Tuto operaci je možné vykonat díky reálným rozměrům zadaných na počátku aplikace uživatelem. Přepočítané rozměry jsou nakonec nastaveny objektu (dvířku).

Obrázek 4.2: **Ilustrace floodFill algoritmu.**

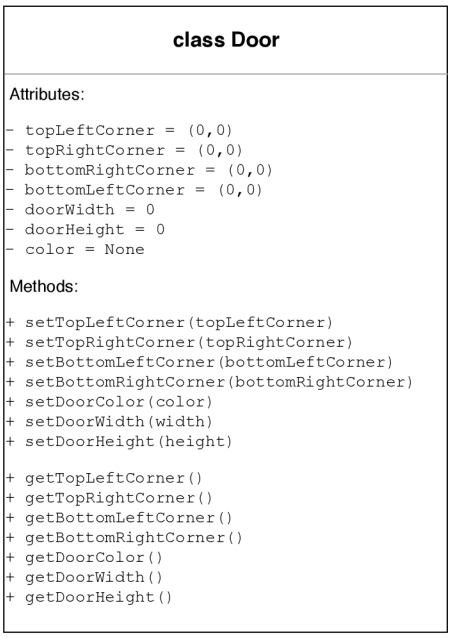
Třída Door

Tato třída slouží k ukládání informací o nábytkových dvířkách. Pomocí getterů a setterů se získává a nastavují potřebné atributy.

Instance třídy Door 4.3 jsou ukládány do pole a navráceny do okna Doors. V okně Doors jsou vykresleny na obrazovku. Uživatel má možnost před výpočtem provést úpravu rohových bodů vykreslených dvířek.

Třída Result

V této třídě je známa poloha rohových bodů dvířek. Je tedy možné přejít k výpočtu. Výpočet je proveden voláním metody `computeDoorsSize` třídy Vision. Tato funkce je popsána výše. Vrací objekty dvířek s reálnými rozměry. Tyto rozměry jsou poté pomocí cyklu vypsány do `QTableWidget`. Dále se v této třídě nachází `computeDoorsPrice`, která vypočítá



Obrázek 4.3: Diagram třídy pro ukládání informací o dvířkách.

cenu nových dvířek. Výpočet probíhá jako násobek naměřených metrů čtverečních s cenou za metry čtvereční podle typu dvířek.

V této kapitole je zmíněno, jak je tato aplikace implementována. Výsledná aplikace je rozdělena na 2 logické části – grafické uživatelské rozhraní a logika aplikace. Je využito programovacího jazyka Python, pomocných knihoven OpenCV pro zpracování obrazu a PyQt pro potřeby grafického rozhraní. V následující části testování 5 je ukázáno, jaké přesnosti algoritmus pro měření dosahuje.

Kapitola 5

Testování

V této části je aplikace již plně funkční a je tedy možné přikročit k testování. Testována bude přesnost měření nábytkových dvířek. Tato část obsahuje 4 typy testů, na kterých jsou ukázány úspěchy a selhání výsledné aplikace. Na začátku testu je ukázka vstupní fotky, poté ukázka jak se povedla segmentace s klasifikací a v poslední části vyhodnocení naměřených rozměrů. Každý test byl proveden 3x a byly z něj vybrány nejlepší výsledky měření. Testovací fotky se nachází ve složce fotkyNabytku. Procenta úspěšnosti klasifikace a výsledky měření jsou zaokrouhleny nahoru.

5.1 Matná dvířka bez hloubkového frézování

Na první testované fotce 5.1 jsou jednoduché rovné dvířka bez tvaru. Jak můžete vidět, je zde celkem malý kontrast mezi hranou dvířka a pozadím. Dále se zde nachází rušivý element v podobě hranatého obkladu. Reálné rozměry měřené oblasti jsou 746 mm x 2348 mm (výška x šířka).

Segmentace a klasifikace

Pokud se povede uzavření obrysů dvířek 5.2, značí to pozitivní výsledek při klasifikaci dvířek 5.3. A tedy pozitivní výsledek při měření.

U testované fotky se kompletní klasifikace povedla 2x a 1x došlo k selhání dvou z pěti dvířek. Selhání nastalo z důvodu neuzavřeného obrysu. Obrys se nepovedlo uzavřít z důvodu nízkého kontrastu mezi hranou objektu a pozadím. Důvod proč se to 2x povedlo a 1x ne je vysvětlen v následujícím textu. Hranová detekce je provedena z obrázku, který prošel perspektivní transformací. Před provedení perspektivní transformace uživatel vybírá oblast, na kterou se perspektivní transformace aplikuje. Výsledný obraz po perspektivní transformaci je pokaždé mírně jiný. Je to ovlivněno ručním výběrem (uživatel klikne myší) krajních bodů oblasti, před provedením perspektivní transformace (okno Homography).

Přesnost měření

Po správném určení hranic objektů se provede měření. Při výběru dvířek jsem postupoval zleva doprava a shora dolů. V tomto pořadí je to i prezentováno v tabulce níže. Bylo vybráno nejlepší měření z tří pokusů a výsledky naleznete v tabulce 5.1.

Obrázek 5.1: **Ukázka fotky 01.**

Tabulka výsledných rozměrů					
Reálná v.	Naměřená v.	Reálná š.	Naměřená š.	Přesnost v. (%)	Přesnost š. (%)
746.00	**746.43**	446.00	**451.64**	0.43	5.64
746.00	**742.65**	446.00	**447.86**	-3.35	1.86
746.00	**735.09**	596.00	**587.69**	-10.91	-8.31
746.00	**736.98**	596.00	**591.47**	-9.02	-4.53
746.00	**736.98**	246.00	**243.77**	-9.02	-2.23

Tabulka 5.1: Tabulka s nejlepším výsledkem prvního testu.

Závěr

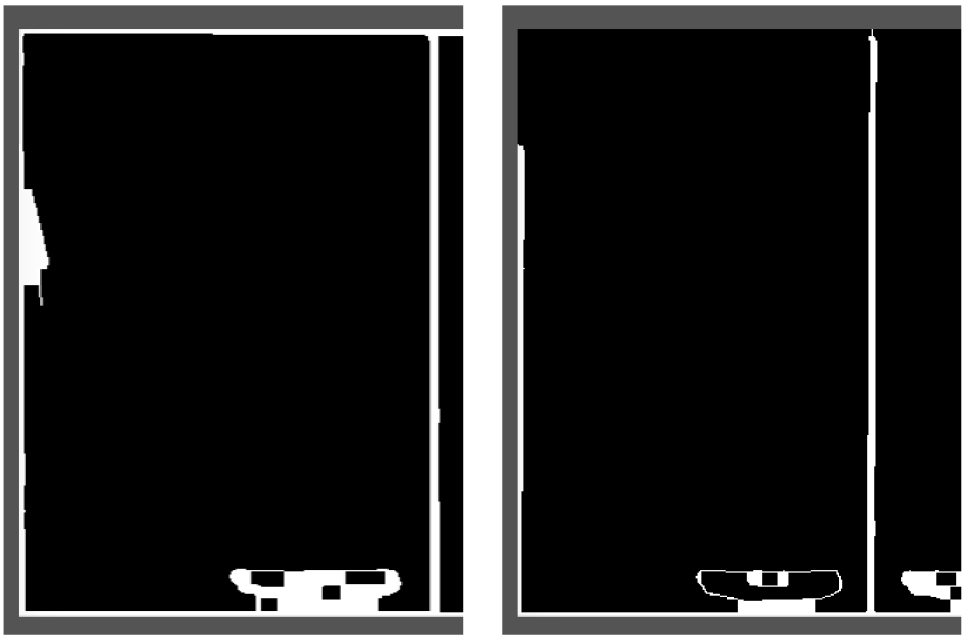
Z naměřených výsledků vyplývá, že průměrná odchylka od reálné výšky je **6.55** mm a **4.51** mm od reálné šířky dvířek. Úspěšnost při automatické klasifikaci objektů je 87 %.

5.2 Lesklá dvířka bez hloubkového frézování

V předešlém testu jsou testovány dvířka bez tvaru v matném provedení. Tento test má za úkol vyzkoušet jak si obstojí lesklé dvířka 5.4. U lesklých dvířek může docházet k různým odrazům světla a tedy obtížnějšímu rozpoznání hledaného objektu. Reálné rozměry měřené oblasti jsou 678 mm x 1590 mm (výška x šířka).

Segmentace a Klasifikace

Byly provedeny 3 pokusy a pokaždé se povedlo nalézt rohové body všech dvířek, tedy úspěšnost při automatické klasifikaci je 100 %. Pokud se provede dobré prahování – odstranění



Obrázek 5.2: **Porovnání úspěšného a neúspěšného uzavření obrysů.**

odlesku z prostoru dvířek je nalezení objektů na velmi dobré úrovni. Porovnání hranové detekce s odleskem a bez odlesku je na obrázcích 5.5 a 5.6.

Přesnost měření

Po správném určení hranic objektů se provede měření. Při výběru dvířek jsem postupoval zleva doprava. V tomto pořadí to je prezentováno v tabulce níže. Bylo vybráno nejlepší měření z tří pokusů a výsledky naleznete v tabulce 5.2.

Tabulka výsledných rozměrů					
Reálná v.	Naměřená v.	Reálná š.	Naměřená š.	Přesnost v. (%)	Přesnost š. (%)
250.00	**248.34**	995.00	984.16	-1.66	-10.84
250.00	**248.34**	995.00	990.29	-1.66	-4.71
250.00	**248.36**	995.00	984.16	-1.64	-10.84
250.00	**248.34**	995.00	991.82	-1.66	-3.18

Tabulka 5.2: Tabulka s nejlepším výsledkem druhého testu.

Závěr

Z tohoto testu je patrné, že aplikace zvládne i dvířka s mírným odleskem na jejich ploše. Při tomto testu je průměrná odchylka od reálných rozměrů 1.66 mm pro výšku dvířek a 7.39 mm pro šířku dvířek. V dalším testu je ukázáno, jak funguje aplikace s dvířky v nějakém složitějším tvaru.

Obrázek 5.3: **Porovnání úspěšného a neúspěšného nalezení rohů objektu.**

Obrázek 5.4: **Ukázka fotky 02.**

5.3 Matná a lesklá dvířka s hloubkovým frézováním

Jednoduchý tvar dvířek je testován ve dvou předešlých testech. V tomto testu je předvedeno, jak si obstojí dvířka se složitějším tvarem 5.7.

Segmentace a Klasifikace

Z obrázku níže 5.8 je patrné, že dvířka s tvarem obdélníku uprostřed, znemožní úplnou automatickou klasifikaci. Tedy u těchto tvarů algoritmus selhává a je nutné vybrat hranici objektu ručním zadáním. Dvířka s jednodušším tvarem jsou úspěšně klasifikována. Dvířka s obdélníkovým tvarem uprostřed se úspěšně nepovedlo klasifikovat ani jednou tedy úspěšnost 0 %. U jednoduššího tvaru (dvířek nalevo) se to povedlo vždy – úspěšnost 100 %.

Obrázek 5.5: **Ukázka hranové detekce s odleskem.**

Obrázek 5.6: **Ukázka hranové detekce bez odlesku.**

Přesnost měření

Po správném určení hranic objektů se provede měření. Při výběru dvířek jsem postupoval zleva doprava. V tomto pořadí to je prezentováno i v tabulce níže. Bylo vybráno nejlepší měření z tří pokusů a výsledky naleznete v tabulce 5.3.

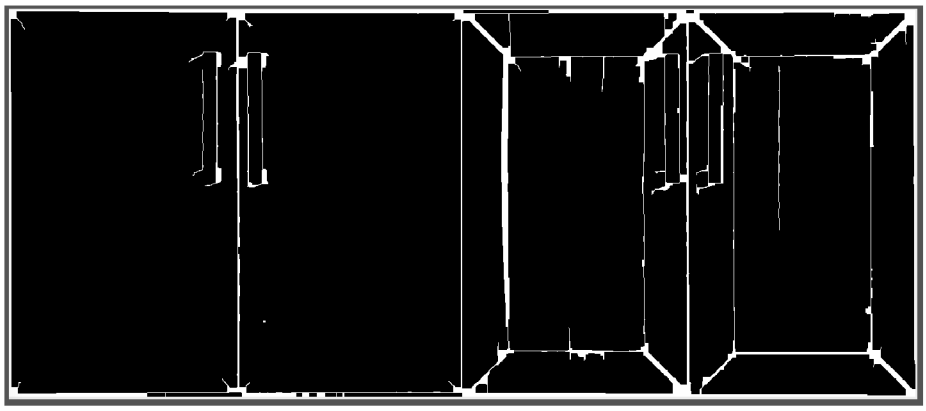
Tabulka výsledných rozměrů					
Reálná v.	Naměřená v.	Reálná š.	Naměřená š.	Přesnost v. (%)	Přesnost š. (%)
678.00	**674.54**	396.00	**396.23**	-3.46	0.24
678.00	**669.39**	396.00	**393.66**	-8.64	-2.34
678.00	**669.37**	396.00	**389.82**	-8.63	-6.18
678.00	**671.94**	396.00	**396.24**	-6.06	0.24

Tabulka 5.3: Tabulka s nejlepším výsledkem třetího testu.

Závěr

Tímto testem jsem poukázal na skutečnost, že výsledná aplikace selhává při složitějším tvaru dvířek – přesněji obdélníkovém tvaru uprostřed. Tyto dvířka ještě navíc obsahují texturu, která může také představovat problém. Při tomto testu bylo dosaženo průměrné odchylky 6.70 mm od reálné výšky a 2.25 mm od reálné šířky dvířek. Při automatické klasifikaci bylo dosaženo 50 % úspěšnosti. Další test bude tedy brát v úvahu texturovaná nábytková dvířka.

Obrázek 5.7: **Ukázka fotky 03.**



Obrázek 5.8: **Ukázka hranové detekce u třetího testu.**

5.4 Matná texturovaná dvířka bez hloubkového frézování

V tomto testu jsou testována dvířka s texturou dřeva s plochým tvarem 5.9. Rozměry měřené oblasti jsou 1949 mm x 1649 mm (výška x šířka).

Segmentace a Klasifikace

Při nastavování prahu je nutné co nejvíc odstranit vnitřní texturu. Je však potřeba brát ohled na hrany. Proto se může stát, že část textury zůstane, avšak nesmí být spojená. Ne vždy se podaří dobře odstranit texturu a zároveň zachovat hrany. U testovaného obrázku 5.10 se nepovedlo uzavřít všechny obrysy a zároveň rozpojit vnitřní texturu. Je tedy nutné zadat vrcholy dvířka rozpojených obrysů ručně. Úspěšnost automatické klasifikace je 47 %.

Přesnost měření

Po správném určení hranic objektů se provede měření. Při výběru dvířek jsem postupoval zleva doprava, shora dolů. V tomto pořadí to je prezentováno i v tabulce níže. Bylo vybráno nejlepší měření z tří pokusů a výsledky naleznete v tabulce 5.4.

Obrázek 5.9: **Ukázka fotky 04.**

Tabulka výsledných rozměrů					
Reálná v.	Naměřená v.	Reálná š.	Naměřená š.	Přesnost v. (%)	Přesnost š. (%)
1196.00	**1208.34**	446.00	**453.08**	12.34	7.08
295.00	**299.15**	596.00	**598.3**	4.15	2.3
1196.00	**1208.34**	596.00	**604.14**	12.34	8.14
746.00	**729.05**	446.00	**444.37**	-16.95	-1.63
746.00	**726.1**	596.00	**584.81**	-19.9	-12.19

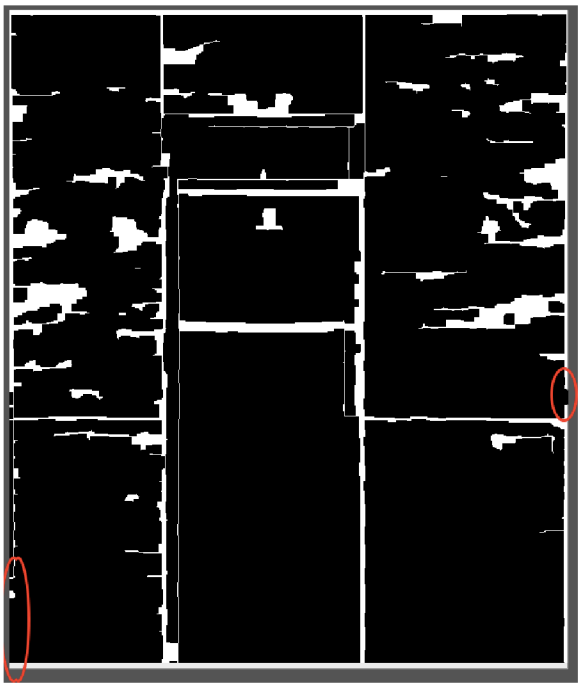
Tabulka 5.4: Tabulka s nejlepším výsledkem čtvrtého testu.

Závěr

Pokud se otevřou obrysy dvířka, jak je vidět na 5.10, tak automatická klasifikace daného objektu je neúspěšná. Z toho plyne, že pro texturované objekty je možné provést automatickou klasifikaci jen částečně. Je tedy téměř vždy nutné provést i manuální úpravu či zadání rohových bodů manuálně. Při tomto testu bylo dosaženo průměrné odchylky 13.14 mm od reálné výšky a 6.27 mm od reálné šířky dvířek. Automatická klasifikace byla z 47 % úspěšná.

Vyhodnocení testování

V této kapitole jsou provedeny čtyři testy různého charakteru. První test se zabývá dvířky s plochým tvarem bez textury. Tento pokus dopadl nadprůměrně a úspěšnost automatického nalezení objektů byla 87 %. Při měření je dosaženo průměrné odchylky **6.55** mm od reálné



Obrázek 5.10: **Ukázka hranové detekce s rozpojeným obrysem. Oblast, kde došlo k rozpojení, je zvýrazněna červenou elipsou.**

výšky a **4.51** mm od reálné šířky dvířek. V druhém testu se testuje opět plochý tvar dvířek s lesklým povrchem. Tento test ukázal, že při dobrém prahování nemá malý odlesk žádný vliv na výslednou detekci objektů. Bylo dosaženo 100 % úspěšnosti při automatické klasifikaci. Při měření je dosaženo průměrné odchylky **1.66 mm** od reálné výšky dvířek a **7.39** mm od reálné šířky dvířek. Třetí test se zabývá dvířky se složitějším tvarem frézování. Zde se povedlo automaticky klasifikovat pouze dvířka s jednodušším tvarem, tedy úspěšnost klasifikace byla 50 %. Body u dvířek se složitým tvarem, jsou zadány ručně. Při měření je dosaženo průměrné odchylky **6.70 mm** od reálné výšky dvířek a **2.25 mm** mm od reálné šířky dvířek. Čtvrtý test měří dvířka s plochým tvarem a s texturou dřeva. Texturované plochy jsou velmi obtížné, protože tvoří falešné hrany. Při třech pokusech se klasifikace povedla s 47 % úspěšností. Při měření je dosaženo průměrné odchylky **13.14 mm** od reálné výšky dvířek a **6.27 mm** mm od reálné šířky dvířek.

Aplikace tedy funguje velmi dobře pro hladké tvary jak matné tak lesklé bez textury. Při hloubkovém frézování dvířek či texturovaných dvířkách aplikace z velké části selhává a je potřeba zásah uživatele.

Kapitola 6

Závěr

Cílem práce bylo vytvořit aplikaci pro měření nábytkových dvířek pomocí fotografie a z naměřených výsledků vypočítat cenu dvířek nových.

K dosažení cíle bylo nutné nastudovat, jak funguje snímání reálného světa pomocí fotoaparátu. Bylo zjištěno, že při snímání může vlivem čočky docházet ke zkreslení reálných objektů. Toto zkreslení se dá odstranit pomocí perspektivní transformace, která je provedena pomocí matice homografie. Dále předzpracování obrazu, při kterém je nutné odstranit šum z důvodu zvýraznění zašumělých hran a poté provést hranovou detekci. Následně metodu FloodFill a Houghovy transformace. Tyto metody slouží k segmentaci objektů. V poslední řadě jsem nastudoval euklidovskou vzdálenost, pomocí které je provedeno finální měření vzdáleností.

Po prostudování teoretických informací byla navrhnuta desktopová aplikace pro měření nábytkových dvířek z fotografie. Tento návrh byl pomocí jazyka Python implementován.

Pokud se běžnému uživateli podaří uzavřít obrysy, velmi dobře pracuje aplikace s dvířky, které nemají hloubkové frézování, ať jsou matná či lesklá. V tomto případě bylo dosaženo 93 % úspěšnosti při automatické klasifikaci. Při nábytkových dvířkách s hloubkovým frézováním aplikace selhává. Automatická klasifikace dvířek s hloubkovým frézováním dosáhla úspěšnosti 50 %. U dvířek s texturou se automatická klasifikace povedla na 47 %. U hloubkového frézování a texturovaných dvířek je tedy uživatel nucen zadávat rohové body ručně. Při měření bylo dosaženo průměrné odchylky 6 mm od reálných rozměrů.

Prvkem pro zlepšení je přesnost meření. Tu by se dalo zpřesnit více informacemi o dané scéně. Tedy nevycházet z jedné fotky, ale použít rozšířenou realitu či video, které by uživatel natočil.

Na závěr je nutno říct, že tato úloha je velmi komplexní a pro její kompletní vyřešení by muselo být zkombinováno několik algoritmů zabývajících se zpracováním obrazu. Pro lepší klasifikaci hledaných objektů by se dalo využít umělé inteligence.

Literatura

[1] Apple Inc.: *ARKit 2*. 2019, [Online; navštíveno 24.04.2019].
URL https://developer.apple.com/arkit/

[2] Dr. Gerhard Roth: *Homography*. [Online; navštíveno 20.03.2019].
URL http://people.scs.carleton.ca/~c_shu/Courses/comp4900d/notes/homography.pdf

[3] Forsyth David, A.: *Computer vision a modern approach*. New Jersey: Prentice-Hall, vyd. 1 vydání, 2003, ISBN 0-13-191193-7.

[4] Hlaváč, V.: Hledání hran. [Online; navštíveno 11.03.2019].
URL http://6.869.csail.mit.edu/fa12/lectures/lecture13ransac/lecture13ransac.pdf

[5] Hlaváč, V.: *Počítačové vidění*. Praha: Grada, 1992, ISBN 80-85424-67-3.

[6] MIT CSAIL: *Homographies and RANSAC*. [Online; navštíveno 20.03.2019].
URL http://cmp.felk.cvut.cz/cmp/courses/33zsl1zima2005/slidy/DetekceHran.pdf

[7] OpenCV dev team : *Canny Edge Detector*. 2011-2014, [Online; navštíveno 24.04.2019].
URL https://docs.opencv.org/2.4.13.7/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html

[8] OpenCV dev team : Hough Line Transform. 2011-2014, [Online; navštíveno 25.04.2019].
URL https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/hough_lines/hough_lines.html

[9] OpenCV dev team : Miscellaneous Image Transformations. 2011-2014, [Online; navštíveno 25.04.2019].
URL https://docs.opencv.org/2.4.13.7/modules/imgproc/doc/miscellaneous_transformations.html

[10] OpenCV dev team : OpenCV API Reference. 2011-2014, [Online; navštíveno 9.05.2019].
URL https://docs.opencv.org/3.0-beta/modules/refman.html

[11] OpenCV team: About. 2019, [Online; navštíveno 24.04.2019].
URL https://opencv.org/about/

[12] Python Software Foundation: Graphical User Interfaces with Tk. 2001-2019, [Online; navštíveno 24.04.2019].
URL https://docs.python.org/3/library/tk.html#graphical-user-interfaces-with-tk

[13] Riverbank Computing Limited: *What is PyQt? 2018, [Online; navštíveno 24.04.2019].*
URL
https://www.riverbankcomputing.com/software/pyqt/intro/index.html

[14] Sublime HQ Pty Ltd: Sublime Text. [Online; navštíveno 24.04.2019].
URL https://www.sublimetext.com

[15] The Qt Company Ltd: Qt Designer Manual. 2019, [Online; navštíveno 24.04.2019].
URL https://doc.qt.io/qt-5/qtdesigner-manual.html

[16] Viktorin P.,Hrončok M. a další: GUI v Pythonu: PyQt5. 2016-2017, [Online; navštíveno 24.04.2019].
URL https://naucse.python.cz/lessons/intro/pyqt/

[17] Šonka, M.: Image processing, analysis, and machine vision. *Toronto: Thomson, třetí vydání, 2008, ISBN 978-0-495-08252-1.*

Příloha A

Obsah CD

Na přiloženém CD se nachází následující adresáře a soubory:

- **src** – zdrojové soubory výsledného programu a readme
- **tex** – zdrojové soubory technické zprávy
- **plakat.pdf** – plakát prezentující práci
- **video.mp4** – video prezentující práci
- **TechnickaZprava.pdf** – technická zpráva
- **TechnickaZpravaTisk.pdf** – technická zpráva pro tisk

Příloha B

Manuál

B.1 Požadavky pro spuštění

Výsledná aplikace byla testována na macOS Mojave 10.14. Pro běh aplikace je využito programovacího jazyka Python 3.7.0 a balíčků/knihoven:

- colorama 0.4.1
- colorlog 4.0.2
- colormap 1.0.2
- easydev 0.9.37
- numpy 1.16.3
- opencv-contrib-python 4.1.0.25
- pexpect 4.7.0
- ptyprocess 0.6.0
- PyQt5 5.12.2
- PyQt5-sip 4.19.17

B.2 Spuštění aplikace

Aplikace se spustí příkazem `python Main.py`. Pro pochopení jak aplikace funguje je nutné se podívat na přiložené video (video.mp4).