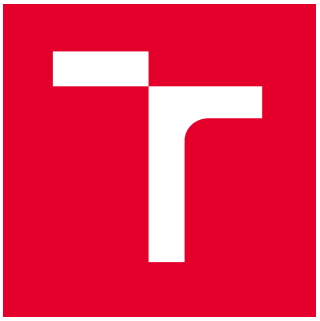


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

DETEKCE ZMĚN V OBRAZE

DETECTION OF CHANGES IN THE IMAGE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — **Jan Čech**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Miloslav Richter, Ph.D.**

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Jan Čech ***ID:*** 211416

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Detekce změn v obraze

POKYNY PRO VYPRACOVÁNÍ:

V obraze pořízeném pohybující se kamerou je nutné vyznačit změny, které se udály od posledního průchodu (pohledu kamery) stejným místem.

1) Nastudujte metody zpracování obrazu a identifikace objektů.

2) Navrhněte testovací pracoviště pro ověřování metod a nasnímejte několik sad pro testování.

3) Navrhněte a realizujte algoritmy pro extrakci přítomných objektů, zjištění shodných pozic v jednotlivých průchodech, a následné zjištění rozdílů. Zvolte vhodnou reprezentaci dat.

4) Otestujte realizované algoritmy a postupy na testovací sadě. Zhodnoťte výsledky.

DOPORUČENÁ LITERATURA:

Gonzalez R.C., Woods R.E.: Digital Image Processing, 4th edition, Pearson, 2017, ISBN 978-0133356724

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 22.5.2023

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

doc. Ing. Václav Jirsík, CSc.
předseda rady studijního programu

Abstrakt

Cílem této práce je nalezení změn ve dvou podobných obrazech pořízených v různých okamžicích. Tato problematika spadá do oboru počítačového vidění. K řešení tohoto problému je použito několik různých metod počítačového vidění v několika verzích řešení. Program pracuje se dvěma scénami. Ve finální verzi jsou objekty z obou scén extrahovány a vzájemně porovnány. Výsledek srovnání je zapsán do logovacího souboru.

Klíčová slova

Scéna, klíčový bod, SURF, detekce změn, AR tag

Abstract

The aim of this thesis is finding changes in two similar pictures made in different periods. This task belongs to the field of computer vision. To solve this problem different methods of computer visions are used in several solution versions. Program is working with two scenes. In final version are objects from both scenes compared. The result is written into logging file.

Keywords

Scene, point of interest, SURF, change detection, AR tag

Bibliografická citace

ČECH, Jan. Detekce změn v obraze. Brno, 2023. Dostupné také z: https://www.vut.cz/studenti/zav-prace/detail/151565. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Miloslav Richter.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	*Jan Čech*
VUT ID studenta:	*211416*
Typ práce:	*Bakalářská práce*
Akademický rok:	*2022/23*
Téma závěrečné práce:	*Detekce změn v obraze*

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

.

V Brně dne: 22. června 2023 ----------

podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Miloslavu Richterovi, Ph.D. za ochotu vést mě při zpracování mé bakalářské práce.

V Brně dne: 22. července 2023 ----------

podpis autora

Obsah

SEZNAM OBRÁZKŮ

Úvod

Mnoho lidí by chtělo mít co největší kontrolu nad svým majetkem. Příkladem může být vlastník obchodu se starožitnostmi velké peněžní hodnoty. V takové situaci byste nejraději měli svůj majetek pod kontrolou celý den. To je ale nepraktické vzhledem k nutnosti vaší nepřetržité přítomnosti.

Možným řešením by mohl být program, který kontroluje stav zboží ve vašem obchodě. Můžete tedy obchod bez problémů svěřit zaměstnanci s vědomím, že program vás upozorní v případě změny ve vašem obchodě.

O možném zpracování takového programu pojednává tato práce. K realizaci tohoto systému je použit obor počítačové vidění, který se zabývá zpracováním digitálních obrazů a získáváním informací z něj. V počítačovém vidění je většinou využívána následující posloupnost operací, snímání, předzpracování, segmentace, popis.

Nejprve je potřeba pořídit snímek scény, pak je potřeba tento snímek upravit (filtrace, změna jasu, …), následně je provedena segmentace, což je hrubé rozdělení objektů na snímku a nakonec je proveden popis, což je vlastně hlavní zdroj informací ze snímku.

Aby bylo kde snímky pořizovat, je potřeba navrhnout testovací pracoviště. Testovací pracoviště by mělo co nejlépe simulovat pracoviště reálné, tudíž by mělo obsahovat objekty, na které můžeme narazit i v praxi.

Mezi operace předzpracování v tomto systému se řadí například převedení do černobílé. Převedení do černobílé je provedeno kvůli metodě SURF a Homografii, tyto metody pracují s černobílými obrázky.

Pro potřebu realizovaného systému je potřeba popsat objekty vyskytující se na prvním snímku a následně je najít na snímku druhém. Následně je třeba rozeznat změny poloh těchto objektů. Pro reprezentaci výstupu bude využita míra shody pro indikaci, do jaké míry jsou situace shodné.

1. CÍL PRÁCE

V této kapitole jsou vytyčeny cíle, které jsou potřeba ke splnění zadání této práce.

1.1 Navrhnout ideální pracoviště

Testovací pracoviště by mělo simulovat pracoviště z praxe, které obsahuje nástroje, které potřeba inventarizovat nebo zboží, které je třeba hlídat. Bude několik variant pracovišť.

1.2 Pořídit databázi fotek

Pro vývoj softwaru je potřeba mít, na čem testovat. Proto, na výše navrženém pracovišti bude pořízena série fotek s různými předměty, v různých polohách.

1.3 Ošetřit různé úhly snímání

V praxi se může nezřídka stát, že dojde ke změně úhlu mezi snímání prvního a druhého snímku.

1.4 Detekce objevení a zmizení objektu

Problematika je rozfázována do několika kroků. Jako první bude realizována detekce přítomnosti nového objektu nebo naopak detekce zmizení již přítomného objektu.

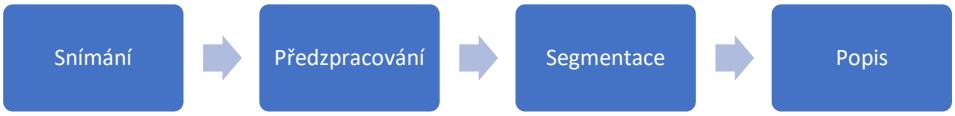
1.5 Detekce jednotlivých objektů

Finální program by měl být schopen rozpoznat změny týkající se jednotlivých objektů, namísto soustředění se na detekci prostých změn. To umožní poskytnout detailnější a více vypovídající výstup.

2. TEORIE

K řešené této práce je využito počítačové vidění. Počítačové vidění je disciplína, která se zabývá zpracováním digitálních obrazů. Digitální obraz je možné vnímat jako dvoudimenzionální funkci $f(x, y)$, kde její amplituda označuje intenzitu pixelu na souřadnicích x a y.

Při zpracování obrazu je uplatněn řetězec základních operací, který vede k výsledku. Mezi tyto operace patří snímání, předzpracování, segmentace a popis.



Obrázek 2.1 Řetězec základních operací počítačového vidění

2.1 Předzpracování

Předzpracování je důležitá operace počítačového vidění a u většiny aplikací je nutná. Zabývá se připravením obrazu na další zpracování, díky čemuž je následně dosaženo lepších výsledků. Mezi operace předzpracování se řadí například vyrovnání histogramu, filtrace šumu nebo i převedení do černobílé.[1]

2.2 Segmentace

Segmentace patří mezi základní operace zpracování obrazu. Slouží k rozdělení obrazu na základní regiony či objekty. Detailnost rozdělení záleží na povaze řešeného problému. Segmentace je ukončena, jakmile jsou objekty nebo požadované regiony detekovány. Mezi využití segmentace v praxi se řadí například teplem naváděná detekce cíle.

Pro metody segmentace jsou využívány různé principy. Většina metod pracuje však s jednou ze dvou vlastností intenzity pixelů a to buď se spojitostí, anebo s podobností. Spojitost je využívána při hledání hran, kdy se hledá prudký skok v intenzitě pixelů. Podobnost pixelů je využívána při rozdělování na definované regiony, například prahování. V praxi je rovněž využíváno kombinace různých segmentačních metod.[13]

2.3 Detekce klíčových bodů

V současnosti se využití počítačového vidění dramaticky zvyšuje a v mnoha aplikacích, které ho využívají, je potřeba detekovat klíčové body. Využívá se například při rozpoznávání obličejů.

Detekce klíčových bodů tvoří základ výsledného programu této práce. Využívá se jak pro homografii, tak pro detekci samotnou.

Mezi nejvyužívanější metody pro detekci klíčových bodů patří SIFT (Scale Invariant Feature Transform), SURF (Speeded Up Robust Features) nebo BRISK (Binary Robust Invariant Scalable Keypoints).[6]

2.3.1 SURF

V této práci byla vybrána metoda SURF, protože je rychlejší než SIFT a má nejširší pole použití ze zmíněných algoritmů.

Metoda SURF byla navržena H. Bayem v roce 2008 a vychází z metody SIFT. Jedná se o rychlý a robustní algoritmus pro stanovení podobnosti, který je navíc invariantní vůči rotaci a měřítku. Metoda se skládá ze dvou fází, detekce a popisu.

Detekce

K detekci se používá Hessova matice. To je čtvercová matice parciálních derivací funkce o více proměnných. Využívá se k určení inflexních bodů nebo kritických bodů, zde je využívána k detekci klíčových bodů. Pokud je uvažována spojitá funkce dvou proměnných f(x, y), Hessova matice bude mít tvar:

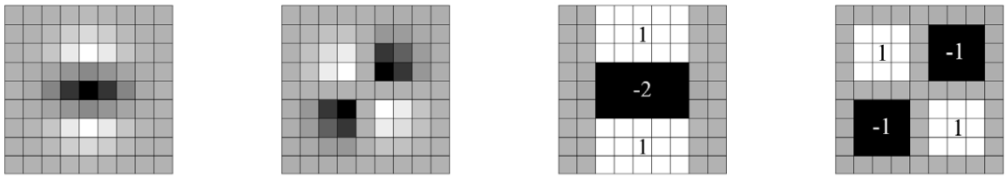
$$H\big(f(x,y)\big) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x^2} & \dfrac{\partial^2 f}{\partial x \partial y} \\ \dfrac{\partial^2 f}{\partial x \partial y} & \dfrac{\partial^2 f}{\partial y^2} \end{bmatrix} \qquad (2.1)$$

Tato matice je základem Hessenského detektoru. U Hessovy matice jsou nahrazeny funkce f(x, y) intenzitami pixelů I(x, y). Hessova matice H(X, σ) v každém bodě X(x, y) obrázku I a v měřítku σ může být definována jako:

$$\mathrm{H}(\mathrm{X},\sigma) = \begin{bmatrix} \mathrm{L_{xx}}(\mathrm{X},\sigma) & \mathrm{L_{xy}}(\mathrm{X},\sigma) \\ \mathrm{L_{xy}}(\mathrm{X},\sigma) & \mathrm{L_{yy}}(\mathrm{X},\sigma) \end{bmatrix} \qquad (2.2)$$

Kde L_{xx} je konvoluce druhé derivace Gaussovy funkce $\frac{\partial^2}{\partial x^2} g(\sigma)$ obrázku I v bodě X, podobně pro $L_{xy}(X, \sigma)$ a $L_{yy}(X, \sigma)$.

Maticové filtry 9x9 na obrázku 1.5 jsou aproximace pro druhou derivaci Gaussovy funkce s σ = 1,2 a reprezentací našeho nejmenšího měřítka.



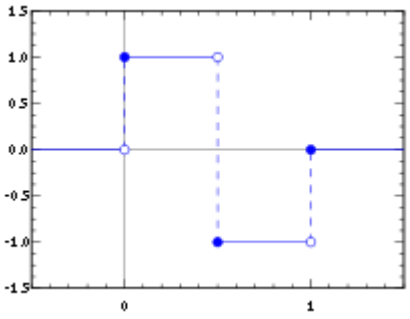
Obrázek 2.2 Druhá derivace Gaussovy funkce

Aproximace jsou značeny D_{xx}, D_{yy} a D_{xy}. Váhy aplikované na obdélníkové regiony jsou jednoduché kvůli efektivitě výpočtu.[7]

Popis

Deskriptory metody SURF jsou podobné deskriptorům metody SIFT. Při vytváření deskriptoru je nejprve potřeba určit orientaci, která je založená na informacích poskytnutých okolím klíčového bodu. Následně vytyčíme čtvercový region zarovnaný podle vybrané orientace a z něj získáme deskriptor.

Za účelem nezávislosti na rotaci, je určen směr klíčových bodů. Pro tento úkol jsou nejprve spočítány odezvy Haarovy vlnky, ve směrech x, y a v kruhovém okolí klíčového bodu o poloměru 6*s*, kde *s* je měřítko, se kterým byly detekovány klíčové body. Vzorkovací krok je také závislý na měřítku a jeho velikost je *s*. Ke spočítání odezev v jakémkoliv měřítku je potřeba 6 operací. Délka strany jedné vlny je 4*s*.



Obrázek 2.3 Haarova vlnka

Jakmile jsou odezvy a váhy spočítány, jsou reprezentovány pomocí vektorů podél os x a y. Po sečtení odezev všech dvojic vektorů je výsledná orientace určena podle nejdelšího, nově vzniklého vektoru.

Pro získání deskriptoru je prvním krokem konstrukce čtvercového regionu kolem klíčového bodu s orientací určenou podle postupu výše. Velikost regionu je 20s. Tento region je rovnoměrně rozdělen na menší 4x4 sub-regiony. Tento proces uchová důležité

prostorové informace. Pro každý sub-region je spočítáno několik jednoduchých rysů. Pro jednoduchost nazýváme d_x odezvu na Haarovu vlnku v horizontálním směru a d_y ve vertikálním. Pro zvýšení robustnosti proti geometrické deformaci a lokalizačním chybám jsou nejprve určeny váhy pomocí Gaussovy funkce (σ=3,3s) se středem v klíčových bodech.

Následně jsou odezvy d_x a d_y sečteny v každém sub-regionu a vytvoří se první vstupy vektoru rysů. Za účelem získání informací o změně polarity a intenzity jsou sečteny také absolutní hodnoty d_x a d_y. Nyní má každý sub-region čtyřrozměrný deskriptorový vektor v.

$$v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|) \qquad (2.3)$$

Výsledkem je vektor (o délce 64 prvků) pro všechny 4x4 sub-regiony. Invariance vůči kontrastu bude docíleno převedením deskriptorů na jednotkový vektor.

Realizace v programu

Knihovna OpenCV disponuje mnoha metodami zabývající se SURFem. Nejprve je potřeba vytvořit detektor. Následně pomocí něj nalézt klíčové body a z nich vytvořit deskriptory, obě tyto operace realizuje metoda *detectAndCompute*.

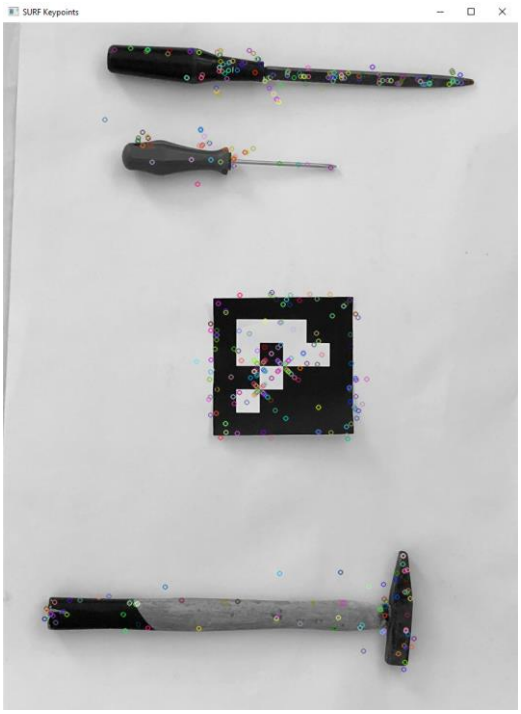
```
void SURFDetect()
{
      int minHessian = 400;
      Ptr<SURF> detector = SURF::create(minHessian);

      Detector->detectAndCompute(object, noArray(), kptsO ,desO);

}
```

Tato metoda najde klíčové body a spočítá z nich deskriptory.
Jako argumenty metody jsou:

- object … Obrázek ke zpracování.
- kptsO … Proměnná, do které se uloží klíčové body daného obrázku.
- desO … Proměnná, do které se uloží deskriptory daného obrázku.



Obrázek 2.4 Body detekované metodou SURF

Následně je potřeba spočtené deskriptory sesouhlasit. K tomu je potřeba vytvořit *DescriptorMatcher* a následně použijeme funkci *knnMatch*.

```
void SURFDetect()
{
      Ptr<DescriptorMatcher> matcher =
          DescriptorMatcher::create(DescriptorMatcher::FLANNBASED);
      vector<vector<DMatch>> knn_matches;
      matcher->knnMatch(a.descriptors, b.descriptors, knn_matches,2);

}
```

Ta naplní proměnnou *knn_matches*. Tato proměnná obsahuje jednak indexy sesouhlasených bodů a jednak také údaj o vzájemné poloze bodů.

2.4 Homografie

Homografie pracuje s údaji o předmětech na snímané scéně, jako je jejich tvar, účel a vzájemná vzdálenost. Zpracováním těchto dat je možné rekonstruovat okolí a umožnit tak například efektivní řízení pohybu mobilního robotu, nebo sledování předem definovaného pohyblivého předmětu. V této práci je využita k sesouhlasení významných bodů obou obrázků, aby je bylo možné dále srovnávat.

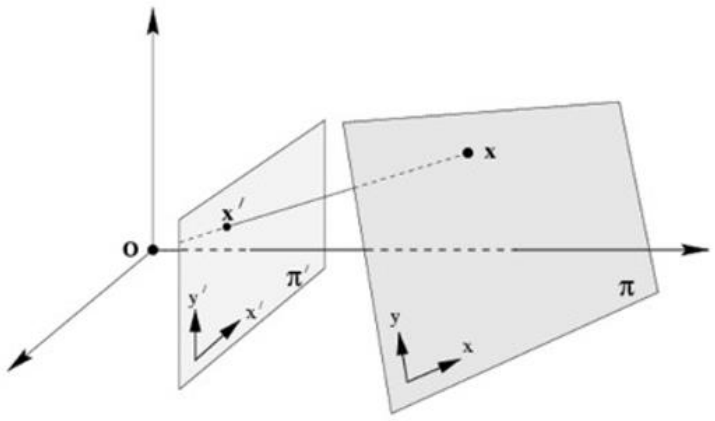
2.4.1 Princip

Homografie je transformace mezi dvěma rovinami. Je reprezentována maticí 3x3

$$s\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{2.4}$$

Kde x

Každý bod v obrázku má svůj průmět do dalšího obrázku v rovině s homogenními souřadnicemi, kde obsahuje stejné informace v transformované perspektivě.[9]



Obrázek 2.5 Bod promítnutý ve dvou různých rovinách

Homogenní souřadnice

Homogenní souřadnice jsou systémem souřadnic, používaných v projektivní geometrii. Tyto souřadnice mají tu výhodu, že je lze reprezentovat pomocí konečných souřadnic. Pokud jsou homogenní souřadnice vynásobeny nenulovým skalárem, pak

výsledné souřadnice představují stejný bod. Protože homogenní souřadnice jsou také dány body v nekonečnu, počet souřadnic potřebných k povolení tohoto rozšíření je o jeden více, než rozměr projektivního prostoru.[10]

Pin Hole Camera Model

Pin Hole Camera Model ukazuje vztah mezi 3D prostorem a 2D obrázkem. Tento vztah vyjadřuje následující rovnice

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{2.5}$$

Kde první matice převádí rovinu obrazu na pixely a druhá matice obsahuje informace o pozici kamery. Tyto dvě matice můžeme zkombinovat do jedné pomocí roznásobení. Tuto matici nazýváme matici kamery.

$$\mathrm{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}, \tag{2.6}$$

Derivací této matice získáme matici homografie.

Výpočet Homografie

Hodnoty matice homografie můžeme spočítat pomocí kalibračního procesu. Ten spočívá v tom, že použijeme odpovídající body z obou

$$\begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (1) \qquad \begin{bmatrix} \hat{x} \\ \hat{y} \\ 1 \end{bmatrix} = \frac{1}{z_a} H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ 1 \end{bmatrix} = \frac{1}{z_a} \begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} \quad (2) \qquad z_a \begin{bmatrix} \hat{x} \\ \hat{y} \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3)$$

Obrázek 3.2 Vztah mezi dvěma páry odpovídajících bodů

Záměrem je nalézt transformační matici, pro níž platí:

$$2\, x_i' = H x_i \tag{2.7}$$

x_i' a Hx_i se však číselně nerovnají, protože se liší v měřítku (daném w_i'). Přesto však můžeme zapsat:

$$(x_i')_\times H x_i = 0 \tag{2.8}$$

Pokud nahradíme $(x_i')_\times$ dostaneme soustavu:

$$\begin{pmatrix} 0^T & -w_i' x_i^T & y_i' x_i^T \\ w_i' x_i^T & 0^T & -x_i' x_i^T \\ -y_i' x_i^T & x_i' x_i^T & 0^T \end{pmatrix} h = 0 \tag{2.9}$$

Tyto rovnice mají podobu:

$$A_i h = 0 \tag{2.10}$$

Kde A_i je matice 3x9 a vektor h se rovná:

$$h = (h_{11}; h_{12}; h_{13}; h_{21}; h_{22}; h_{23}; h_{31}; h_{32}; h_{33})^T \tag{2.11}$$

A_i má hodnost, tudíž je každá korespondence vyjádřena dvojicí rovnic.**Chyba! Nenalezen zdroj odkazů.**

Složením rovnic pro čtyři body vzniká matice A jejíž hodnost je 8 a tvoří lineární homogenní soustavu rovnic pro 9 neznámých. Z toho plyne, že stačí, když matice obsahuje 8 lineárně nezávislých řádků. Řešením je potom nulový prostor této matice. Body musí být zvoleny tak, aby žádné tři neležely na stejné přímce.

$$Ah = \begin{pmatrix} x_1 & x_1 & 1 & 0 & 0 & 0 & -x_1 x_1' & -y_1 x_1' & -x_1' \\ 0 & 0 & 0 & x_1 & x_1 & 1 & -x_1 y_1' & -y_1 y_1' & -y_1' \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2 x_2' & -y_2 x_2' & -x_2' \\ 0 & 0 & 0 & x_2 & y_2 & 0 & -x_2 x_2' & -y_2 y_2' & -y_2' \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_n x_n' & -y_n x_n' & -x_n' \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_1 y_n' & -y_n y_n' & -y_n' \end{pmatrix} = 0 \tag{2.12}$$

V praxi se používá korespondencí více než 4, aby se zmenšil vliv chyby při určení korespondujících bodů. Kvůli chybám je však také nulový prostor matice A prázdný a vektor h se hledá ve smyslu nejmenších čtverců, tj. určí se jako:

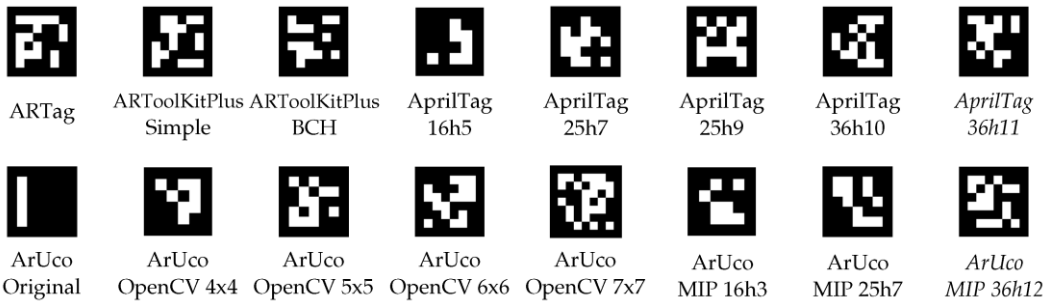
$$h = \underset{||h^*||=1}{\operatorname{argmin}} ||Ah^*||^2 \tag{2.43}$$

Kde ||X|| je Euklidovská norma $\sqrt{\sum X_i^2}$. Nalezení vektoru h se v takovém případě udělá pomocí **SVD** (Singular Value Decomposition).[8][10]

2.5 AR tag

AR tag (Augmented reality) je čtvercová binární značka. Tyto tagy jsou dnes hojně využívány v oblasti rozšířené reality, robotiky, aj. Používá se ke zjednodušení určování polohy detekovaných objektů na snímku.[4]

Existuje mnoho různých kolekcí AR tagů, jejichž vlastnosti se mírně liší. Tato práce používá ArUco OpenCV 6x6 dictionary.[5]



Obrázek 2.6 Různé druhy AR tagů

Výhoda ArUco knihovny je, rychlost a snadná detekovatelnost. Každá značka se skládá ze čtvercové binární mřížky. Ta obsahuje rámeček, který se v detekci neuplatňuje, pro detekci samotnou tedy zůstává pouze vnitřek, například v případě použití 6x6 mřížky 16 buněk. Z těchto 16 buněk jsou vždy pouze dvě v řadě použity pro binární kód, zbytek slouží pro detekci chyb. Máme tedy k dispozici 256 možných ID. Pro tuto aplikaci je AR tag výhodný, protože je dostačující a oproti například QR kódu, není přehnaně složitý.

Obrázek 2.7 QR kód

2.5.1 Detekce

Detekce AR tagu začíná nalezením jeho hranic (rámečku) pomocí adaptivního prahování. Hranice, které nesplní určité požadavky pro hledání značky, jsou vyfiltrovány. Zbývající objekty jsou porovnány, a jestli jsou dva čtyřúhelníky příliš blízko sebe, tak je jeden odstraněn. Toto je primárně proto, aby se eliminovaly čtyřúhelníky vevnitř značky a detekoval se vnější.

2.5.2 Popis

První krok popisu je Homografie, díky které je odstraněna perspektivní deformace. Následně je použit další prahovací algoritmus, k odfiltrování šumu. Vyfiltrovaný obrázek je zanalyzován, zda se jedná o značku. Nakonec je stanoven rámeček a následně vnitřní mřížka, která obsahuje binární kód.[14]

3. VLASTNÍ ŘEŠENÍ PROBLÉMU

V této kapitole je popsán vývoj programu. Nejprve jsou zde popsány použité vývojové nástroje a poté vývoj samotný, který probíhal v několika verzích.

3.1 Vývojové nástroje

V této kapitole jsou představeny vývojové nástroje, které jsou použity při vývoji

3.1.1 Microsoft Visual Studio 2022 Comunity

Jako vývojové prostředí je využita bezplatná verze softwaru Visual Studio. Jde o populární prostředí, které bylo vyvinuto společností Microsoft. Visual Studio podporuje až 36 programovacích jazyků, včetně C++, které je použito v této práci.

První verze tohoto programu byla vyvinuta v roce 1997 a sjednocovala jazyky, Visual Basic, Visual C++, FoxPro a Visual J++.

Toto vývojové prostředí jsem si vybral, protože v něm pracuji již od střední školy a dobře jej znám.[2]

3.1.2 OpenCV

Je open source softwarová knihovna zaměřená na počítačové vidění a strojové učení. Tato knihovna byla vyvinuta firmou Intel a obsahuje více, než 2500 algoritmů, které používá více než 47 000 lidí. Navíc je pro vývoj použit také dodatečný modul contrib. [3]

OpenCV contrib

OpenCV contrib je sbírka přídavných modulů k OpenCV. Poskytují rozšíření funkcí pro komplexnější úlohy. Součástí tohoto modulu jsou funkce realizující SURF, která se používá v této práci k detekci objektů, a funkce pracující s AR tagy, které se používají k rozlišení scén.[3]

3.2 Návrh scény

Program bude pracovat s několika scénami identifikovanými pomocí AR tagů. Tyto scény budou představovat snímky různých míst na pracovišti, díky AR tagům tyto místa budou nezaměnitelná. Na každé pracoviště bude patřit určitá sada objektů. Pracovištěm je vnímáno místo, kde se odkládají nástroje v dílně nebo v tovární hale, program však není omezen jen na tyto dvě využití. Lze jej využít prakticky všude, kde je potřeba mít kontrolu nad věcmi. Předpokládá se, že pracoviště je v jedné výškové hladině, bez výškových schodů. Program rovněž dosahuje nejlepších výsledků s monotónním pozadím, kvůli hranové filtraci, která tvoří vstup pro další zpracování.

Obrázek 3.1 Příklad scény

3.3 První verze

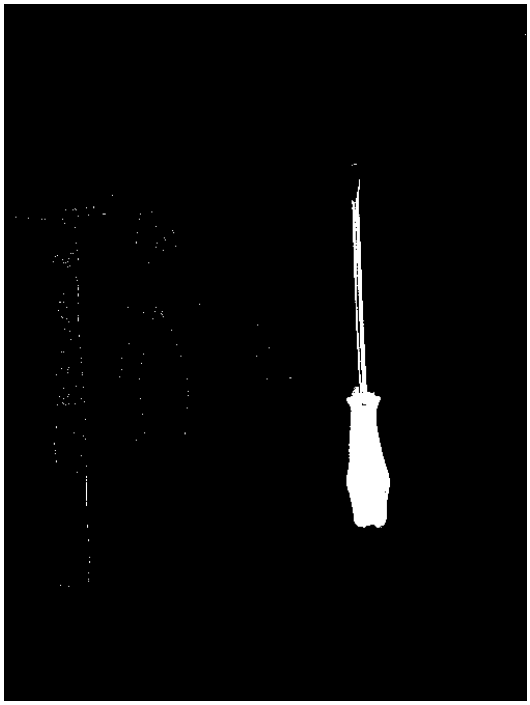
První verze programu pojala problematiku problému co nejvíce přímočaře. Funguje dobře pro detekci prostých změn, jako je zmizení, objevení nebo přesun objektu.

3.3.1 Funkčnost

Nejprve je třeba načíst obrázky scén, které jsou převedeny do černobílé. Následně je aplikována Homografie, která ošetří možnosti focení z různých úhlů.

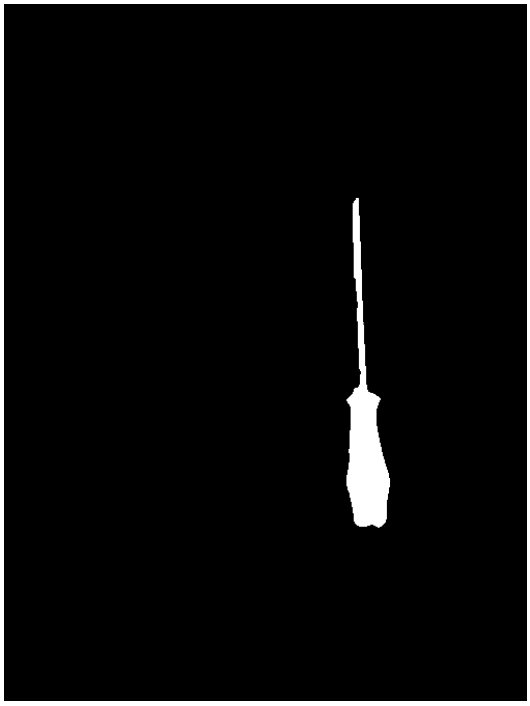
Obrázek 3.2 Scény na vstupu

Následně jsou obrázky zbinarizovány. Binarizace je provedena pomocí funkce *threshold(gray, binary, 100, 255, THRESH_BINARY)*, což je prahovací funkce, kde jsou nastaveny mezní hodnoty na 100 a 255. Tyto hodnoty jsem stanovil na základě testování a fungují nejlépe za předpokladu, že má scéna bílý podklad. Byla testována rovněž varianta s automatickým stanovením prahů na základě metody Otsu, avšak výsledky byl nevyhovující. Rozdíl binarizovaných scén je zobrazen na obrázku 3.3.



Obrázek 3.3 Rozdíl binárních obrazů

Z obrázku 3.3 je patrné, že rozdíl binárních obrazů obsahuje šum, proto je provedena filtrace.



Obrázek 3.4 Rozdíl binárních obrazů po filtraci

Takto vyfiltrovaný rozdíl je zobrazen na jedné, ze vstupních scén.

Obrázek 3.5 Výstup první verze

3.3.2 Zhodnocení

Problém tohoto řešení spočívá v tom, že se nesoustředí na objekty samotné, ale pouze na obecné změny. Na výstupu tedy nejsou poskytnuty žádné bližší informace o tom, co se stalo, pouze jsou zvýrazněny změny, které jsou dost dobře patrné i pouhým okem.

3.4 Druhá verze

Pro druhou verzi programu je potřeba mít nafocenou databázi předmětů, které mají být ve scénách detekovány. Tyto data jsou uloženy v konfiguračním souboru. Každá scéna, se kterou tento program pracuje, obsahuje AR tag, který scéně přiřadí identifikační číslo. Na každou identifikovanou scénu budou příslušet určité nástroje. Pokud scéna nebude obsahovat všechny nástroje nebo bude některý načtený nástroj přebývat, tak se zapíše varovná zpráva do logu s informací, o kterou scénu a o který předmět jde. Na rozdíl od první verze, je zde počítáno s vložením vždy jedné scény, tudíž v této verzi není aplikována Homografie.

3.4.1 Konfigurační soubor

Jména a cesty k těmto předmětům jsou uložené v souboru typu .json.
Tento soubor obsahuje jednak jména načtených objektů a cesty k nim, ale také cesty ke scénám a výčet předmětů, přiřazených ke scéně, ve které se mají nacházet.

```
{
    "brush": "c:\\Users\\Cechj\\Documents\\other\\pics\\brush.JPG",
    "hammer": "c:\\Users\\Cechj\\Documents\\other\\pics\\hammer.JPG",
    "knife": "c:\\Users\\Cechj\\Documents\\other\\pics\\knife.JPG",
    "metalBrush":
"c:\\Users\\Cechj\\Documents\\other\\pics\\metalBrush.JPG",
    "pliers": "c:\\Users\\Cechj\\Documents\\other\\pics\\pliers.JPG",
    "screwdriver":
"c:\\Users\\Cechj\\Documents\\other\\pics\\screwdriver.JPG",
    "wrench": "c:\\Users\\Cechj\\Documents\\other\\pics\\wrench.JPG",

    "scenes":
    [
        "c:\\Users\\Cechj\\Documents\\other\\pics\\angleScene.JPG",
        "c:\\Users\\Cechj\\Documents\\other\\pics\\angleScene2.JPG",
        "c:\\Users\\Cechj\\Documents\\other\\pics\\closeScene.JPG",
        "c:\\Users\\Cechj\\Documents\\other\\pics\\farscene.JPG",
        "c:\\Users\\Cechj\\Documents\\other\\pics\\objects1.JPG",
        "c:\\Users\\Cechj\\Documents\\other\\pics\\objects2.JPG"
    ],

    "scene1":
    [
        "brush",
        "metalBrush",
        "knife"
    ],

    "scene2":
    [
```

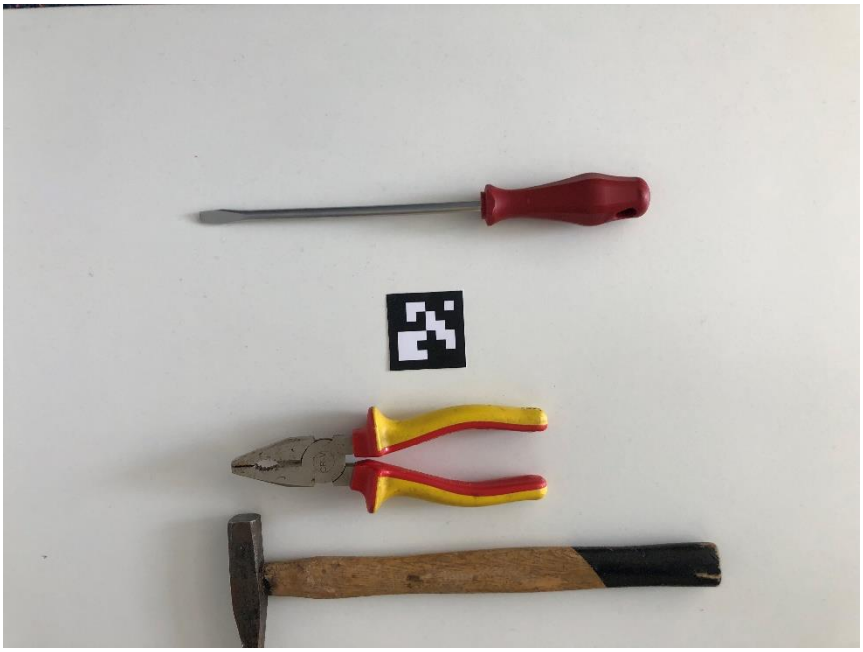
```
        "hammer",
        "pliers",
        "screwdriver",
        "wrench"
    ]
```

3.4.2 Databáze snímků

Databáze snímků obsahuje jednak snímky objektů a dále pak snímky scén. Snímky objektů jsou pořízeny v ideálních podmínkách a v kódu jsou proměnné, v nichž jsou data o nich uloženy, pojmenovány podle typu objektu (např. screwdriver, pen,…). Jména jsou stejné kvůli přehlednosti a také kvůli logování, popsané níže.

Obrázek 3.6 Příklad snímku objektu

Snímky scén vždy obsahují AR tag, kvůli identifikaci pracoviště. Ke každé scéně je přiřazeno několik objektů, které by se tam měli vyskytovat.



Obrázek 3.7 Příklad scény

3.4.3 Zpracování dat

Při načtení scény se detekují její klíčové body, pomocí metody Surf. Sesouhlasením klíčových bodů a jejich srovnáním se zjistí, zda se načtené objekty ve scéně vyskytují. Přípustná shodnost klíčových bodů na snímku byla v této verzi nastavena na 75 %. Tato hodnota byla nastavena na základě testování.

3.4.4 Výstup programu

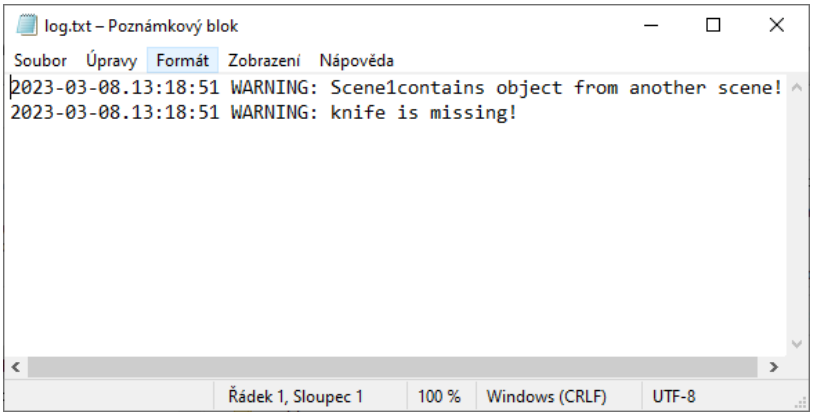
Výstup programu bude realizován pomocí zápisu do textového souboru, takzvaného logu. Součástí zápisu bude vždy čas zápisu, typ a zpráva (*2023-01-09.11:09:36 WARNING: {message}*). Zpráva může být dvou typů, buď informuje o objektu, který na scéně chybí a měl by tam být, anebo o objektu, který je ve scéně navíc a neměl by tam být.

3.4.5 Příklad použití

V tomto příkladu použijeme následující scénu.

Obrázek 3.8 Scéna na vstupu

Ar tag má id 0, jedná se tedy o scénu, která by měla obsahovat štětec, železný kartáč a nůž. Scéna na obrázku 3.8 obsahuje pouze štětec a kartáč, nůž zde chybí. Tato scéna rovněž obsahuje jeden objekt navíc, a sice kladivo. Program projde všechny načtené objekty a pokusí se je vyhledat ve scéně, výsledek zapíše do logu.
Po zpracování této scény log vypadá následovně.



Obrázek 3.9 log po zpracování scény na obrázku 3.8

3.4.6 Zhodnocení

Tato verze je, oproti předchozí, soustředěná na objekty a ne jen na prosté změny. Uživatel má rovněž více možností, jako například konfigurace jednotlivých scén, výstup je rovněž více vypovídající. Úskalím této verze je nutná konfigurace scén a nafocení jednotlivých objektů.

3.5 Třetí verze

Třetí verze pracuje bez načtené databáze. Objekty jsou detekovány na první scéně, které jsou následně srovnávány s objekty na scéně druhé. K detekci objektů bylo využito několik různých metod.

3.5.1 Pozice SURF deskriptorů

Objekty jsou detekovány na základě pozic deskriptorů, nalezených metodou SURF. Deskriptory jsou nejprve seřazeny podle sloupců pixelů v obrázku, počínaje v levém horním rohu. Následně jsou porovnávány vzdálenosti s předem určenou konstantou, která představuje maximální míru rozdílu souřadnic v dané ose. Pokud vzdálenost, více jak pěti bodů po sobě, splňuje podmínky, jsou tyto body považovány za objekt. Poté dojde k seřazení deskriptorů i v ose řad a k opakování popsaného procesu.

Tato metoda se ukázala jako značně nespolehlivá a nekonzistentní. U některých scén byly výsledky dobré.

Obrázek 3.10 Dobrý případ

Na obrázku 3.10 lze vidět, že nalezené klíčové body náleží pouze k jednomu objektu, tudíž program, v tomto případě, pracuje správně.

U jiných scén však program objekty nedetekoval přesně.

Obrázek 3.11 Špatný případ

V tomto případě body, které program našel, náleží několika různým objektům. Proto byla tato metoda vyhodnocena jako nevhodná.

3.5.2 Watershed

Byla vyzkoušena také detekce objektů pomocí segmentační metody Watershed. Segmentační metoda Watershed je metoda segmentace obrazu, která se používá k oddělení různých objektů v obrazu. Tato metoda je založena na předpokladu, že všechny pixely v obrazu lze chápat jako vrcholy v topografické mapě, kde údolí vznikají v místech s nízkými intenzitami obrazu a hory naopak v místech s vysokými intenzitami obrazu.

Metoda Watershed začíná tím, že se vytvoří tzv. transformace vzdálenosti, která každému pixelu přiřadí hodnotu, která udává vzdálenost tohoto pixelu od nejbližšího bodu s nulovou intenzitou (tzv. nultý bod). Poté se na základě této transformace vzdálenosti vytvoří markerový obraz, kde jsou nulté body zvýrazněny jako markery. Tyto markery slouží jako začátek segmentace.

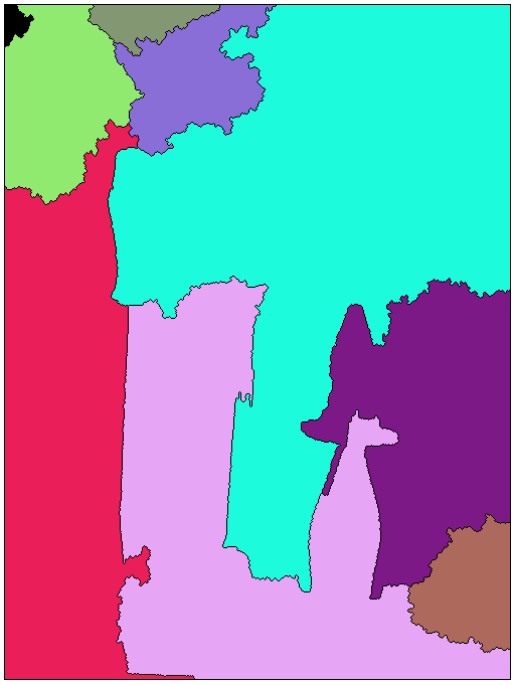
Následně se provede tzv. transformace vody, která spočívá v simulaci napouštění údolí v topografické mapě vody a následném rozlití vody do okolních oblastí. Tímto způsobem se vytvoří oblasti, které jsou odděleny od sebe vodními přehradami, a které tvoří segmenty objektů v obrazu.

Segmentace pomocí metody Watershed je velmi účinná pro segmentaci obrazů s vysokou kontrastní rozdílností mezi objekty a pozadím. Avšak pokud je kontrastní rozdílnost nízká, může být segmentace s pomocí této metody neúspěšná. Dále je třeba poznamenat, že výsledná segmentace může obsahovat segmenty, které neodpovídají žádnému reálnému objektu v obrazu, což může být problémem při následném zpracování dat.[12]

Tato metoda byla použita k oddělení bílého pozadí obrázku .

Obrázek 3.12 Obrázek na vstupu

Výstup je však následující:



Obrázek 3.13 Výstup programu s obrázkem 3.12 na vstupu

Tento výstup je však značně nevypovídající. Proto byla tato metoda vyhodnocena jako nevhodná.

3.5.3 Lazy snapping

Lazy Snapping je segmentační metoda, která umožňuje jednoduše segmentovat objekt z obrázku, za pomoci malého uživatelského vstupu. Uživatel udělá několik čar na obrázku, které indikují popředí a pozadí obrázku. Na základě těchto čar algoritmus automaticky spočítá mapu pravděpodobnosti, která přiřadí každému pixelu pravděpodobnost, že náleží do pozadí či popředí.

Uživatel může později upřesnit výběr dalšími čárami nebo upravením samotné mapy pravděpodobnosti. Nakonec je uživatel schopen extrahovat objekt z obrázku.

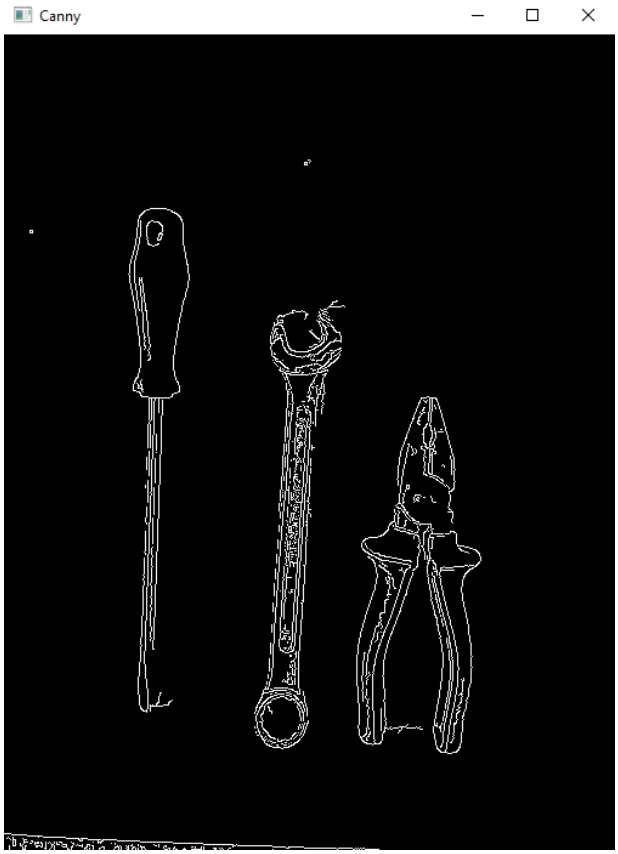
Lazy Snapping je poloautomatická segmentační technika, která je hojně využívána v oblasti počítačového vidění. Je užitečná hlavně pro segmentaci složitějších tvarů nebo ve členitém pozadí, kde by plně automatické metody selhaly.[11]

3.5.4 Cannyho hranová detekce

Tato metoda detekuje náhlé změny v obraze, a protože ve scéně je uvažováno monotónní pozadí, ideálně bílé, je tato metoda vhodná.

Metoda Cannyho hranové detekce byla vynalezena v roce 1986 a je široce používána v robotice či medicíně. Metoda pracuje v několika krocích.

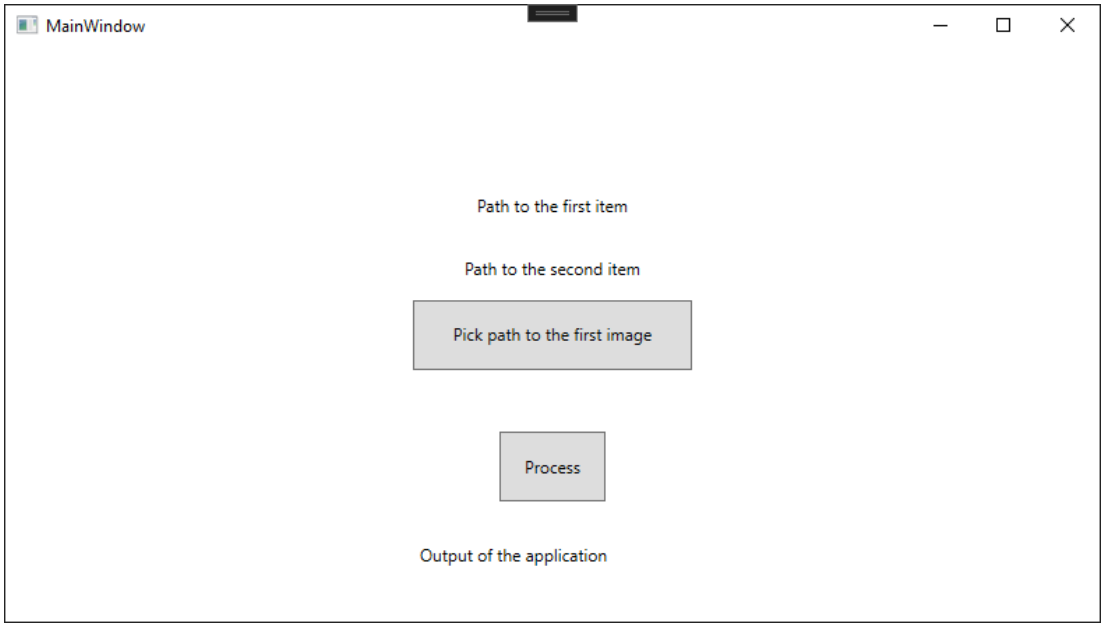
1. Vyhlazení obrázku, typicky Gaussovým filtrem k odstranění šumu a zvýraznění hran
2. Výpočet gradientu. K výpočtu gradientu je využit Sobelův operátor.
3. Zúžení hran. Tento krok se týká eliminace pixelů v okolí hran, které nejsou součástí hrany samotné
4. Dvojité prahování. Eliminace hodnot menší intenzity. Tato operace je provedena dvakrát, s různými prahy, pro zjištění výrazných a méně výrazných hran.
5. Sjednocení hran je posledním krokem algoritmu a spočívá ve spojení výrazných hran s jemnějšími, aby vznikl celek.



Obrázek 3.14 Výstup Cannyho hranové detekce

3.5.5 Uživatelské rozhraní

V této verzi bylo rovněž přidáno uživatelské rozhraní. Skrz toto rozhraní probíhá výběr obrázků, které budou zpracovány.



Obrázek 3.15 Uživatelské rozhraní

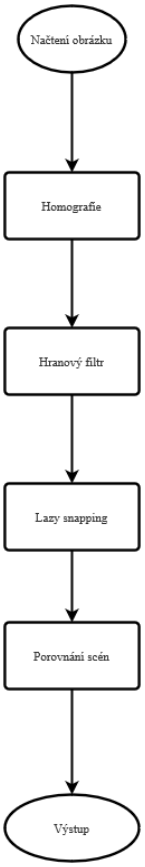
Jakmile uživatel stiskne *Pick path to the first image*, vyskočí promt, přes který je uživatel schopen vybrat cestu, nejprve k prvnímu a následně i ke druhému snímku. Jakmile jsou cesty vybrány, uživatel stiskne proces a snímky jsou zpracovány.

Po zpracování je výsledek vypsán

Rozhraní bylo programováno v jazyku c# wpf, z důvodu lepší podpory a jednoduššího vytváření desktopových aplikací, než na straně c++.

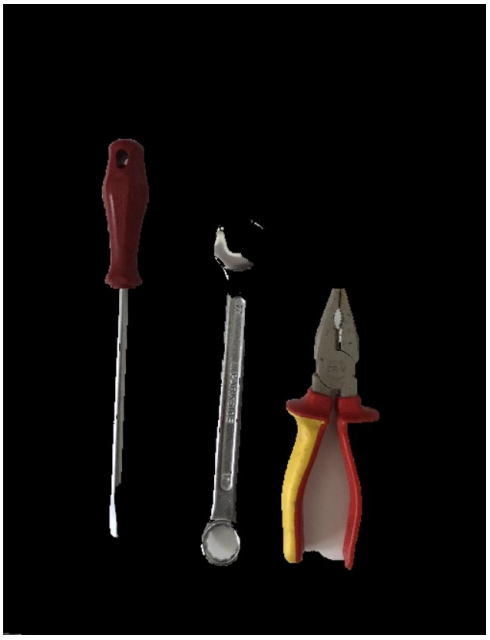
3.5.6 Finální algoritmus

Ve finálním řešení je Metoda Lazy Snapping zkombinována s Cannyho hranovou detekcí. A protože na rozdíl od druhé verze jsou opět vkládány dvě scény, je opět aplikována homografie na ošetření různých úhlů snímání.



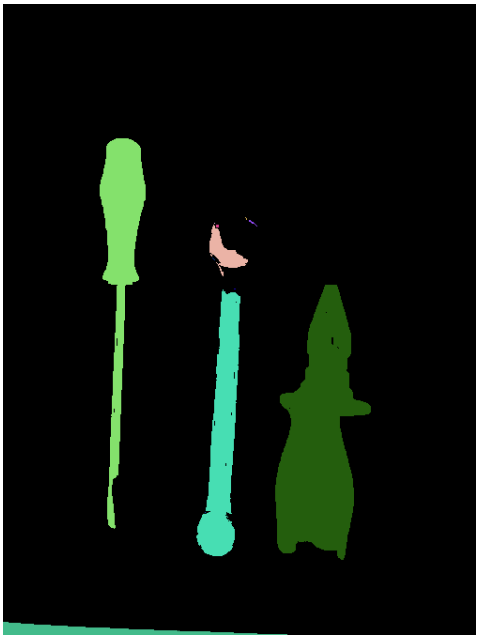
Obrázek 3.16 Vývojový diagram algoritmu

Cannyho hranová detekce slouží jako náhrada k uživatelskému vstupu. Při vstupu na obrázku 3.10 je tedy výstup zobrazen na obrázku 3.11.



Obrázek 3.17 Výstup funkce Lazy Snapping

Dále je objekty potřeba od sebe oddělit. K tomu byla použita knihovní funkce openCV *connectedComponentsWithStats*, která separuje jednolité objekty.



Obrázek 3.18 Separované objekty

Separované objekty jsou následně uloženy zvlášť do proměnných,

Obrázek 3.19 Nalezený objekt

Stejný postup je uplatněn na druhé scéně. Objekty jsou pak vzájemně porovnány a chybějící objekty jsou zaznamenány do logu, jako v předchozí verzi. Do logu je rovněž zaznamenáno, pokud není detekováno stejné množství objektů.

Rozdíl, oproti přechozí verzi je ten, že nejsou použita jména objektů pouze index objektu. Rovněž tato verze nerozeznává id scény, protože nejsou předem nakonfigurovány, tudíž není známo, co kam patří.

4. ZÁVĚR

Na začátku práce je vytyčeno několik cílů, v závěru je diskutována jejich realizace.

4.1 Navržení pracoviště

Pracoviště bylo navrženo tak, aby se podobalo reálným pracovištím v továrních halách nebo v dílnách. V této práci scénu představuje bílá deska, na kterou jsou pokládány různorodé nástroje. V praxi může být použita i jiná monotónní deska, ale je potřeba zachovat dostatečný kontrast a vyvarovat se vzorům, kvůli kvalitě segmentace.

4.2 Pořídit databázi fotek

Databáze fotek představuje kolekci snímků dříve navržené scény s různými objekty. Snímky jsou pořizovány hlavně staticky, kvůli povaze plánované scény, jsou zde ale i snímky focené z úhlu. Databáze je k dispozici na přiloženém CD.

4.3 Detekce objevení a zmizení objektu

Tento cíl byl splněn již v první verzi programu, kdy byly detekovány prosté změny. Tyto změny byly ve výstupu vyznačeny zvýrazněním obrysů oblasti, kde došlo ke změnám. Další verze se soustředily více na detekci objektů, proto jsou rozebrány v dalším bodě.

4.4 Detekce jednotlivých objektů

Na splnění tohoto bodu cílily verze dva a tři. Verze dva je více náročná na uživatelský vstup. Je nutné nakonfigurovat soubor typu .json, aby fungovala správně a také je potřeba nasnímat objekty, které se na scéně budou vyskytovat. Rovněž je nutné nakonfigurovat objekty na jednotlivých číslovaných scénách. Dostaneme avšak detailní výstup, který nám jednak řekne, zda nějaký předmět chybí v dané scéně, ale i jestli jsou na scéně všechny objekty anebo jestli se zde vyskytuje nějaký cizí. Výstup tvoří textový soubor, kde jsou popsány detekované události.

Verze třetí oproti tomu vyžaduje minimální uživatelský vstup a je vybavena uživatelským rozhraním. Je tedy ze všech verzí uživatelsky nejpřívětivější. Je schopna rozpoznat jednotlivé objekty a chybějící uloží jako obrázky, k bližšímu zkoumání. Výstup zapisuje jednak do textového souboru, jako verze dva, ale rovněž do uživatelského rozhraní. Nevýhoda oproti verzi dva spočívá v neschopnosti operaci s číslovanými scénami.

4.5 Možné rozšíření

Možné pokračování práce by mohlo spočívat v zapojení umělé inteligence při rozpoznávání objektů. Aplikace umělé inteligence by umožnila větší robustnost programu a rozpoznávání různých typů nástrojů, bez omezení na předem použité. Pro použití umělé inteligence je však třeba ji nejprve naučit, což vyžaduje velké množství dat.

Další možné pokračování by se mohlo týkat rozšíření působnosti algoritmu i na video, popřípadě na noční scény.

LITERATURA

[1] GONZALES, Rafael C. a Richard E. WOODS. *Digital Image Processing* [online]. 3rd edition. Pearson Education Internantional, 2010 [cit. 2023-05-05]. Dostupné z: https://sde.uoc.ac.in/sites/default/files/sde_videos/Digital%20Image%20Processing%203rd%20ed.%20-%20R.%20Gonzalez,%20R.%20Woods-ilovepdf-compressed.pdf

[2] Visual Studio. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2022 [cit. 2022-10-24]. Dostupné z: https://en.wikipedia.org/wiki/Visual_Studio

[3] *OpenCV* [online]. Wakefield, Massachusetts, USA: The Apache Software Foundation, 2000 [cit. 2022-10-24]. Dostupné z: https://opencv.org/

[4] *ARTag Revision 1. A Fiducial Marker System Using Digital Techniques* * [online]. Canada: National Research Council Canada Institute for Information Technology Conseil national de recherches Canada Institut de technologie de l'information, 2004 [cit. 2023-05-17]. Dostupné z: https://www.cs.cmu.edu/afs/cs/project/skinnerbots/Wiki/AprilTags/NRC-47419.pdf

[5] AR Tags and their Applications in Computer Vision Tasks. *NUS Information Technology* [online]. Singapore: National University of Singapore [cit. 2023-01-27]. Dostupné z: https://nusit.nus.edu.sg/technus/ar-tags-and-their-applications-in-computer-vision-tasks/

[6] Keypoint Descriptors in SIFT and SURF for Face Feature Extractions. *Researchgate* [online]. [cit. 2022-12-06]. Dostupné z: https://www.researchgate.net/publication/323362742_Keypoint_Descriptors_in_SIFT_and_SURF_for_Face_Feature_Extractions

[7] *SURF: Speeded Up Robust Features* [online]. In: . s. 4 - 8 [cit. 2022-12-06]. Dostupné z: https://people.ee.ethz.ch/~surf/eccv06.pdf

[8] Homografie a epipolární geometrie. *Trilobit* [online]. Zlín: Fakulta aplikované informatiky UTB ve Zlíně, 2010 [cit. 2022-12-06]. Dostupné z: http://trilobit.fai.utb.cz/homografie-a-epipolarni-geometrie#_ftn1

[9] Homography Transform — Image Processing. *Medium* [online]. [cit. 2022-12-06]. Dostupné z: https://mattmaulion.medium.com/homography-transform-image-processing-eddbcb8e4ff7

[10] *Homogenní souřadnice - Homogeneous coordinates* [online]. [cit. 2022-12-06]. Dostupné z: https://wikijii.com/wiki/Homogeneous_coordinates

[11] LI, Yin, Jian SUN, Chi-Keung TANG a Heung-Yeung SHUM. *Lazy snapping* [online]. [cit. 2023-05-17]. Dostupné z: https://home.cse.ust.hk/~cktang/sample_pub/lazy_snapping.pdf

[12] Use of watershed in contour detection. People [online]. [cit. 2023-03-30]. Dostupné z: https://people.cmm.minesparis.psl.eu/users/beucher/publi/watershed.pdf
[13] Image Segmentation. *Mathworks* [online]. Natick, Massachusetts, USA: The MathWorks [cit. 2023-04-23]. Dostupné z: https://www.mathworks.com/discovery/image-segmentation.html
[14] KULICH, Tim. *Indoor navigation using vision-based localization and augmented reality* [online]. Uppsala: Teknisk- naturvetenskaplig fakultet UTH-enheten, 2019 [cit. 2023-05-09]. Dostupné z: https://uu.diva-portal.org/smash/get/diva2:1345994/FULLTEXT01.pdf

OBRÁZKY

[1] *SURF. People [online]. [cit. 2020-12-24]. Dostupné z: https://people.ee.ethz.ch/~surf/eccv06.pdf*
[2] *Haarova vlnka. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2020-12-23]. Dostupné z: https://en.wikipedia.org/wiki/Haar_wavelet*
[3] *Homografie a Epipolární Geometrie. Trilobit [online]. Zlín: Univerzita Tomáše Bati, 2010 [cit. 2020-12-26]. Dostupné z: http://trilobit.fai.utb.cz/homografie-a-epipolarni-geometrie#_ftn1*
[4] *Různé druhy AR tagů* [online]. In: . [cit. 2023-05-16]. Dostupné z: https://www.mdpi.com/1424-8220/22/21/8548
[5] KULICH, Tim. *Indoor navigation using vision-based localization and augmented reality* [online]. Uppsala: Teknisk- naturvetenskaplig fakultet UTH-enheten, 2019 [cit. 2023-05-09]. Dostupné z: https://uu.diva-portal.org/smash/get/diva2:1345994/FULLTEXT01.pdf

SEZNAM SYMBOLŮ A ZKRATEK

Zkratky:

FEKT	Fakulta elektrotechniky a komunikačních technologií
VUT	Vysoké učení technické v Brně
SURF	Speeded-Up Robust Features
AR	Augmented Reality
JSON	JavaScript Object Notation

Symboly:

H	Matice Homografie	(-)
A	Hodnost matice	(-)