# ABSTRAKT

# ABSTRACT

# KLÍČOVÁ SLOVA

Deskriptory, redukce šumu, zlepšování obrazu, detekce oblastí zájmu, přiřazování klíčových oblastí, detekce tvarů

# KEY WORDS

Descriptors, noise reduction, image enhancement, detection of reagions of interrest, key regions matching, shape detection

# BIBLIOGRAFICKÁ CITACE

LABUDOVÁ, K. Zpracování obrazu pro ovládání robotické ruky. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2017. **x** s. Vedoucí bakalářské práce Ing. Vratislav Harabiš Ph.D.

# PROHLÁŠENÍ

Prohlašuji, že svoji seminární práci na téma Zpracování obrazu pro ovládání robotické ruky jsem vypracovala samostatně a pod vedením vedoucího seminární práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci této práce. Jako autorka uvedené seminární práce dále prohlašuji, že v souvislosti s vytvořením této seminární práce jsem neporušila autorská práva třetích osob, zejména jsem nezasáhla nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědoma následků porušení ustanovení S 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne 3.ledna 2017 ………………………..

podpis autora

# PODĚKOVÁNÍ

Děkuji vedoucímu bakalářské práce Ing. Vratislavu Harabišovi Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé seminární práce.

V Brně dne 3. ledna 2017 ………………………….

podpis autora

# OBSAH

[ÚVOD 8](#_Toc470026116)

[1 PŘEDZPRACOVÁNÍ OBRAZU 9](#_Toc470026117)

[1.1 Normalizace obrazu 9](#_Toc470026118)

[1.1.1 Detekce rohů displeje 9](#_Toc470026119)

[1.1.2 Normalizace jasu 10](#_Toc470026120)

[1.2 Redukce šumu 11](#_Toc470026121)

[2 DETEKCE OBRAZOVEK 13](#_Toc470026122)

[2.1 Detekce zajímavých bodů v obraze 15](#_Toc470026123)

[2.1.1 Harissův detektor rohů 15](#_Toc470026124)

[2.1.2 FAST – Feature from Accelerated Segment Test 16](#_Toc470026125)

[3 AKTUÁLNÍ STAV 18](#_Toc470026126)

[4 ZÁVĚR 19](#_Toc470026127)

[5 SEZNAM LITERATURY 20](#_Toc470026128)

[6 SEZNAM ZKRATEK A PŘÍLOH 22](#_Toc470026129)

[6.1 Seznam zkratek 22](#_Toc470026130)

# SEZNAM OBRÁZKŮ

[Obr. 1 – Mezikroky v algoritmu pro nalezení rohů displeje 9](#_Toc470026066)

[Obr. 2 - Snímek obrazovky před (vlevo) a po(vpravo) transformaci kontrastu 10](#_Toc470026067)

[Obr. 3 – Příklady problematických obrazovek 11](#_Toc470026068)

[Obr. 4 – Schéma filtrace ve výkonovém spektru 12](#_Toc470026069)

[Obr. 5 – Výsledek filtrace šumu typu moaré 12](#_Toc470026070)

[Obr. 6 – Výsledek filtrace šumu typu moaré na monotónním obraze 12](#_Toc470026071)

[Obr. 7 – Obrazová pyramida 14](#_Toc470026072)

[Obr. 8 – Reprezentace vlastností objektů v obraze na základě vlastních hodnot matice M 16](#_Toc470026073)

[Obr. 9 – Test pro detekci rohů v segment 12x12 pixelů [10] 17](#_Toc470026074)

# ÚVOD

Tato práce se zabývá zpracováním obrazu pro účely navádění robotické ruky po dotykovém displeji embedded zařízení tiskáren. Tento systém s robotickou rukou pomáhá ve firmě Y Soft Corporation testovat software SafeQ určený pro tiskový management. Celé oddělení výzkumu a vývoje funguje na bázi kontinuálního vývoje, a proto je několik vybuildovaných verzí SafeQ týdně. Ideálně by se mělo provést regresní testování pro každou verzi a každého výrobce tiskáren, kterých máme v dnešní době 11, pro podchycení chyb vzniklých v průběhu vývoje. Toto je velmi náročné až nemožné splnit, kvůli časové náročnosti a počtu lidí, kterých by bylo potřeba. Navíc buildování se děje večer a nejlepší by bylo zjistit případné problémy v noci, aby je mohli vývojáři začít opravovat už ráno. Robot je dobrým řešením, protože může pracovat přes noc, neunaví se a může provádět repetitivní úkony, aniž by ztrácel pozornost jako lidští testeři. Další pozitivum je škálovatelnost a finanční úspora.

Tento robot už je několik let ve fázi vývoje. Robotická ruka je naváděná kamerou, která snímá displej embedded zařízení. Část pro zpracování obrazu je naimplementovaná, ale neobsahuje předzpracování obrazu a není dostatečně rychlá ani spolehlivá. Náplň této práce je vybrat vhodné metody pro předzpracování obrazu a implementovat je. Dále nastudovat algoritmy pro rozpoznávání a klasifikaci obrazů, vybrat ty nejslibnější na základě odborné literatury a poté je implementovat a porovnat. Nejlepší metoda se použije v praxi. Na závěr se zhodnotí rychlost a spolehlivost řešení.

# PŘEDZPRACOVÁNÍ OBRAZU

## Normalizace obrazu

### Detekce rohů displeje

První krok předzpracování je výřez displeje z obrazu. Je tak dáno z několika důvodů, hlavním je snížení plochy pro další zpracování a tím snížení náročnosti na paměť a CPU. Další je normalizace rozměrů pro určení vzdálenosti na dipleji a určení polohy ovládacích prvků z databáze bez nutnosti automatické detekce. Aktuálně je naimplementovaná afinní transformace [1], ale rohy displeje se volí ručně přes GUI. Toto není ideální z hlediska nutnosti zásahu obsluhy robota. Další nevýhodou je, že se transformace provádí jen jednou, na začátku, a pokud se kamera během testování posune, je zbytek testování zbytečné. Proto byla naimplementována automatická detekce rohů displeje v obraze a podle pozice rohu se bude program kalibrovat v pravidelných intervalech.

Jednotlivé mezikroky zpracování je možné vidět na sérii obrázků Obr. 1. Nejprve se obraz převede na šedotónový, pak na binární s vhodným prahem. Binární obraz se nasegmentuje [1] a dále se provedou morfologické operace otevření a uzavření [1] kvůli redukci malých segmentů způsobených nerovnoměrným osvětlením a kvůli zajištění celistvosti větších segmentů. Pomocí kritérií pro délku kontury kolem segmentu a obsahu segmentu se vybere oblast, kde je s největší pravděpodobností displej. Nakonec se vyberou čtyři body v segmentu, jejichž Euklidovská vzdálenost je nejmenší k rohům obrazu.

|  |  |
| --- | --- |
|  |  |
|  |  |
| Obr. – Mezikroky v algoritmu pro nalezení rohů displeje – vstupní obraz (vlevo nahoře), binarizovaný obraz (vpravo nahoře), segmentovaný obraz (vlevo dole), výsledek (vpravo dole) | |

### Normalizace jasu

Tato část je důležitá, protože většina algoritmů je závislá na hodnotách jasu a pro lepší porovnání obrazovek je potřeba, aby měly podobný kontrast a tón barev. Ideální by bylo kalibrovat kameru pomocí referenční obrazovky, která by obsahovala saturovanou červenou, zelenou a modrou a alespoň 5 odstínů šedé. Taková obrazovka bohužel v aplikaci není k dispozici, proto se alespoň využívá faktu, že je v každé obrazovce bílá a černá. Jas se normalizuje po kanálech pomocí následujícího vzorce *(1)*. Kde je původní obraz, je pak transformovaný.

|  |  |  |
| --- | --- | --- |
|  |  | () |

Tím, že se kontrast transformuje po kanálech, dosahuje se i částečného vyvážení bílé a redukuje se vliv barvy osvětlení.

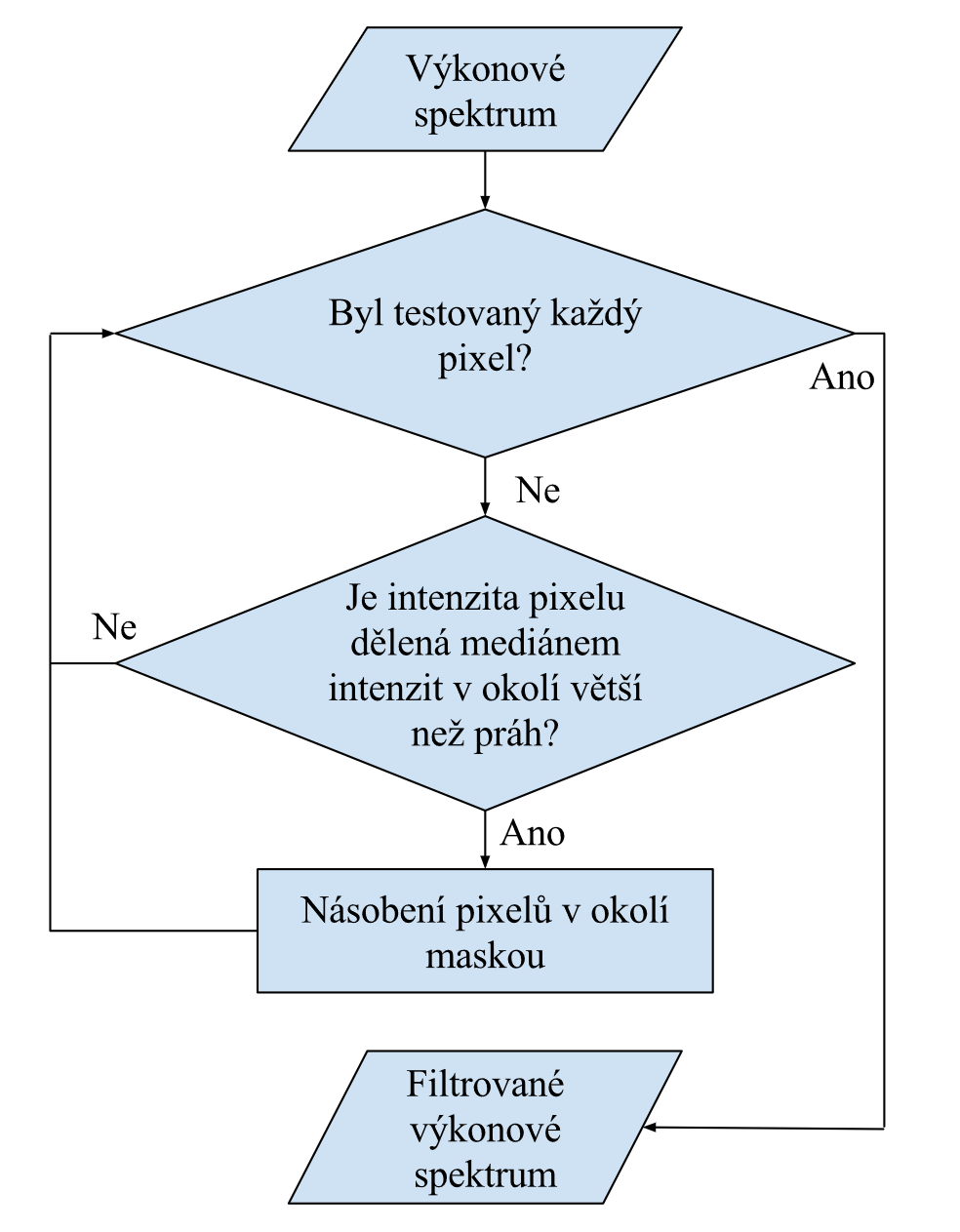
|  |  |
| --- | --- |
|  |  |
| Obr. - Snímek obrazovky před (vlevo) a po(vpravo) transformaci kontrastu | |
| Překážkou normalizace jasu jsou monotónní obrazovky, které značí načítání. Po tyto nemá normalizace smysl, proto se dopředu rozpoznají a žádná další analýza ani zpracování na nich neprobíhá. Kvůli občasnému výraznému moaré šumu a velkému množství různých podob monotónní obrazovek se není možné spolehnout jen na intenzitu pixelů v obraze nebo jejich barevnost. Pro rozpoznání monotónní obrazovky se tedy používá absolutní zastoupení hran v obraze a histogram intenzit pixelů po barevných kanálech. Hrany se získávají Cannyho hranovým detektorem[2].  U určení monotónnosti obrazovky bylo dosáhnuto stoprocentní úspěšnosti.   |  |  |  | | --- | --- | --- | |  |  | | |  | |  |   Obr. – Příklady problematických obrazovek: a) jinak barevný okraj b) výrazný šum c) nízký jas (obrazovka je reálně bílá) d) nemonotónní obrazovka s výrazným zastoupením bílé, která musí pokračovat k analýze | |
|  | |

## Redukce šumu

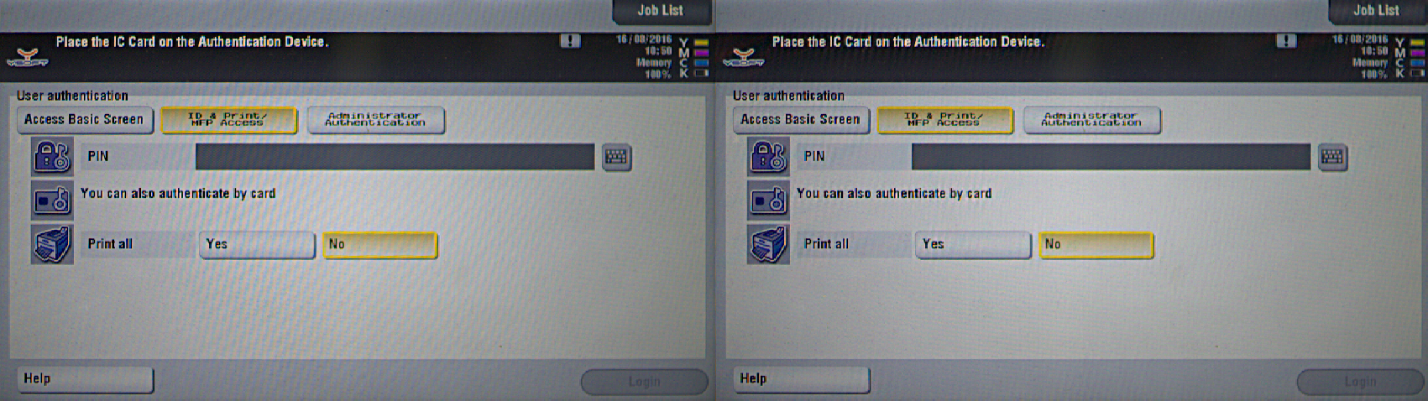
Při snímkování displeje je největším problémem šum typu moaré. Moaré vzniká v důsledku pravidelného uspořádání pixelů na displeji i na čipu kamery do matice. Tyto dvě mřížky spolu interferují, a tak vznikají pruhy. Vzhledem k výpočetní náročnosti odstranění tohoto typu šumu by bylo ideální použít hardwarový antialiasingový filtr, který ale nemáme momentálně k dispozici. Pro odstranění šumu se ukázala jako nejefektivnější filtrace ve spektru pásmovou Gaussovou zádrží. Filtrace byla provedena pro každý barevný kanál zvlášť. Pro vygenerování masky filtru byly použity následující vztahy[3]:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |
|  |  | *(3)* |
|  |  | *(4)* |

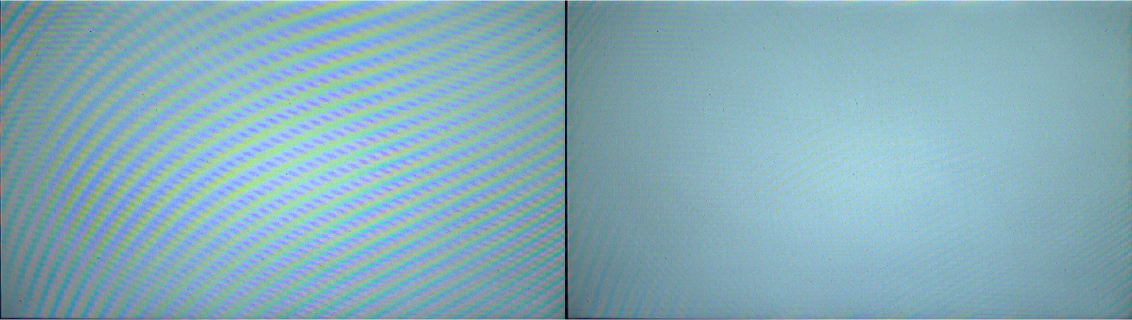
Po testování a optimalizaci vyšly nejlépe koeficienty C=1, a=0,01, práh = 4, rozměr masky = 11x11. Kvalita filtrace byla hodnocena čistě subjektivně, vizuálním porovnáním filtrovaného a původního obrazu.



Obr. – Schéma filtrace ve výkonovém spektru



Obr. – Výsledek filtrace šumu typu moaré



Obr. – Výsledek filtrace šumu typu moaré na monotónním obraze

# DETEKCE OBRAZOVEK

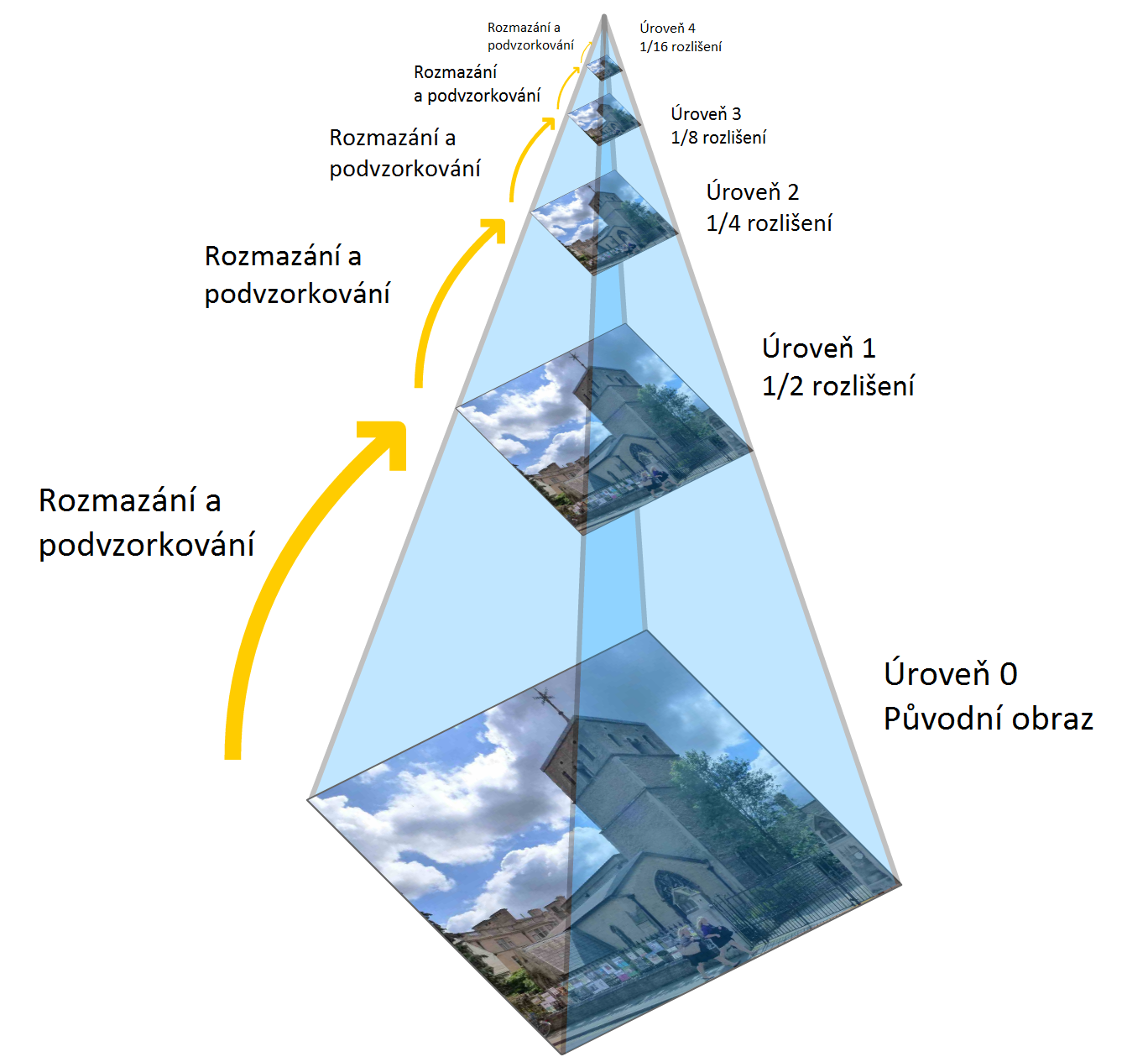
Algoritmus pro rozpoznání obrazovek na emdedded zařízení musí splňovat několik požadavků. Musí být dostatečně rychlý pro použití v reálném čase a úplně automatický, za předpokladu, že je k dispozici databáze srovnávacích dat. Musí být spolehlivý, protože na něm závisí funkce celé robotické ruky a navíc jsou výsledky z běhu každého testování použity na hodnocení kvality produktu, který se testuje. Dalším důležitým kritériem jsou autorská práva. V této chvíli není rozpočet na zaplacení patentovaných algoritmů, proto se bude vybírat jen z těch, které jsou volně ke stažení nebo použití. Z předzpracování vyplývá, že jsou obrazovky normalizovány, částečně zbaveny šumu a program je zkalibrovaný na jejich velikost, resp. souřadnice.

Pro detekci obrazovek jsem byly uvažovány nejdříve nízkoúrovňové metody, jako je porovnávání hranové reprezentace[2], 2D korelaci[4], porovnávání histogramů[5], ale všechny tyto metody byly otestovány a trpěly značnou nespolehlivostí. Další metody jako vyhledávání kruhů a obdélníků pomocí Houghovy transformace [2], prahování a segmentace[5], vyhledávání kontur a segmentace [5] byly o poznání spolehlivější, ale vyžadovaly spoustu apriorní znalosti obrazovek. To znamená, že pokud by došlo k nějaké významnější změně vzhledu embedded aplikace, musely by se všechny vlastnosti znovu nadefinovat a prakticky by bylo potřeba znovu vyvinout celou aplikaci. Navíc jsou výše zmíněné metody velmi výpočetně náročné. Nakonec byla pro detekci a klasifikaci obrazovek zvolena třída metod, která používá deskriptory. Deskriptory se vyznačují svým obecným využitím, relativně rychle se dají vytvořit srovnávací data, a proto je jejich použití pro tuto aplikaci výhodné. Deskriptory jsou také způsobem, jak zobecnit aplikaci pro mnoho různých embedded zařízení, vždy stačí jen vybrat správnou množinu srovnávacích obrazů. Navíc jsou až o řád rychlejší na výpočet než výše zmíněné metody a poskytují dobrý poměr přesnost vs. výpočetní náročnost.

Klasifikace obrazovek obyčejně probíhá v několika fázích. První fáze je detekce bodů zájmu. Důležitá je jejich opakovatelnost, která koreluje se spolehlivostí[6]. Opakovatelnost hodnotí, v jaké míře se detekují ty samé body v několika podobných snímcích jednoho objektu. Další důležitou vlastností je jedinečnost. Je nežádoucí, aby detektory našly několik bodů se stejnými vlastnostmi v okolí, protože je dojde k nežádoucím falešným shodám při přiřazování a srovnávání. Detektory bodů zájmu pracují na několika různých principech, které se mohou prolínat. První skupina pracuje s parciálními gradienty v okolí bodu jako např. Harrisův detektor[7], Lowova Aproximace Laplacianů Gausianů[8], Moravcův detektor rohů[7], detektory založené na výpočtu determinantu Hessiánské matice[6] nebo detektory odvozené od zmíněných. Další skupinou jsou metody pracující na základě porovnávání intenzity centrálního pixelu a jeho okolí jako například SUSAN[9], FAST[10] nebo AGAST[11]. Třetí skupina využívá entropie v okolí pixelu a výpočtu vlastních čísel Hessiánské matice jako je Kadirův a Bradyův detektor asymetričnosti[12]. Dále je mnoho metod velmi podobných od zmíněných nebo které kombinují výše uvedené přístupy.

Po detekci bodů zájmu se výsledky zpřesňují a dále se vylučují nepravděpodobné body. To přispívá k robustnosti celého algoritmu. Vylučují se zejména body ležící v oblasti hran, které jsou nežádoucí, protože jsou málo jedinečné a zavádí falešně pozitivní shody při srovnávání[13]. Podle síly gradientu nebo podle rozdílů intenzit se může počítat skóre bodů a pro další zpracování vybrat jen ty nejlepší[14].

Způsobem, jak zajistit invariantnost deskriptoru vůči zvětšení, resp. zmenšení, je přiřazení měřítka ke každému bodu zájmu. Měřítko se obecně vyhodnocuje pomocí obrazové pyramidy. Jde o dekompozici na vrstvy střídavým rozmazáním Gaussovým filtrem a podvzorkováním[8]. V každé takové vrstvě se detekují body zájmu a podle toho, v které vrstvě se detekují, se k nim přiřadí odpovídající měřítko.



Obr. – Obrazová pyramida

V obecných aplikacích počítačového vidění se také přiřazuje orientace každému bodu zájmu. V této práci a aktuální aplikaci je to přímo nežádoucí, a to z několika důvodů. Prvním je redundance. Takové vyhodnocení není potřeba, protože jsou všechny obrazy normalizované a stejně otočené. Výpočet orientace zvyšuje čas výpočtu, a naopak snižuje robustnost. V aplikaci se nachází hodně textu a rámečků. Pokud detektor najde levý dolní roh rámečku, pak v nerotovaném smyslu je jasně vidět jeho pozice a orientace. Na rozdíl od toho, pokud se hledá jakkoli orientovaný bod se stejnými vlastnostmi, pak se najdou zbylé tři rohy rámečku. Tím by vznikaly falešně pozitivní shody. Z uvedených důvodů se budou vybírat deskriptory, které nejsou invariantní vůči otočení.

Dalším krokem je výpočet deskriptoru. Způsoby výpočtu jednotlivých deskriptorů jsou uvedeny dále v kapitole 2.2. Deskriptory můžou být binární nebo nebinární, s různou dimenzionalitou. Binární deskriptory mají velkou výhodu ve srovnávání deskriptoru aktuálního snímku s deskriptory snímků v databázi, protože se může využít Hammingova vzdálenost[15]. Výpočet Hammingovy vzdálenosti je velmi rychlý a také jednoduchý. Pro výpočet stačí binární operace XOR následovaná sumou bitů. Na rozdíl od toho, na srovnání nebirnárních deskriptorů je potřeba složitějších a výpočetně náročnějších metod. Příkladem je výpočet Euklidovské vzdálenosti, následovaný Houghovou transformací pro detekci klastrů, které patří k jednomu objektu. Nakonec se výsledek verifikuje pomocí metody nejmenších čtverců kvůli konzistenci pozičních parametrů (více v [8]). Dimenzionalita deskriptoru prakticky znamená, jak dlouhý je vektor příznaků pro každý bod zájmu. Obecně platí, čím vyšší dimenzionalita, tím větší robustnost a přesnost, ale zároveň delší čas na výpočet a srovnání deskriptoru s deskriptory snímků v databázi.

Deskritory s vyšší dimenzionalitou dosahují lepších výsledků, ale s počtem snímků v databázi se dramaticky zvyšuje čas pro srovnávání[16]. To se úspěšně řeší redukcí dimensionality pomocí metod jako je PCA[17] a LDA[18] nebo hashovacích funkcí, jako například LSH[19]. Využití těchto metod není s této aplikaci klíčové, zejména kvůli rozsahu databáze srovnávacích snímků (v našem případě desítky snímků).

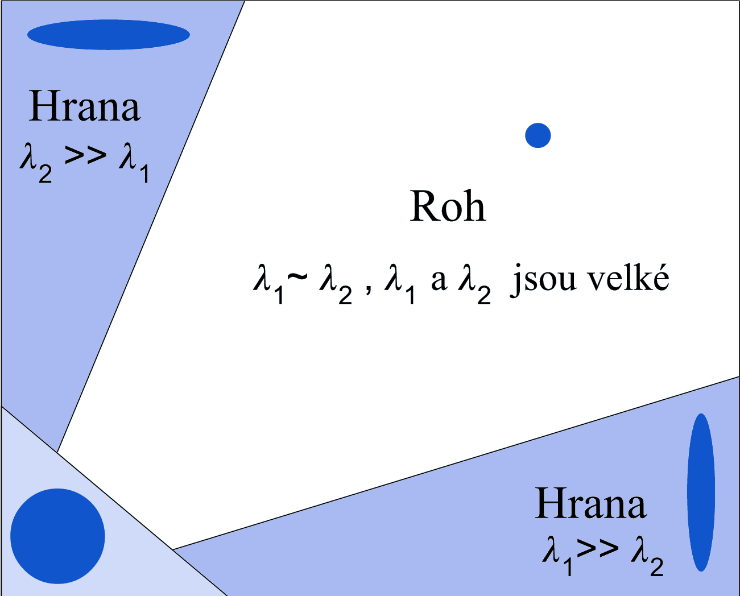
## Detekce zajímavých bodů v obraze

Budou otestovány dva detektory bodů zájmu v obraze. Oba fungují na velmi odlišném principu a budou reprezentovat dvě hlavní skupiny detektorů, ty, které fungují na základě gradientu v okolí bodu a ty, které jen srovnávají intenzity pixelů v okolí. Z těchto skupin byly vybrány právě Harrisův detektor a FAST, a to hlavně kvůli jejich hojnému využití v praxi a dobrým výsledkům ve spojení s vybranými deskriptory (více v kapitole 2.2).

### Harissův detektor rohů

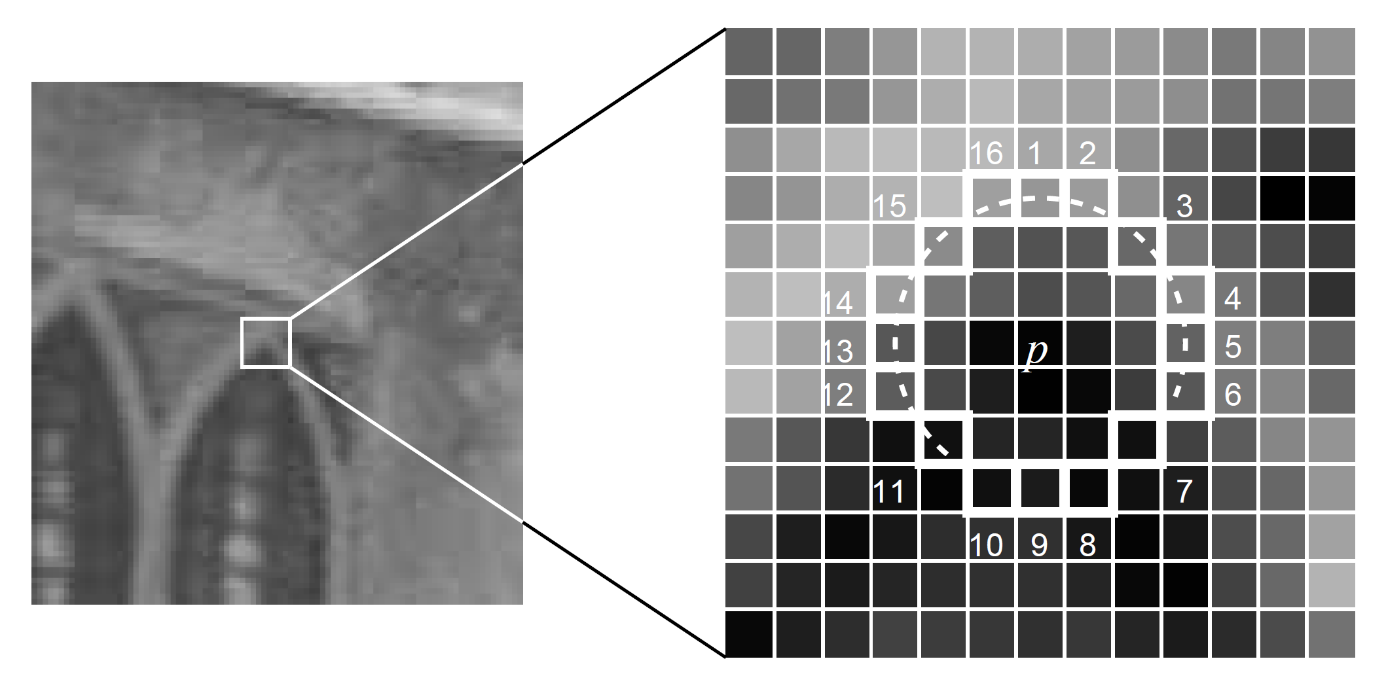
Harrisův detektor je už přes dvě desetiletí standardně používaným detektorem bodů zájmu, a to hlavně kvůli jeho robustnosti a opakovatelnosti. Naopak jeho nevýhodou je pomalý výpočet bodů a relativní citlivost na šum. Pro vyhodnocení bodů zájmu využívá parciální gradienty *(5), (8)* v okolí bodu váhované Gaussovým oknem *w (4), (6), (7)*. K vyhodnocení vlastností v okolí bodu se využívá poměru vlastních hodnot matice *M (4)*. Z Obr. 7 je jasné, že čím jsou obě vlastní čísla větší, tím je bod zájmu silnější.[7]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | |  | | *(5)* | |
|  | () | |  | | () | |
|  | *(8)* | |  | | () | |
|  | *(10)* | |  | |  | |
|  |  | |  | |  | |



Obr. – Reprezentace vlastností objektů v obraze na základě vlastních hodnot matice M

### FAST – Feature from Accelerated Segment Test

Feature from Accelerated Segment Test, dále jen FAST, je jeden z nejrychlejších detektorů bodů zájmu v dnešní době. FAST detektor prochází obraz pixel po pixelu a vyhodnocuje jeho NxN okolí. Jeho základním principem je srovnávání jasu centrálního pixelu Ip s pixely, které leží ve vzdálenosti *r* kolem něj. Pro větší rychlost se nejdříve vyhodnotí čtyři pixely (nahoře, dole, vlevo a vpravo). Pokud alespoň tři z nich mají hodnotu jasu vyšší než Ip+t nebo nižší než Ip-t, kde *t* je prahová hodnota, místo je vyhodnoceno jako potenciální příznak a postupuje se evaluací ostatních pixelů. Pokud se najde spojitý segment pixelů o dané délce *m*, které splňují podmínku Ip+t nebo Ip-t, pak je místo vyhodnoceno jako bod zájmu[10]. Podle délky segmentu a počtu pixelů v kruhu se jednotlivé modifikace FAST detektoru označují. Například FAST 9-16 a FAST 5-8 pro segment od délce 9 resp. 5 po sobě jdoucích pixelů z šestnácti, resp. Osmi pixelů.

Obr. – Test pro detekci rohů v segment 12x12 pixelů. Zvýrazněné čtverce jsou pixely v určité vzdálenosti od středu potenciálního příznaku p a jsou používány pro jeho hodnocení. Část kružnice, která je naznačená čárkovaně, označuje pixely, které jsou alespoň o prahovou hodnotu jasnější než p. [10]

Výhodou tohoto detektoru je vysoká rychlost (až 20x rychlejší než Harrisův detektor). Nevýhodou je, že má vysokou odezvu kolem hran, která může být nežádoucí. Dále je detekce jednotlivých příznaků nezávislá, proto nezajišťuje uniformní pokrytí scény. Další nevýhodou je závislost na prahu. Při nastavení konstantního prahu pro celou sérii obrazů lze dojít k jinému počtu bodů vlivem rozdílného kontrastu snímků.

# AKTUÁLNÍ STAV

Softwarová část funguje na Windows servrech a je připojená k databázi. V databázi se nachází referenční snímky obrazovek, zajímavé regiony a text obražený ve všech obrazovkách. Referenční obrazovky jsou uložené ve stromové struktuře podle kontextu (verze SafeQ, výrobce a model tiskárny). Dále se tam nachází informace o pozici ovládácích prvků a na jakou obrazovku má aplikace přejít po stisknutí ovládacího prvku.

Samotné rozpoznávání obrazovek se děje ve třech fázích. Nejdříve se ověří, jestli se po stisku tlačítka objevila požadovaná obrazovka. Pro detekci příznaků se používá metoda FAST[10] a pak jednoduché srovnání s požadovaným obrazem na základě procentuální shody detekovaných bodů. Pokud je výsledek nepřesvědčivý, nastoupí analýza pomocí descriptorů. V současné době je implementovaná kombinace ORB[20] a BRISK[21] deskriptoru. Počítá se skóre pro každý deskriptor zvlášť, a pak se výsledek průměruje. Pokud i deskriptory selžou, provádí se analýza textu v obraze pomocí OCR[22] a výsledek se srovnává s databází textu na obrazovkách. Každá obrazovka má unikátní slovo, které se nachází jen na příslušné obrazovce a nikde jinde. Pokud se najde pomocí OCR unikátní slovo, pak je jistá detekovaná obrazovka, v opačném případě se počítá procentuální zastoupení slov.

# ZÁVĚR

Předzpracování je z velké části hotové. V dalším semestru se chci zaměřit na extrakci textu z obrazu pro zvýšení rychlosti OCR[22].

Nastudovala jsem problematiku deskriptorů a na základě literatury jsem pro moji aplikaci vybrala BRIEF[14] a UD-BRISK[21] deskriptory. U-SUFT[13] by měl mít podobné výsledky jako předchozí dva deskriptory, ale jelikož je patentovaný, budu dále implementovat jen BRIEF a UD-BRISK. Pro detekci bodů zájmu chci srovnat FAST[10] a Harrisův dektor[7] a vyberu ten, který bude více robustní. Celou aplikaci pro zpracování a analýzy obrazu budu vyvíjet v jazyce C# za pomoci OpenCV knihovny, která funguje v .NET wrapperu OpenCVSharp.

V dalším semestru chci výše zmíněné deskriptory vyzkoušet v praxi, optimalizovat jejich parametry, srovnat jejich výpočetní náročnost a robustnost, popřípadě je zkusit zkombinovat.

# SEZNAM LITERATURY

[1] I. P. Walek, I. M. Lamoš, and J. Jan, *Analýza biomedicínských obrazů počítačová cvičení FEKT VUT v Brně Auto ři textu :* .

[2] J. Jan, *Medical Image Processing , Reconstruction and Restoration Concepts and Methods*. 2006.

[3] Z. Wei, J. Wang, H. Nichol, S. Wiebe, and D. Chapman, “A median-Gaussian filtering framework for Moiré pattern noise removal from X-ray microscopy image,” *Micron*, vol. 43, no. 2–3, pp. 170–176, 2012.

[4] J. Jan, “Číslicová Filtrace, Analýza a Restaurace Signálů.” p. 427, 2002.

[5] R. Szeliski, “Computer Vision : Algorithms and Applications,” *Computer (Long. Beach. Calif).*, vol. 5, p. 832, 2010.

[6] K. Mikolajczyk and C. Schmid, “Scale & affine invariant interest point detectors,” *Int. J. Comput. Vis.*, vol. 60, no. 1, pp. 63–86, 2004.

[7] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” *Procedings Alvey Vis. Conf. 1988*, pp. 147–151, 1988.

[8] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[9] S. Smith and J. Brady, “SUSAN—a new approach to low level image processing,” *Int. J. Comput. Vis.*, vol. 23, no. 1, pp. 45–78, 1997.

[10] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3951 LNCS, pp. 430–443, 2006.

[11] "Elmar Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger", “Adaptive and Generic Corner Detection Based on the Accelerated Segment Test",” *“European Conf. Comput. Vis. (ECCV’10),”* pp. 1–14.

[12] T. Kadir and M. Brady, “Scale Saliency: a novel approach to salient feature and scale selection,” *Vis. Inf. Eng.*, pp. 25–28, 2003.

[13] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3951 LNCS, pp. 404–417, 2006.

[14] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: Binary robust independent elementary features,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6314 LNCS, no. PART 4, pp. 778–792, 2010.

[15] M. Norouzi, D. J. D. D. J. Fleet, R. Salakhutdinov, and D. M. Blei, “Hamming distance metric learning,” *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.

[16] K. Mikolajczyk, K. Mikolajczyk, C. Schmid, and C. Schmid, “A performance evaluation of local descriptors,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, 2005.

[17] B. Moore, “Principal component analysis in linear systems: Controllability, observability, and model reduction,” *IEEE Trans. Automat. Contr.*, vol. 26, no. 1, pp. 17–32, 1981.

[18] D. M. Blei, B. B. Edu, A. Y. Ng, A. S. Edu, M. I. Jordan, and J. B. Edu, “Latent Dirichlet Allocation,” *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[19] L. S. Hashing, “Locality-Sensitive Hashing,” pp. 1–6, 2014.

[20] E. Rublee and G. Bradski, “ORB - an efficient alternative to SIFT or SURF,” 2011.

[21] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK : Binary Robust Invariant Scalable Keypoints.”

[22] J. Mantas, “An overview of character recognition methodologies,” *Pattern Recognit.*, vol. 19, no. 6, pp. 425–430, 1986.

# SEZNAM ZKRATEK A PŘÍLOH

## Seznam zkratek

LoG – Laplacian of Gaussian

DoG – Difference of Gaussian

FAST - Feature from Accelerated Segment Test

SIFT - Scale Invariant Feature transform

SURF - Speeded-Up Robust Features

BRIEF – Binary Robust Independent Elementary Features

BRISK – Binary Robust Invariant Scalable Keypoints