VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

Rozpoznání typu střelné zbraně v obraze

Firearm Type Identification in an Image

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — ONDŘEJ ČECH
AUTHOR

VEDOUCÍ PRÁCE — ING. MICHAL DVOŘÁK
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Čech Ondřej**
Program: Informační technologie
Název: **Rozpoznání typu střelné zbraně v obraze**
Firearm Type Identification in an Image
Kategorie: Zpracování obrazu

Zadání:

1. Prostudujte literaturu týkající se klasifikace objektů v obrazu a seznamte se s nejčastěji využívanými algoritmy.
2. Připravte anotovanou databázi snímků zbraní.
3. Navrhněte algoritmus pro stanovení druhu zbraně dle délky (dlouhá a krátká) a mechanismu nabití (jednoranová, víceranová, opakovací, samonabíjející/samočinná).
4. Postup navržený v předchozím bodu implementujte. Otestujte Vaše řešení na databázi vytvořené v bodě 2.
5. Shrňte a zhodnoťte dosažené výsledky. Diskutujte možnosti budoucího vývoje.

Literatura:

- RUSS, J.C. The Image Processing Handbook. Boca Raton : CRC Press, 1995. 674 p. ISBN 0-8493-2516-1.
- SONKA, Milan, HLAVAC, Vaclav, BOYLE, Roger. Image Processing, Analysis and Machine Vision. 3rd edition. Toronto : Thomson, 2008. 829 p. ISBN 978-0-495-08252-1

Pro udělení zápočtu za první semestr je požadováno:

- Body 1,2 a 3.

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

Vedoucí práce: **Dvořák Michal, Ing.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 1. listopadu 2018

Abstrakt

Cílem této práce je navrhnout, implementovat a otestovat metody pro klasifikaci typu zbraně v obraze do kategorií na krátké a dlouhé, a dále pak na jednoranové, víceranové, opakovací a samonabíjecí nebo samočinné. Tento problém byl řešen použitím SVM klasifikátoru spolu s Harrisovým rohovým detektorem, deskriptorem FREAK a metodou Bag of Words. Výsledný program dosahuje přesnosti 13,3 %.

Abstract

Main goal of this work is to design, implement and test an approach for classifying firearms in an image into categories with short and long fireams, and then with single shot, multi-barreled, repeating and semi-automatic/automatic firearms. This problem was solved using SVM classifier together with Harris corner detector, FREAK descriptor and Bag of Words method. Accuracy of final program is up to 13,3 %.

Klíčová slova

zbraň, klasifikace, obraz, scéna, Harris, FREAK, BoW, SVM

Keywords

gun, firearm, classification, image, scene, Harris, FREAK, BoW, SVM

Citace

ČECH, Ondřej: Rozpoznání typu střelné zbraně v obraze, bakalářská práce, Brno, FIT VUT v Brně, 2019, vedoucí práce Ing. Michal Dvořák

Rozpoznání typu střelné zbraně v obraze

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Inženýra Michala Dvořáka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Ondřej Čech
15. 5. 2019

Poděkování

Chtěl bych poděkovat svému vedoucímu práce panu Ing. Michalu Dvořákovi za pomoc při zpracovávání práce.

© Ondřej Čech, 2019
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1 Úvod

Zbraně hrají nejen v dnešní době velkou roli. Téměř každý den je v nějaké podobě vidíme, ať už se jedná o filmy a seriály, počítačové hry, hračky nebo repliky pro airsoft a paintball, a samozřejmě také zbraně v rukou bezpečnostních složek, jako policie a armáda, nebo civilistů, kteří je mohou mít pro lov, sebeobranu, nebo sportovní střelbu. A bohužel také mezi různými zločinci a teroristy.

Zbraně dokáží pozitivně i negativně ovlivnit bezpečnost ve společnosti. Zejména kvůli riziku zneužití zbraně k trestné činnosti existují v dnešní době různé nástroje, které umožňují bezpečnostním složkám pomoci detekovat zbraně. Například různé detektory kovů a rentgeny, ale i programy, které dokáží detekovat zbraň třeba na záznamu bezpečnostní kamery.

Zrak je pro člověka velmi důležitý smysl, kterým získává naprostou většinu informací o svém okolí. Právě napodobit funkci oka, které získává informace, a mozku, který je následně vyhodnocuje, je cílem disciplíny zvané počítačové vidění. Oko v tomto vztahu zastupuje kamera, jako mozek funguje procesor.

V této bakalářské práci se budeme věnovat určení typu střelné zbraně v obraze. Rozlišovat budeme zbraně na krátké a dlouhé, a dále pak na jednoranové, víceranové, opakovací a samonabíjecí/samočinné. Toto rozdělení vychází z českého zákona č. 119/2002 Sb., zkráceně nazývaného Zákon o zbraních. Pro tento účel byla vytvořena a anotovaná databáze zbraní rozdělených podle těchto kategorií.

Určit typ zbraně může pomoci například bezpečnostním složkám v odhadnutí palebné síly pachatele a naplánování vhodných opatření a postupu pro jeho zastavení. Automatizace tohoto úkonu umožní prohledávat velké množství dat v reálném čase či zpětně analyzovat video záznamy.

První kapitola se bude zabývat samotnými zbraněmi, jejich klasifikací a vztahu těchto klasifikací a vzhledu zbraní. Následně je probrána v rámci práce sestavená databáze zbraní.

Ve druhé kapitole ze budeme zabývat obrazem a jeho počítačovým zpracováním. Definujeme, co je to obraz, jak vzniká a proč je potřeba ho zpracovávat, poté probereme základní předzpracování obrazu a následně se budeme věnovat detektorům význačných bodů v obrazu a deskriptorům, ukážeme si jejich často používané zástupce a jejich principy. Také budou vysvětleny principy Bag of words a SVM, a jejich vztah ke klasifikaci objektů v obraze.

Tématem třetí kapitoly bude návrh řešení a použitý hardware a software. Čtvrtá kapitola se bude věnovat samotné implementaci řešení. V páté kapitole bude řešeno testování implementovaného programu, následně zhodnoceny výsledky a možný budoucí postup této práce.

2 Zbraně

Tato kapitola se bude zabývat základními pojmy ohledně zbraní, jejich rozřazení do kategorií na základě zákonů České Republiky a problematikou jejich rozpoznávání na základě vzhledu. Nakonec přiblíží anotovanou databázi zbraní, která byla v rámci práce sestavena, její zdroje a způsoby, jak zvětšit její rozsah.

Zbraně jsou jedny z nejstarších nástrojů, které člověk začal používat. Za nejstarší známou zbraň lze považovat dřevěné kopí nalezené v Schöningenu v Německu, která jsou okolo 400 000 let stará [30]. Nejstarší známou palnou zbraní je potom bronzový ruční kanón původem ze severovýchodní Číny, který je datován do 13. století [31].

Definice zbraně dle § 118, zákona č. 40/2009 zní: „zbraní se tu rozumí, pokud z jednotlivého ustanovení trestního zákona nevyplývá něco jiného, cokoli, čím je možno učinit útok proti tělu důraznějším.“ Zbraň tedy vzniká z dané věci v okamžiku, kdy je použita za účelem útoku na cíl nezávisle na tom, k čemu byla určena původně. Dle definice zákona může být zbraní také třeba automobil nebo pes, v běžné mluvě se ale slovo zbraň chápe jako předmět nebo stroj, který byl navržen a vyroben právě s důrazem na schopnost působit zranění nebo poškození, tedy věci jako pistole, puška, luk, meč či nůž. Zbraně lze dělit podle mnoha kritérií. V následující podkapitole jsou uvedeny možnosti rozdělení, které budou dále v práci nějak klíčové.

2.1 Dělení zbraní

1) Dělení dle způsobu, jímž je ranivá energie předávána cíli [10]:
- Chladné zbraně: Na cíl působí čepelí, vedenou jílcem, ratištěm (násada kopí, oštěpu, apod.), nebo topůrkem. Dále se dělí na bodné a sečné. Jde o oštěp, dýku, meč, sekeru, kopí apod..
- Úderné zbraně: Působí na cíl přeměnou pohybové energie na destrukci zasaženého místa. Jde o kyj, palcát apod..
- Střelné zbraně: Vymršťují na dálku střelu rozrušující svou dopadovou energií zasažený cíl. Zde je důležité zmínit pojem palná zbraň - střelná zbraň, u které je funkce odvozena od okamžitého uvolnění chemické energie. Jako střelná zbraň je totiž klasifikován i například luk.

2) Dělení podle přenosnosti zbraní [10]:
- Ruční zbraně: Jsou uzpůsobené tak, aby je mohla ovládat a přenášet jedna osoba.
- Lafetované zbraně: Umístěné na podstavci (lafetě), zpravidla ovládané více lidmi.

Z různých důvodů existují lafety i pro zbraně ruční. Skupina zbraní označovaných jako univerzální kulomety je konstruována tak, aby umožňovala plnit funkci jak lehkých tak těžkých kulometů. Lehké kulomety odpovídají ručním zbraním, zatímco těžké jsou typicky montovány na lafetu, tudíž se řadí mezi lafetované. Univerzální kulomet může tedy být ruční i lafetovaná zbraň, v závislosti na konkrétní konfiguraci.

V této práci se budeme zaměřovat na zbraně ruční a střelné, konkrétněji palné. Ty budeme klasifikovat podle následujících dělení.

3) Dělení podle délky zbraně [12]:
- Krátké zbraně: Palné zbraně, jejichž délka hlavně nepřesahuje 300 mm nebo jejichž celková délka nepřesahuje 600 mm.
- Dlouhé zbraně: Palné zbraně, které nejsou krátkými zbraněmi.

4) Dělení podle charakteru střelby [12]:
- Jednoranové zbraně: palné zbraně bez zásobníku nebo jiného podávacího ústrojí, u nichž se opětovné nabití děje ručním vložením náboje do nábojové komory, hlavně nebo nábojiště.
- Víceranové zbraně: palné zbraně bez zásobníku nebo jiného podávacího ústrojí, s 2 nebo více hlavněmi, u níž se opětovné nabití děje ručním vložením nábojů do nábojových komor, hlavní nebo nábojišť.

- Opakovací zbraně: palné zbraně se zásobníkem nebo jiným podávacím ústrojím, u níž se opětovné nabití děje v důsledku ručního ovládání závěru nebo mechanického otočení revolverového válce.
- Samonabíjecí zbraně: palné zbraně, u nichž se opětovné nabití děje v důsledku předchozího výstřelu a u kterých konstrukce neumožňuje více výstřelů na jedno stisknutí spouště.
- Samočinné zbraně: palné zbraně, u nichž se opětovné nabití děje v důsledku předchozího výstřelu a u kterých konstrukce umožňuje více výstřelů na jedno stisknutí spouště.

2.2 Rozpoznávání typu zbraně podle vzhledu

Určování typu zbraně čistě pomocí vzhledu může být problematické, protože z definicí zde uvedených je znatelné, že nejsou až založené na vzhledu zbraní v jednotlivých kategoriích, ale na principu funkce. Tento problém je nejznatelnější na kategoriích samočinných a samonabíjecích, kde je používán stejný systém opětovného nabití a liší se pouze princip spouště. V současné době jsou samočinné zbraně vyráběny především pro bezpečnostní složky, zejména armádu, ale výrobci zbraní často vedle této vojenské produkce vyrábějí i samonabíjecí verze vzhledově stejné s samočinnou variantou.

V České Republice je to například kulovnice CZ 858, samonabíjecí varianta zbraně Samopal vzor 58 [17]. Z krátkých zbraní lze u tohoto problému uvést samočinnou pistoli Glock 18 odvozenou od samonabíjecího Glocku 17 [18]. Dalším příkladem může být CZ 75 Automatic. Zároveň je možné zbraně vyrobené v jedné kategorii konvertovat na jinou, takto se například upravují vyřazené armádní zbraně na samonabíjecí a následně prodávají na civilním trhu. A také se obráceně upravují civilní zbraně na střelbu dávkou, například pro potřeby filmu [18] nebo ilegálně. Poměrně známá je přestřelka v Severním Hollywoodu, kde bylo použito několik původně civilních zbraní předělaných na střelbu dávkou. Z těchto důvodů bylo rozhodnuto rozpoznávat samočinné a samonabíjecí zbraně jako jednu kategorii.

Obr. 2.1: samočinná zbraň Samopal vzor 58 vlevo [19], samonabíjecí kulovnice CZ858 vpravo [20]

Obr. 2.2: samonabíjecí Glock 17 vlevo, samočinná Glock 18 uprostřed, Glock 17 upravená na samočinnou vpravo [18]

Norma ČSN 39 5002-1 slučuje kategorie „samonabíjecí“ a „samočinná“ do jedné s označované jako „automatická“. Tato kategorie není v zákoně č. 119 nikde zmíněna a často dochází k označování zbraně samočinné jako automatické s tím, že samonabíjecí zbraně jsou nazývány poloautomatické, z anglického *semi-automatic weapon.* Různé prameny uvádějí rozdílné definice pojmu automatická zbraň, mnohé si v těchto definicích vzájemně odporují. Pojmy „samonabíjecí“ a „samočinná“ užité v této práci jsou používané podle jejich definice v zákoně č. 119, pojem „automatická“ pak vychází z definice v normě ČSN 39 5002-1 a je použit pro společnou kategorii samočinných a samonabíjecích zbraní klasifikovaných v této práci na základě dříve popsaného problému.

Dále je třeba zmínit, že pojmy krátká a dlouhá zbraň někdy označují, zda je zbraň konstruována pro střelbu za použití jedné nebo obou rukou [11]. Dle definice zákona č. 119 totiž do kategorie krátkých spadá i poměrně dost kompaktních zbraní konstruovaných pro střelbu obouruč, jako například PDW Heckler & Koch MP7 (délka hlavně asi 180 mm [13]), CZ Skorpion 61 S (délka hlavně 115 mm [14]), nebo dokonce některé kompaktní varianty pušek, jako FN SCAR-L CQC (délka hlavně 254mm [15]), AKS-74U (délka hlavně 210 mm [16]), v kategorii opakovacích zbraní pak například brokovnice Serbu Super-Shorty (délka hlavně nejkratší varianty 165 mm [23]). Právě v této kategorii se často jedná o zbraně, které jsou vyráběny ve více variantách, z nichž většina svou délkou spadá už do kategorie dlouhých zbraní a jen ty nejkratší do krátkých. Jelikož mezi těmito variantami nejsou kromě délky velké rozdíly ve vzhledu, tyto rozdíly nemusí být na zkoumaném obrazu vůbec vidět a schopnosti těchto krátkých zbraní jsou srovnatelné s dlouhými, rozhodl jsem se tyto kompaktní obouruční zbraně řadit jako zbraně dlouhé.

Obr. 2.3: krátká (hlaveň 254 mm) FN SCAR-L CQC vlevo, dlouhá (hlaveň 368 mm) FN SCAR-L STD vpravo [15]

Poslední kategorií zbraní, na kterou by v rámci rozpoznávání typu zbraně z obrazu bylo vhodné upozornit, jsou pušky s válcovým závěrem, jednoranové a opakovací. Opakovací pušky jsou totiž často konstruované tak, že jejich zásobník není vidět a nejde tak úplně jistě určit, zda se jedná o jednoranovou nebo opakovací zbraň.

Obr. 2.4: jednoranová Savage Youth Rascal vlevo [21], opakovací Savage Arms B 22 FVSS s kapacitou 10 ran vpravo [22]

Další problémy při rozpoznávání z obrazu mohou vznikat, pokud je zbraň vybavena doplňky, jako jsou kolimátory a optické zaměřovače, rukojeti, podvěsné granátomety a další. Jejich použití mění vzhled zbraní a může vést k zmatení rozpoznávacího programu, který se zaměří na tyto doplňky místo samotných důležitých částí zbraní. Tyto doplňky totiž zejména v poslední době používají unifikovaný systém připojení na zbraň a jejich přítomnost tedy odvodit kategorii zbraně nemůže nijak usnadnit, spíše naopak.

Každá kategorie zbraní má na základě své kategorie určité znaky, které ji mohou pomoci určit. Automatická puška má zpravidla zásobník a okénko vyhazovače nábojnic, a jako dlouhá zbraň bude mít na rozdíl od krátké pistole pažbu a také celkový tvar bude daleko delší.

Obr. 2.5: znaky automatických zbraní [51][50]

U některých jednoranových zbraní jsou často vidět mechanismy iniciace, jako jsou otočné kohouty.

Obr. 2.6: znaky jednoranových zbraní [51]

Typickým znakem víceranových zbraní je samozřejmě počet hlavní.

Obr. 2.7: znaky víceranových zbraní [50][51]

Opakovací zbraň musí mít nějaký mechanismus, kterým střelec vyhodí prázdnou nábojnici a nabije novou. Nejznámější tyto nabíjecí systémy jsou asi válcový válec s výraznou klikou závěru

na jedné straně, dále pákový, který se vyznačuje rozšířeným pohyblivým lučíkem, a princip „pumpa“ s pohyblivým předpažbím. Typickým zástupcem krátkých opakovacích zbraní je revolver s výrazným revolverovým válcem.

Obr. 2.8: znaky opakovacích zbraní [50][51]

Opakovací zbraně také mají zásobník, čímž se s předchozím textem dostáváme k velkému problému: Žádný z těchto znaků není definitivní, a žádný z nich nemusí být na obrázku vidět. Jak již bylo řečeno, zásobníky mohou být skryté, mohou být ze zbraně na snímku odstraněny. Znaky mohou být na obraze skryty za jiným objektem, na snímku nemusí být zbraň kompletně vidět, může být z úhlu, ze kterého není znak vidět a spousta dalších faktorů včetně problémů při kategorizaci podle vzhledu zmíněných výše předurčují samotné hledání těchto znaků obecně jako nevhodné. Nalezení takového znaku na obraze ale může pomoci zúžit okruh možných kategorií.

2.3 Databáze zbraní

V rámci práce byla pro natrénování rozpoznávání sestavena databáze obrazů střelných zbraní roztříděných do skupin podle kritérií probíraných v kapitole 2. Obrazy byly získány z následujících zdrojů:

- IMFDB.ORG[1]: Internet Movie Firearms Database. Stránka se zabývá identifikací zbraní použitých zejména v televizní tvorbě (filmy, seriály) a v počítačových hrách. Obsahuje zejména snímky scén filmů, kde se nějaká zbraň vyskytuje, a dále také snímky samostatných zbraní pro porovnání. Stránka funguje na principu wiki, tedy že uživatelé mohou přidávat i měnit obsah. Proto je vhodné si informace o zbrani na obraze ověřit, protože nemusejí být úplně pravdivé. Dále je nutno ze snímků ze scény vybrat tu část se zbraní, protože pro tuto práci musí snímek obsahovat pouze zbraň.
- Google.com[2]: Populární internetový vyhledávač umožňuje snadno najít velké množství digitálních obrazů. Stejně, jako v případě IMFDB je často nutné obrazy upravit a následně ručně kategorizovat. Značná část snímků nalezených pomocí google.com pochází z různých obchodů se zbraněmi.
- Collectors Firearms[3]: Americký internetový obchod se zbraněmi, s rozsáhlou nabídkou včetně těch historických. Nabízené zbraně jsou kvalitně nafocené z více stran, proto jsem tuto stránku využil, jako další zdroj snímků do databáze.

Další použitou metodou na získávání snímků do databáze bylo generování snímků pomocí 3D modelů. K tomu byl použit nástroj SYDAGenerator[4]. Ten vykresluje vybraný objekt formátu g3db

1 http://www.imfdb.org/wiki/Main_Page
2 https://www.google.com/
3 http://www.collectorsfirearms.com
4 http://www.fit.vutbr.cz/~igoldmann/app/SYDAGenerator

a obrázek pozadí. Jako zdroj 3D modelů byla použita internetová stránka Free3D.com[5], která poskytuje ke stažení 3D modely zbraní v různých rozšířených formátech, ale použití formátu g3db není časté, proto bylo nutné stažené modely konvertovat. K tomu byl využit nástroj fbx-conv[6], který podporuje konverzi z typů obj a fbx do g3db. Taktéž byl použit 3D editor Blender v2.76[7] s pluginem pro export modelu do formátu g3db LibGDX Blender G3D Exporter[8].

Kromě ukládání nových obrázků je možnost rozšíření databáze i pomocí augmentace už existujících obrazů. Zvětšení databáze pomocí augmentace pomáhá při rozpoznávání tak, že snižuje šanci na mylné rozpoznání nesprávného příznaku a pomáhá ke zobecnění úlohy. Pro augmentaci obrazů můžeme použít několik transformací.

• Rotace: náhodné otočení obrázku o 0° až 180° po směru nebo proti směru hodinových ručiček

Obr. 2.9: příklad otočení obrazu [24]

• Posun: obrázek lze posunout po ose X nebo Y. Může dojít ke ztrátě části informace obsažené v původním obraze.

Obr. 2.10: příklad posunutí [24]

5 https://free3d.com/3d-models/weapons

6 https://github.com/libgdx/fbx-conv

7 https://www.blender.org

8 https://github.com/Dancovich/libgdx_blender_g3d_exporter

- Překlopení: podle vodorovné nebo svislé osy.

Obr. 2.11: příklad překlopení [24]

- Škálování: zvětšování nebo zmenšování původního obrázku. Stejně jako při posunu může při zvětšování dojít ke ztrátě informací obsažených v původním obrázku.

Obr. 2.12: příklad škálování [24]

Při rotaci, posunu a zmenšování vznikne prázdné místo, které by mohlo generovat nechtěné hrany a narušovat průběh rozpoznávání. Z tohoto důvodu je potřeba tuto plochu nějak zakrýt tak, aby nenarušovala původní strukturu obrázku. Vhodné řešení je vyplnit tuto plochu pixely z původního okraje obrázku.

Aktuální databáze obsahuje 2531 snímků ve všech kategoriích.

Problémem při sestavování databáze byla především variabilita, proměnlivost a nerovnoměrnost v jednotlivých kategoriích zbraní. Automatické zbraně, zejména ty dlouhé existují v současnosti v obrovské spoustě vzhledových variací, konstrukcí (například zbraně typu bull-pup, tedy ty, které mají nabíjecí mechanismus spolu s zásobníkem umístěný až za spouští) a konfigurací, a to ať už výrobcem zbraně nebo jejím samotným uživatelem. Tento fakt ještě znásobuje problémy s přesným určením kategorie zbraně uváděné v podkapitole 2.2 a pro zajištění odpovídající kvality klasifikace by bylo třeba skutečně velké množství příkladů, které není zas takový problém sehnat, jelikož právě tyto zbraně se často objevují ve filmech, počítačových hrách a obecně jsou častěji užívány. Naproti tomu kupříkladu krátké jednoranové nebo víceranové zbraně zahrnují především vzácnější historické nebo méně často používané typy, které nejsou velmi často k vidění, a tudíž i jejich fotografie jsou často hůře dohledatelné.

Obr. 2.13: Některé z extrémnějších případů podob dlouhých automatických zbraní pro porovnání. FN F2000 [47], Calico 960A [48] a FN Minimi [49]

3 Obraz a jeho zpracování

Informace obsažené v této kapitole pocházejí ze zdrojů [1], [2] a [3]

Obrazem se zde myslí optický obraz v běžném smyslu, tedy například obraz na sítnici lidského oka nebo obraz viděný televizní kamerou. Tyto obrazy jsou dvourozměrné. Matematicky lze obraz definovat jako funkci f(i,j), kde i, j jsou souřadnice v ploše. Funkce f(i,j) se nazývá obrazová funkce.

Prostředí kolem nás má trojrozměrnou povahu. Obrazová funkce může vznikat perspektivním zobrazením části trojrozměrného prostředí. Přitom se ale ztratí velká část informace, kterou je velmi těžké zjistit zpětně a jsou potřeba další dodatečné znalosti o problému. Některé zpracovávané obrazy mohou mít přímo dvojrozměrnou povahu, například obraz znaků na papíře.

Digitální obraz je reprezentace obrazu pomocí hodnot, se kterými je počítač schopný pracovat, tedy bity. Existují dvě varianty, jak lze tohoto docílit. Rastrové nebo také bitmapové obrazy, kde je nejmenší jednotka obrazový element. Pro něj se běžně používá slovo pixel vzniklé z anglického termínu picture element. Tento element může nabývat více rozměrů, například v běžném barevném digitálním obraze se používají tři hodnoty pro červenou, zelenou a modrou barvu (RGB), běžným rozsahem je 24 bitů na pixel, tedy 8 bitů na jednu barvu. Z toho vyplývá, že jedna barevná složka je reprezentována hodnotami 0-255. Jelikož už bylo zmíněno, že jsou používány barvy červená, zelená a modrá, je zřejmé, že se používá aditivní míchání barev. Počet pixelů v obraze je vždy konečný. Od počtu pixelů v obraze se odvíjí rozlišení obrazu, jeden z faktorů kvality výsledného obrazu. Druhou možností reprezentace jsou vektorové obrazy. Vektorový obraz se skládá z jednotlivých grafických, matematicky definovaných objektů, jako úsečky, křivky, mnohoúhelníky, písmena a další. Vektorové obrazy jsou značně složitější na získání a je pro ně typické, že vznikají za velké pomoci uživatele. Jedná se například o tvorbu ilustrací nebo sazbu textu. Takové obrazy obvykle není nutné zkoumat metodami zpracování obrazu, případně většinou pro tento účel existuje nějaké efektivnější řešení, pokud je potřeba. Oproti tomu typickým příkladem rastrového obrazu je fotografie, která je daleko běžnější reprezentace obrazu, a daleko častěji bude využívána právě k zpracování.

Abychom mohli obrazovou funkci zpracovat počítačem, potřebujeme napřed získat její digitální ekvivalent. Toho dosáhneme pomocí vzorkování obrazu v matici M*N bodů a kvantování spojité úrovně vzorku do K intervalů. Kvalita výsledného digitálního obrazu se odvíjí od jemnosti vzorkování a kvantování, tedy čím větší M, N a K, tím kvalitnější aproximace původní spojité funkce. Důležité je také plošné uspořádání bodů pro vzorkování. Nejčastější je pravidelná čtvercová mřížka. Digitální obraz je tedy datová struktura reprezentovaná jako matice, kde prvek matice je obrazový element, který odpovídá jednomu vzorkovacímu bodu.

Zpracování digitálního obrazu je dnes velmi rozšířená věc. Běžně se editují digitální snímky a videa, rozšířené je rozpoznávání obličeje v fotoaparátech. V továrnách lze využít automatickou vizuální kontrolu výrobku, Z leteckých a satelitních snímků lze získat data o přírodních zdrojích nebo sledování objektů v rámci vojenského průzkumu. V lékařství se hojně používá zpracování obrazů z CT a magnetické rezonance. Poslední dobou je zpracování obrazu také nutné třeba při vývoji autonomního řízení vozidel [4].

3.1 Předzpracování obrazu

Zkoumaný obraz s největší pravděpodobností nebude dokonalý. Například může obsahovat šum, nemusí být správně zaostřený, vliv hraje i správné nasvětlení při pořizování obrazu a velké množství dalších faktorů, které výsledek ovlivní. Proto je důležité pomoci následnému rozpoznávání úpravou vlastností tohoto vstupního obrazu.

Existují různé operace, jak vstupní obraz upravit a zvýraznit vlastnosti, které pomohou k rychlejšímu nebo přesnějšímu rozpoznávání. Dají se rozdělit na jednobodové a prostorové. Jednobodové operace pracují pouze s jedním bodem a jeho hodnotou, v případě barevného obrazu hodnotami, vícebodové operace do výpočtu zahrnují i okolní body.

3.1.1 Bodové, lokální a globální jasové transformace

Jasová transformace znamená změnu hodnot obrazové funkce vstupního obrazu podle daného pravidla, aniž by se nějak měnily jeho parametry, jako rozlišení nebo bitová hloubka. Bodová transformace znamená, že nová hodnota pixelu je vypočítána pouze z původní hodnoty daného pixelu. Výsledek lokální transformace pak závisí na hodnotách lokálního okolí pixelu a globální na hodnotách celého obrazu. Asi nejběžnější, nejjednodušší a nejznámější jsou úprava jasu, kontrastu a obecně histogramu. Úprava jasu znamená, že ke všem stávajícím hodnotám barev se přičte konstanta. V případě kontrastu se k nadprůměrně světlým hodnotám konstanta přičítá a od podprůměrných se odečítá.

Operací velmi častou pro zpracování obrazu je převedení do stupňů šedi. V rámci této práce už v kapitole 2 byly ukázány obrazy některých zbraní. Na těchto zbraních je vidět, že zbraň zdaleka nemusí být jenom černá. Dodnes se u starších zbraní používají různé druhy dřev na pažbení, kovové součástky mohou být pro změnu poniklované, vyrobené z nerezu nebo dokonce pozlacené, polymerové díly moderních vojenských i civilních zbraní se pro zase vyrábějí v odstínech pískové, hnědé nebo zelené kvůli maskování, a u zbraní pro civilní trh se už žádné meze v barvách zbraní nekladou, a tak není výjimkou třeba puška v kombinaci bílé a růžové. To vede k závěru, že barvy v obraze při identifikaci zbraně nenesou zas až tak velké množství informace, jak by se mohlo zdát. Navíc zpracování všech barevných kanálů zpomaluje algoritmus zpracování. Převod na šedotónový obraz tedy spočívá v tom, že z dosavadních hodnot barev vypočítáme jednu výslednou. Pro tento výpočet se používá vzorec [5]:

$$Y = 0{,}299 * R + 0{,}587 * G + 0{,}114\, B \,[5] \qquad (3.1)$$

kde Y je výsledná hodnota pixelu, R je hodnota červené složky pixelu, G zelená složka a B modrá složka.

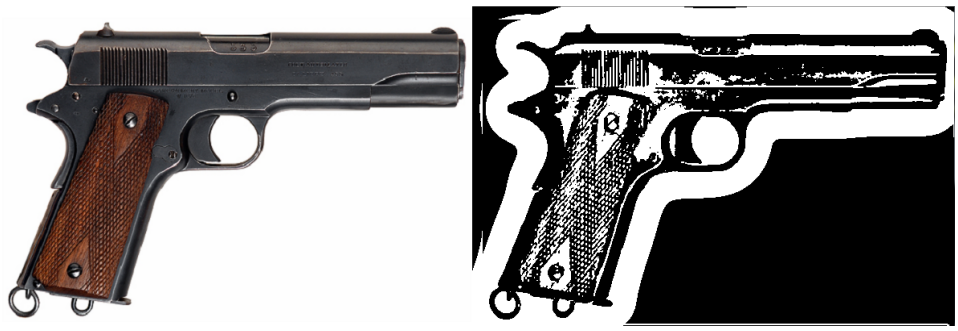
Obr. 3.1: převod do stupňů šedi [24]

Používána je i operace prahování. Je to nejčastěji používaná a nejjednodušší metoda segmentace obrazu. Na začátku je třeba určit hodnotu prahu v histogramu, poté jsou hodnoty nižšího jasu (pod prahem) převedené na černou barvu a nad prahem na bílou. V ideálním případě tím vznikne

jasně rozdělený obraz na dvě části, například se tím oddělí pozadí od popředí. Kvůli různým vlastnostem obrazů byly vyvinuty různé varianty adaptivního prahování, které mohou být účinnější v nehomogenních obrazech.

Obr. 3.2: Segmentace prahováním [24]

Adaptivní prahování nepoužívá jedinou hodnotu prahu pro celý obraz, ale pro jeho část nebo každý pixel vypočítá tuto hodnotu zvlášť. K tomuto výpočtu se používají původní hodnoty prahovaného bodu a bodů v jeho okolí.

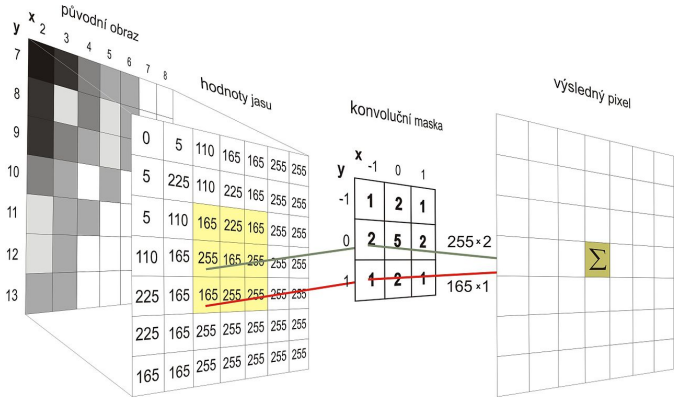


Obr. 3.3: použití adaptivního prahování [24]

3.1.2 Prostorové operace

Další z používaných operací nad obrazy jsou detektory hran. Hrana je místo v obraze, kde výrazně mění hodnota jasu mezi pixely. Na fotce to tedy nemusí být jen v místech okrajů objektů, ale i třeba v místech přechodu mezi světlem a stínem. Pokud platí, že se mezi jednotlivými pixely výrazně mění hodnota jasové funkce, pak zde bude vycházet i vysoká hodnota derivace této funkce, a nejvyšší možná bude směrem kolmo proti této hraně. Ale detekovat hrany ve 360° by bylo velmi náročné na výpočet, i vzhledem k tomu, že struktura obrazu je běžně čtvercová mřížka, jak již bylo zmíněno v kapitole 3. Běžně se proto detekují hrany jen ve dvou směrech a to vodorovném a svislém. Derivace se nejčastěji aproximuje využitím konvoluce, proto se tyto filtry často zapisují ve tvaru podobnému matici nebo tabulce, který nazýváme jádro. Vzorec pro výpočet hodnoty bodu pak má tvar:

$$I(x,y)=\sum_{i=-k}^{k}\sum_{j=-k}^{k} h(i,j)*f(x-i,y-j) \tag{3.2}$$



Obr. 3.4: Princip konvoluce [28]

kde x, y jsou souřadnice cílového bodu. Hodnota k pak závisí na rozměrech masky filtru tak, že k=(a-1)/2, kde a je rozměr čtvercového filtru. Jeden z filtrů často používaný právě na detekci je hran je Sobelův operátor. Ten má v zápisu ve tvaru jádra tvar:

1	0	-1
2	0	-2
1	0	-1

1	2	1
0	0	0
-1	-2	-1

Obr. 3.5: matice Sobelova operátoru

Z toho je vidět, že se vlastně jedná o dva filtry pro detekci svislých a vodorovných hran. Filtry pro detekci hran se také nazývají horní propusti, protože propouštějí signál vyšších frekvencí.

Obr. 3.6: použití Sobelova operátoru [24]

Ke zpracování obrazu se využívají i dolní propusti, které dokáží snížit šum v obraze. Princip funkce je stejný, jako u detektorů hran. Jako příklady lze uvést box filtr, který počítá nový bod jako průměr bodů a Gaussův filtr na podobném principu, který ale dává větší váhu původní hodnotě bodu.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

Obr. 3.7: Matice Gaussova filtru

Tyto filtry se nazývají lineární. Jejich nevýhodou je, že vyhlazují celý obraz, tedy i hrany, které bychom při zpracování obrazu mohli potřebovat. Naproti tomu nelineární filtr je k odstranění šumu vhodnější. Příkladem nelineárního filtru je mediánový filtr, který z okolí cílového bodu vybere hodnotu mediánu, kterou uloží jako novou hodnotu pixelu.

Obr. 3.8: použití Gaussova filtru [24]

3.2 Detektor významných bodů

Nejprve je třeba vysvětlit, co se myslí pojmem významný bod. Jak již název napovídá, je to místo, které se od ostatních nějak významně odlišuje. Z pohledu rozpoznávání obrazu je nutné, aby tyto body[6]:

- měly jasnou, nejlépe matematicky dobře podloženou definici.
- měly jasně definovanou pozici v prostoru obrazu.
- měly strukturu obrazu okolo sebe bohatou na obsah informací vhodných pro další stupně zpracování.
- byly stabilní z hlediska působení lokálních a globálních deformací, včetně deformace perspektivou, změny osvětlení objektu, a tím pádem zajišťovaly vysoký stupeň znovupoužitelnosti v různých obrazech.
- byly navzájem dostatečně odlišné, aby šlo oddělit detekované významné body odpovídající fyzicky různým bodům.

Mezi nejdůležitější takováto místa v obraze patří hrany a rohy. Jelikož detekce hran už byla zmíněna v předchozí kapitole, lze přejít rovnou k detekci rohů. Roh je typicky místo v obraze, kde se střetávají hrany. Pro toto místo je charakteristické, že dochází ke změně jasové funkce ve všech směrech, na rozdíl od hrany, kde ke změně nedochází, pokud postupujeme souběžně s danou hranou. Právě detekce rohů je základem detekce významných bodů. Spolu s detekcí bodů se používá i detekce významných ploch, nazývaná blob detection, která se zabývá hledáním oblasti, která má oproti okolním oblastem odlišnou barvu nebo jas.

3.2.1 Harrisův detektor

Tato podkapitola čerpá ze zdrojů [7], [8] a [9].

Harrisův detektor byl vyvinut v roce 1988 Chrisem Harrisem a Mikem Stephensem v Plessey Research Roke Manor ve Velké Británii, odtud také pochází občas používané označení „Plessey operátor“. Cílem jejich práce bylo najít a přiřadit k sobě body na rozdílných obrazech získaných pohybující se kamerou. Jako základ použili 11 let starý Moravcův detektor. Ten funguje na základě detektoru rohů posuvným obdélníkovým oknem a počítá rozptyl intenzity jasu při posunutí v různých směrech, z toho je schopen určit hranu, která má velkou změnu v jednom směru a minimální nebo žádnou ve směru kolmém, roh, který má změny ve všech směrech, a plochu, která změny nemá. Hlavní nevýhodou tohoto řešení je rotační invariance, protože počítáme změnu intenzity vlastně jen v osmi směrech. Harrisův detektor místo toho používá kruhové okno s gaussovským rozložením vnitřních hodnot, přičemž největší váha připadá na hodnoty v blízkosti zkoumaného bodu.
Harrisův operátor je i v dnešní době používaný, i když má svoje nevýhody. Zejména je citlivý na šum, proto je třeba před jeho použitím použít některou z metod redukce šumu, které byly zmíněny v kapitole 3.1.2.

Základem matematické specifikace Harrisova detektoru je následující vzorec:

$$f(x,y)=\sum_{(x,y)\in W}(w(x,y)*[I(x,y)-I(x+\Delta x,y+\Delta y)])^2 \quad (3.3)$$

kde W je okno okolo zkoumaného bodu,
x,y jsou souřadnice zkoumaného bodu v obraze,
w je váhová funkce,
I je jasová funkce,
Δx, Δy značí lokální posun okna.

Váhová funkce je volitelná, běžně se používá gaussovská nebo konstantní nabývající hodnoty 1 v případě, že souřadnice náleží do do okna W, a hodnoty 0, pokud do okna nenáleží. Už bylo zmíněno, že okno nemusí být jen čtverec a obdélník, ale třeba kruh.

Jako další krok je třeba aproximovat výraz $I(x+\Delta x, y+\Delta y)$ značící intenzitu jasové funkce na posunutých souřadnicích. To lze provést pomocí Taylorova rozvoje. Nechť I_x a I_y jsou parciální derivace I v osách x a y tak, že:

$$I(x+\Delta x,y+\Delta y)\approx I_x(x,y)\Delta x+I_y(x,y)\Delta y \quad (3.4)$$

tudíž platí:

$$f(x,y)\approx\sum_{(x,y)\in W}(w(x,y)*[I_x(x,y)\Delta x+I_y(x,y)\Delta y])^2 \quad (3.5)$$

což lze zapsat ve tvaru matice jako:

$$f(x,y)\approx(\Delta x\,\Delta y)M\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix} \quad (3.6)$$

kde:

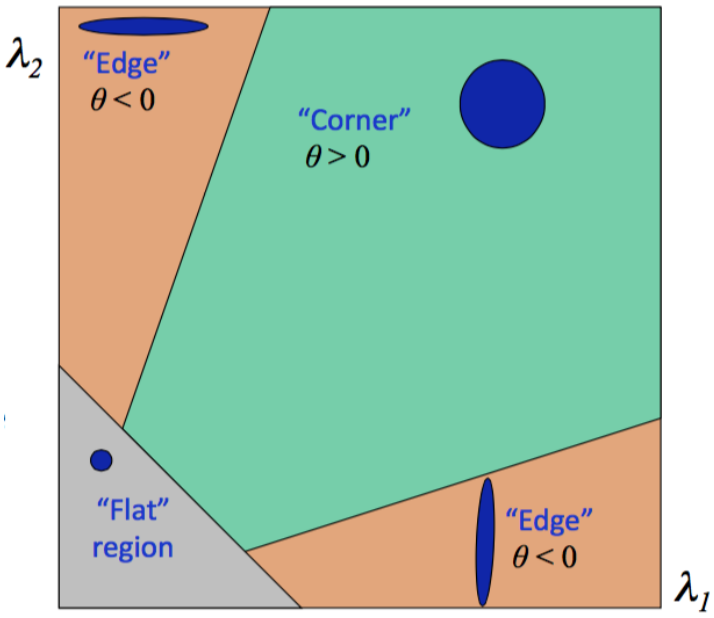
$$M=\sum_{(x,y)\in W}\begin{bmatrix}I_x^2 & I_xI_y\\ I_xI_y & I_y^2\end{bmatrix}=\begin{bmatrix}\sum_{(x,y)\in W}I_x^2 & \sum_{(x,y)\in W}I_xI_y\\ \sum_{(x,y)\in W}I_xI_y & \sum_{(x,y)\in W}I_y^2\end{bmatrix} \quad (3.7)$$

V praxi se ale využívá následující vzorec, takzvaná Harrisova funkce:

$$R=det(M)-k(trace(M))^2 \quad (3.8)$$

Operace *trace* se vypočítá jako součet prvků na hlavní diagonále matice M.
Konstanta k je stanovena experimentálně, doporučované jsou hodnoty mezi 0,04 a 0,06 [8]
Determinant se vypočítá běžným způsobem.
Hodnota R rozhoduje, zda se jedná o plochu, hranu nebo roh:

- Pokud |R| je blízké nule, což znamená, že I_x a I_y jsou nízké, tedy malé změny jasu, zkoumaný bod je v ploše.
- Pokud R<0, což se stane, pokud $I_x >> I_y$ nebo obráceně, je zde velká změna z jednoho směru a minimální z na ni kolmého, tedy hrana
- Pokud R>0, znamená to, že I_x a I_y jsou velké a srovnatelné, tedy změna ze všech stran a tudíž roh.

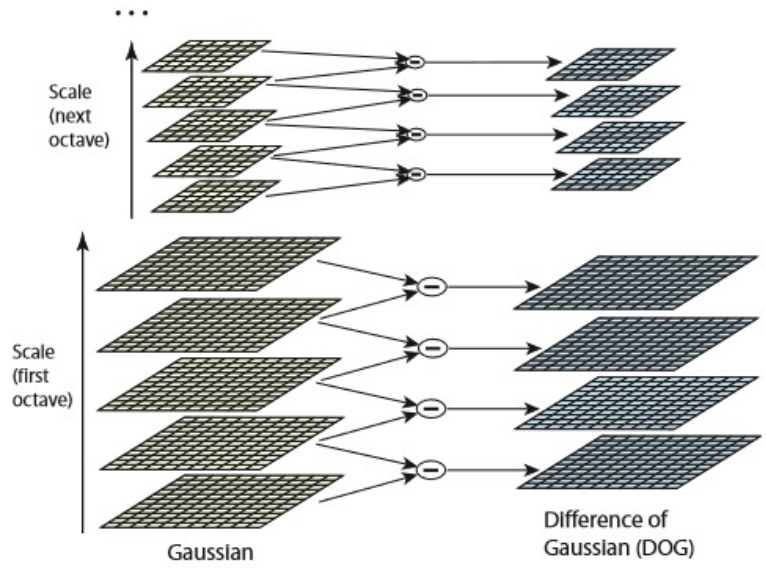


Obr. 3.9: Klasifikace významných oblastí podle Harrisovy funkce [25]

3.2.2 SIFT

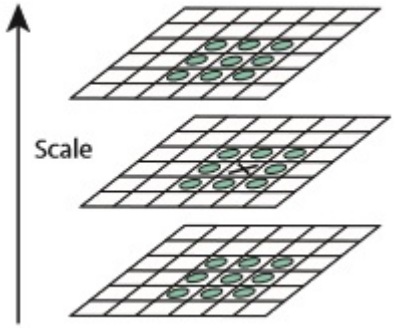
Podkapitola vychází z [8] a [37].
SIFT (Scale-Invariant Feature Transform) je jeden z modernějších detektorů, vyvinutý v roce 2004 Davidem Lowem na University of British Columbia v Kanadě. V podstatě jde o kombinaci deskriptoru a detektoru, kterým je věnována následující kapitola. Cílem metody je nezávislost na měřítku, což znamená, že řešení je velmi komplexní a náročné na výpočet. Algoritmus tedy není úplně vhodný na výpočty v reálném čase, ale i tak se často uplatňuje v metodách počítačového vidění.

Hlavní myšlenka spočívá v nalezení bodů, které zůstávají stabilní při změně měřítka. Je tedy sestavena funkce, která postupně převádí vstupní obraz do různých měřítek pomocí konvoluce s Gaussovým filtrem, kdy další měřítko vzniká konvolucí filtru s výsledkem předchozí konvoluce. Výsledky „sousedních“ konvolucí se od sebe odečtou, toto se nazývá Rozdíl Gaussiánů (Difference of Gaussians, DoG).



Obr. 3.10: Princip tvorby DoG [37]

Následně jsou vyhledány lokální extrémy. Ty jsou určeny tak, že bod v DoG je porovnáván se svými osmi sousedy a následně s 9 sousedy v rozdílu vyšších měřítek a s dalšími 9 v rozdílu nižších. Takto jsou porovnány všechny úrovně kromě té nejvyšší, která nemá vyšší rozdíl, a té nejnižší, která nemá nižší rozdíl.



Obr. 3.11: Princip vyhledávání lokálního extrému [37]

Tyto body jsou kandidáti na významné body. Jsou popsány pomocí funkcí popisující okolí bodu pomocí Taylorova rozvoje:

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x \tag{3.9}$$

kde $x=(x, y, \sigma)^T$ jsou souřadnice bodu identifikovaného jako lokální maximum a D(x) je aproximace DoG. Rovnice se položí rovna nule a je vypočítán extrém:

$$x = \frac{-\partial^2 D^{-1}}{\partial x^2} \frac{\partial D}{\partial x} \tag{3.10}$$

Pokud je x větší, než 0,5 v jakékoli dimenzi, znamená to polohu extrému blíže k sousednímu bodu, je třeba vyměnit bod za bod sousední a opakovat výpočet. Pokud je hodnota menší, pak je hodnota x přičtena k souřadnicím zkoumaného bodu.

Z těchto vybraných bodů je třeba odfiltrovat takové s nízkým kontrastem, které jsou závislé na šumu. Také body ležící v blízkosti hran nejsou vhodné, tyto lze odfiltrovat pomocí Hessovy matice 2x2. Zbylé body jsou jednoznačně lokalizovatelné a tedy invariantní vůči měřítku. Těmto bodům je přiřazena orientace, aby byly invariantní i vůči rotaci. Pro každý bod v okolí jsou spočítány velikost gradientu:

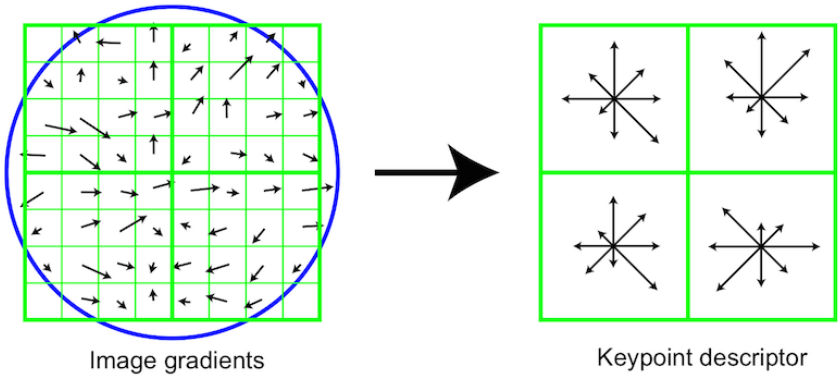
$$m(x,y)=\sqrt{(L(x+1,y)-L(x-1,y))^2-(L(x,y+1)-L(x,y-1))^2} \quad (3.11)$$

a orientace:

$$\theta(x,y)=\tan^{-1}\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)} \quad (3.12)$$

Následně je zkonstruován histogram orientací sestávající se z gradientů spočtených v okolí významného bodu.

Deskriptor se sestavuje z gradientů v blízkém okolí významného bodu. Body v okolí popisovaného významného bodu jsou rozděleny do oblastí, a pro každou takovouto oblast je spočítán histogram orientací. Získané histogramy jsou poté natočeny podle určené dominantní orientace významného bodu, čímž je zajištěna nezávislost na rotaci. Rozčlenění okolí do několika oblastí má výhodu v tom, že výsledný deskriptor je tak i odolný vůči malým posunům obrazu.



Obr. 3.12: princip tvorby deskriptoru SIFT [32]

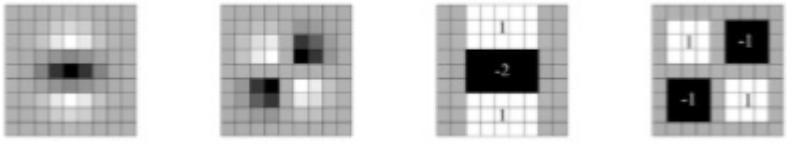
3.2.3 SURF

Další metoda kombinující detektor s deskriptorem. Autoři Herbert Bay, Tinne Tuytelaars a Luc Van Gool vycházeli z metody SIFT, ale s cílem urychlit výpočet. Odtud pochází název Speeded Up Robust Features.

Urychlení výpočtu mělo pomoci vytvoření detektoru, který nebude detekovat hrany. SIFT detektor totiž hrany detekuje a následně je odstraňuje. Detektor používá Hessovu matici kvůli dobrému poměru doby výpočtu a přesnosti, od toho se odvíjí jeho označení Fast-Hessian. Je definován následující maticí:

$$H(x,\sigma)=\begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix} \quad (3.13)$$

kde $L_{xx}(x,\sigma)$ je konvoluce druhé derivace Gaussiánu s obrazem I v bodě x. Místo druhé derivace Gaussiánu se používá její diskrétní aproximace čtvercovými filtry.



Obr. 3.13: vlevo druhé derivace Gaussovské funkce ve směru y a xy, vpravo jejich aproximace. bílá=1, černá=-1, šedá=0 [46]

Vyšší měřítka jsou generovány zvětšováním masky na 15x15, 21x21, 27x27, atd. Extrakce významných bodů je podobná metodě užité v SIFTu, také jsou porovnány body v okolí 3x3x3.

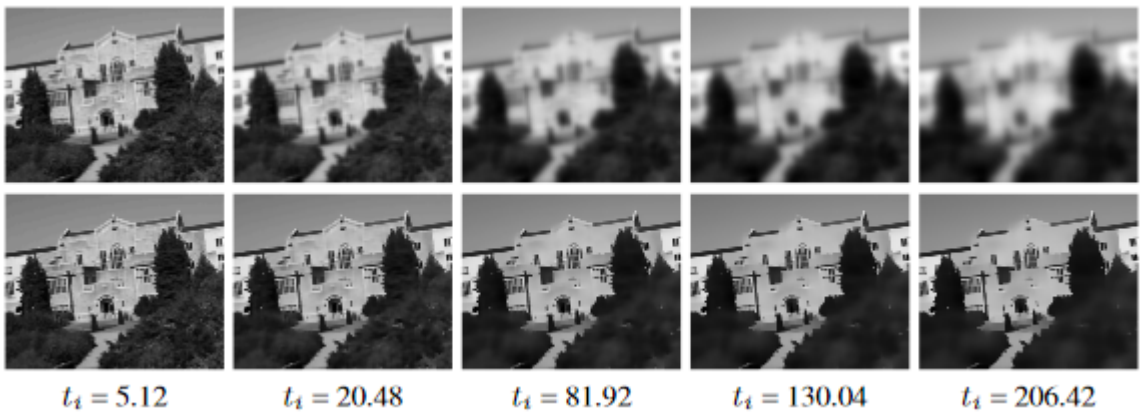
I určení orientace vychází z postupu SIFTu, ale zde je kvůli zrychlení uvažovaný pouze dominantní směr gradientu. Pro výpočet orientace se používá kruhové okolí bodu s poloměrem 6σ. Nejprve jsou spočítány odezvy Haarovy vlnky v směrech x a y. Ty jsou následně váženy Gaussovým filtrem o poloměru 2,5. Dominantní směr je odhadnut součtem všech odezev a následným zařazením do jednoho ze 6 směrů, Jednotlivé směry jsou proti sobě o $\pi/3$.

Poté je třeba vytvořit deskriptor. Okolo významného bodu je vytvořeno okno o velikosti 20σ s rotací odpovídající bodu, to je rozděleno na 16 5x5 oblastí, pro každý bod oblasti je spočítána odezva na Haarovu vlnku v obou směrech a ta je opět vážena Gaussovou funkcí. Pro každou oblast jsou sečteny všechny odezvy d_x, d_y, také $|d_x|$ a $|d_y|$, to vytváří 4 čísla pro každou oblast, 64 hodnot pro bod. Kvůli invariantnosti na kontrastu jsou hodnoty nakonec normalizovány.

3.2.4 KAZE

Podkapitola vychází z [44] a [45].

Poměrně nová kombinace detektoru a deskriptoru vyvinutá v roce 2012, jeho autory jsou Pablo F. Alcantarilla, Adrien Bartoli a Andrew J. Davison. Slovo „kaze“ pochází z japonštiny a znamená „vítr“. Myšlenka je detekovat a popsat klíčové body v nelineárně škálovaném prostoru. Protože obvykle používané Gaussovo rozmazání rozmazává hrany objektů, KAZE využívá nelineární difúzní filtrování spolu s metodou AOS (Additive Operator Splitting), která reguluje šum, ale ponechává hrany. Metoda je podobná už popsanému algoritmu SIFT.



Obr. 3.14: ukázka Gaussovského scale-space a nelineárního difúzního filtrování [43]

Algoritmus hledá body invariantní vůči rotaci posunu. Stejně jako SIFT vytvoří varinaty obrázku s různým měřítkem, ale místo rozdílu Gaussova scale-space se požívá nelineární difúzní filtrování spolu s funkcí:

$$c(x,y,t)=g(|\nabla\ L_\sigma(x,y,t)|) \tag{3.14}$$

kde L je původní obrázek a t je čas. Namísto rozdílu gaussiánů se v Kaze používá AOS a pro generování různých měřítek se používá původní obrázek. Klíčové body jsou určeny výpočtem odezvy normalizovaného determinantu Hessovy matice. Orientace bodů je podobná jako u SIFTu, ale

gradienty jsou reprezentovány jako body v vektorovém prostoru, nejdelší vektor je vybrán jako orientace daného bodu.

Deskriptor je M-SURF upravený pro nelineární scale-space. Oproti SURFu používá větší okno o velikosti 24σ rozdělené do 16 sekcí po 9x9, s přesahem 2σ.

3.2.5 MSER

Podapitola vychází z [8] a [36].

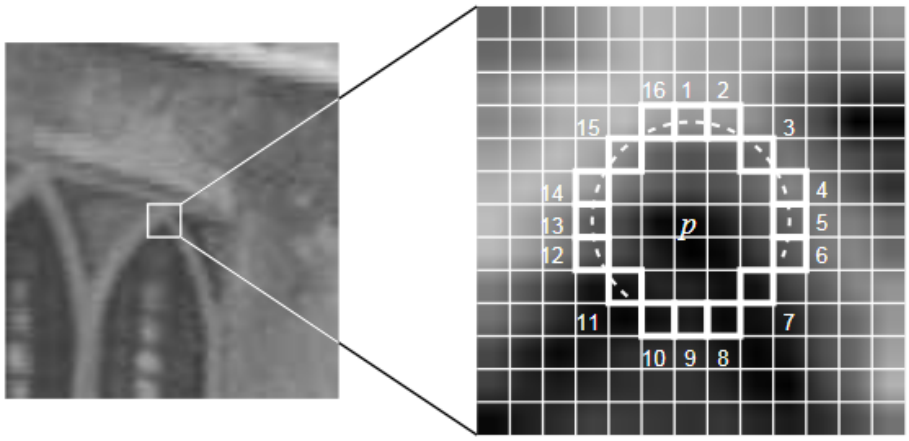
Tento detektor vznikl ve spolupráci ČVUT v Praze a University of Surrey v Guildfordu ve Velké Británii. Detektor MSER (Maximally Stable Extremal Regions) se od detektorů popisovaných v předchozích kapitolách liší tím, že se nezabývá významnými body, ale místo toho hledá v obraze významné oblasti, pro které se používá označení blob. Tyto oblasti ale také musí vykazovat vlastnosti podobné významným bodům.

Princip algoritmu spočívá v prahování s různými hodnotami prahu. Běžný šedotónový obraz zmiňovaný na začátku kapitoly má 256 stupňů šedi (hodnoty 0-255). Pokud je na tento obraz aplikováno prahování s hodnotou prahu 255, tak bude výsledek celý černý. Při nižších hodnotách prahu se na výsledných obrazech začnou objevovat bílé body a plochy, které se s dalším snižováním hodnoty prahu budou zvětšovat a spojovat, až nakonec získáme jako výsledek celý bílý obraz. Významné oblasti jsou takové, které jsou stabilní pro co nejvíce hodnot prahu. Existující řešení využívají ke kontrole neměnnosti oblastí porovnání jejich okrajů, těžiště nebo změnu elipsy v nich vepsané.

3.2.6 FAST

Podapitola vychází z [8] a [34].

Metoda Features from Accelerated Segment Test byla vyvinuta Tomem Drummondem a Edwardem Rostenem v roce 2006. Princip této metody spočívá v porovnání hodnot jasu zkoumaného bodu a bodů v jeho okolí. Tyto body se zkoumaným bodem přímo nesousedí, ale nacházejí se na kružnici o poloměru 3 pixely, jejíž středem je zkoumaný bod.



Obr. 3.15: body na kružnici pro porovnání v algoritmu FAST [34]

Cílem je zjistit, jestli alespoň dvanáct navzájem sousedících z celkového počtu šestnácti pixelů na kružnici má vyšší hodnotu intenzity jasu, než zkoumaný bod. Stejně tak může mít těchto dvanáct bodů intenzitu nižší. Jako další možnost pro konkrétnější výsledky lze určit hodnotu, které musí dosáhnout rozdíl mezi jasem zkoumaného bodu a bodu na kružnici, aby byl bod na kružnici považován za dostatečně rozdílné intenzity. Urychlení metody spočívá v tom, že nejsou testovány všechny pixely, ale pouze některé. Konkrétně byly zvoleny ty, které jsou na obrázku 3.9 označeny čísly 1, 5, 9 a 13. Pokud totiž má být splněna podmínka, že dvanáct libovolných sousedních pixelů má mít vyšší případně nižší intenzitu jasu, stejnou vlastnost budou vykazovat alespoň tři z těchto čtyř

bodů. Jestliže se toto nesplní, zkoumaný bod nemůže být rohem a lze přejít na další zkoumaný bod, což ušetří čas potřebný na zkoumání zbylých 12 bodů. V opačném případě je ale vhodné alespoň část ostatních bodů prověřit.

Výhoda algoritmu spočívá v jeho jednoduchosti a zároveň opakovatelnosti detekce bodu. Bohužel je velmi citlivý vůči šumu.

3.3 Deskriptor

Poté, co jsme našli významné body v obraze, je třeba je pro další výpočty tyto body zobecnit, obohatit je o opakovatelnost, robustnost, invariantnost vůči posunutí nebo změně světelných podmínek. K tomu slouží deskriptory. Úkolem deskriptoru je matematicky popsat tyto body. Některé řešení, jako třeba SIFT, který je již popsán v kapitole 3.2.2 v sobě spojují detektor a deskriptor.

3.3.1 FREAK

Informace v této kapitole pocházejí ze zdrojů [8] a [27].

Deskriptor Fast Retina Keypoint vznikl v roce 2012 na Ecole Polytechnique Fédérale de Lausanne (EPFL), ve Švýcarsku. Jeho autory jsou Alexandre Alahi, Raphael Ortiz a Pierre Vandergheynst. Jejich cílem bylo vytvořit rychlý, robustní a kompaktní deskriptor. Základem jejich práce byl starší deskriptor BRIEF (Binary robust independent elementary features). Ten je založen na porovnání intenzity zkoumaného bodu vůči bodům okolí. Dá se popsat vzorcem:

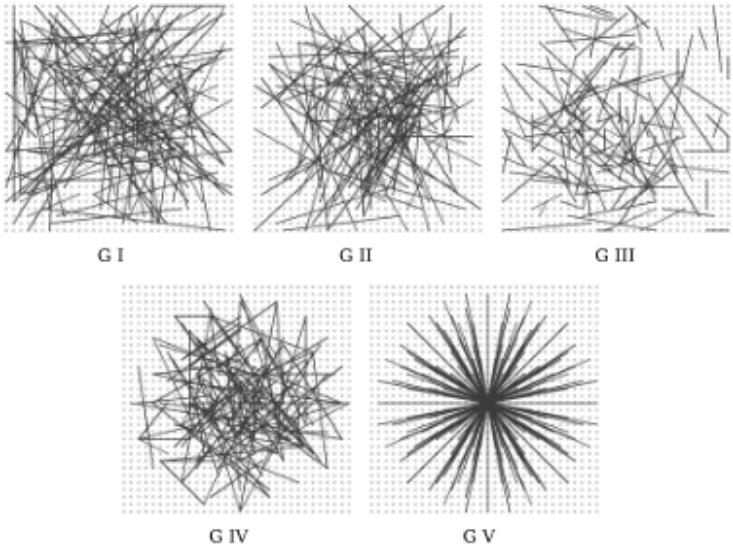
$$\tau(p,x,y)=\begin{cases}1 \dots pokud\ p(x)<p(y)\\ 0 \dots\dots\dots\dots\dots\dots jinde\end{cases} \tag{3.15}$$

kde p(x) je intenzita pixelu $x=(u,v)^T$. Následuje vytvoření deskriptoru ve tvaru bitového řetězce. Ten lze definovat následujícím vztahem:

$$f_{n_d}(p)=\sum_{1\leqslant i\leqslant n_d} 2^{i-1}\tau(p,x,y) \tag{3.16}$$

kde n_d může nabývat hodnot 128, 256 nebo 512, proto se používá označení BRIEF-*k*, kde *k* značí počet bytů nutných k uložení deskriptoru.

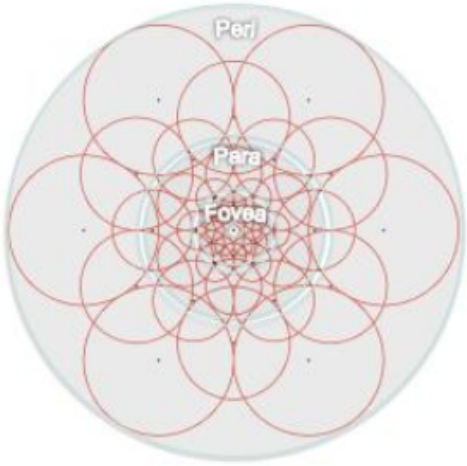
Před samotnou operací je nutné na obraz aplikovat vhodný filtr na redukci šumu a vybrat rozložení bodů p(y) pro porovnávání. Autoři toto rozložení vybírali na základě experimentu, kde vytvořili pět možných rozložení ukázaných na obrázku 3.10.



Obr. 3.16: rozložení vytvořená při tvorbě deskriptoru BRIEF [26]

Experiment ukázal, že rozložení G1 až G4 měly podobné výsledky, G5 bylo nejhorší.

Autoři deskriptoru FREAK se zaměřili na problém výběru správných bodů pro porovnání intenzity. Jejich řešení se zakládá na tom, že se ve své funkci snaží napodobit sítnici lidského oka. Odtud také pochází název tohoto deskriptoru (retina znamená sítnice).



Obr. 3.17: ilustrace vzorkovací mřížky pro deskriptor FREAK [27]

Vzorkovací mřížka napodobuje rozmístění retinálních ganglionových buňek na sítnici oka. Tedy používá kruhový tvar a hustota bodů od středu exponenciálně klesá.
Řetězec deskriptoru pak vzniká podobným způsobem, jako u deskriptoru BRIEF.

$$F = \sum_{0 \leqslant a \leqslant N} 2^a T(P_a) \tag{3.17}$$

kde P_a je dvojice vybraných oblastí, N je požadovaná délka deskriptoru a kde:

$$T(P_a) = \begin{cases} 1 \ldots pokud\ I(P_a^{r_1}) > I(P_a^{r_2}) \\ 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots jinak \end{cases} \tag{3.18}$$

přičemž $I(P^{r1}{}_a)$ je intenzita bodu po aplikování filtru na redukci šumu. Problémem je, že tento postup vede k situaci, kdy by se měly porovnat tisíce párů, přičemž většina z nich nejspíš nebude hrát velkou roli v popisu obrazu. Proto autoři využívají následující postup, který vybírá jen žádané množství párů s nízkou vzájemnou korelací.:

1. Sestavit matici D ze skoro padesáti tisíc významných bodů. Každý řádek odpovídá významnému bodu reprezentovanému deskriptorem sestaveným ze všech možných párů.
2. Spočítat průměr každého sloupce, průměr 0,5 je nejlepší výsledek, podle toho se seřadí sloupce a vybere se ten nejlepší. Iterativně se přidávají další sloupce s nízkou korelací.

Experimentem autoři zjistili, že nejrelevantnějších je prvních 512 párů, přidávání dalších už nezvyšuje výkon. Podobný princip používá i deskriptor ORB, který taktéž vychází z BRIEF. Tento deskriptor je předmětem kapitoly 3.3.3.

3.3.2 HoG

Podkapitola čerpá ze [8] a [33].
Histogram of Oriented Gradients vychází z myšlenky, že objekt lze popsat lokálním rozložení gradientů intenzity. Jeho prinicip spočívá v tom, že obraz se rozdělí na malé navzájem propojené oblasti, kterým se říká buňky. Pro každou buňku je vytvořen histogram orientovaných gradientů a právě tyto histogramy tvoří deskriptor. Pro výpočet gradientů se potom používá aplikace derivačních filtrů, tudíž velmi podobný postup jako při aplikaci Sobelova operátoru, který byl zmiňován v kapitole 3.1.2. Autoři tohoto algoritmu jako nevhodnější určili následující jádra[33]:

-1	0	1

1
0
-1

Obr. 3.18: matice Sobelova operátoru použitá pro HoG

Následně je určena amplituda a orientace gradientu, a z těchto znalostí lze pro každou buňku sestavit histogram. Výhodou tohoto deskriptoru je jeho invariantnost vůči posunům a změnám jasu, bohužel není invariantní vůči otáčení.

3.3.3 ORB

Zdroji pro podkapitolu jsou [8] a [35]
ORB je další z kombinovaných řešení deskriptoru a detektoru. Jeho název je zkratka Oriented FAST and Rotated BRIEF, z čehož vyplývá, že jde o kombinaci a modifikaci detektoru FAST popsaného v podkapitole 3.2.4 a desriptoru BRIEF, který byl popsán v první části podkapitoly 3.3.1. Cílem autorů bylo vytvořit výpočtově méně náročnou alternativu k SIFT, který je popisován v podkapitole 3.2.2. Tento přístup měl přinést schopnosti SIFTu na méně výkonná zařízení, například chytré telefony, a také do rozpoznávání obrazu v reálném čase.

Spolu s algoritmem FAST je zde využit i další detektor, Harrisův. Ten slouží k vybrání nejvhodnějších významných bodů detekovaných algoritmem FAST. Pro nalezení N významných bodů se FAST nastaví tak, aby nalezl více než N bodů, a následně je použit Harrisův detektor, podle jehož výsledků se body detekované algoritmem FAST seřadí podle jejich „rohovosti“ (v [35] „cornerness“), a horních N bodů jsou vybrané významné body. Pro nezávislost na měřítku je vytvořena pyramida různých měřítek zkoumaného obrazu, kde jsou pro každý stupeň detekovány významné body výše popsaným způsobem. Pro nezávislost na rotaci je využita technika intenzity těžiště. Ta předpokládá, že intenzita těžiště je posunuta mimo svůj střed a tento vektor lze použít k určení orientace. Pomocí následujících vzorců se vypočítá těžiště C a úhel orientace θ:

$$M_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{3.19}$$

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right) \tag{3.20}$$

$$\theta = \arctan 2(m_{01}, m_{10}) \tag{3.21}$$

Následně je aplikován deskriptor BRIEF popsaný v první části kapitoly 3.3.1 s výběrem párů s nízkou s nízkou vzájemnou korelací, jehož princip je popsán na konci stejné kapitoly.

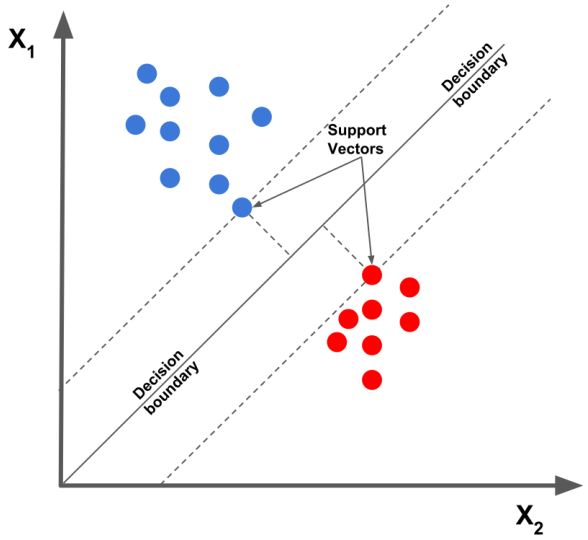
3.4 Klasifikátor

Při klasifikaci je úkolem zařadit nový vzorek do jedné nebo více kategorií na základě trénovacích dat. Trénovací data jsou vzorky, u nichž je známa jejich kategorie. Jedná se tedy o strojové učení s učitelem. Klasifikátor je algoritmus, který implementuje řešení tohoto problému.

3.4.1 Support vector machines

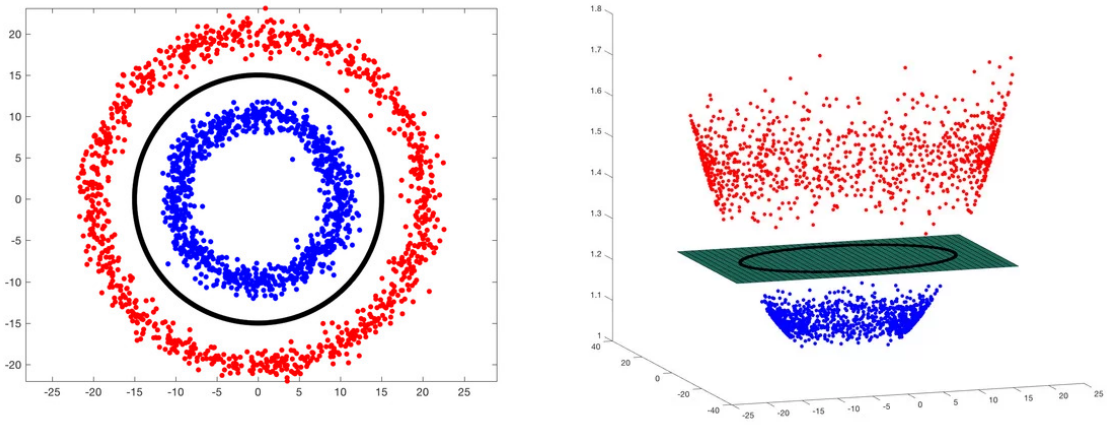
Podkapitola vychází z [38], [39], [40] a [41].
Support vector machines neboli metoda podpůrných vektorů je metoda, jejíž základem je lineární klasifikátor do dvou tříd. Základní myšlenkou je nalezení nadroviny, která rozdělí n-dimenzionální data na dvě třídy ideálně tak, že se trénovací data různých kategorií nacházejí na protějších stranách této nadroviny. Nadrovina je podprostror s dimenzí o 1 nižší, než je dimenze prostroru. Pro jednodimenzionální prostror je to bod, pro dvoudimenzionální prostor přímka, pro trojdimenzionální plocha a tak dále. Optimální nadrovina je taková, která má co největší minimální vzdálenost od bodů

na obou stranách. Kolem nadroviny tak vede pásmo, ve kterém se nenacházejí žádné body. Body, které leží na okraji tohoto pásma se nazývají *podpůrné vektory* a popisují danou nadrovinu, odtud také pochází název metody.



Obr. 3.19: ukázka funkce SVM [38]

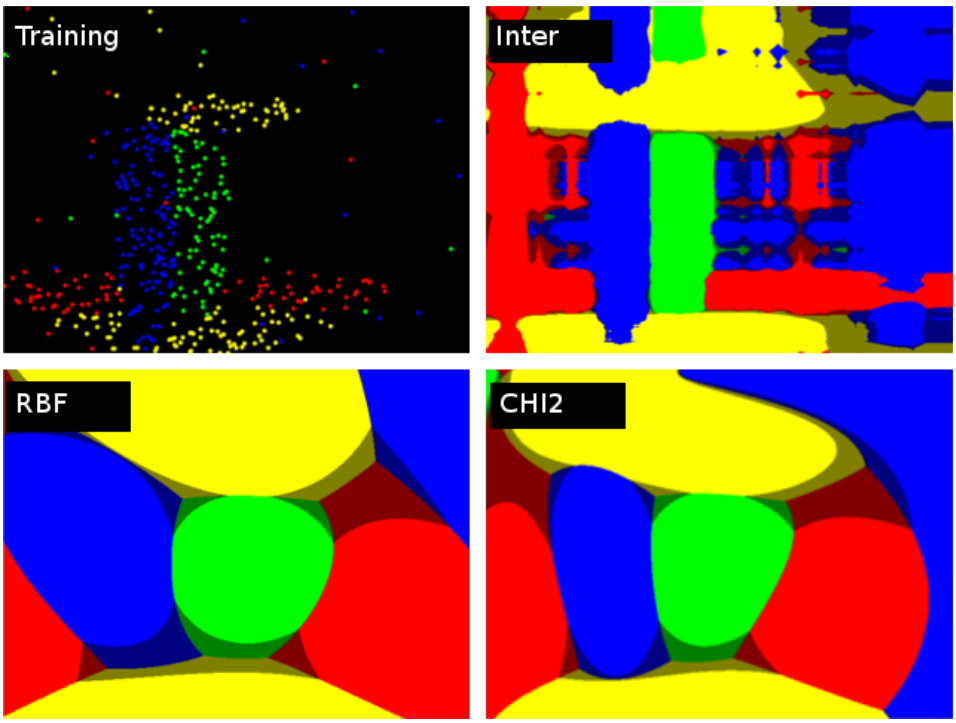
Příklad na obrázku 3.12 je lineárně separabilní, což je vcelku jednoduše řešitelný problém. Na řešení lineárně neseparabilních úloh využívá SVM takzvaný „kernel trick“. Jeho principem je rozšíření prostoru o další dimenzi tak, aby bylo možné prostor lineárně rozdělit.



Obr. 3.20: příklad lineárně neseparabilních dat a jejich rozdělení s pomocí kernel tricku [38]

V tomto případě byla pro výpočet hodnoty nové dimenze použita funkce:

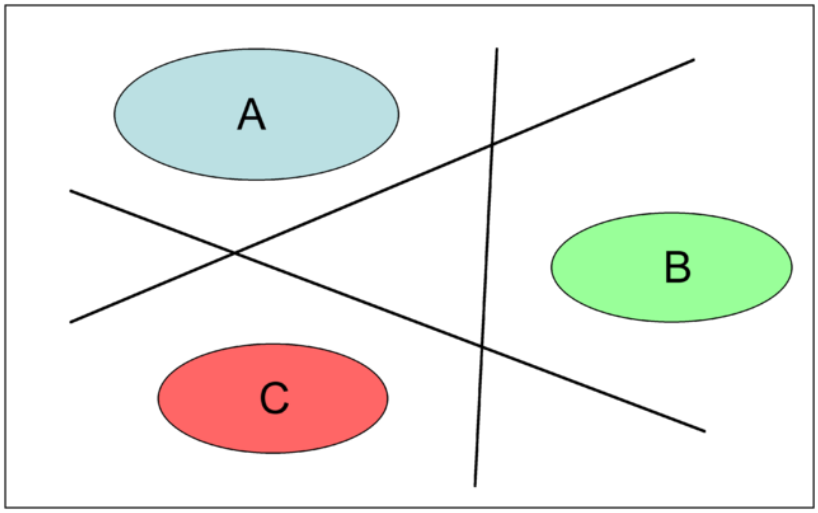
$$z = e^{-\gamma\left(x^2+y^2\right)} \tag{3.22}$$



Obr. 3.21: porovnání jader pro SVM v knihovně OpenCV [39]

Jak již bylo řečeno, základní SVM klasifikátor dokáže klasifikovat pouze do dvou tříd, typicky na pozitivní a negativní výsledek. Existuje několik přístupů, jak s pomocí SVM vytvořit klasifikátor do více kategorií.

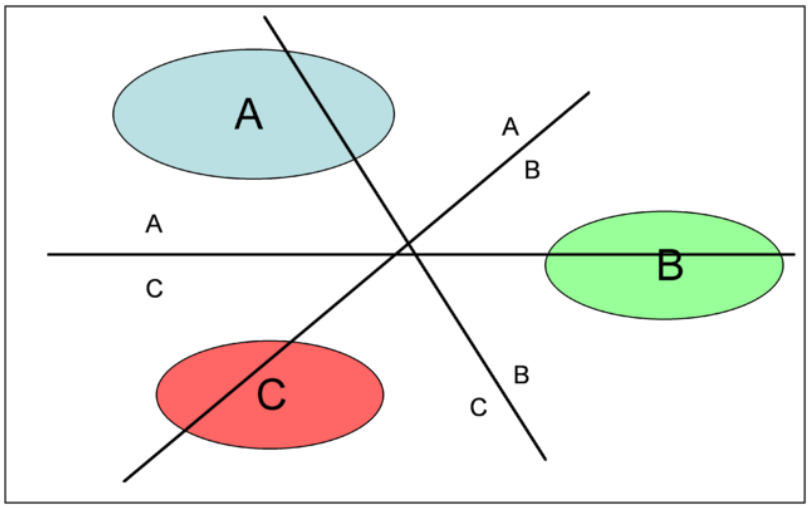
Metoda „Jeden proti všem“ (One-against-all) pro klasifikaci do k kategorií spočívá ve vytvoření k SVM klasifikátorů, z nichž každý dostane trénovací data jedné kategorie jako pozitivní a zbytek dat jako negativní výsledky. Prvek je vyhodnocen jako náležící do kategorie, pokud ho klasifikátor s odpovídající kategorií vyhodnotí jako pozitivní a ostatní klasifikátory jako negativní. Tento přístup může vcelku dobře fungovat v prostorech, kde jsou shluky testovacích dat blízko u sebe, ale v prostoru vzniká místo, které nenáleží k žádné kategorii



Obr. 3.22: Rozdělení prostoru při použití metody "Jeden proti všem" plocha uprostřed trojúhelníku nenáleží žádné kategorii [40]

Poměrně používaná úprava metody „Jeden proti všem“ spočívá v použití spojitých rozhodovacích funkcí SVM, zkoumaný prvek pak náleží do té kategorie, která má nejvyšší hodnotu rozhodovací funkce, tedy ke kterému shluku je nejblíže. Díky tomu jsou všechny prvky někam zařazeny.

Další metodou je „Jeden proti jednomu“. Ta spočívá v porovnání jednotlivých dvojic trénovacích kategorií proti sobě. Pro k kategorií je vytvořeno *k*(k-1)/2* SVM klasifikátorů tak, že každý klasifikátor dostane jednu z možných dvojice kategorií. Existuje několik možností výběru, do které kategorie zkoumaný prvek náleží, jednodušší řešení vyberou kategorii, do které byl prvek zařazen nejvícekrát, pokročilejší metody mohou určovat kategorii například formou podobnou vyřazovacímu systému.



Obr. 3.23: Rozdělení prostoru při použití metody „Jeden proti jednomu“[40]

Zvlášní variantou této metody je DAGSVM (Directed Acyclic Graph Support Vector Machines), která ve cvičící fázi pracuje stejně jako výše popsaná metoda, a v testovací fázi používá binární strom (odtud název metody) s *k*(k-1)/2* vnitřními uzly a *k* listy, přičemž každý uzel odpovídá jednomu z SVM klasifkátorů a list kategorii. Testování prvku pak odpovídá průchodu stromem, kde při vstupu do uzlu je vypočítána rozhodovací funkce a na základě výsledku je volena další cesta, nalezený list je řešení.

3.4.2 Matcher

Kapitola vychází z informací v [52].
Úkolem matcheru je najít podobnosti mezi dvěma obrázky, přesněji řečeno mezi jejich klíčovými body, které byly detekovány na obou obrázcích některým z detektorů, jejichž zástupci byli uvedeni v kapitole 3.2. Tyto klíčové body rovněž musí být popsány deskriptorem, jejichž příklady jsou uvedeny v kapitole 3.3. Právě od deskriptoru se odvíjí i výběr a nastavení matcheru. Knihovna OpenCV implementuje dva typy matcheru, a sice Brute-Force Matcher a FLANN matcher.

Brute-Force matcher je velmi jednoduchý. Vstupem mu jsou dva sety deskriptorů, a tento matcher pro každý z deskriptorů prvního setu projde všechny deskriptory druhého setu a přiřadí mu ten, který je mu nejbližší podle zvoleného výpočtu vzdálenosti (distance measurement). Zde je totiž nutno poznamenat, že obecně lze rozdělit deskriptory na dvě skupiny podle hodnot výsledku. Jsou zde floating point deskriptory, jako je SIFT, SURF nebo KAZE, a binární deskriptory, což jsou například BRIEF, FREAK nebo ORB. Pro binární deskriptory se pro tento výpočet používá Hammingova vzdálenost. Hammingova vzdálenost dvou posloupností značí na kolika pozicích se vzájemně liší. Například následující binární řetězce *11010111001* a *10010101101* mají vzájemnou

Hammingovu vzdálenost 3, protože se od sebe liší na třech pozicích (2., 7. a 9. znak zleva). Floating Point deskriptory pak používají k výpočtu Euklidovskou vzdálenost.

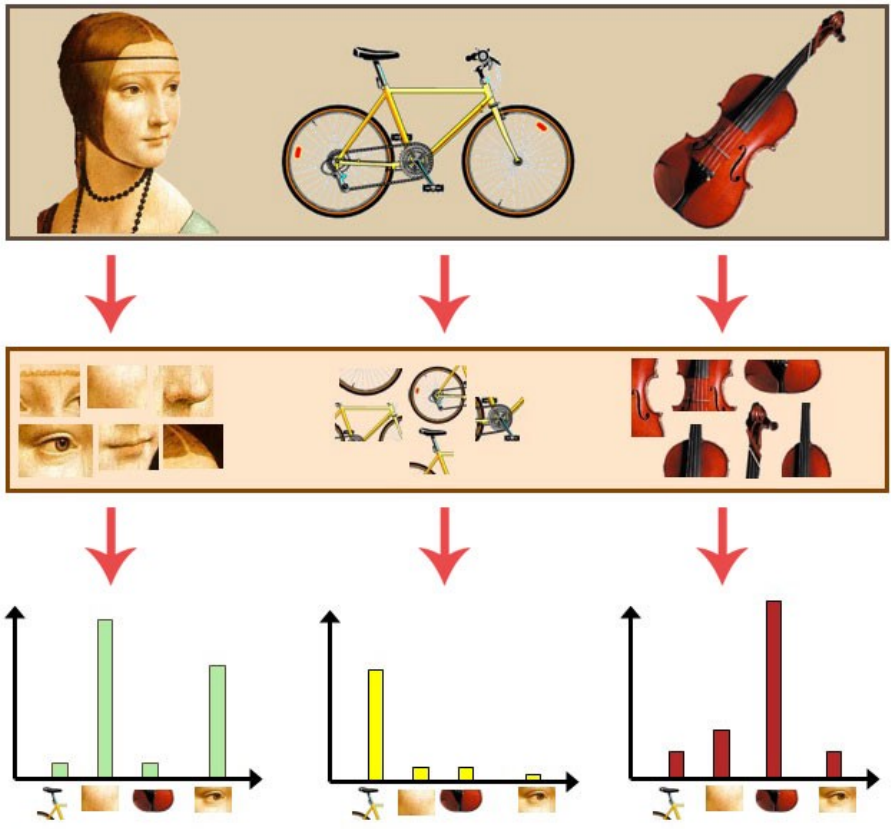
FLANN (Fast approximate nearest neighbor) obsahuje soubor několika algoritmů pro hledání nejbližšího souseda ve vysokodimenzionálních prostorech a systém, který na základě datasetu vybírá nejvhodnější z těchto algoritmů i jeho optimální parametry.

3.4.3 Bag of visual words

Podkapitola vychází z [42] a [43].
Bag of words je metoda zprvu používána zejména na zpracování přirozené řeči, například pro klasifikaci dokumentů, nicméně v obměněné variantě se používá i pro počítačové vidění.

Princip metody spočívá ve vytvoření slovníku, což je seznam všech možných slov. Každý text jde popsat jako na základě slovníku tak, že je vytvořen vektor, který označuje, kolikrát se dané slovo slovníku v textu objevuje. Nezávisí na posloupnosti slov ve větě, od toho se odvíjí název metody. Podle použitých slov a jejich frekvence lze vytvořit histogramy slov a následně texty vyhodnocovat a třídit. Pouhé počty jednotlivých slov je ale obvykle třeba nějak interpretovat, problémem mohou být například v angličtině slova jako „the“, „a“ nebo „to“, které jsou v textu velmi časté, ale nemají žádnou důležitost pro klasifikaci.

Pro počítačové vidění se používá obdobná myšlenka, akorát místo slov jsou použity prvky obrázků. Proto se pro tuto metodu používají názvy „Bag of visual words“ a „Bag of features“. Tyto vlastnosti se zpravidla získávají pomocí detektorů a deskriptorů, kterým byly věnovány části kapitoly 3.2 a 3.3. Pro zobecnění se z těchto deskriptorů vytvoří shluky, střed shluku bude sloužit jako položka slovníku. Pro shlukování se používají běžné shlukovací algoritmy jako K-means nebo třeba DBSCAN. Databázi pro klasifikaci obrazu pak tvoří histogramy četností prvků ze slovníku pro jednotlivé obrázky v databázi.



Obr. 3.24: Ilustrace principu Bag of visual words [42]

3.4.4 K-means a K-Majority

OpenCV používá ve své implementaci Bag of Words shlukování pomocí K-means. Tento algoritmus je jeden z nejjednodušších metod strojového učení bez učitele. Jeho úkolem je rozdělit vstupní množinu dat do *K* shluků (*K* je známé před začátkem výpočtu) podle určitých podobností mezi jednotlivými prvky. Algoritmus vybere *K* počátečních středů shluků a každý prvek přiřadí do právě jednoho nejbližšího shluku. Obvykle se „nejbližším" shlukem myslí ten, který má od daného prvku nejmenší Euklidovskou vzdálenost, ale pro různé potřeby lze použít i jiné vlastnosti prvků. Poté, co jsou takto vytvořeny shluky, je třeba přepočítat jejich středy, tak aby byly skutečně ve středu shluku. Po přepočítání středů je dalším krokem znova prověřit všechny prvky, zda některý není blíž jinému středu, než tomu svého současného shluku. Pokud se tak stane, prvek se přesune do shluku, jehož středu je nyní nejblíž. Pokud během prověřování prvků došlo alespoň k jedné takovéto změně, je třeba opakovat přepočítání středů a prověřit příslušnosti prvků k shlukům. Pokud k žádné změně nedošlo, shlukování je dokončeno. Obvykle je ještě u výpočtu možnost omezit počet cyklů, po kterých je algoritmus ukončen, i když shluky nemusejí být ještě stabilní. [53]

Jak již bylo zmíněno, tento algoritmus lze použít pro shlukování v případě floating point deskriptorů, ale binární deskriptory potřebují jiný typ vzdálenosti. Pro ty je určen například algoritmus K-majority, který z K-means vychází. Při výpočtu vzdálenosti od středu je používána Hammingova vzdálenost a přepočítávání středu shluku funguje tak, že jsou postupně procházeny bity všech prvků v shluku. Nový *i*-tý bit středu shluku bude takovou hodnotu, jakou má *i*-tý bit většiny prvků ve shluku, odsud pochází *majority* (anglicky většina) v názvu algoritmu. Pokud dojde k tomu, že jsou počty prvků ve shluku s 1 a s 0 na *i*-tém bitu vyrovnané, je nový *i*-tý bit středu vybrán náhodně.

4 Návrh řešení

V této kapitole bude určen potřebný hardware a software, na kterém bude práce vytvořena a testována. Poté se bude věnovat návrhu postupu od předzpracování až po vyhodnocení výsledku. Kapitola bude vycházet z poznatků v kapitolách 2 a 3.

4.1 Použitý hardware a software

Jako první krok je nutné vybrat vhodné nástroje pro řešení daného problému, a také zvolit programovací jazyk. Tento výběr může velmi ovlivnit složitost dosažení výsledku i jeho výsledky.

Pro implementaci bude použitý programovací jazyk C++ v kombinaci s knihovnou OpenCV verze 3.4.5, která nabízí mnoho funkcí nejen pro zpracování a rozpoznávání obrazu. Implementace bude provedena jako projekt SDK Microsoft Visual Studio 2017, protože velmi usnadňuje jak linkování s knihovnou OpenCV, tak i celkový vývoj a ladění projektu.

K testování bude použitý notebook Lenovo B70 se dvoujádrovým procesorem intel i5-5200U 2,20 GHz a pamětí RAM 12 GB s grafickou kartou Intel HD Graphics 5500, a s nainstalovaným 64bitovým OS Windows 7 Professional.

4.2 Postup řešení

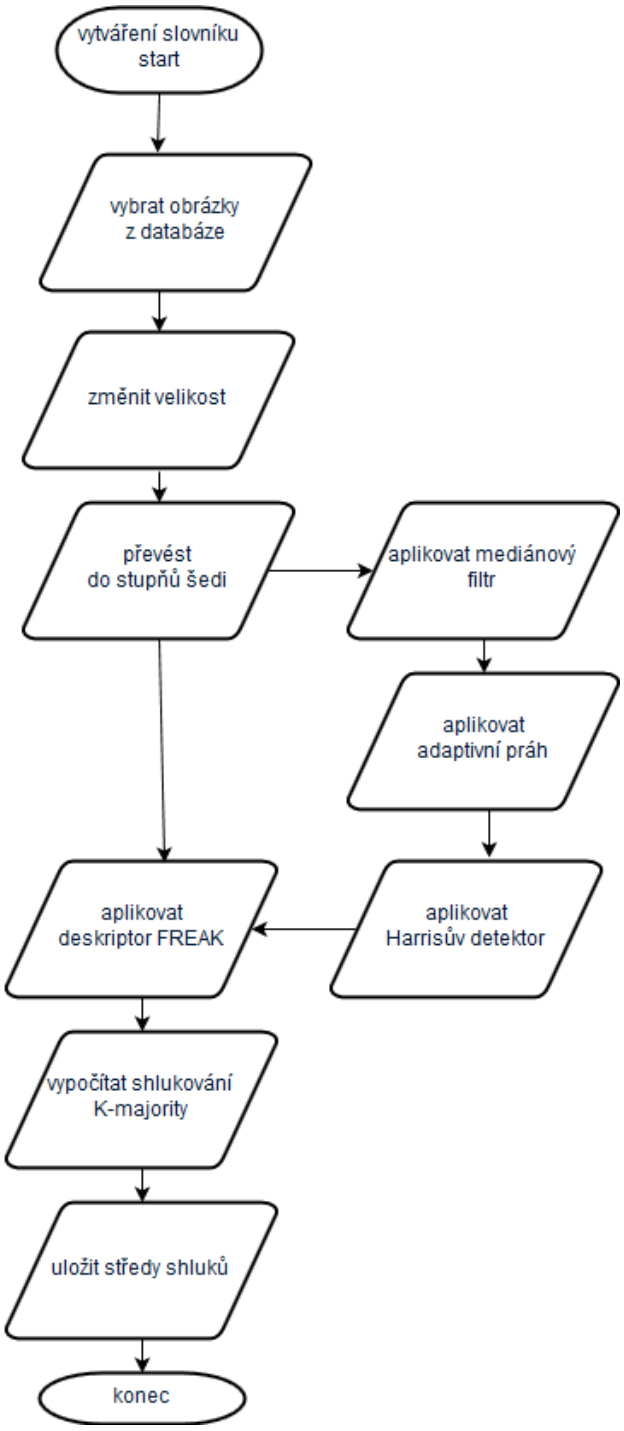
Inspirací pro řešení mi byla práce [29], která se ale zabývá detekcí zbraně v obraze a to pouze pistolí.

V rámci předzpracování obrazu budou použity redukce šumu pomocí mediánového filtru a převod do stupňů šedi. Pro účely detekce hran bude na obraz použité adaptivní prahování a na takto přizpůsobený obraz následně aplikován Harrisův detektor. Vstupem pro deskriptor FREAK bude ale zase původní obrázek pouze převedený do stupňů šedi, Protože prahováním se z původního obrázku

ztratí poměrně velké množství informací, které by pro deskriptor mohly být užitečné. Takto budou předzpracovány vstupní obrázek, obrázky při vytváření slovníku i databáze pro SVM.

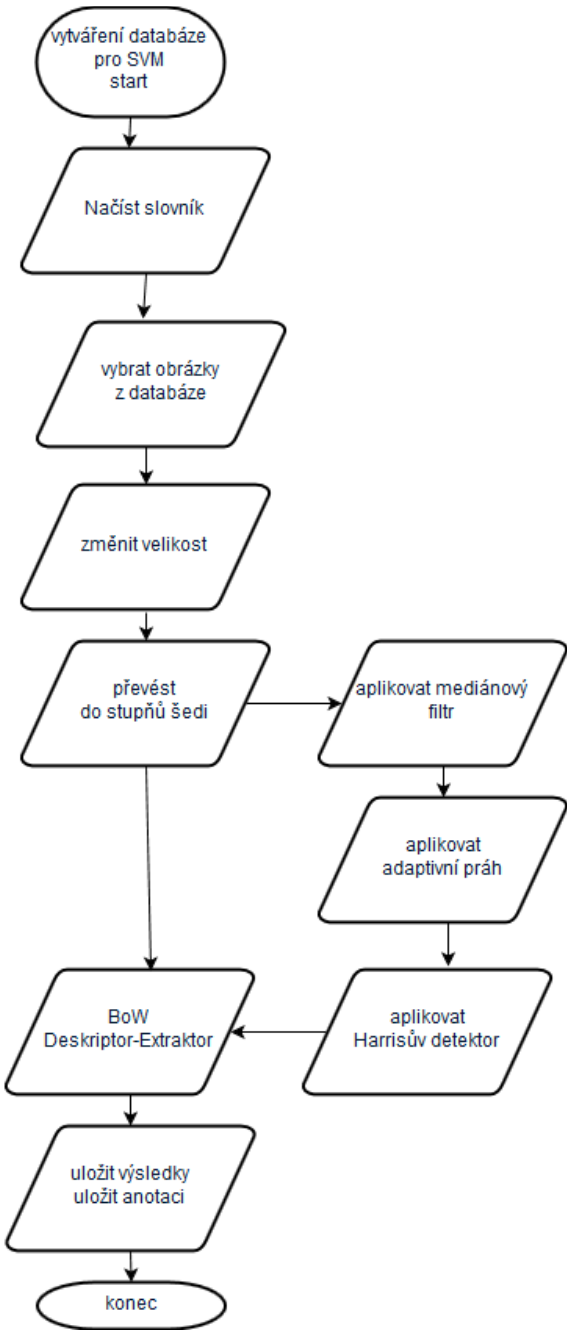
Detektor významných bodů by měl zejména najít rohy zbraně na obraze. Tyto rohy definují tvary zbraně a některé znaky zmiňované v kapitole 2.2, jako jsou zásobníky, mechanismy iniciace a nábojové válce, které vedou k určení kategorie. Od tvaru zbraně se zas odvíjí délka zbraně. Krátké zbraně mají menší rozdíl mezi délkou a výškou/šířkou, u dlouhých zbraní je délka naprosto dominantní rozměr, který může i několikanásobně přesahovat ostatní dva. Do slovníku by se měly dostat znaky, které se objevují na více obrázcích, a ty by tedy nějak měly charakterizovat větší skupinu. Dá se tedy očekávat, že se do slovníku dostanou nějaké části, které charakterizují, opět by se mohlo jednat o znaky pomáhající určit kategorii, například spojení mezi dvěma hlavněmi víceranové zbraně.

Před samotným určením je třeba mít připravenou databázi určených příkladů pro SVM klasifikátor. Napřed je třeba vytvořit slovník. Program vybere z každé kategorie náhodně obrázky, aplikuje na ně předzpracování, Harrisův detektor a deskriptor FREAK, jak bylo popsáno v předchozích odstavcích. Nad těmito výsledky bude provedeno shlukování. K tomu bude použit algoritmus K-majority, který je odvozený od K-means, ale uzpůsobený pro použití s binárními deskriptory. Výsledky shlukování, tedy středy jednotlivých shluků budou uloženy do souboru, aby šly použít při opětovném spuštění a nemusely být pokaždé vytvářeny znovu.



Obr. 4.1: Diagram postupu tvorby slovníku

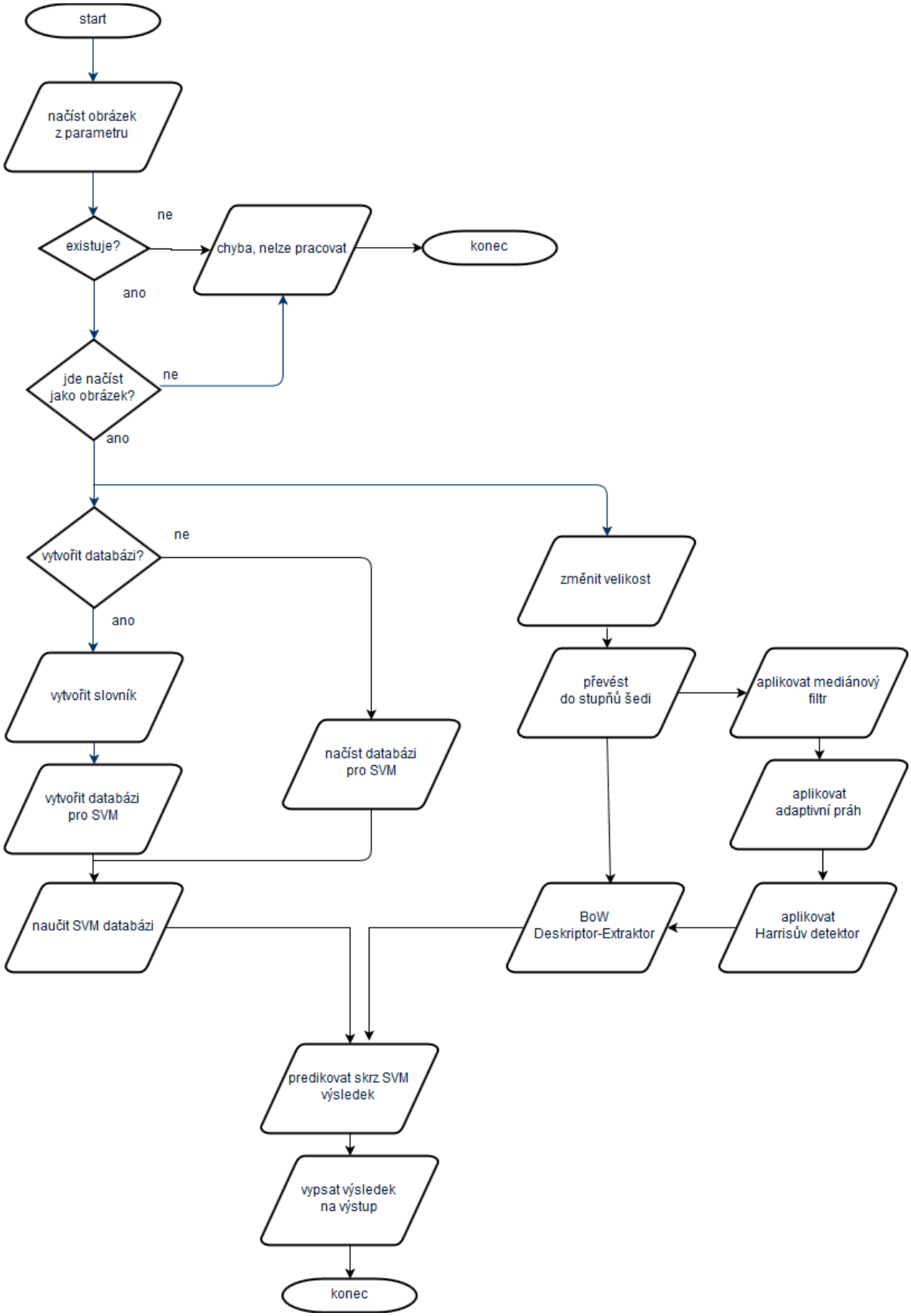
Samotná databáze je pak vytvořena jako výsledky porovnání většího počtu databázových snímků proti centroidům ve slovníku pomocí Matcheru. Databáze je po vygenerování taktéž uložena do souboru pro pozdější využití.



Obr. 4.2: Diagram postupu tvorby databáze pro SVM

Podobnost mezi řetězci vytvořenými deskriptorem na zkoumaném obraze a referenci v databázi se určí pomocí matcheru implementovaného v třídě BruteForceMatcher v OpenCV. Deskriptor FREAK je binární deskriptor, takže jako distance measurement bude použita Hammingova vzdálenost. Pomocí Matcheru bude vstupní obrázek porovnán s slovníkem Bag of words a na základě výsledků pak bude pomocí SVM predikována kategorie zbraně na zkoumaném obraze.

SVM tedy dostane anotované porovnání databázových snímků vůči slovníku a porovnání zkoumaného snímku vůči stejnému slovníku, na základě toho rozhodne, do které kategorie zkoumaný snímek patří.



Obr. 4.3: Diagram celého návrhu

5 Implementace

Celý projekt byl tvořen s důrazem na modularitu a znovupoužitelnost jednotlivých funkcí. Soubor *Source.cpp* obsahuje pouze funkci *main()*, která při běhu volá jednotlivé funkce implementace z oddělené knihovny *GR_Lib.h*. Další přítomné soubory pak implementují převzaté algoritmy pro Deskriptor FREAK popsaný v kapitole 3.3.1 a BoWKmajorityTrainer implementující metodu Bag of Words s použitím shlukovacího algoritmu K-majority.

Program se spouští s parametrem s cestou k obrázku, který bude klasifikován. Napřed je prověřeno, že je tento parametr přítomen a že vede k souboru, který dokáže otevřít jako obrázek. Uživatel je poté tázán, zda chce vygenerovat novou databázi, nebo použít již vytvořenou. Toto řešení bylo zvoleno hlavně pro debugování na Visual Studiu, kde je jednodušší mít stále stejné předem nastavené parametry spouštění. Pro běžné užití by tento dotaz šel řešit i volitelným parametrem při spuštění.

5.1 Předzpracování

Funkce *change_size_gray(Mat img)* a *Preprocess(Mat img)* jsou implementovány pomocí funkcí knihovny OpenCV pro úpravy obrazu.

Funkce *change_size_gray(Mat img)* převede vstupní obrázek do stupňů šedi a změní velikost tak, aby delší strana obrázku měla 250 pixelů, kvůli sjednocení jednotlivých obrázků a snížení nároků na výpočet v případě, že vstupní obrázek má vysoké rozlišení. Také malé obrázky budou zvětšeny, což přispěje k jednotnosti databáze.

Funkce *Preprocess(Mat img)* pak na vstupní obrázek aplikuje mediánový filtr a adaptivní práh. Zatímco výsledek funkce *change_size_gray(Mat img)* bude nadále používán i pro výpočet deskriptoru, výsledek *Preprocess(Mat img)* je pouze pro upřesnění práce detektoru. Principům těchto funkcí se věnovala kapitola 3.1. Obě funkce jsou implementovány v souboru GR_Lib.h.

5.2 Harris detektor a FREAK deskriptor

Funkce *detect(Mat input)* implementuje Harrisův detektor popisovaný v kapitole 3.2.1. Funkce nejprve použije Sobelův operátor k vypočítání derivací, poté spočítá snímky s I_x^2 , I_y^2 a I_xI_y, k tomu jsou použity funkce *pow* a *multiply* knihovny OpenCV. Na každý z těchto výsledků je následně použito Gaussovské rozostření a spočítáno R podle vzorce 3.8. Výsledek je normalizován do hodnot 0-255, aby se s ním dalo jednodušeji pracovat, podobně jako s obrázkem.

Úkolem funkce *Mat2KP(Mat map, int thresh)* je převést výsledek funkce *detect(Mat input)* do tvaru očekávaného pro *FREAK*, což je vektor struktur typu Keypoint. Prochází matici *map* a ukládá body, které mají hodnotu vyšší než *thresh* do vektoru keypointů s velikostí 1. Tento vektor je funkcí vrácen. Funkce detect() je rovněž implementována v souboru GR_Lib.h.

Výhodou této implementace oproti využití detektoru v knihovně OpenCV je zejména možnost adaptovat některé parametry dle aktuální potřeby, jako velikosti filtrů používaných pro výpočet. Díky tomu dokáže tento detektor detekovat i více zaoblené rohy, u čehož byl s detektorem knihovny OpenCV problém.

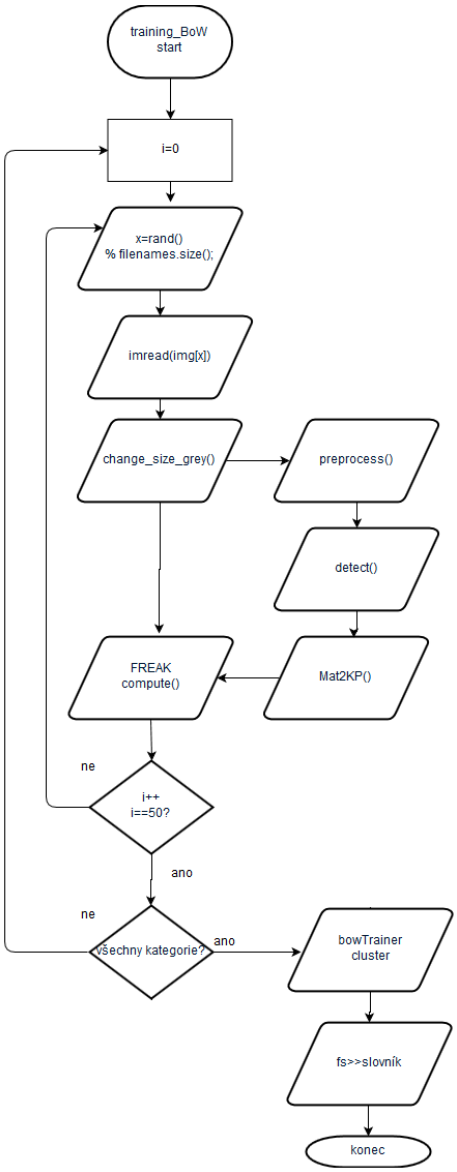
Celou třídu *FREAK* implementovali v rámci [27] autoří práce A. Alahi, R. Ortiz, a P. Vandergheynst. Je volně dostupná na GitHubu[9]. Funkce compute vypočítá pro vstupní vektor klíčových bodů na základě vstupního obrázku deskriptor FREAK podle popisu v kapitole 3.3.1.

9 https://github.com/kikohs/freak

5.3 Tvorba slovníku

Při generování databáze je třeba napřed vytvořit slovník. To je zajištěno funkcí *training_BoW(int thresh)*, kde parametr *thresh* je hodnota, která bude předávána funkci aplikující Harrisův detektor. *Training_BoW* vrací 0, pokud nebyly zaznamenány žádné problémy, při chybách vrací -1. Tato funkce je specificky vytvořená pro tento projekt a zaštiťuje užití obecnější funkce *training_BoW_byOne(string locationFrom, int thresh, Mat *vocab, int num_img),* která vypočítá deskriptory pro obrázky uložené ve složce.

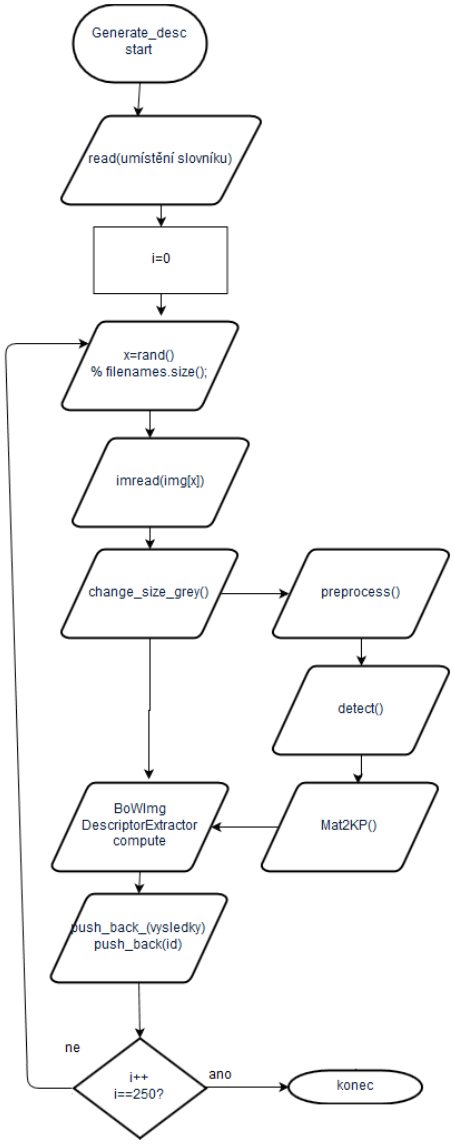
LocationFrom je cesta ke složce se soubory databáze, *thresh* je opět hodnota prahu, **vocab* je ukazatel na matici, ve které budou ukládány vygenerované data a *num_img* je počet obrázků, kolik má funkce na generování slovníku použít. Tato funkce náhodně vybere z dané složky daný počet obrázků, každý obrázek předzpracuje podle kapitoly 5.1, aplikuje na něj Harrisův detektor, vypočítá deskriptor a výsledek uloží do matice, vrací 0 při úspěšném průchodu, -1 pokud došlo k chybě.



Obr. 5.1: Diagram funkce Training_BoW

Funkce *training_BoW* tedy volá *training_BoW_byOne* pro vygenerování dat z osmi složek obrázků ve složce „guns“ umístěné v kořenovém adresáři projektu, a tyto data předá třídě BOWKMajorityTrainer, která nad ní provádí shlukování na základě algoritmu k-majority popsaného v kapitole 3.4.3. Tato třída je implementována v souboru BOWKmajorityTrainer.cpp a je také dostupná na GitHubu[10]. Po úspěchu této operace jsou vygenerovaná data slovníku uložena do souboru bowvocab.yml ve formě matice, která má 400 řádků (počet slov ve slovníku) a 64 sloupců (délka deskriptoru).

5.4 Tvorba dat databáze



Obr. 5.2: diagram funkce generate_desc()

Program pokračuje voláním funkce *generate_desc_ForAll(int thresh, int num),* která generuje databázi pro SVM na základě slovníku BoW. Tato funkce má strukturu velmi podobnou již popsané

10 https://github.com/berak/opencv_smallfry/tree/master/bow

training_BoW, tedy volá obecnější *generate_desc(string locationFrom, int thresh, int grp_num, int num_img, Mat *descr, Mat *classes),* která načte slovník vygenerovaný v předchozím kroku, náhodně vybírá *num* obrázků z vybrané složky, každý z těchto obrázků předzpracuje, aplikuje Harrisův detektor a na základě deskriptoru FREAK a Brute-Force matcheru s použitím Hammingovy vzdálenosti vygeneruje data databáze a její identifikaci pro SVM. K tomu je použita třída *BOWImgDescriptorExtractor* z knihovny OpenCV. Po vygenerování dat ze všech složek jsou data uložena do souboru database.yml. Obsahem souboru jsou dvě matice, matice iD obsahuje výsledky porovnání středů shluků ze slovníku s obrázky databáze pomocí Brute-Force matcheru, jednořádková matice iDL pak obsahuje označení, do které skupiny obrázek databáze patří.

5.5 Rozpoznávání

Funkce *Load_iD(string file)* a *Load_iDL(string file)* načítají ze souboru *file* data databáze a jejich identifikaci do matic. Tento soubor vznikl v předchozích krocích. Načtené matice jsou připraveny pro SVM implementovanou v knihovně OpenCV. To je v následující části vytvořeno, jsou nastaveny jeho parametry a dostane načtenou databázi. Je zvolen vysoký důraz na přesnost určení a jádro INTER (viz. Obr. 3.18). Následně je zde provedeno předzpracování vstupního obrazu, aplikace detektoru již známými funkcemi *change_size_gray(), Preprocess(), detect(),* výpočet deskriptoru a matcheru ve funkci *compute* třídy *BOWImgDescriptorExtractor* podobně, jako byly vytvořeny položky v databázi, které se SVM učí ve funkci *train()*. SVM má za úkol v této funkci najít rozdělení trénovacích dat databáze podle jejich příslušnosti ke kategoriím. Implementace SVM v knihovně OpenCV využívá metodu „jeden proti jednomu“. Poté SVM zařadí zkoumaný obrázek do jedné z kategorií ve funkci *predict()* tak, že ho přidá do prostoru a jelikož byl tento prostor již ve funkci train() rozdělený do kategorií, SVM vrátí kategorii, ve které se zkoumaný prvek objevil. Protože výstup SVM je číselný, program na základě tohoto čísla vypíše textové označení kategorie na standardní výstup s využitím selekce switch().

5.6 Změny v průběhu implementace

V průběhu implementace byly zkoušeny další kombinace detektoru a deskriptoru, jmenovitě FAST+FREAK a Harrisův detektor implementovaný v knihovně OpenCV. Oba detektory generovaly poměrně velké množství možných klíčových bodů, což sice ve finále zpomaluje výpočet, jelikož je třeba na tyto body aplikovat deskriptor a nebo je zařadit do výpočtu shlukování. Výpočty dat databáze tak v některých případech trvaly i přes osm hodin, a přitom výsledky byly velmi slabé. Detektor FAST často opomíjel oblasti blíže ke kraji obrázku a zaměřoval se na jeho střed, což nutně vedlo k neobsažení informace například o okrajích zbraně na obrázku. Implementace Harrisova detektoru v OpenCV zase nedetekovala zaoblenější rohy, a navíc byla velmi citlivá na šum. Z těchto důvodu byly oba tyto detektory vyhodnoceny jako nevhodné pro použití v programu ještě před jeho kompletní implementací. Další ze zkušených detektorů byl KAZE s BoWKmeansTrainer a matcherem s použitím výpočtu pro vzdálenost v NORM_L2, který měl ale jako float deskriptor značně delší dobu výpočtu dat databáze, učení i predikce, a přitom výsledky odpovídaly tehdy současným verzím s využitím Harrisova detektoru implementovaného v rámci práce spolu s deskriptorem FREAK, navíc některé obrázky použité pro generování databáze algoritmus ignoroval a do dat nic nevkládal, což vedlo k vyvolávání výjimek za běhu. Proto bylo i od tohoto detektoru upuštěno.

Na internetových stránkách jako stackoverflow.com a learnopencv.com se ohledně použití Bag of Words v součinnosti s binárním deskriptorem objevují rady převést výsledek do podoby čísla s plovoucí řádovou čárkou a to pak předat funkci compute třídy BoWKMeansTrainer, která je součástí knihovny OpenCV, nicméně to je naprostý nesmysl, jelikož princip popisu bodu binárním

deskriptorem je naprosto odlišný od popisu float deskriptorem. Implementace podle těchto rad byla z principu špatně, proto nemohla produkovat žádné použitelné výsledky.

6 Výsledky

Pro ověření funkce programu byla připravena databáze popsaná v kapitole 2.3. Protože tato databáze má velké rozdíly mezi počty obrázků v jednotlivých kategoriích, bylo třeba přistoupit k jejímu zarovnání, aby každá z kategorií měla stejný počet příkladů. Tento počet byl zvolen na základě nejméně početné kategorie 250 příkladů pro každou kategorii, tedy 2000 celkem. Rovněž pro Bag of Words je náhodně vybíráno 50 obrázků z každé kategorie, tedy 400 celkem. Klíčové body jsou shlukovány do 400 shluků, k těmto hodnotám bylo přistoupeno hlavně z důvodu časové náročnosti výpočtu. 120 příkladů (15 na kategorii) bylo předtím odebráno pro následné ověření funkčnosti. Tyto příklady pak byly hotovému programu předkládány k určení kategorie. Doba výpočtu dat pro databázi na počítači uvedeném v kapitole 4.1 byla u finální zhruba 5 hodin, učení SVM pak trvá 4-5 minut. Testování probíhalo na dvou vygenerovaných databázích, celkový počet testovacích snímků v následujících tabulkách je tedy 240.

Před počátkem testování probíhaly zejména testy detekcí bodů na snímcích databáze, kdy bylo odhadem zkoumáno, zda vůbec detekované body mohou vhodně popsat obrázek a zda jsou použitelné v kombinaci se zvoleným deskriptorem. Tomu byla věnována část předchozí kapitoly. První testování ukázalo úspěšnost jen okolo 5 %. Na základě těchto výsledků byl odhalen problém s použitím BoWKmeansTrainer spolu s binárním deskriptorem zmiňovaný v předchozí kapitole. Díky tomu byl BoWKmeansTrainer nahrazen BoWKmajorityTrainer, který je skutečně pro binární detektory určen. Nicméně bylo nutné projít všechny obrázky databáze, protože tento modul s některými z nich nedokázal pracovat. Nevhodné obrázky byly odstraněny z databáze a nahrazeny vhodnějšími. Další testovací iterace přinesla celkovou úspěšnost 12,5 %, nicméně to bylo způsobeno tím, že SVM vše klasifikoval do jedné skupiny, která tím pádem měla úspěšnost 100 %, zatímco všechny ostatní selhaly. Tento problém byl řešen skrze dvě testovací iterace, kdy bylo změněno jádro SVM z lineárního na INTER a provedeny změny ve výpočtech slovníku.

Nakonec byla jako příčina odhalena nevyvážená databáze, tak došlo k jejímu vyvážení. Po vyvážení databáze vycházela celková úspěšnost okolo 12 %. V této době byl taktéž pokusně použit detektor KAZE zmiňovaný v předchozí kapitole, který ale nepřinesl lepší výsledky. Během další testovací iterace byl upraven vstup obrázku do výpočtu deskriptoru, který nyní není upravován mediánovým filtrem. S průběhem iterací se také adaptovaly parametry Harrisova deskriptoru a předzpracování obrazu, aby docházelo k co nejlepší detekci významných bodů. Ve finální verzi programu bylo dosaženo následujících výsledků:

	klasifikováno jako:		
	Dlouhé zbraně	Krátké zbraně	úspěšnost
Dlouhé zbraně	62	58	51,7 %
Krátké zbraně	44	76	63,3 %

Tabulka 6.1: výsledky klasifikace pro dělení dle délky

kde hodnota „úspěšnost“ je vypočítána jako:

$$úspěšnost = \frac{počet\ úspěšných\ identifikací}{počet\ příkladů\ v\ kategorii} * 100 \quad (5.1)$$

Celková úspěšnost je pak vypočítána jako:

$$celková\ úspěšnost = \frac{celkový\ počet\ úspěšných\ identifikací}{celkový\ počet\ příkladů} * 100 \quad (5.2)$$

Celková úspěšnost klasifikace pro dělení dle délky je 57,5 %

klasifikováno jako:

	automatické	opakovací	jednoranové	víceranové	úspěšnost
automatické	24	4	6	26	40 %
opakovací	24	8	3	25	13,3 %
jednoranové	12	20	9	19	15 %
víceranové	11	11	4	34	56,7 %

Tabulka 6.2: výsledky klasifikace pro dělení dle kategorie

Celková úspěšnost klasifikace pro dělení dle kategorie je 31,3 %

	Správně určená délka	Špatně určená délka
Správně určená kategorie	32	43
Špatně určená kategorie	106	59

Tabulka 6.3: celkové výsledky klasifikace

Z těchto výsledků vychází celková úspěšnost klasifikace podle obou kritérií 13,3 %

Tento celkový výsledek veskrze ukazuje navrhovaný přístup jako stále nevhodný, jelikož výsledná pravděpodobnost správné klasifikace je pouze o málo jistější, než náhodný výběr z nabízených osmi kategorií, kde pravděpodobnost získání správného výsledku odpovídá 12,5 %. Alespoň trochu použitelných hodnot úspěšnosti bylo dosaženo pouze u klasifikace do kategorie krátkých zbraní, a v omezené míře v případě víceranových zbraní.

Podle výsledků se zdá, že SVM nedokázal dostatečně separovat jednotlivé kategorie tak, aby následně dokázal přiřadit neznámý obrázek do správné kategorie. To mohlo být způsobeno problémy při klasifikaci zbraní zmiňovaných v podkapitolách 2.2 a 2.3. Snímky zbraní často musejí obsahovat velké množství pozadí, které při jejich klasifikace nemá hrát žádnou roli, ale detektor v něm generuje množství klíčových bodů, které budou klasifikátor spíše mást, než aby pomohly v detekci samotné zbraně na obraze. Rohové detektory obecně mají problémy s detekcí rohů zaoblených a také těch rohů, které svírají tupý úhel, oba tyto prvky se na zbraních vyskytují poměrně často. To například vede k situaci, kdy jsou správně detekovány spodní okraje zásobníku, ale nikoliv části zásobníkové šachty, které mívají zaoblenější tvary. Rovněž detekce okrajů hlavně byla často problémová. Jak bylo upozorněno 5.6, Harrisův detektor implementovaný v rámci práce sice ukazuje lepší detekci zaoblených rohů oproti implementaci v OpenCV, ale v nemalé části případů ani toto nastavení nedokázalo detekovat vhodný bod.

Z zatím neznámého důvodu má klasifikátor velkou tendenci klasifikovat zbraně jako víceranové, přestože jimi nejsou. Špatná klasifikace zbraní opakovacích jako automatických a jednoranových jako opakovacích je nejspíš způsobena problémy při klasifikaci, kterým byla věnována kapitola 2.2.

Obr. 6.1: ukázka detekce klíčových bodů funkcí detect() *na testovacím obrázku*

6.1 Možnosti dalšího postupu

Přesnost určení by se dala zvětšit větší sadou vstupních dat. Program už je na to předem připraven a nové snímky do databáze lze přidat bez zásahu do kódu pouze vložením do jedné ze složek. Pouze při zvětšení zarovnání databáze je třeba změnit definovanou konstantu v souboru GR_Lib.h. V případě zvětšení databáze by ale bylo vhodné upravit zejména algoritmus shlukování pro podporu paralelního výpočtu, například pomocí funkcí knihovny OpenMP.

Možnosti postupu v oboru počítačového vidění se neustále rozvíjejí, takže další možností by mohlo být použití jiného postupu pro klasifikaci, jako jsou třeba dnes velmi používané neuronové sítě nebo K-nearest-neighbor klasifikátor. Také by šlo využít pokročilejší detektory jako třeba SIFT, který je ale chráněn patentem a proto jeho implementace není součástí OpenCV.

Díky poměrně modulární konstrukci velké části programu lze soubor GR_Lib.h využít jako knihovnu i při dalších úlohách rozpoznávání a klasifikace. A Například třeba funkce *Mat2KP()* je vhodná i pro užití s implementací některých detektorů v OpenCV, které vrací výstup ve formě matice.

7 Závěr

V práci byly shrnuty pro práci důležité informace z nauky o zbraních a jejich klasifikace podle zákona ČR. Také byla vytvořena anotovaná databáze zbraní důležitá pro implementaci a testování programu. Dále se práce věnovala obrazu a jeho digitálnímu zpracování, byly uvedeny některé zásadní postupy předzpracování obrazu a vysvětleny dnes využívané detektory, deskriptory, a metody Bag of Words a SVM. Taktéž byl vytvořen návrh řešení problému klasifikace zbraní dle délky a kategorie, který byl následně v bakalářské práci implementován a testován.

Testování postupem času ukázalo změnu k lepšímu, nicméně výsledná přesnost 13,3 % stále není dostatečná pro vážně míněné použití programu. Byly shrnuty možné důvody proč tomu tak je, a jak by se dalo na programu pokračovat tak, aby se zvýšila jeho přesnost.

Literatura

[1] Russ, J.C. The Image Processing Handbook. Boca Raton : CRC Press, 1995. 674 p. ISBN 0-8493-2516-1.
[2] Hlaváč V., Šonka, M.: Počítačové vidění. Praha: Grada, 1992, ISBN 80-85424-67-3.

[3] Sonka, Milan, Hlavac, Vaclav, Boyle, Roger. Image Processing, Analysis and Machine Vision. 3rd edition. Toronto : Thomson, 2008. 829 p. ISBN *978-0-495-08252-1*

[4] Tinku Acharya, A. K. R.: Image Processing: Principles and Applications. Wiley-Interscience, 2005, ISBN 978-0471719984.

[5] OpenCV: color conventions, [Online; navštívené 14.1.2019]. Dostupné z: https://docs.opencv.org/3.1.0/de/d25/imgproc_color_conversions.html

[6] Lindeberg, T. J Math Imaging Vis (2015) 52: 3. *https://doi.org/10.1007/s10851-014-0541-0*

[7] Harris C., Stephens M., A Combined Corner and Edge Detection, 1988, Proceedings of The Fourth Alvey Vision Conference. [Online; navštívené 14.1.2019]. Dostupné z: http://www.ic.unicamp.br/~rocha/teaching/2013s1/mc851/aulas/additional-material-harris-detector.pdf
[8] Žilka, F. Detektory a deskriptory oblastí v obrazu. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2016. Vedoucí diplomové práce byl Ing. Karel Horák, Ph.D..
[9]OpenCV: Harris corner detection, [Online; navštívené 15.1.2019]. Dostupné z: https://docs.opencv.org/3.4.2/dc/d0d/tutorial_py_features_harris.html

[10] JUDr. Libor Kovárník, Ph.D. Otázky a odpovědi – vymezení pojmu zbraň. [Online; navštívené 11.12.2018]. Dostupné z: http://bezpecnostni-sbory.wbs.cz/clanky/5-2010/otazky_a_odpovedi_____vymezeni_pojmu_zbran.htm#_ftn2

[11] Jankových, R.: Hlavňové zbraně a střelivo. Brno: VUT Brno, 2012, ISBN 978-80-260-2384-5.

[12]Česká republika. Příloha č. 1 k zákonu č. 119/2002 Sb. [Online; navštívené 11.12.2018]. Dostupné z: https://www.zakonyprolidi.cz/cs/2002-119#prilohy

[13] MP7A1 | Technical Data, [Online; navštívené 11.12.2018]. Dostupné z: https://www.heckler-koch.com/en/products/military/submachine-guns/mp7a1/mp7a1/technical-data.html

[14] CZ SCORPION 61 S, [Online; navštívené 11.12.2018]. Dostupné z: https://www.czub.cz/cz/cz-skorpion-61-s.html

[15] FN SCAR®-L, [Online; navštívené 11.12.2018]. Dostupné z: https://www.fnherstal.com/en/product/fn-scar-l

[16] SOV - AKS-74U, [Online; navštívené 11.12.2018]. Dostupné z: https://forum.valka.cz/topic/view/17472/SOV-AKS-74U

[17] SAMOPAL vz. 58, [Online; navštívené 13.12.2018]. Dostupné z: https://www.zbraneamysliveckepotreby.cz/myslivecke-potreby-cz/eshop/47-1-SAMOPAL-vz-58

[18] Glock pistol series, [Online; navštívené 13.12.2018]. Dostupné z: http://www.imfdb.org/wiki/Glock#Glock_18

[19] SA Vz.58 assault rifle, [Online; navštívené 15.12.2018]. Dostupné z: http://www.imfdb.org/wiki/SA_Vz.58_assault_rifle

[20] CZ 858 Synthetic, [Online; navštívené 15.12.2018]. Dostupné z: http://www.imfdb.org/wiki/File:CZ_858_Synthetic.jpg

[21] Savage Youth Rascal Single-Shot Bolt Action Rifle, [Online; navštívené 15.12.2018]. Dostupné z: https://www.cabelas.ca/product/36733/savage-youth-rascal-single-shot-bolt-action-rifle

[22] Savage Arms B 22 FVSS Bolt Action 22 LR Rifle, [Online; navštívené 15.12.2018]. Dostupné z: https://alsimmonsgunshop.com/product/savage-arms-b-22-fvss-bolt-action-22-lr-rifle-70202/

[23] Serbu Firearms: Super-Shorty, [Online; navštívené 14.1.2019]. Dostupné z: https://serbu.com/products/super-shorty-1

[24] http://www.imfdb.org/images/5/5c/M1911-RAFIssue455.jpg

[25] http://ai.stanford.edu/~syyeung/cvweb/tutorial2.html

[26] Calonder, M., Lepetit, V., Strecha, C. Fua, P., Brief: Binary robust independent elementary features. Pattern Analysis and Machine Intelligence, IEEE Transactions. Volume 34. Issue7. 2011. s. 1281-1298. [Online; navštívené 15.1.2019]. Dostupné z: https://www.robots.ox.ac.uk/~vgg/rg/papers/CalonderLSF10.pdf

[27] Alahi. A., Ortiz, R., Vandergheynst, P., FREAK: Fast retina keypoint. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference. s. 510–517. [Online; navštívené 15.1.2019]. Dostupné z: http://infoscience.epfl.ch/record/175537/files/2069.pdf

[28] https://cs.wikipedia.org/wiki/Konvoluce

[29] Tiwari R. K., Verma, G. K., A Computer Vision based Framework for Visual Gun Detection Using Harris Interest Point Detector, Procedia Computer Science, Volume 54, 2015, Pages 703-712 [Online; navštívené 15.1.2019]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1877050915014076#bibl0005

[30] Oldest Wooden Spear [Online; navštívené 13.3.2019]. Dostupné z: http://humanorigins.si.edu/evidence/behavior/getting-food/oldest-wooden-spear

[31] The Heilongjiang hand cannon manufactured no later than 1288 is the world's oldest surviving firearm [Online; navštívené 13.3.2019] https://www.thevintagenews.com/2016/08/04/priority-heilongjiang-hand-cannon-manufactured-no-later-1288-worlds-oldest-surviving-firearm/

[32] http://www.computervisionblog.com/2015/01/from-feature-descriptors-to-deep.html

[33] Dalal, N., Triggs, B., Histograms of Oriented Gradients for Human Detection, CVRP 2005, s. 886-893, Dostupné online: http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

[34] Drummond, T., Rosten, E., Machine learning for high-speed corner detection. In Proceedings of the 9th European conference on Computer Vision -Volume Part I(ECCV'06), 2006. s. 430-443 Dostupné z: http://www.edwardrosten.com/work/rosten_2006_machine.pdf

[35] Rublee, E., et al. ORB: An efficient alternative to SIFT or SURF. Computer Vision International Conference. 2011. s.2564-2571.Dostupné z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6126544

[36] Matas, J., Chum, O., Urban, M., Pajdla, T., Robust Wide Baseline Stereo from Maximally Stable Extremal Regions, Proceedings of the British Macine Vision Conference. 2002. s. 36.1-36.10. Dostupné z: http://cmp.felk.cvut.cz/~matas/papers/matas-bmvc02.pdf

[37] Lowe, D., Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91–110, 2004. Dostupné z: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[38] Support Vector Machines (SVM) – Learn OpenCV [Online; navštívené 26.4.2019]. Dostupné z: https://www.learnopencv.com/support-vector-machines-svm/

[39] Support Vector Machines – OpenCV documentation [Online; navštívené 26.4.2019]. Dostupné z: https://docs.opencv.org/2.4/modules/ml/doc/support_vector_machines.html

[40] AISEN, B., A Comparison of Multiclass SVM Methods, [Online; navštívené 26.4.2019]. Dostupné z: https://courses.media.mit.edu/2006fall/mas622j/Projects/aisen-project/

[41] HSU, C.-W., LIN, C.-J., A Comparison of Methods for Multi-class SupportVector Machines, IEEE Transactions on Neural Networks, 13(2002), 415-425. Dostupné z: https://www.csie.ntu.edu.tw/~cjlin/papers/multisvm.pdf

[42] Bethea Davida - Bag of Visual Words in a Nutshell, [Online; navštívené 28.4.2019], Dostupné z: https://towardsdatascience.com/bag-of-visual-words-in-a-nutshell-9ceea97ce0fb

[43] Jocelyn D'Souza - An Introduction to Bag-of-Words in NLP [Online; navštívené 28.4.2019], Dostupné z: https://medium.com/greyatom/an-introduction-to-bag-of-words-in-nlp-ac967d43b428

[44] ALCATNTARILLA, P. F., BARTOLI, A., DAVISON, A. J., KAZE Features, Universite d'Auvergne, Clermont Ferrand, France, 2012, dostupné z: https://www.doc.ic.ac.uk/~ajd/Publications/alcantarilla_etal_eccv2012.pdf

[45] ANDERSON, O., MARQUEZ, S. R., A comparison of object detection algorithms using unmanipulated testing images, Stockholm, Sweden, 2016, dostupné z: https://kth.diva-portal.org/smash/get/diva2:927480/FULLTEXT01.pdf

[46] BAY, H., TUYTELAARS, T., VAN GOOL, L.. SURF: Speeded up robust features. In Proceedings of European Conference on Computer Vision, pages I: 404–417, 2006. Dostupné z: http://www.vision.ee.ethz.ch/~surf/eccv06.pdf

[47] http://www.imfdb.org/wiki/FN_F2000

[48] http://www.imfdb.org/wiki/File:Calico960a.jpg

[49] http://www.imfdb.org/wiki/File:MinimiPara.jpg

[50] http://www.imfdb.org/wiki/Category:Screenshot

[51] http://www.imfdb.org/wiki/Category:Gun_Image

[52] OpenCV – Feature matching, [Online; navštívené 8.5.2019] dostupné z: https://docs.opencv.org/3.4.3/dc/dc3/tutorial_py_matcher.html

[53] Understanding K-means Clustering in Machine Learning, [Online; navštívené 8.5.2019] dostupné z: https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1