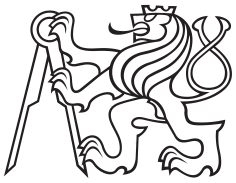
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra kybernetiky

Analýza obrazu gelové elektroforézy

Josef Grus

Školitel: Prof. Dr. Ing. Jan Kybic
Zaměření: Kybernetika a robotika
Květen 2020

Poděkování

Chtěl bych poděkovat Prof. Dr. Ing. Janu Kybicovi za vedení této práce, a za rady a připomínky při řešení problémů s touto prací souvisejících.

Také bych chtěl poděkovat Státnímu zdravotnímu ústavu, konkrétně Národní referenční laboratoři pro lymeskou borreliózu za poskytnuté obrázky sloužící jako podklad bakalářské práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne .....

Podpis: .....

Abstrakt

Cílem této bakalářské práce je popsat použité metody zpracování obrazu gelové elektroforézy. Práce se zabývá detekcí značek zachycených na snímcích gelové elektroforézy a jejich následným zpracováním. To se skládá z vybrání značek tvořících žebříčky s referenčními hodnotami počtu bází, opravou vstupní rotace v obrázcích a zkreslení způsobeného nehomogenitou proudu, fitování nalezených žebříčků do referenční struktury, a odhad počtu bází neznámých značek na základě jejich polohy vzhledem k referenčním žebříčkům. Výsledky práce jsou vizualizovány pomocí jednoduše ovladatelného programu.

Klíčová slova: elektroforéza, MOSSE filtr, kaskádní detektor, problém přiřazení

Školitel: Prof. Dr. Ing. Jan Kybic

Abstract

The aim of this bachelor thesis is to describe used methods of image processing of gel electrophoresis. The thesis deals with the detection of markers - bands, captured on the images of gel electrophoresis and their subsequent processing. This processing consists of selecting the ladder markers with base pairs reference values, correcting the image rotation and current inhomogeneity distortion, fitting the found ladders to a reference structure, and estimating the number of base pairs of the unknown markers based on their position relative to the reference ladders. The results are visualized using an easy-to-use program.

Keywords: electrophoresis, MOSSE filter, cascade detector, assignment problem

Title translation: Gel electrophoresis image analysis

Obsah

1 Úvod **1**

2 Rozbor problematiky **3**

3 Vstupní data a způsob vyhodnocování **7**

3.1 Vstupní data 7

3.2 Způsob vyhodnocování 8

4 Metody detekce **11**

4.1 Korelační filtry 11

4.1.1 Předzpracování vstupních obrázků 12

4.1.2 Konstrukce korelačního filtru 14

4.1.3 Zpracování výstupu korelace . 17

4.1.4 Výběr optimální hodnoty prahu 18

4.1.5 Kombinace dvou typů filtru . 20

4.2 Kaskádní detektor 21

4.2.1 Deskriptory založené na Haar wavelets 22

4.2.2 Deskriptory založené na LBP 23

4.2.3 Postprocessing výstupu kaskádního detektoru s MB-LBP deskriptory 25

4.2.4 Rozšíření negativního datasetu 29

5 Zpracování nalezených značek **31**

5.1 Rozdělení značek do žebříčků . . . 31

5.2 Korekce rotace vstupního obrázku 35

5.3 Fitování žebříčků do referenčního modelu 36

5.3.1 Odhad parametrů 38

5.3.2 Definice kriteriální funkce . . . 41

5.3.3 Formulace optimalizační úlohy přes množinu přiřazení 42

5.3.4 Optimalizace přes množinu přiřazení pomocí metod lineárního programování 42

5.3.5 Optimalizace přes množinu přiřazení pomocí Maďarského algoritmu 43

5.3.6 Formulace optimalizační úlohy přes množinu parametrů afinní funkce 45

5.3.7 Optimalizace přes množinu parametrů afinní funkce pomocí metod lineárního programování . . 45

5.3.8 Optimalizace přes množinu parametrů afinní funkce pomocí lineárních nejmenších čtverců . . . 46

5.3.9 Výběr parametrů fitování . . . 47

5.4 Korekce chyb způsobených nehomogenitou proudu 52

5.5 Odhad počtu bází neznámých značek 55

5.5.1 Odhad fitováním hyperboly . 55

5.5.2 Odhad počtu bází pomocí hyperboly v opačném smyslu a odhad lineární interpolací 58

6 Výsledky **61**

6.1 Hodnocení detekčních metod na reálných datech 61

6.2 Zhodnocení výsledků detekce na syntetických datech 66

6.3 Zhodnocení výsledků algoritmů na zpracování detekovaných značek . . 70

6.4 Zhodnocení použitelnosti implementace 75

7 Aplikace **77**

7.1 Popis uživatelského rozhraní . . . 77

7.2 Popis souborů a spuštění aplikace 79

8 Závěr **81**

A Ilustrace výsledků detekce značek a fitování žebříčků **83**

A.1 PCR-INTSPA-01-11-06 84

A.2 PCR-LD5-27-07-05 85

A.3 PCR-EHR709-KL-25-04-07 87

B Přílohy na CD **89**

C Návod k uživatelskému rozhraní **91**

D Bibliografie **97**

E Zadání práce **101**

Obrázky

1.1 Nákres aparatury k provedení gelové elektroforézy, z [1] 2

1.2 Příklad vstupního obrázku 2

2.1 Typický tvar značek 3

2.2 Poloha wells v obrázku gelové elektroforézy, z [3] 4

2.3 Obrázek z datasetu bez dobře viditelných zdrojnic 4

3.1 Příklad z datasetu obrázků 8

4.1 Vstupní obrázek před normalizací 13

4.2 Výsledek lokální normalizace s parametry $\sigma_x = 5, \sigma_y = 9$ 13

4.3 Výsledek lokální normalizace s parametry $\sigma_x = 2, \sigma_y = 3$ 13

4.4 Korelační filtry pro běžný typ značky 15

4.5 Korelační filtry pro širší typ značky 15

4.6 Příklady MOSSE filtrů s jinými parametry σ 16

4.7 Výsledek korelace s MOSSE filtrem 17

4.8 ROC křivka pro běžný typ značky 19

4.9 ROC křivka pro širší typ značky 19

4.10 Zdvojování detekovanýcg extrémů na krajích značek v heatmap při použití nevhodného filtru 20

4.11 Vizualizace výpočtu IoU, [9] . . 21

4.12 Obdélníkové deskriptory dle [10] 22

4.13 Ukázka možných MB-LBP deskriptorů, [14] 24

4.14 Příklady positivních okének pro trénink kaskádního detektoru 24

4.15 Příklady negativních okének pro trénink kaskádního detektoru 24

4.16 Výstup samotné multiscale detekce, červeně označeny detekce, zeleně označeny anotace 26

4.17 Výsledek dilatace se strukturním prvkem S_1 26

4.18 Výsledek uzavření se strukturním prvkem S_2 26

4.19 Vstupní binární obrázek před vzdálenostní transformací 27

4.20 Výsledek vzdálenostní transformace 27

4.21 Výsledek detekce po provedení segmentace rozvodím, zeleně označeny reference, červeně detekce 28

4.22 ROC křivka kaskádního detektoru pro threshold τ 28

4.23 Příklady nových negativních okének pro trénink kaskádního detektoru 29

4.24 Výsledek samotné detekce detektorem s MB-LBP deskriptory trénovaném na rozšířeném datasetu, zeleně označeny reference, červeně detekce 30

5.1 Vstupní obrázek pro rozdělení značek do lanes 32

5.2 Histogram p_G filtrovaných jednotkových impulzů se segmentací osy x do lanes - přechody pozadí odpovídají prahům segmentů 32

5.3 Příklad různých variant histogramu v závislosti na na sm. odchylce filtru σ 33

5.4 Ukázka rozdělení bodů na žebříčky a neznámé značky (zeleně označena lane žebříčku, červeně lane zbylých značek) 34

5.5 Závislost chyby v rozpoznávání žebříčků na parametru ω 34

5.6 Obrázek před provedením opravné rotace 36

5.7 Výsledek opravné rotace 36

5.8 Vliv tolerančního parametru u na relativní chybu 48

5.9 Vliv počtu iterací algoritmu 1 na relativní chybu 49

5.10 Vliv nastavení parametru λ na výsledky algoritmu RANSAC s použitím l_1 normy 49

5.11 Vliv nastavení parametru λ na výsledky algoritmu RANSAC s použitím l_2 normy 50

5.12 Vliv nastavení parametru λ na výsledky algoritmu Lo-RANSAC s použitím l_1 normy 50

5.13 Vliv nastavení parametru λ na výsledky algoritmu Lo-RANSAC s použitím l_2 normy 50

5.14 Porovnání fitování při použití různých norem, při vybrání optimální hodnoty λ z grafů 5.10 a 5.11 51

5.15 Vliv délky výpočtu na medián relativní chyby 52

5.16 Vrstevnice korigujícího polynomu a značky žebříčku, jimiž mají vrstevnice procházet, pro 1 žebříček 53

5.17 Vrstevnice korigujícího polynomu a značky žebříčku, jimiž mají vrstevnice procházet, pro 2 žebříčky 54

5.18 Vrstevnice korigujícího polynomu a značky žebříčku, jimiž mají vrstevnice procházet, pro 3 žebříčky 54

5.19 Příklad použití polynomu z $\mathcal{Q}_3$ pro obrázek s 2 žebříčky 54

5.20 Vrstevnice polynomu při použití kvadratického členu y^2 55

5.21 Vrstevnice polynomu stejného obrázku při absenci y^2 55

5.22 Fitování hyperbolické funkce do dat expertního žebříčku 56

5.23 Příklad 1 - fitování hyperboly . 57

5.24 Příklad 2 - fitování hyperboly . 57

5.25 Příklad 3 - fitování hyperboly . 57

5.26 Chybný odhad fitovanou hyperbolou 58

5.27 Porovnání hyperbolických křivek pro různé varianty optimalizace . . . 59

5.28 Výsledky odhadu počtu bází pomocí hyperbolické funkce dle rovnice 5.45 59

5.29 Ukázka odhadu počtu bází lineární interpolací 60

6.1 Histogram přesností pro různé metody detekce 62

6.2 ROC křivky obou detektorů 63

6.3 Závislost průměrných recall a precision na hodnotě geometrického prahu 64

6.4 Detekce MOSSE filtrem v obrázku s problematickými „šmouhami“ . . 65

6.5 Detekce kaskádním detektorem v obrázku se „šmouhami“ 65

6.6 Problematické dvojité detekce kaskádního detektoru 65

6.7 Příklad detekce MOSSE filtrem na obrázku zatíženém drobným šumem 66

6.8 Detekce MOSSE filtrem na obrázku zatíženém většími chybami 66

6.9 Vliv šumu na výsledky detekce . 67

6.10 Heatmap MOSSE filtru při vstupním šumu se $\sigma = 5$ 67

6.11 Vliv velikosti kernelu průměrovacího filtru na výsledky detekce 68

6.12 Výsledek detekce kaskádním detektorem při šířce kernelu průměrovacího filtru 20 68

6.13 Heatmap korelovaného obrázku po prahování 69

6.14 Výsledek detekce zašuměného obrázku s použitím původní hodnoty prahování 69

6.15 Výsledek detekce zašuměného obrázku s použitím nové hodnoty prahování 69

6.16 Vstupní obrázek před rotací . . . 71

6.17 Obrázek po detekci a provedení rotace 71

6.18 Vizualizace přesnosti odhadu odchylky úhlu 71

6.19 Nedostatečně vyhlazený histogram pro rozdělení značek . . . 72

6.20 Chybné fitování na levém žebříčku 73

6.21 Odhad počtu bází hyperbolickou funkcí 73

6.22 Absolutní chyba lineární interpolace v závislosti na poloze v žebříčku 74

6.23 Odhad počtu bází hyperbolickou funkcí - LD primery 74

6.24 Odhad počtu bází lineární interpolací - LD primery 75

7.1 Úvodní okno programu 77

7.2 Okno nastavování žebříčku 79

A.1 Výsledky detekce - Př. 1 84

A.2 Rozdělení do lanes - Př. 1 84

A.3 Fitování žebříčku a odhady počtů bází - Př. 1 85

A.4 Výsledky detekce - Př. 2 85

A.5 Porovnání detekcí s anotacemi - Př. 2 86

A.6 Fitování žebříčku a odhady počtů bází - Př. 2 86

A.7 Výsledky detekce - Př. 3 87

A.8 Porovnání detekcí s anotacemi - Př. 3 87

A.9 Fitování žebříčku a odhady počtů bází - Př. 3 88

C.1 Zobrazení tlačítek 93

C.2 Horní menu aplikace 93

C.3 Tlačítka nastavování nového žebříčku 94

C.4 Formáty textových souborů žebříčků 95

C.5 Formát výstupního souboru ... 95

Tabulky

5.1 Průměrné doby výpočtu optimálního přiřazení v závislosti na rozdílu značek detekovaného a referenčního žebříčku 45

6.1 Výsledky odhadů počtu bází ... 75

Kapitola 1

Úvod

Gelová elektroforéza je základní separační metoda, která umožňuje separovat různé makromolekuly, například molekuly DNA, na základě jejich velikosti a elektrického náboje. Gelová elektroforéza nachází uplatnění v mnoha oborech, například v biochemii, genetice nebo molekulární biologii.

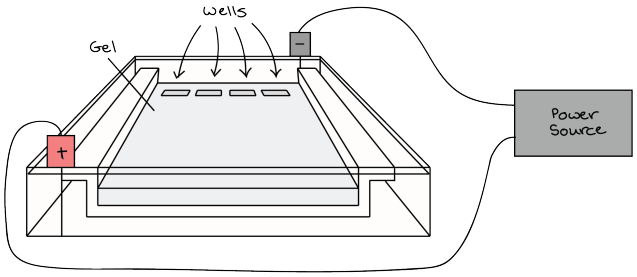
Gelová elektroforéza využívá faktu, že zkoumané molekuly mají nenulový elektrický náboj. Experiment probíhá v nádobě s polysacharidovým gelem, a nesetříděné molekuly jsou na počátku umístěny do zdrojnic (angl. wells). Aplikací elektrického pole se molekuly dají do pohybu ve směru kladné elektrody a rychlost tohoto pohybu je závislá na velikosti molekuly a jejím náboji. Po uplynutí stanovené doby jsou pořízeny snímky, na nichž jsou díky použití barviva molekuly viditelné jako značky (angl. bands). Obvykle je součástí experimentu také žebříček značek známých velikostí, které slouží k určení velikostí značek neznámých.

Motivací k automatizaci zpracování snímků získaných z experimentů gelové elektroforézy je zkrácení času, který je nutný vynaložit na ruční analýzu, snímku kvůli určení velikosti neznámých značek, což je operací, která nevyžaduje expertní znalosti, nutné k pozdější práci se získanými daty.

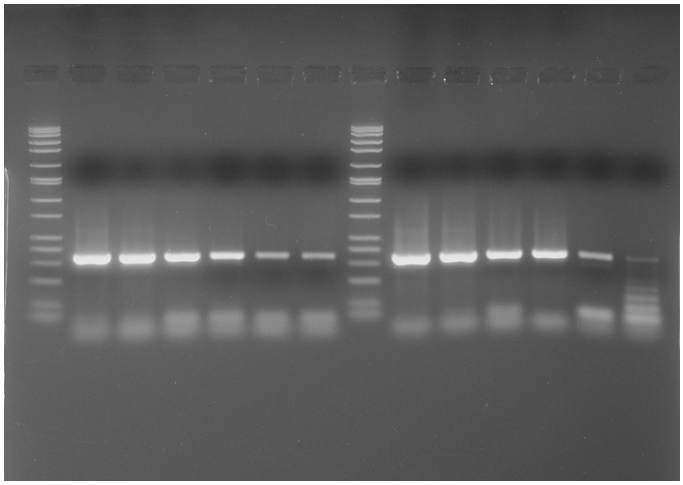
Cílem práce je navrhnout a otestovat metody pro zpracování obrázků gelové elektroforézy. Tyto metody lze rozdělit do dvou hlavních etap podle toho, v jaké části zpracování se použijí. První etapa zpracování se zabývá detekcí jednotlivých značek v obrázku. V druhé etapě zpracování se provádějí operace, na jejichž konci by mělo být doplnění neznámých značek o jejich velikost, resp. počet bází, které příslušnou makromolekulu tvoří. Tyto operace musí

být schopny rozdělit nalezené značky na značky referenční (tj. ty, které jsou součástí referečních žebříčků) a značky neznámé, opravit vstupní rotaci v obrázku anebo zkreslení způsobené nehomogenitou proudu, a nakonec určit počet bází neznámých značek na základě polohy značky vůči referenčním žebříčkům, ke kterým je počet bází značek znám.

Pro vizualizaci výsledků práce je úkolem vytvořit jednoduše ovladatelný program, který implementuje algoritmy pro zpracování a který umožňuje provádět analýzy snímků bez nutnosti nastavovat velké množství parametrů programu.



Obrázek 1.1: Nákres aparatury k provedení gelové elektroforézy, z [1]

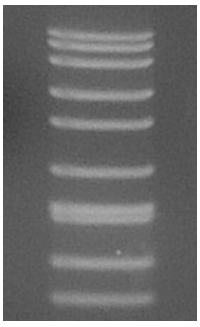


Obrázek 1.2: Příklad vstupního obrázku

Kapitola 2

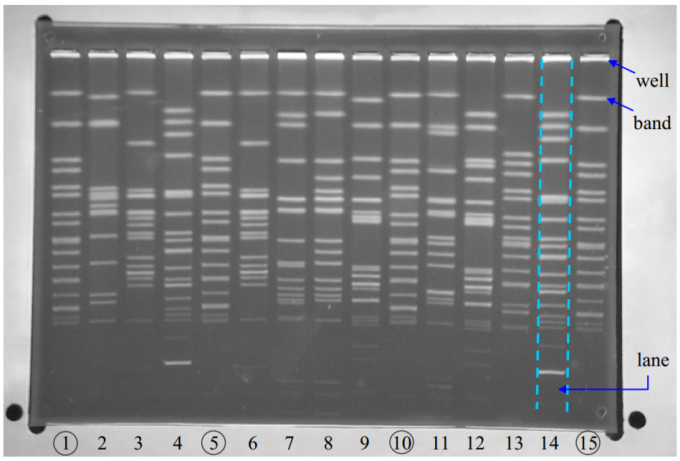
Rozbor problematiky

První fáze zpracování obrázků gelové elektroforézy je detekční úlohou. Značky makromolekul v gelu jsou typické podlouhlým tvarem, viz obr. 2.1. Kontrast mezi značkami a gelovým pozadím se však výrazně liší experiment od experimentu, detekční metoda musí být rovněž dostatečně robustní, aby případný šum ve snímcích nezpůsobil vznik falešně positivních detekcí.



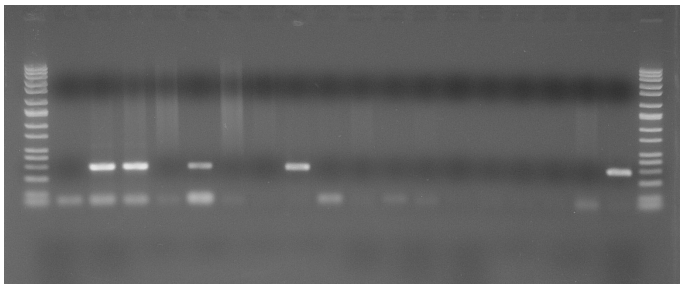
Obrázek 2.1: Typický tvar značek

Problematikou automatického zpracování obrázků gelové elektroforézy se již v minulosti zabývalo několik týmů. Většina automatizovaných řešení, které jsou popsané v [2] a [3], využívá pro detekci předpoklad, že v obrázku se nachází rozpoznatelná viditelné wells (zdrojnice makromolekul DNA) v jedné řadě, viz obr. 2.2. Poté se již problém nalezení značek ve dvoudimenzionálním prostoru obrázku zjednodušuje na 1D prohledávání vytipovaných lanes. Poloautomatizované řešení dle [4] podobně lokalizuje jednotlivé značky v lanes právě potom, co operátor programu lane ručně zaměří.



Obrázek 2.2: Poloha wells v obrázku gelové elektroforézy, z [3]

Po prohlídce obrázků dodaných pro vypracování bakalářské práce se však ukázalo, že takový předpoklad se na většinu z nich nedá aplikovat, viz příklad 2.3. Proto se v této práci využívá principiálně opačného postupu. Nejprve jsou detekovány jednotlivé značky pomocí metod založených na korelaci obrázku s filtrem, nebo za použití detektorů založených na strojovém učení. Po nalezení značek dojde následně k jejich sdružení do lanes.



Obrázek 2.3: Obrázek z datasetu bez dobře viditelných zdrojnic

Po úspěšném provedení detekce je možné pokračovat ve zpracování opravením rotace v obrázku, která se projevuje vychýlením os referenčních žebříčků od vertikální osy obrázku. Druhým typem jsou chyby způsobené nehomogenitou proudu. Použití nehomogenního elektrického pole má za důsledek to, že rychlost značek s identickým nábojem a velikostí, a tedy i jejich uražená dráha, je různá v závislosti na poloze značky v experimentální nádobě. Tento efekt se nejvýrazněji projevuje na okrajích snímku.

Pro odhad počtu bází neznámých značek je ještě nutné nalézt mapování mezi známou strukturou žebříčku, definovanou dvojicemi absolutní poloha značky

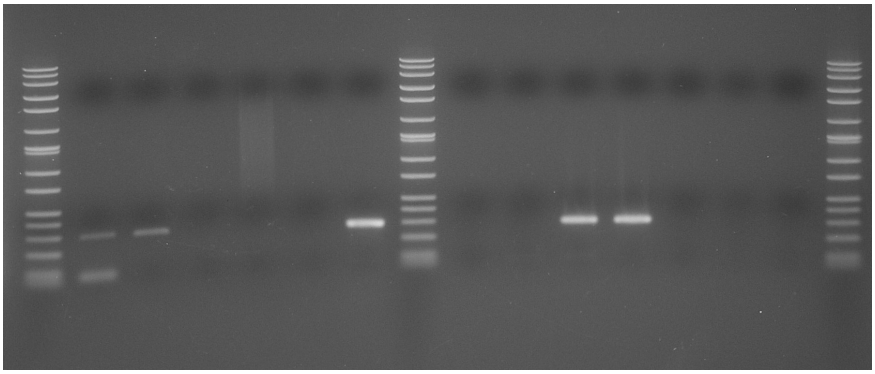
a její počet bází v nějakém předem anotovaném případě, a nově detekovaným žebříčkem. Polohy značek v nově detekovaném a původním žebříčku se nutně liší měřítkem - scale - a pevným posunutím - bias. Rovněž je nutné vzít v úvahu možnost, že detekční algoritmus některé značky žebříčku nedetekuje, nebo jsou do žebříčku navíc přiřazeny falešně positivní detekce. Proto je nutné optimální mapování mezi žebříčky získat jako řešení optimalizační úlohy.

Kapitola 3

Vstupní data a způsob vyhodnocování

3.1 Vstupní data

Vstupní data byla tvořena obrázky různých experimentů gelové elektroforézy, která však nebyly následně anotovány, resp. jejich anotace u obrázků chyběla. Obrázky z experimentů byly vytvořeny Národní referenční laboratoří pro lymeskou borreliózu ze Státního zdravotního ústavu. Z toho důvodu byla pro potřeby testování vytvořena anotace pro 105 obrázků, která byly nadále používány pro testování metod zpracování. Nedostatkem této osobní neexpertní anotace je to, že zejména u málo viditelných značek (malý kontrast mezi domnělou značkou a pozadím) byla volba, zda se pořád jedná o skutečnou značku, založena pouze na subjektivním zhodnocení. Je tedy rovněž pravděpodobné, že z důvodu neexpertní anotace byly metody detekce založené na učení částečně zatížené chybou vstupních dat. Rovněž nebyla vytvořena anotace počtu bází jednotlivých značek, ani typu použitého referenčního žebříčku.



Obrázek 3.1: Příklad z datasetu obrázků

3.2 Způsob vyhodnocování

Po provedení detekce libovolným algoritmem slouží k vyhodnocování výsledků detekce rozdělení značek do skupin True positive (TP), False positive (FP) a False negative (FN). Postup pro rozdělení je následující. Pro všechny značky v anotovaném obrázku (tj. ground-truth značky) je pro všechny nalezené značky vypočítána následující funkce:

$$\mathcal{V}(\mathbf{r_1}, \mathbf{r_2}) = \sqrt{\min\{\max(d_x, 0), \max(d_y, 0)}, \tag{3.1}$$

$$d_x = \frac{w - |r_{1,x} - r_{2,x}|}{w}, \tag{3.2}$$

$$d_y = \frac{h - |r_{1,y} - r_{2,y}|}{h}, \tag{3.3}$$

kde $\mathbf{r_1}, \mathbf{r_2}$ jsou polohy dané dvojice bodů. Tento způsob výpočtu byl vybrán proto, aby pro příliš vzdálené dvojice značek byla hodnota nulová a aby se nebyla změna v této funkci $\mathcal{V}$ v případě, kdy se značky v poloze liší o malé hodnoty. Parametry w, h určují práh, který odlišuje FP a TP případy. Ačkoli se některé značky liší v šířce, výška značek je velmi podobná v celém datasetu. Proto byla pro parametr h zvolena hodnota 15. Při experimentech v kapitole 4 byla pro jednoduchost nastavena $w = 25$, která po subjektivním vyhodnocení výsledků splňovala rozdělení detekcí na FP a TP. V kapitole 6 byla pak hodnota w měněna, a jako jeden ze způsobů vyhodnocování byla vykreslována závislost recall a precision (viz rovnice 3.6) na měnící se šířce prahu w.

Nejprve je vybrána pro každou z anotovaných značek nejbližší detekovaná značka $\mathbf{r_n}$, tj. taková, která maximalizuje pro aktuální anotovanou značku $\mathbf{r_r}$ funkci $\mathcal{V}(\mathbf{r_r}, \mathbf{r_n})$. Pokud je hodnota funkce pro nejbližší detekovanou značku

nenulová, pak je výsledek vyhodnocen jako TP, a hodnota TP se zvýší o $\mathcal{V}(\mathbf{r_r}, \mathbf{r_n})$, tj. hodnota TP neodpovídá přesně počtu True positive detekcí, ale je jeho dolním odhadem. Pokud je $\mathcal{V}(\mathbf{r_r}, \mathbf{r_n})$ nulová, pak se o 1 zvýší hodnota FN. Po provedení této operace pro všechny anotované značky $\mathbf{r_r}$ jsou zbylé detekované značky, pro které neexistovala anotovaná značka s kladnou $\mathcal{V}$, označeny jako FP a hodnota FP je zvýšena o hodnotu odpovídající tomuto počtu.

Při vyhodnocování výsledků detekcí je použita hodnota přesnosti:

$$\text{acc} = \frac{TP}{TP + FP + FN}. \tag{3.4}$$

Standardní způsob výpočtu přesnosti, například při klasifikační úloze, vychází navíc z True negative (TN) případů:

$$\text{acc} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3.5}$$

Kvůli špatnému definování True negative pro úlohu detekce bylo tedy v rovnici pro přesnost použití TN vypuštěno. Pro použití v ROC křivkách jsou pak počítány hodnoty recall, resp. precision následovně:

$$\text{recall} = \frac{TP}{TP + FN}, \ \text{precision} = \frac{TP}{TP + FP}. \tag{3.6}$$

Kapitola 4

Metody detekce

Pro detekování jednotlivých značek jsou v této práci použity 2 metody. První metoda je založena na korelačních filtrech typu MOSSE. Popis předzpracování dat pro tento typ filtru je v sekci 4.1.1, trénink filtru a zpracování jeho výstupu je popsán v sekci 4.1.2, resp. 4.1.3. Druhý způsob, popsaný v sekci 4.2, využívá kaskádního detektoru založeného na LBP deskriptorech.

4.1 Korelační filtry

Časta metoda pro detekci je korelace vstupního obrázku s malým hledaným vzorkem, a následnou analýzou výsledku korelace lze lokalizovat značky jako lokální maxima. Jedná se o relativně snadno implementovatelné metody, v nejjednodušším případě lze provádět korelaci s výstřižkem značky získáným z jednoho z obrázků. Problémem této základní metody je však velmi malá robustnost, neboť prostým „vystřižením“ filtru je pak takřka nemožné efektivně používat stejný filtr na obrázky z elektroforéz, kde se zkoumají například výrazně širší značky.

Postupů, jak navrhnout komplexní korelační filtr k úloze detekce anebo sledování, je více druhů, v této práci jsou zmíněny korelačí filtry typu ASEF (Average of Synthetic Exact Filters), popsané v [5] a [6], a korelační filtry typu MOSSE (Minimum Output Sum of Squared Error) popsané v [7]. Tyto metody ve velké míře využívají konvolučního teorému, tedy možnosti efektivně vypočítat korelaci dvou vstupních signálů jako jejich součin ve frekvenční

oblasti. Převod do frekvenční oblasti je navíc urychlen využitím algoritmu Rychlé Fourierovy transformace (FFT). Díky tomu tyto metody velmi rychle vypočítají exaktní korelační filtr na základě podílu Fourierova obrazů očekávané odezvy signálu a samotného vstupu. Pomocí různých postupů pak tyto metody kombinují exaktní filtry různých vstupních signálů do jediného robustnějšího filtru.

4.1.1 Předzpracování vstupních obrázků

Před aplikací filtrů je nutné provést předzpracování vstupních obrázků. Nejprve je provedeno převedení jasových hodnot obrázků do intervalu $\langle 0; 1\rangle$. Tímto se zajistí použitelnost programu pro různé reprezentace jasu v obrázku. Nechť f je pole reprezentující obrázek. Pak transformovaný obrázek $\tilde{f}_1$ je získán následovně:

$$\tilde{f}_1 = \frac{f - \min f}{\max f - \min f}, \tag{4.1}$$

kde operátory $\min, \max$ určují globální minimum, resp. maximum jasu v obrázku. Daná operace je po prvcích aplikována pro celé pole f.

Následně je provedena lokální normalizace celého obrázku. Důvodem je snaha zmenšit varianci jasu v rámci jednoho obrázku. Optimálním výsledkem operace by měl být sjednocený kontrast mezi značkou a pozadím nad celým zkoumaným obrázkem. Nechť f je vstupní obrázek. Lokální střední hodnota μ a variance σ^2 se získá pomocí konvoluce vstupního pole $\tilde{f}_1$ s gaussovským filtrem g_f, jehož konstrukce je popsaná například v [8]. Pro výpočet variance je nutno ještě získat odhad druhého obecného momentu μ_2:

$$\mu = \tilde{f}_1 * g_f, \tag{4.2}$$

$$\mu_2 = \tilde{f}_1^2 * g_f, \tag{4.3}$$

$$\sigma^2 = \mu_2 - \mu^2. \tag{4.4}$$

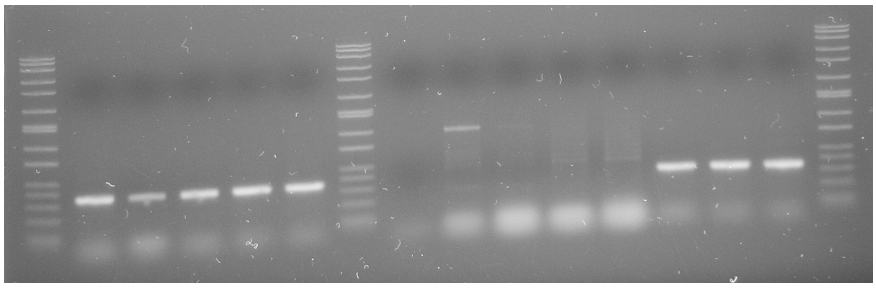
Operace rozdíl i mocnina jsou prováděny nad jednotlivými položkami polí. Operátor $*$ značí konvoluci. Následně je výstupní pole $\tilde{f}_2$ určeno následovně:

$$\tilde{f}_2 = \frac{\tilde{f}_1 - \mu}{\sigma + \epsilon}, \tag{4.5}$$

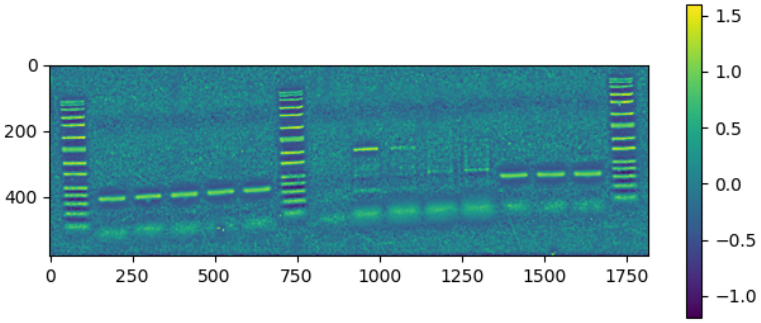
kde $\epsilon > 0$ je parametr zabraňující dělení nulou. Optimální výsledkem předzpracování je stav, kdy jsou hodnoty jasu v okolí značek přibližně stejné v rámci celého obrázku. Výsledek normalizace je výrazně závislý na volbě variance gaussovského filtru g_f. Obecně pro filtr g_f platí, že:

$$g_f(x, y) = c \cdot \mathrm{e}^{-\frac{x}{2\sigma_x^2} - \frac{y}{2\sigma_y^2}}, \tag{4.6}$$

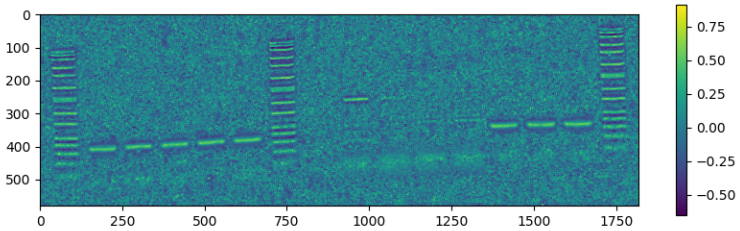
kde c je normalizační konstanta a σ_x, σ_y variance filtru v osách x, y. Při příliš nízké hodnotě variance dojde k přílišnému potlačení detailů v obrázku, neboť pro variance σ_x^2, σ_y^2 blížící se 0 se odezva filtru stává Diracovým impulsem. Pro vysoké hodnoty variancí pak klesá význam celé normalizace, neboť lokální střední hodnota se pro σ_x^2, σ_y^2 blížící se nekonečnu blíží průměru jasu v celém obrázku. Experimentálně byly zvoleny hodnoty směrodatných odchylek (tj. odmocnin variancí) $\sigma_x = 5, \sigma_y = 9$, které pro zkoumané vzorky vracely dostatečně kvalitní výstupy normalizace.



Obrázek 4.1: Vstupní obrázek před normalizací



Obrázek 4.2: Výsledek lokální normalizace s parametry $\sigma_x = 5, \sigma_y = 9$



Obrázek 4.3: Výsledek lokální normalizace s parametry $\sigma_x = 2, \sigma_y = 3$

4.1.2 Konstrukce korelačního filtru

Konstrukce korelačního filtru probíhá pomocí známých poloh středů značek. Daný obrázek f, nad kterým již byly provedeny kroky předzpracování popsané výše, je reprezentován pomocí ground-truth značek následovně. Nechť g_I je 2D pole o rozměrech f, která má na pozicích reálných značek hodnotu 1, všude jinde 0. Pak cílové ground-truth pole g je získáno pomocí konvoluce g_I s gaussovským filtrem popsaným v rovnici 4.6, postup je popsán v [5]. Exaktní korelační filtr je takový filtr h, pro který platí:

$$g = f \star h, \tag{4.7}$$

kde $\star$ je operátor korelace. Ve Fourierově doméně tato operace přechází ve vztah (dle [5]):

$$G = F \odot H^*, \tag{4.8}$$

kde F, G, H jsou výsledky Fourierovy transformace daných polí, $\odot$ reprezentuje násobní položku po položce (element-wise) a operátor * značí operaci komplexního sdružení. Pak dle [5] je pro N dvojic vstupních obrázků a ground-truth polí f_i, g_i a pro jejich Fourierovy obrazy získán filtr typu ASEF následovně:

$$H^*_{\text{ASEF}} = \frac{1}{N} \sum_{i=1}^{N} \frac{G_i \odot F_i^*}{F_i \odot F_i^* + \epsilon}. \tag{4.9}$$

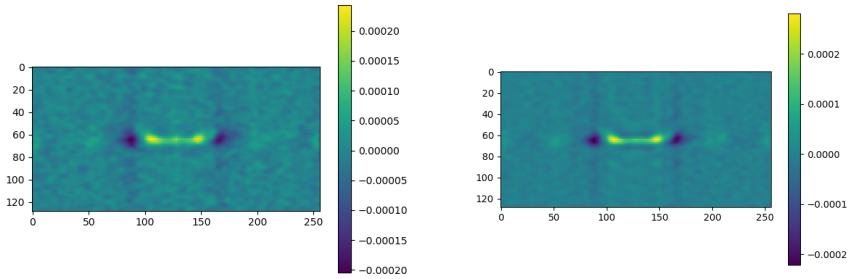
Parametr $\epsilon > 0$ zabraňuje dělení nulou. Obdobným způsobem je filtr typu MOSSE zkonstruován dle [7] jako:

$$H^*_{\text{MOSSE}} = \frac{\sum_{i=1}^{N} G_i \odot F_i^*}{\sum_{i=1}^{N} F_i \odot F_i^* + \epsilon}. \tag{4.10}$$

Je patrné, že zatímco filtr typu ASEF je získán prostým průměrováním několika exaktních filtrů, filtr typu MOSSE je výsledkem optimalizační úlohy popsané v [7].

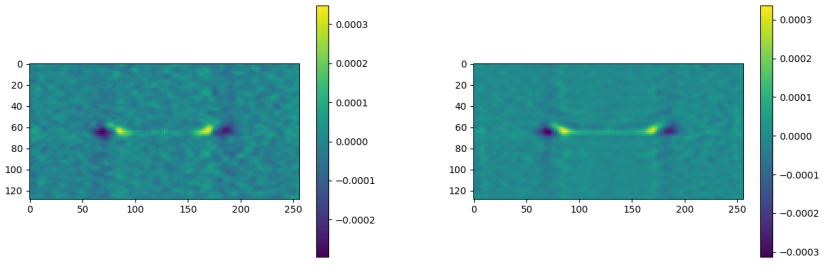
Protože vstupní obrázky nabývají velkých rozměrů, filtry nejsou trénovány nad celým obrázkem v jediném kroku. Místo toho je kolem každé značky v ground-truth poli zkonstruován obdélník o rozměrech 256x128, který je následně z polí „vystřihnut“. Nad těmito zmenšenými poly je následně prováděna Fourierova transformace. Generalizováním velikosti okna se zmenšilo množství dat nutných pro uložení filtru a zjednodušilo se průměrování dat, neboť pole reprezentující výstupy Fourierovy transformace (konkrétně Rychlé Fourierovy transformace FFT) měly všechny stejný rozměr nezávisle na rozměrech vstupního obrázku. Velikost obdélníku byla zvolena tak, aby bez problému zahrnovala značky a jejich okolí ve všech používaných testovacích obrázcích.

Při práci s korelačními filtry se ukázalo, že v datasetu některé obrázky obsahují typ značky, která je širší než běžný typ značky, při zachování téže výšky. Protože tento typ nebyl v datasetu dostatečně zastoupen, výsledný filtr nebyl schopen později správně detekovat tento typ značek. Z toho důvodu byl pro širší typ značky vytvořen separátní korelační filtr. Výsledné filtry, trénované v případě filtrů pro běžný typ značky na 65 vstupních obrázcích, a v případě širšího typu značky na 15 obrázcích, byly následující.



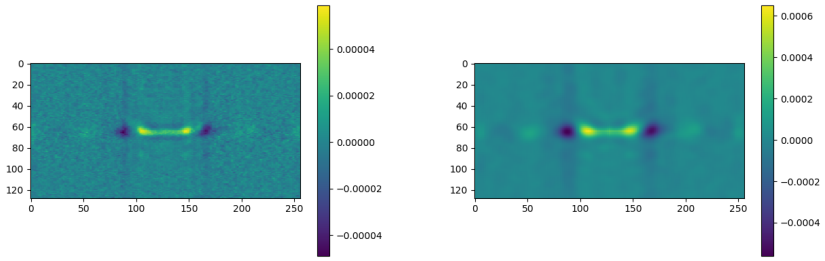
(a) : Filtr ASEF **(b) :** Filtr MOSSE

Obrázek 4.4: Korelační filtry pro běžný typ značky



(a) : Filtr ASEF **(b) :** Filtr MOSSE

Obrázek 4.5: Korelační filtry pro širší typ značky



(a) : MOSSE filtr pro $\sigma = 1$ **(b) :** MOSSE filtr pro $\sigma = 5$

Obrázek 4.6: Příklady MOSSE filtrů s jinými parametry σ

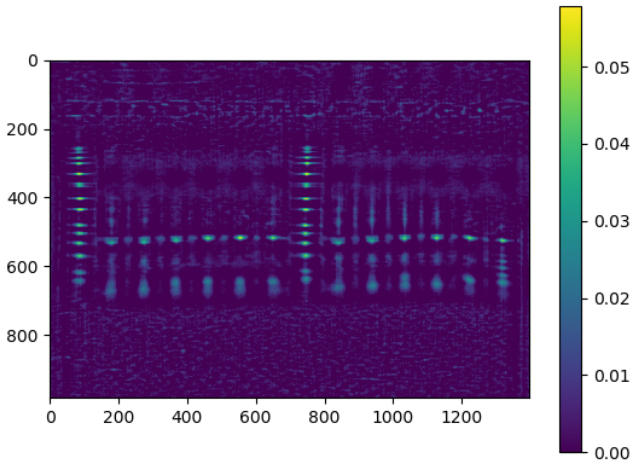
Na obou případech 4.4b, 4.5b je patrné, že se potvrdila teze z článku [7], tj. filtr typu MOSSE vyžaduje výrazně méně trénovacích vzorků, aby se ve výsledném filtru dostatečně potlačil šum. Navíc je na obrázcích vidět, že výsledný filtr skutečně kopíruje svým tvarem obecný tvar značky, kterou má detekovat, tedy je schopen detekovat i zkreslené a zašuměné značky. Při testech bylo pole ground truth konstruováno symetrickým gaussovským filtrem s $\sigma = \sigma_x = \sigma_y = 2.5$. Hodnota byla takto zvolena, protože pro nižší hodnoty již byl výsledný filtr výrazněji zašuměný (viz 4.6a), pro vyšší hodnoty se pak ztrácela přesnost detekce v žebříčcích a na obrázcích, kde jsou značky blízko sebe (dochází ke splývání odezev filtru na blízké značky).

Ačkoli to v [7] není explicitně specifikováno, k výpočtu filtru není nutné vybírat pouze positivních dat. Správnou manipulací s cílovým ground-truth polem lze ve filtru vynutit, aby například pro daný typ vzorků měl odezvu zápornou. V takovém případě je pole ground-truth g zkonstruováno opět pomocí rozmázání gaussovským filtrem, ale v poli impulsů g_I jsou tyto negativní vzorky reprezentovány hodnotou -1. Proto byl proveden experiment s doplněním trénovacích obdélníků o kritické body na některých obrázcích, které při předchozích testech způsobily vznik falešně positivních detekcí. Protože však takových negativních vzorků bylo oproti positivním vzorkům vybráno výrazně méně, změna na výsledném filtru byla zanedbatelná. Pokud by byly předem kritické body označeny při konstrukci anotací obrázků tak, aby počty positivních a negativních vzorků nebyly o několik řádů rozdílné, je pravděpodobné, že by filtr po tréninku na datasetu, který by obsahoval také negativní vzorky, dosáhl lepších výsledků.

4.1.3 Zpracování výstupu korelace

Obvyklý výstup korelace g vstupního obrázku f s MOSSE filtrem h je na obrázku 4.7. Výsledek korelace g je následně pomocí konvoluce filtrován gaussovským filtrem s velikostí kernelu 2 pro potlačení drobných špiček, které by byly problematické při výběru extrémů. Následně je provedeno prahování. Hodnota prahování θ je dána konvexní kombinací 85% percentilu a maxima v heatmap, dle rovnice:

$$\theta = \psi \cdot \max g + (1 - \psi) \cdot \mathrm{perc}_{85}\, g. \tag{4.11}$$



Obrázek 4.7: Výsledek korelace s MOSSE filtrem

Hodnoty $g(x, y) > \theta$ se zachovají, ostatní jsou vynulovány. Následně je nad oprahovaným g proveden následující iterativní postup. Nejprve je nalezeno maximum v g. Nechť maxima nabývá pole g v bodě x, y, pak se dvojice (x, y) uloží jako extrém. Pak je z tohoto bodu provedeno prohledávání do šířky, kdy každý bod v osmiokolí, který má nenulovou hodnotu heatmap g menší než aktuální bod g, je zařazen do fronty. Po přiřazení všech bodů z okolí x, y do fronty je $g(x, y)$ nastaveno na nulu. Po dokončení prohledávání je vybráno nové maximum, postup se opakuje, dokud existuje nenulové maximum.

Kromě volby závislosti θ na vybrané konvexní kombinaci veličin obrázku byly testovány také varianty s použitím $\theta = \psi \cdot \max g$, resp. $\theta = \psi$ i použití

samotného percentilu $\theta = \psi \cdot \text{perc}_i\, g$. Při použití max jsou výsledky detekcí silně závislé na největší špičce v heatmap, při použití pevného prahování pro všechny obrázky je pak nutný apriorní předpoklad, že charakter heatmap je takřka identický při všech možných vstupech. Ačkoli samostatné použití vybraného percentilu dávalo dobré výsledky ve smyslu rovnice 3.4 na testovacích obrázcích, při zatížení šumem nebo rozmázání došlo ke změně v rozložení hodnot v g tak, že nakonec práh θ přesáhl svou hodnotou maximum v g (důvodem bylo, že optimální hodnota ψ při samotném použití percentilu byla větší než 1). Při použití konvexní kombinace je zajištěno, že práh θ nepřesáhne maximum g, a zároveň se zachová kvalita použití percentilu pro výpočet.

4.1.4 Výběr optimální hodnoty prahu

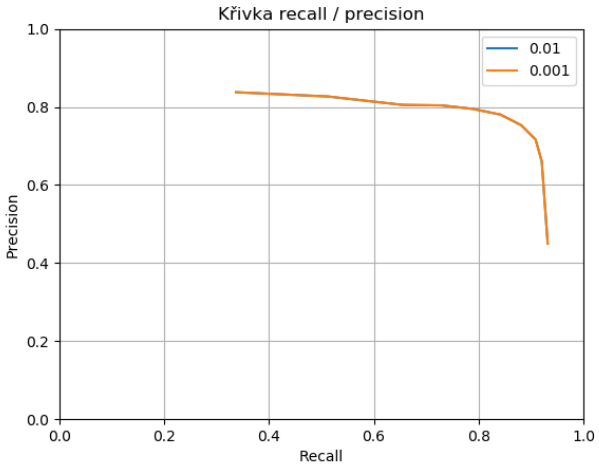
Hodnota ψ, použitá při výpočtu prahování θ daného obrázku, má hlavní vliv na rozdělení, zda se kladná část heatmap bude interpretovat jako nalezená značka, nebo nikoli. Pro její výběr bylo testovány různé hodnoty následujícím způsobem.

Nejprva byla vybrána hodnota ψ_i pro otestování. Následně byly provedeny detekce nad skupinou anotovaných obrázků, které odpovídaly typem značky trénovanému filtru, pro každý obrázek byly pak pomocí známého rozložení skutečných značek vypočteny hodnoty $TP_{i,j}, FP_{i,j}, FN_{i,j}$ dle rovnice 3.2, kdy j odpovídá indexu obrázku ve skupině. Následně byly vypočteny globální hodnoty příslušných veličin nad celou testovací skupinou pro dané ψ_i:

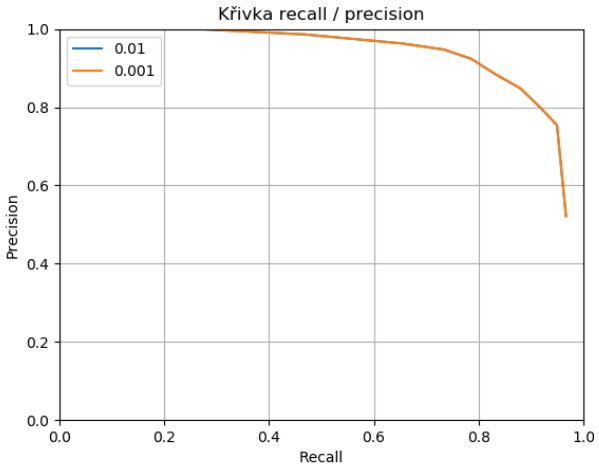
$$TP_i = \sum_j TP_{i,j},\ FP_i = \sum_j FP_{i,j},\ FN_i = \sum_j FN_{i,j}. \tag{4.12}$$

Z těchto akumulovaných hodnot pak je vynesena ROC křivka, kdy na ose x je vynesena hodnota recall, na ose y je vynesena hodnota precision, jejichž výpočty jsou popsány rovnicemi 3.6. Jako optimální hodnota prahování je zvoleno takové ψ_i, pro které je vzdálenost ROC křivky od bodu $(1, 1)$, reprezentujícího ideální detektor, minimální.

Optimální threshold ψ byl pro běžnou velikost značky zvolen na hodnotu $\psi = 0.45$, pro širší typ značky byla zvolena hodnota $\psi = 0.4$, dle výsledků testování. Množina obrázků pro trénink prahování běžného typu značky obsahovala 65 vzorků, pro širší typ značky množina obsahovala 15 obrázků, jednalo se o stejné množiny obrázků použití pro trénink korelačního filtru dle sekce 4.1.2. ROC křivky pro oba experimenty jsou vyneseny níže.



Obrázek 4.8: ROC křivka pro běžný typ značky

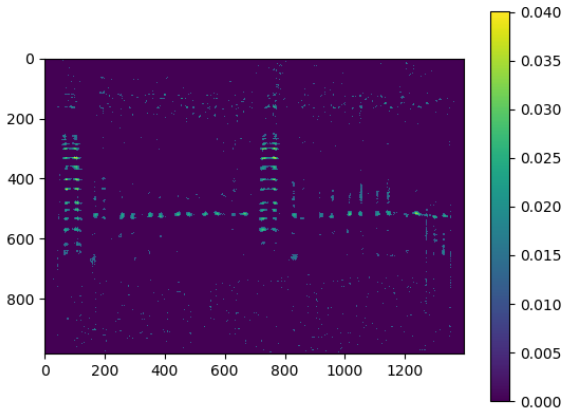


Obrázek 4.9: ROC křivka pro širší typ značky

Při testování byl navíc zkoumán vliv regulačního parametru ϵ v rovnici 4.10. Pro oba typy značky byly vytvořeny MOSSE filtry na stejné množině obrázků, s jiným nastavením parametru ϵ. Poté byl vybrán optimální threshold pro dané MOSSE filtry, a jednotlivé ROC křivky byly vyneseny do stejných grafů. Tyto křivky pro různé parametry ϵ jsou v grafu vyneseny různými barvami. Ukázalo se však, že i přes řádový rozdíl ve volbách ϵ nedošlo ke změnám filtru, které by se projevily na charakteru ROC křivky (křivky pro různé ϵ zcela splývají). Nadále byly používány filtry s nastavením $\epsilon = 0.001$.

4.1.5 Kombinace dvou typů filtru

Ačkoli jsou pro různé typy značek zkonstruovány unikátní korelační filtry, stále je nutné vytvořit postup, kterým je vybírán filtr pro daný typ obrázku, bez apriorní znalosti typu značky od operátora. Pokud byl aplikován chybný filtr, pak docházelo ke zdvojování detekcí na krajích detekované značky, viz obr. 4.10. Protože tento fakt platil v obou případech chybného použití filtru (tj. filtr pro širší typ značky na obrázku s běžným typem značek, a naopak), bylo možné provést test pro rozlišování typů značek na základě výsledků detekce pomocí obou filtrů. Pokud by totiž kolem detekcí byly zkonstuovány obdélníky, přibližně aproximující rozměry značky pro značku danou vybraným filtrem, pak v případě korelace se špatným typem filtru bude výrazně větší množství značek přesahovat svou částí do jiné detekované značky.



Obrázek 4.10: Zdvojování detekovanýcg extrémů na krajích značek v heatmap při použití nevhodného filtru

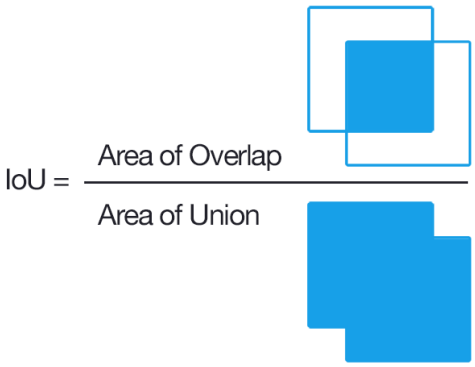
Pro určení tohoto přesahu je vypočtena hodnota IoU (Intersect over Union). Tedy po aproximaci značek pomocí obdélníků, jejichž středy odpovídají pozicím detekcí, je pro každou dvojici značek vypočtena plocha průniku obou obdélníků I a plocha jejich sjednocení U, viz obrázek 4.11. Pak hodnota IoU je rovna:

$$\mathrm{IoU} = \frac{I}{U}. \tag{4.13}$$

Pokud hodnota IoU přesáhne práh δ, pak je dvojice značek vyhodnocena jako kolize. K rozlišení typu značky je ve vstupním obrázku slouží poměr kolizí ku celkovému počtu detekcí. Při zahájení detekce nového obrázku se pak postupuje následovně. Vstup je korelován ve frekvenční doméně (pro přechod je použita FFT) s filtry pro oba typy značek. Nad výslednou dvojicí

heatmap, za použití prahovacích hodnot ψ příslušných danému filtru je provedeno prohledávání a vybrání extrémů, značících detekce. Pro každou dvojici detekcí je pak vypočtena hodnota IoU. Pokud ta přesáhne threshold δ, je inkrementováno počítadlo kolizí. Následně je vybrán výsledek toho filtr, jenž má menší poměr kolizí ku počtu všech detekcí.

Rozměry aproximujících obdélníků byly vybrány na hodnoty 160×20 pro běžný typ značky, a rozměry 240×20 pro širší typ značky, tak aby aproximující obdélníky obsahovaly rovněž i blízké okolí značek. δ bylo stanoveno na 0.4. Při testech na ručně anotované množině 105 obrázků, i na doplněné neanotované množině 20 obrázků, tento postup neselhal. Při rozšíření datasetu o další typ značek by bylo potřeba navrhnout robustnější postup, jak rozhodnout o typu použitého filtru.



Obrázek 4.11: Vizualizace výpočtu IoU, [9]

4.2 Kaskádní detektor

Jako detekční metoda, založená na strojovém učení, je použit kaskádní detektor, popsaný v [10]. Princip je založen na sliding window, tedy celý vstupní obrázek je procházen tak, že je po něm posouváno subokénko pevných rozměrů, a každé takové subokénko je následně klasifkováno. Pro řešení detekce různě škálovaných objektů je obrázek několikrát zmenšen, nebo i zvětšen, aby se do dané pevné velikosti detekčního subokénka vešly různé velikosti hledaných objektů.

Článek [10] formuluje názvosloví a obecný návrh kaskádního detektoru následovně. Kaskádní detektor se skládá z kaskády silných klasifikátorů. Každé subokénko je pak klasifikováno postupně jednotlivými klasifikátory kaskády. Pokud klasifikátor libovolné úrovně kaskády vyhodnotí subokénko

jako část pozadí, zbytek kaskády se již nevyhodnocuje a výsledkem klasifikace subokénka je pozadí. Tedy aby byla výsledkem klasifikace značka, musí subokénko úspešně projít všemi testy v kaskádě. Díky této struktuře lze trénovat rychlé detektory, jež v prvních fází detekce „zahodí“, většinu okének obsahujících pozadí, a složitější klasifikátory pozdějších fází kaskády jsou využity pouze u složitějších příkladů.

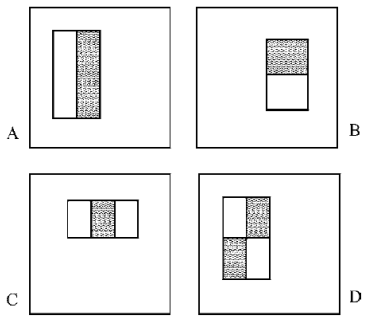
Jednotlivé silné klasifikátory jsou tvořeny soustavou slabých klasifikátorů, které jsou trénovány do jednoho silného klasifikátoru pomocí algoritmu AdaBoost. K tréninku je třeba jak positivních, tak negativních subokének. V případě zde popsaných metod se slabý klasifikátor sestává z deskriptoru a způsobu, jak převést výstup deskriptoru na třídu objektu (pozadí nebo značka). AdaBoost vybírá nejlepší slabý klasifikátor pro zadané váhy vstupních dat, spolu s parametry prahování pro minimalizaci chyby. AdaBoost rovněž vybírá koeficienty lineární kombinace, která spojuje slabé klasifikátory do klasifikátoru silného.

4.2.1 Deskriptory založené na Haar wavelets

Detektor Viola-Jones popsaný v [10], využívá jednoduché obdélníkové deskriptory, viz 4.12. Pro libovolné vstupní okénko je hodnota deskriptoru rovna rozdílu sumy jasů v bílých obdélnících a sumy jasů v černých obdélnících. Pro efektivní výpočet těchto hodnot je navíc použita reprezentace vstupního obrázku jako tzv. integrální obrázek. Pro vstupní obrázek f je integrální obrázek h takový, který splňuje:

$$h(i,j) = \sum_{s=0}^{i} \sum_{t=0}^{j} f(s,t). \tag{4.14}$$

Pomocí této reprezentace obrázku pak pro vyhodnocení sumy hodnot jasu v libovolném segmentu deskriptoru stačí čtyřikrát přistoupit do pole h, dle [10].



Obrázek 4.12: Obdélníkové deskriptory dle [10]

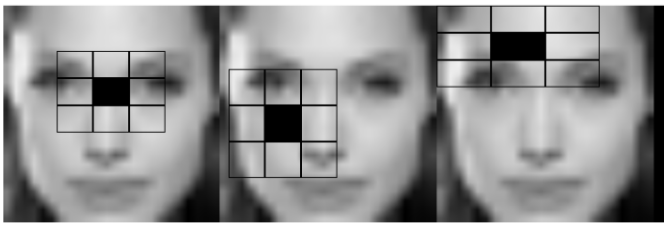
Pro trénink kaskádního klasifikátoru je používána knihovna OpenCV [11]. Bohužel se ukázalo, že trénink pro zvolenou velikost detekčního okénka (66x26) je v rozumném čase neproveditelný. V postulovaném příkladu v [10] je použito okénko o velikosti 24x24 pixelů. V tu chvíli již počet unikátních obdélníkových deskriptorů dosahuje 160 000. Pro vyšší velikosti okének již počet různých deskriptorů roste velmi rychle. Důvod pro vybrání detekčního okénka takto velkých rozměrů vyplynul z úvodní anotace získaných obrázků. Obrázky byly vzhledem ke směřování detekčního systému k použití korelačních filtrů anotovány pouze označením středu dané značky, což bylo pro trénink MOSSE filtru dostačující. Pro použití v tréninku kaskádního klasifikátoru by bylo vhodnější obrázek segmentovat, nebo označit značky pomocí bounding box. Kvůli znalosti pouhého středu značky je obtížné volit velikost subokénka tak, aby obsahovalo celou značku, a zároveň aby okénko nebylo zbytečně velké. Variance v šířce značky měla rovněž vliv na volbu velikosti okénka. Je potřeba vybrat takovou velikost okénka, do kterého se vejde širší typ značky včetně alespoň blízkého okolí, a zároveň aby pro užší značky do subokénka nezasahovaly i další značky. Kvůli velkému rozlišení původního obrázku bylo rovněž před vybráním tréninkových subokének provedeno snížení rozlišení na polovinu. Ačkoli se tím zmenšily možnosti výběru klasifikátoru a tím pádem i výsledná kvalita detektoru, bylo k tomuto kroku přistoupeno, protože použitá implementace pro trénink detektorů neumožňovala nahradit vyhledávání optimálního slabého klasifikátoru přes všechny možné například náhodným výběrem.

4.2.2 Deskriptory založené na LBP

Stejná implementace OpenCV [11], použitá pro trénink detektoru Viola-Jones, umožňuje použít též deskriptory založené na Local Binary Patterns [12]. Operátor LBP slouží k zakódování informace o poměru mezi jasu nějakého vybraného pixelu a jasů 8 pixelů v jeho lokálním okolí, které jsou seřazeny. Výstupem je 8-bitové číslo, kdy platí, že pokud je hodnota jasu i-tého souseda vyšší než hodnota jasu centrálního pixelu, pak je i-tý bit výstupu 1, jinak je 0. Výstup proto nabývá nejvýše hodnoty 255, když jsou všechny pixely v okolí světlejší než pixel centrální, hodnoty 0, pokud je centrální pixel jasnější než všechny pixely v lokálním okolí.

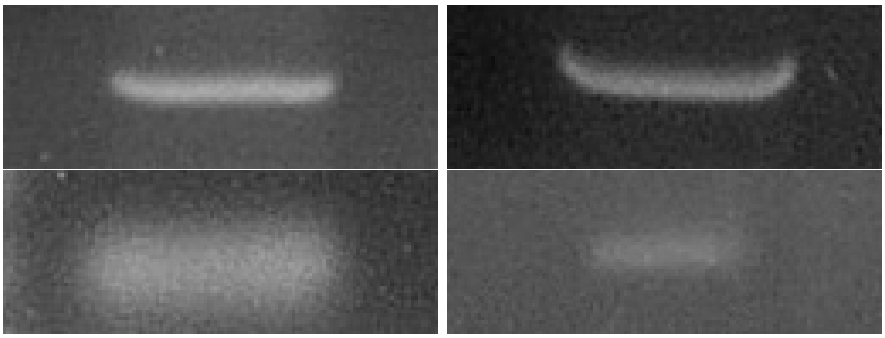
Konkrétní varianta deskriptorů v OpenCV implementaci je popsaná v [13], tzv. Multi-Block Local Binary Patterns (MB-LBP). Tato varianta se neomezuje pouze na jednotlivé pixely, ale porovnává průměrné hodnoty jasu nad většími segmenty obrázku. K výpočtu průměrného jasu opět lze použít integrálního obrázku. Narozdíl od obdélníkových deskriptorů, použitých v detektoru dle [10], slabý klasifikátor užívající MB-LBP deskriptorů nespecifi-

kuje svůj výstup pomocí prahování. Místo toho, dle [13] je pro každou z 256 diskrétních hladin deskriptoru naučen vlastní výstup.

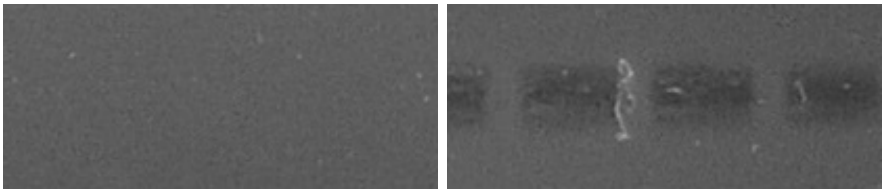


Obrázek 4.13: Ukázka možných MB-LBP deskriptorů, [14]

Pro trénink bylo vygenerováno pomocí vystřihování částí z olabelovaných obrázku 2500 positivních vzorků a 9000 negativních vzorků. Parametry detekčního okénka byly stejné jako v případě detektoru založeného na Haar features. Positivní vzorky byly získány vystřižením okénka dané velikosti vycentrováno na střed anotovaných značek, viz obr. 4.14. Negativní vzorky byly získány výběrem nahodných okének, které uvnitř sebe značky neobsahovaly, nebo je obsahovaly zcela okrajově, viz obr. 4.15.



Obrázek 4.14: Příklady positivních okének pro trénink kaskádního detektoru



Obrázek 4.15: Příklady negativních okének pro trénink kaskádního detektoru

Problémem tohoto generování negativních okének je však velké množství okének vychýlených v okolí značek, které detektor vyhodnotí positivně. Varianta řešení, založená na agregaci značek pomocí morfologických operací, je

popsána v sekci 4.2.3. Rovněž byla zkoušena varianta s doplněním negativního datasetu o takto vychýlená okénka, které je popsána v sekci 4.2.4.

4.2.3 Postprocessing výstupu kaskádního detektoru s MB-LBP deskriptory

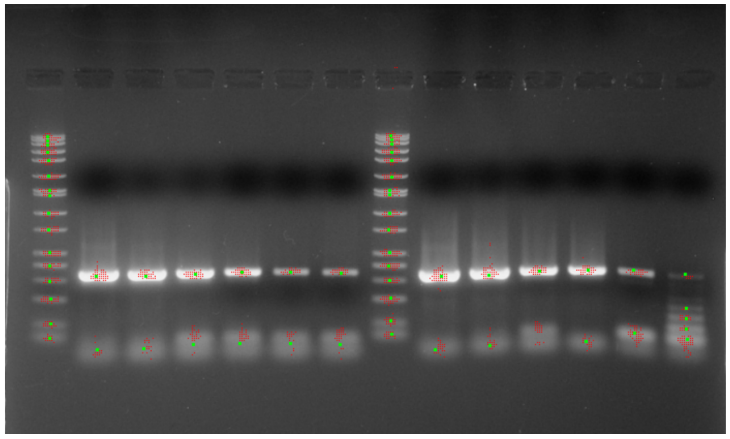
Samotná knihovna OpenCV umožňuje provádět multiscale detekci. Tedy detekce je prováděna nad zvětšeným a zmenšeným obrázkem tak, aby byly pokryty subokénka všech velikostí až do stanovených limitů. Problémem naučeného detektoru se ukázal fakt, že detektor vyhodnocoval velké množství okének v blízkosti značek jako positivní, nikoli pouze okénko vycentrované na střed značky, viz obr. 4.16. Ačkoli díky tomu je každá značka obvykle detekována a označena alespoň v jednom subokénku, počet positivních okének výrazně převyšuje počet značek, a je potřeba nějakým způsobem zredukovat blízká positivní okénka do jednotlivých reprezantací značek. K vyřešení tohoto problému byla použita morfologie. V binárním poli o velikosti obrázku byla položkám v souřadnicích, odpovídajícím středům jednotlivých positivních okének, přiřazena 1. Následně byla provedena dilatace dle [15] se strukturním prvkem S_1:

$$S_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \tag{4.15}$$

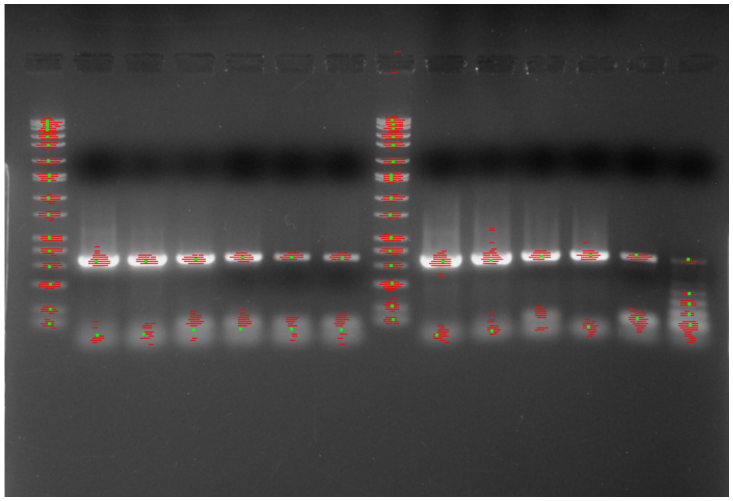
aby došlo k propojení středů okének ve směru osy x, a zamezilo se tím vzniku zdvojených detekcí náležejících jedné značce. Aplikace tohoto stukturního prvku je na obrázku 4.17 Pro spojení detekovaný značek v směru osy y je použito uzavření se strukturním elementem S_2:

$$S_2 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}. \tag{4.16}$$

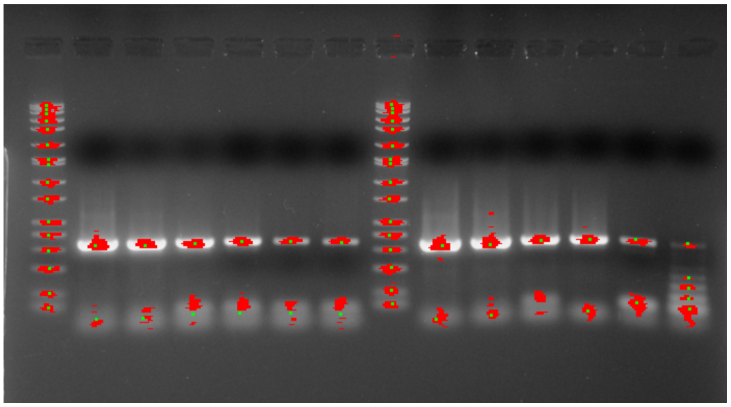
Důvodem pro použití uzavření je zamezení toho, aby se zbytečně nespojovaly blízké značky v žebříčcích, a zároveň aby se propojily detekce náležející jedné značky. Výsledek uzavření je v obrázku 4.18 Je patrné, že ačkoli dojde ke spojení linií náležejících jedné značce, naneštěstí rovněž dochází ke spojení a vytvoření „blobů“ v horních částech žebříčků, kdy ani eroze aplikovaná v druhé části uzavření nedokáže napravit toto chybné spojení. Pro rozdělení těchto „blobů“ se použije segmentace rozvodím (Watershed segmentation, [15]). Pro vyřešení je binární obrázek převeden na spojitý pomocí vzdálenostní transformace, tedy hodnota pixelu transformovaného pole je rovna eukleidovské vzdálenosti pixelu od nejbližšího pixelu pozadí dle [16] (viz obr. 4.19 a 4.20). Po segmentaci do skupin je provedeno prahování pro odstranění chybně detekovaných značek (takových, které měly málo sousedních falešně positivních značek). Pokud pak segmentované skupiny obsahují více pixelů než minimální threshold τ, je výsledek segmentace vyhodnocen jako značka, a jako její poloha je uloženo těžiště segmentované skupiny. Výsledek celé procedury je na obrázku 4.21



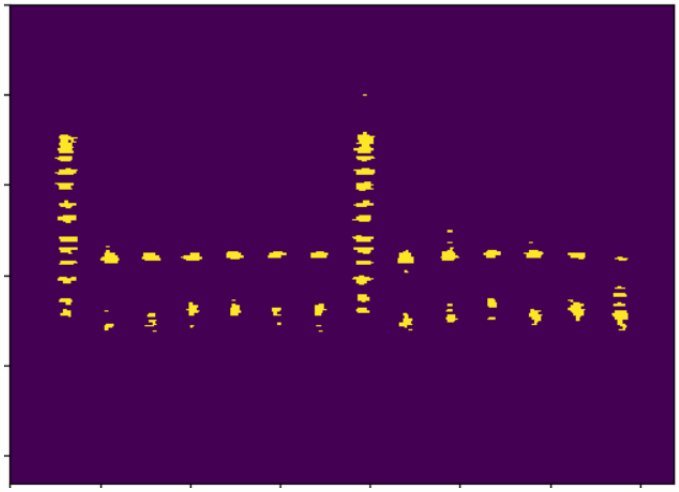
Obrázek 4.16: Výstup samotné multiscale detekce, červeně označeny detekce, zeleně označeny anotace



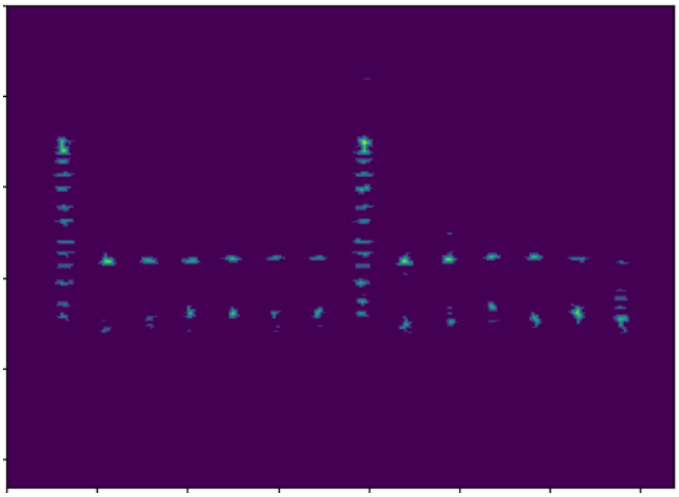
Obrázek 4.17: Výsledek dilatace se strukturním prvkem S_1



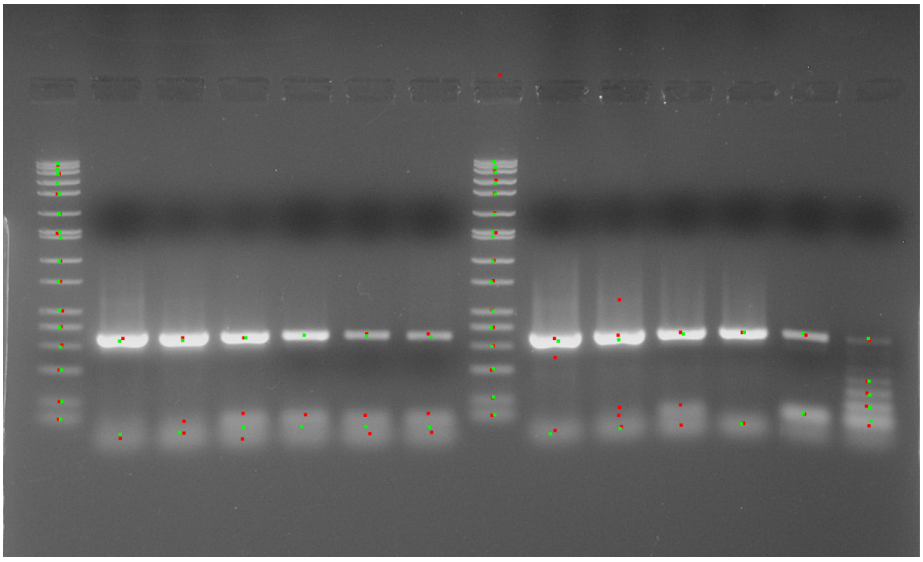
Obrázek 4.18: Výsledek uzavření se strukturním prvkem S_2



Obrázek 4.19: Vstupní binární obrázek před vzdálenostní transformací

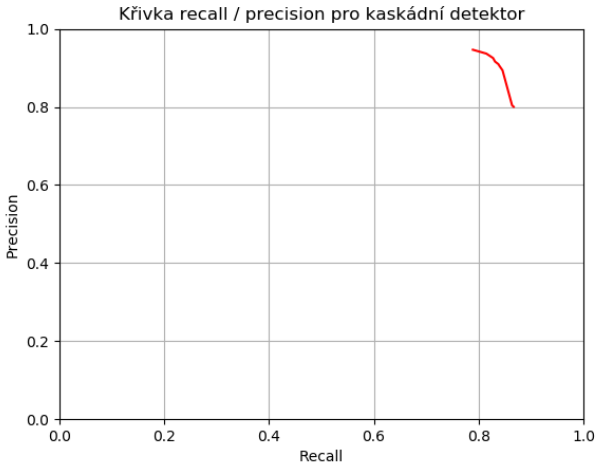


Obrázek 4.20: Výsledek vzdálenostní transformace

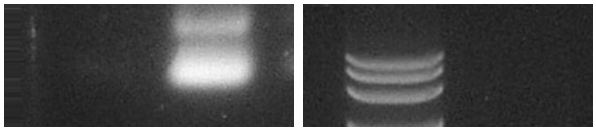


Obrázek 4.21: Výsledek detekce po provedení segmentace rozvodím, zeleně označeny reference, červeně detekce

Různé varianty thresholdu τ byly porovnány pomocí ROC křivky stejným způsobem, jako v případě nastavování učitelných parametrů u MOSSE filtrů. Dle této křivky zobrazené na obrázku 4.22 byla stanovena optimální volba $\tau = 11$ za stejných podmínek jako v případě výběru optimální hodnoty prahování MOSSE filtru.



Obrázek 4.22: ROC křivka kaskádního detektoru pro threshold τ

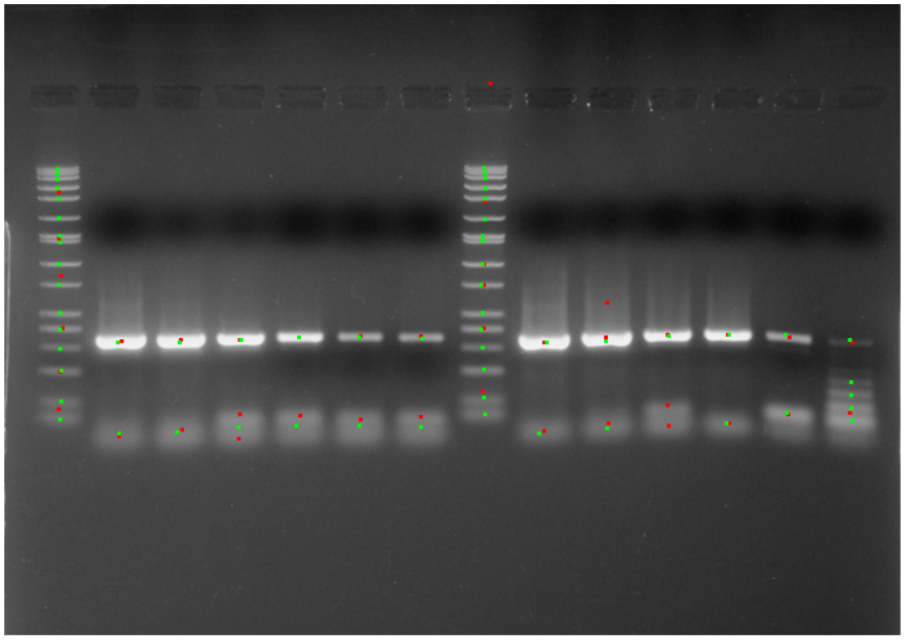


Obrázek 4.23: Příklady nových negativních okének pro trénink kaskádního detektoru

4.2.4 Rozšíření negativního datasetu

Jako druhý pokus k vyřešení problému s velkým množstvím positivních subokének v okolí anotovaných značek bylo rozšíření negativního datasetu. V plánu bylo přidat do datasetu takřka totožná subokénka jako v případě obr. 4.14, kdy by však byl střed značky vychýlen mimo střed okének, viz obrázky 4.23.

Teoreticky se očekávalo, že se detektor naučí diferencovat subokénka tak, aby okénka s vychýlenými středy značek byly klasifikátory v kaskádě odmítnuty. Zde se však opět projevil problém v anotaci značek. Při generování těchto vychýlených okének se posun prováděl o několik různých vychýlení od vycentrované pozice okénka. Detektor se trénoval za využití 14000 negativních vzorků a stejných 3500 positivních vzorků jako v sekci 4.2.2. Výsledný detektor byl pak testován na stejných obrázcích, pro porovnání je uveden výsledek na stejném obrázku jako výše, viz obr. 4.16. Výsledky detekce za stejných podmínek na stejném obrázku jsou zobrazeny na obr. 4.24. Je patrné, že schopnost detekovat značky se výrazně zhoršila, jedním z důvodů tohoto problému může být výsledná nevyváženost počtu positivních a negativních tréninkových vzorků. Daný výsledek je výstupem implementace OpenCV, bez žádných následujících prahování nebo morfologických operací.



Obrázek 4.24: Výsledek samotné detekce detektorem s MB-LBP deskriptory trénovaném na rozšířeném datasetu, zeleně označeny reference, červeně detekce

Přes opakované pokusy se nedařilo natrénovat klasifikátor na rozšířeném datasetu na uspokojivou úroveň. Jedním z možných důvodů selhání je špatné anotování dat ve smyslu absence bounding boxů. V případě anotace pomocí bounding boxů by bylo snadné znormalizovat scale a poměr stran všech různých typů značek do jediného subokénka. Kvůli neschopnosti zlepšit výsledky takto trénovaného detektoru byl nadále používán kaskádní detektor s MB-LBP deskriptory a morfologickým post-processingem.

Kapitola 5

Zpracování nalezených značek

Po detekci značek jsou na získaná data aplikovány algoritmy pro jejich zpracování. Nejprve je provedeno rozdělení značek do lanes, pomocí postupu v sekci 5.1. Poté je určen odhad rotace obrázku a provedena její korekce postupem v sekci 5.2. Po rozdělení do lanes jsou na ty lanes, které obsahují žebříčky, aplikovány algoritmy pro fitování nalezených žebříčků do referenčního žebříčku, postup popsaný v sekci 5.3. Toto fitování spočívá z úvodním odhadu afinní funkce pro transformaci souřadnic mezi detekovaným a nalezeným žebříčkem, a následném zpřesnění tohoto vztahu optimalizačními metodami. Po provedení fitování jsou žebříčky použity pro provedení korekce chyb způsobených nehomogenitou proudu, dle sekce 5.4. Na závěr je proveden odhad počtu bází neznámých značek dle sekce 5.5.

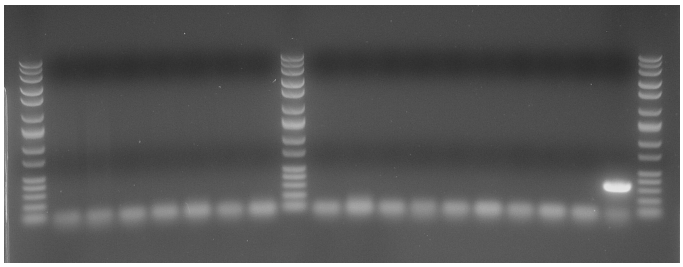
5.1 Rozdělení značek do žebříčků

Po získání detekovaných značek je nejprve provedeno jejich rozdělení do lanes, tedy skupin značek, které leží v jednom sloupcu. Rovněž jsou vybrány lanes, které jsou považovány za žebříčky. K tomu je každá značka reprezentovaná v 1D poli p pomocí projekce svého detekovaného středu do osy x. Nechť (x_i, y_i) jsou souřadnice středů značek. Pak pole impulsů p_I je definováno následovně:

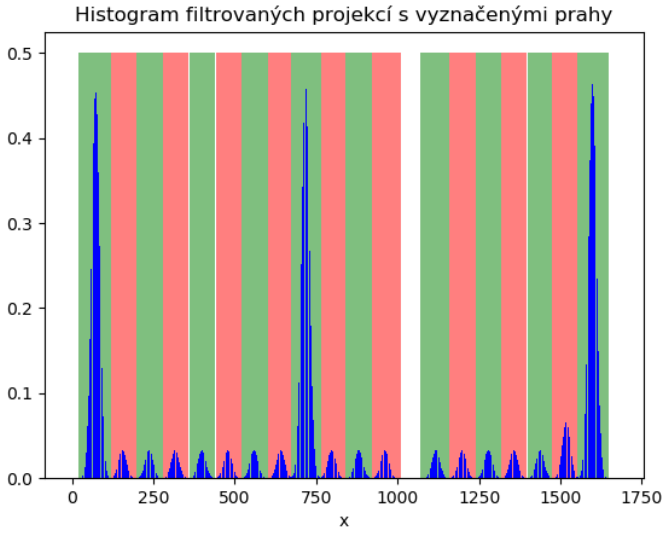
$$p_I(x) = \sum_i \delta(x, x_i), \tag{5.1}$$

kde $\delta(i, j)$ představuje Kroneckerovo delta a součet probíhá přes všechny detekce. Tedy značky jsou reprezentovány jako lineární kombinace jednotko-

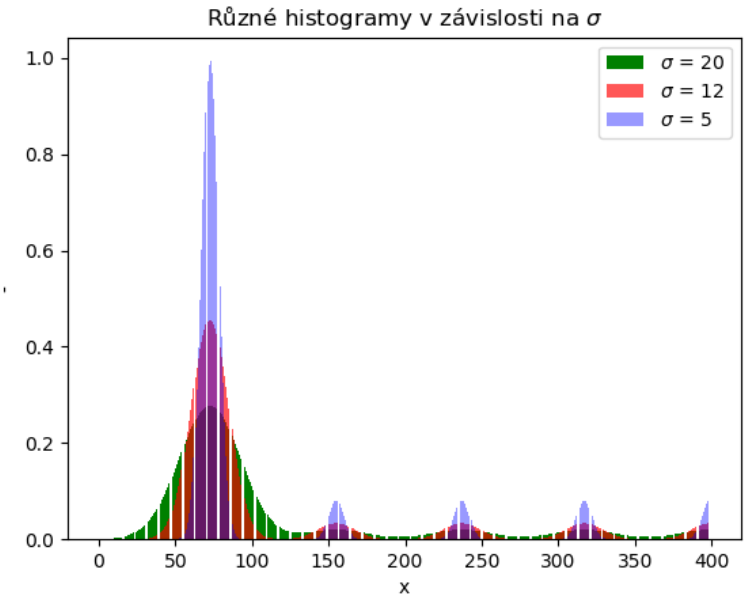
vých impulzů. Následně je provedeno filtrování získáného pole gaussovským filtrem (viz obr. 5.2, náležející vstupu5.1), výsledkem je pak filtrované pole p_G. Hodnota směrodatné odchylky σ definující filtr byla zvolena na $\sigma = 12$, na základě obrázků v datasetu. Je patrné, že pro schopnost algoritmu pracovat s obrázky různého rozlišení by bylo nutné navázat hodnotu σ na rozlišení obrázku, na poskytnutém datasetu ale metoda fungovala i v tomto pevném nastavení. Po provedení filtrace dojde ke splynutí blízkých jednotkových impulsů, a vzniku typických maxim v histogramu. Každé lokální maximum představuje lane. Toto však platí pouze za předpokladu, že obrázek není příliš silně zrotován, pak by totiž docházelo k deformaci histogramu a splývání lanes značek, náležejícím různým lanes, konkrétní výsledky pro rotaci jsou popsány v sekci výsledků 6.3. Následně jsou zkonstruovány prahy mezi jednotlivými maximy, v místech lokálních minim. Příklad segmentace osy x mezi různé lanes je vizualizován na obrázku 5.2 střídáním barvy pozadí. Pro vizualizaci různých možných nastavení σ je přidán obrázek 5.3:



Obrázek 5.1: Vstupní obrázek pro rozdělení značek do lanes



Obrázek 5.2: Histogram p_G filtrovaných jednotkových impulzů se segmentací osy x do lanes - přechody pozadí odpovídají prahům segmentů



Obrázek 5.3: Příklad různých variant histogramu v závislosti na na sm. odchylce filtru σ

Předpokládá se, že v žebříčcích se vyskytuje, resp. je detekováno, větší množství značek než v okolních lanes, proto se očekávají i větší odezvy pole p_G v okolí takové lane. Rozhodování na základě odezvy na filtraci je použito, protože cílem je plně automatizované zpracování obrázků, tedy i bez informace o počtu žebříčků a celkovém počtu drah. Postup rozdělení není kritickou částí zpracování, při předpokladu, že operátor dodá očekávané rozdělení lanes, by při změně implementace tato část zpracování mohla být přeskočena. Po vytvoření prahových hodnot na ose x, rozdělujících jednotlivé segmenty obrázku do lanes, se určí kontrolní konstanta q. Nechť $x_{L,i}, x_{P,i}$ jsou levý a pravý práh i-té lane. Pak pro konstantu q platí:

$$q = \max_i q_i, \tag{5.2}$$

kde každá lane je reprezentována jako součet hodnot $p_G(x)$ pro všechny indexy x mezi svými prahy:

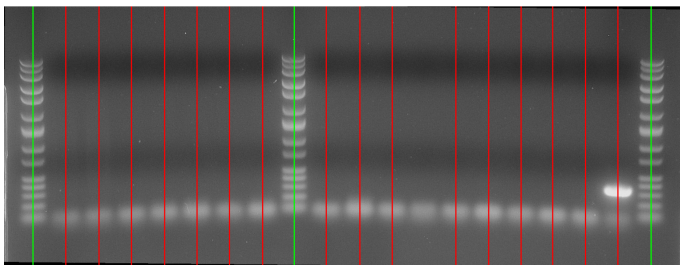
$$q_i = \sum_{j=x_{L,i}}^{x_{P,i}} p_G(j) \tag{5.3}$$

Toto q pak slouží pro určení prahu rozhodování, zda jiné lanes patří také mezi žebříčky, dle rovnice 5.4. Dá se předpokládat, že všechny žebříčky mají stejný počet značek, a proto, až na změny způsobené falešně positivními detekcemi anebo nedetekováním některých značek, by měly být hodnoty q_i příslušné lanes s žebříčky podobné. Proto při rozhodování se každé lane přiřadí hodnota

q_i dle rovnice 5.2. Následně se seřadí lanes podle q_i sestupně, a pokud platí:

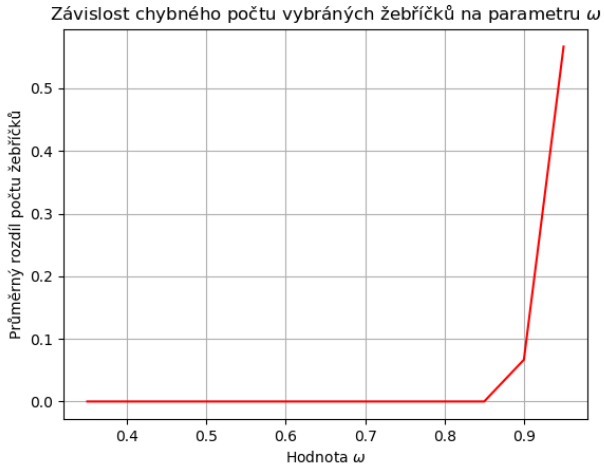
$$p(x) > \omega \cdot q_i, \tag{5.4}$$

pak se příslušná lane prohlásí za žebříček. Nejvyšší počet žebříčků v jednom obrázku je stanoven na 3, důvodem bylo prozkoumání dodaných obrázků, ve kterých se vždy vyskytují 1, 2, nebo 3 žebříčky. Ostatní segmenty se již nepovažují za žebříčky, i v případě, kdy splňují rovnici 5.4. Nadále se považují za skupiny neznámých značek náležející jedné vertikální linii.



Obrázek 5.4: Ukázka rozdělení bodů na žebříčky a neznámé značky (zeleně označena lane žebříčku, červeně lane zbylých značek)

Hodnota ω byla vybrána testováním rozpoznávání žebříčků na 30 obrázcích. Pro každý obrázek byl pro různé hodnoty ω určen počet žebříčků, a chyba se stanovila jako rozdíl tohoto počtu od počtu ručně anotovaných. Výsledky jsou v grafu 5.5, z něhož vyplynulo, že je relativně velká volnost ve výběru ω. Nakonec bylo zvoleno $\omega = 0.65$, což leží přibližně ve středu zkoumaného intervalu hodnot. Po analýze výsledků 100 obrázků se ukázalo, že tento přístup selhal pouze u 2 případů (z přibližně 200 žebříčků, kdy byla důvodem velmi špatná detekce na nekvalitním snímku, a žebříček pak postrádal detekci mnoha značek.



Obrázek 5.5: Závislost chyby v rozpoznávání žebříčků na parametru ω

5.2 Korekce rotace vstupního obrázku

Obrázky na vstupu jsou obvykle mírně rotovány. Efekt je zejména patrný na náklonu žebříčků. Očekává se, že žebříčky budou rovnoběžné s vertikální osou obrázku. Pro provedení opravné rotace je nutné nejprve odhadnout úhel, o který jsou žebříčky, a ve výsledku celý obrázek, vychýleny. Tento úhel se určí ze směrnice přímky, která nejlépe aproximuje žebříček (úloha se řeší jednotlivě pro každý žebříček v obrázku) ve smyslu nejmenších čtverců. Aproximující přímka je afinní podprostor dimenze 1, jenž maximalizuje kvadráty projekcí vektorů značek žebříčků, kdy vektor složky vektoru značky odpovídají detekovaným souřadnicím středu značky. Nechť pro matici $\mathbf{A}$ platí:

$$\mathbf{A}^{\mathbf{T}} = \left[\tilde{\mathbf{a}}_1, \ldots, \tilde{\mathbf{a}}_n\right], \tag{5.5}$$

kde $\tilde{\mathbf{a}}_i$ jsou vektory získané posunutím vektorů značek žebříčku o jejich těžiště, viz rovnice 5.6. Posunutím značek o jejich těžiště se docílí toho, že těžiště, kterým prochází optimální afinní podprostor, je posunuto do počátku, čímž se úloha mění na hledání optimálního lineárního podprostoru. Tedy pro $\tilde{\mathbf{a}}_i$ platí:

$$\mathbf{a}_T = \frac{1}{n}\sum_{i=1}^{n}\mathbf{a}_i, \tag{5.6}$$

$$\tilde{\mathbf{a}}_i = \mathbf{a}_i - \mathbf{a}_T. \tag{5.7}$$

Dle [17] je taková úloha řešitelná jako problém nejmenší stopy matice $\mathbf{A}^{\mathbf{T}}\mathbf{A}$. Úloha formulovaná v rovnici 5.8 má jako řešení bázi podprostoru, který minimalizuje součet kvadrátů rejekcí posunutých vektorů $\tilde{\mathbf{a}}_i$. Tato báze je tvořena vlastním vektorem, který přísluší nejmenšímu vlastnímu číslu matice $\mathbf{A}^{\mathbf{T}}\mathbf{A}$, navíc kritérium úlohy v optimu je rovno tomuto vlastnímu číslu matice. Báze hledaného podprostoru se v tomto případě určí jako kolmý vektor na optimální řešení úlohy, pro tento vektor rovněž platí, že je vlastním vektorem matice $\mathbf{A}^{\mathbf{T}}\mathbf{A}$ příslušný největšímu vlastnímu číslu. Rovněž platí, že optimální vektor $\mathbf{x}^*$ je normálovým vektorem příslušné přímky.

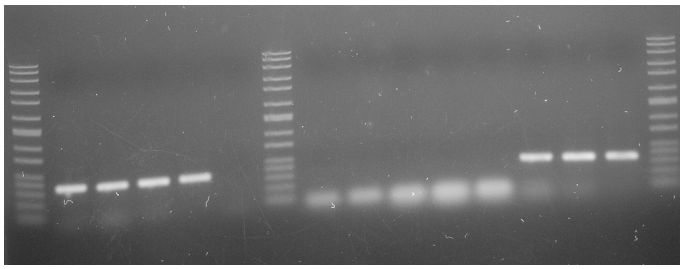
$$\mathbf{x}^* = \arg\min\{\mathbf{x}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{A}\mathbf{x} \mid \mathbf{x}^{\mathbf{T}}\mathbf{x} = 1,\ \mathbf{x} \in \mathbb{R}^2\}. \tag{5.8}$$

Nechť báze hledaného podprostoru, tj. směrnice přímky, je vektor $\mathbf{v} = \begin{bmatrix} v_x \\ v_y \end{bmatrix}$. Pak úhel, o který je potřeba zrotovat obrázek a body, se určí následovně:

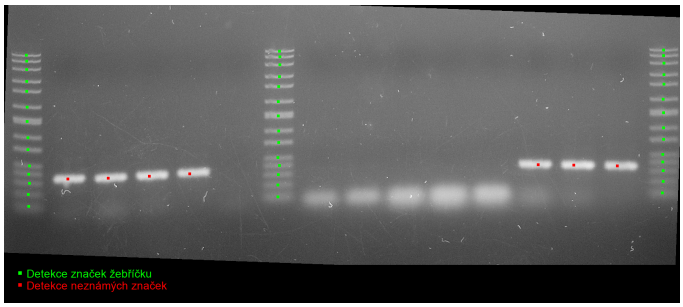
$$\phi = \arctan\frac{v_x}{v_y}. \tag{5.9}$$

Úhel ϕ leží v intervalu $\langle -\pi/2; \pi/2\rangle$. Pokud se v obrázku vyskytuje více žebříčků, pak je výsledný úhel dán jako průměr všech spočtených úhlů. Následně je možné provést rotaci v obrázku tak, že za počátek souřadnic je považován střed obrázku. Ačkoli pro běžné provedení rotace není třeba získat

úhel pro konstrukci rotační matice, kvůli implementaci rotování v knihovně *Pillow* je však výpočet úhlu proveden.



Obrázek 5.6: Obrázek před provedením opravné rotace



Obrázek 5.7: Výsledek opravné rotace

5.3 Fitování žebříčků do referenčního modelu

Poté, co jsou značky rozděleny do skupin, lze provést fitování značek žebříčku do modelu refenčního žebříčku. V tuto chvíli je již provedena oprava vstupní rotace, proto jsou nadále jako poloha jednotlivých značek zkoumána pouze jejich vertikální pozice v obrázku (osa y). Poloha v ose x je přidělena každému žebříčku, a je rovna průměru poloh x značek v příslušném žebříčku. Předpokládá se, že je před samotným zpracováním obrázku poskytnut expertní, též referenční, model žebříčku. Tím je myšlen seznam dvojic (v_i,z_i), kdy v_i je odměřená souřadnice značky v nějakém vzorovém žebříčku, a z_i je počet bází příslušný dané značce. Za předpokladu, že výsledkem detekce je zcela přesně detekovaný žebříček, který obsahuje všechny reálné značky a žádné falešné detekce, a neuvažují-li se chyby způsobené nelinearitami, pak musí existovat afinní funkce, která převede polohu i-tého značky referenčního žebříčku v_i na polohu i-té značky detekovaného žebříčku y_i, tj.:

$$y_i = \alpha + \beta \cdot v_i, \tag{5.10}$$

kdy α odpovídá posunutí (nadále bias) a β měřítku (nadále scale). Protože nelze předpokládat, že detekce budou přesně odpovídat očekávaným polohám

referenčních značek (výsledky jsou zatíženy chybou měření), hledaná optimální afinní funkce, za výše formulovaného předpokladu zanedbání nelinearit, je taková, která minimalizuje obecnou p-normu chybového vektoru, který se získá rozdílem vektoru poloh nalezených značek a vektorem transformovaných poloh referenčních značek. Tedy koeficienty optimální afinní funkce se v případě detekce všech značek získá jako řešení optimalizační úlohy:

$$\arg\min\{\sum_{i=0}^{n-1} |y_i - \alpha - \beta \cdot v_i|^p \;\middle|\; \alpha \in \mathbb{R},\ \beta \in \langle 0; \infty)\}, \tag{5.11}$$

kde n je počet značek v žebříčcích, a p určuje variantu normy. Obvykle používané hodnoty jsou $p = 1$, která specifikuje l_1 (manhattonskou) normu, a $p = 2$ pro l_2 (eukleidovskou) normu.

Taková situace je však velmi výjimečná. Obvyklý výsledkem detekce je žebříček, který část správných značek neobsahuje - detekce nebyla úspěšná např. kvůli blízkosti značek nebo špatným světelným podmínkám experimentu. Rovněž je nutné předpokládat možnost, že některé falešně positivní detekce byly přiřazeny do žebříčku, a jejich zapojení do nalezení optimální afinní funkce by degradovalo výsledek optimalizace. V takto obecném případě je nutné kromě nalezení afinní funkce mezi žebříčky také stanovit přiřazení značek mezi nalezeným a referenčním žebříčkem. Přiřazení je zde použito v následujícím významu. Nechť nalezený žebříček obsahuje n značek, a referenční žebříček obsahuje m značek. Pak přiřazení C je množina dvojic $\{i, c(i)\}$, kdy $c(i)$ je definováno pro celá čísla v intervalu $\langle 0; n)$ a platí:

1. $c(i)$ nabývá celého čísla j v intervalu $\langle 0; m)$, pokud i-tá značka nalezeného žebříčku je přiřazena j-té značce referenčního žebříčku. Navíc platí, že pro nezáporné hodnoty $c(i)$ existuje nejvýše jedno i.
2. $c(i) = -1$, pokud i-tá značka není přiřazena žádné značce referenčního žebříčku.

Při studiu materiálů k bakalářské práci se nepodařilo nalézt způsob, jak vyřešit optimalizační úlohu přes množinu přiřazení a zároveň přes množinu afinních funkcí $h(v) = \alpha + \beta v$, tj. nebyl nalezen způsob, jak s jistotou nalézt globálně optimální řešení úlohy. Proto se při řešení úlohy využívá opakující se posloupnosti dvou typů optimalizací. Nejprve se za pevně zvolených parametrů afinní funkce, tj. za při pevných hodnotách α, β, optimalizuje přes množinu přiřazení, a následně se pro nalezené přiřazení zpřesňují parametry afinní funkce. Tento postup se opakuje do konvergence, tedy dokud se po provedení dvojice optimalizací zmenšuje kriteriální funkce úlohy.

Problémem je rovněž nalezení vhodného počátečního odhadu řešení. Před provedením posloupnosti optimalizací je nutné získat úvodní odhad parametrů

afinní funkce. Úvodní odhad lze získat například vybráním libovolných 2 bodů nalezeného žebříčku y_1, y_2 a dvou bodů referenčního žebříčku v_1, v_2. Pak parametry přesného proložení těchto bodů afinní funkcí jsou rovny řešení následující soustavy rovnic:

$$\alpha + v_1 \cdot \beta = y_1, \tag{5.12}$$

$$\alpha + v_2 \cdot \beta = y_2. \tag{5.13}$$

Úvodní odhad parametrů afinní funkce α, β slouží k prvnímu kroku sekvence optimalizací, kdy je na základě těchto prvotních odhadů nalezeno optimální přiřazení C. Kvalita úvodního nastavení parametrů také určuje, zda algoritmus skončí v lokálním, nebo globálním minimu. Jednou z možností by bylo prohledat všechna možná počáteční nastavení afinní funkce (tj. použít všechny možné dvojice značek z nalezeného a dvojice z referenčního žebříčku), což by však bylo velmi výpočetně náročné z pohledu příliš častého výpočtu optimálních přiřazení a parametrů afinní funkce. Už v případě detekce žebříčku o 10 značkách a jeho fitování do reference rovněž o 10 značkách je počet různých čtveric bodů, které definují počáteční odhad parametrů afinní funkce (s příhlédnutím ke správné orientaci značek), roven:

$$\binom{10}{2}^2 = 2025, \tag{5.14}$$

navíc v obvyklých případech se počet značek žebříčků blíží spíše 15. Proto pro zvýšení kvality odhadů, nad kterým je pak prováděna optimalizace, slouží algoritmus RANSAC.

5.3.1 Odhad parametrů

Úvodní odhad parametrů je založen na algoritmu RANSAC, viz [18]. V každé iteraci algoritmu jsou vybrány dvojice bodů z každého z žebříčků, a odhad parametrů afinní transformace je určen vyřešením rovnice 5.12. Následně je kvalita daného odhadu stanovena určením počtu zbylých značek, které splňují podmínku tolerance (značky, které toto kritérium splní, se označí jako inliers). Počet těchto značek je určen následovně. Každá značka z nalezeného žebříčku y_j je porovnána se všemi značkami v žebříčku referenčním. Pokud pak platí:

$$\min_{i \in \{0,1,\ldots,m-1\}} |y_j - \alpha - \beta \cdot v_i| < u \cdot \Delta v \cdot \beta, \tag{5.15}$$

je j-tá značka nalezeného žebříčku považována za inlier. Tolerance v nerovnice (pravá strana) je dána parametry referenčního žebříčku. Nechť Δv je nejmenší rozdíl poloh mezi dvěma značkami referenčního žebříčku, a u je kladná konstanta, zpřísňující požadavek na blízkost značek, vliv parametr u na úspěšnost

fitování byl zkoumán v sekci 5.3.9. V práci se používaly dva způsoby kombinace odhadování počátečních parametrů afinní funkce a následné posloupnosti optimalizačních úkonů. V prvním případě se provádí N iterací hlavní smyčky. Během každé z iterací je pomocí algoritmu RANSAC vybrán nejlepší odhad parametrů při zadaném počtu iterací běhu algoritmu. Až po ukončení běhu RANSACu jsou nejlepší odhady použity pro zahájení sekvence střídajících se optimalizací, nejprve přes prostor přiřazení, pak přes prostor afinních funkcí. Pokud je hodnota kritéria po dokonvergování sekvence optimalizací menší než předchozí nejlepší, jsou nově vypočtené parametry afinní funkce i přiřazení uloženy. Po dokončení N iterací hlavní smyčky se vrací parametry a přiřazení s nejmenší hodnotou kriteriální funkce. Pseudokód je uveden v algoritmu 1.

Druhá varianta algoritmu spočívá v řešení optimalizační úlohy pokaždé, kdy dojde v rámci algoritmu RANSAC ke zlepšení počtu inliers. V tomto pohledu se algoritmus více odpovídá variantě zvané Lo-RANSAC, popsané v [19]. Pokud počet značek splňující rovnici 5.15 přesáhne počet inliers předchozí nejlepší počet, je provedena sekvence optimalizací, po jejímž konci je model uložen v případě, že dojde ke zlepšení kriteriální funkce, nikoli na základě počtu inliers. Pouze při zlepšení kritéria se rovněž uloží naměřený počet inliers. Tato varianta algoritmu vyžaduje stanovit jeden počet iterací N, který stanovuje počet opakování náhodného výběru. Počet optimalizačních sekvencí není pevně dán, vychází z odhadů parametrů afinní funkce. Pseudokód je uveden v algoritmu 2.

Formulace kriteriální funkce je uvedena v následující sekci 5.3.2.

input : detected ladder $(y_i)_{i=1}^{n}$, reference ladder $(v_i)_{i=1}^{m}$
output : assignment C^*, affine parameters (α^*, β^*)
best criterion $\leftarrow \infty$;
for $i \leftarrow 0$ **to** N **do**
  local parameters $\leftarrow$ None;
  local assignment $\leftarrow$ None;
  best inliers $\leftarrow 0$;
  for $j \leftarrow 0$ **to** n **do**
    estimate parameters α, β from random data;
    compute number of inliers;
    if *inliers* $>$ *best inliers* **then**
      best inliers $\leftarrow$ inliers;
      local parameters $\leftarrow (\alpha, \beta)$;
    end
  end
  repeat
    local assignment $\leftarrow$ optimized assignment C based on local parameters;
    local parameters $\leftarrow$ optimized parameters (α, β) based on local assignment;
  until *criterion converges*;
  if *criterion* $<$ *best criterion* **then**
    best criterion $\leftarrow$ criterion;
    assignment C^*, affine parameters $(\alpha^*, \beta^*) \leftarrow$ local assignment, local parameters;
  end
end

Algoritmus 1: Varianta fitování s optimalizací mimo RANSAC, [18]

input : detected ladder $(y_i)_{i=1}^n$, reference ladder $(v_i)_{i=1}^m$
output : assignment C^*, affine parameters (α^*, β^*)
best criterion $\leftarrow \infty$;
best inliers $\leftarrow 0$;
for $i \leftarrow 0$ **to** N **do**
  estimate parameter α, β from random data;
  compute number of inliers;
  if *inliers* $>$ *best inliers* **then**
    local parameters $\leftarrow (\alpha, \beta)$;
    local assignment $\leftarrow$ None;
    repeat
      local assignment $\leftarrow$ optimized assignment C based on local parameters;
      local parameters $\leftarrow$ optimized parameters (α, β) based on local assignment;
    until *criterion converges*;
    if *criterion* $<$ *best criterion* **then**
      best criterion $\leftarrow$ criterion;
      best inliers $\leftarrow$ inliers;
      assignment C^*, affine parameters $(\alpha^*, \beta^*) \leftarrow$ local assignment, local parameters;
    end
  end
end

Algoritmus 2: Varianta fitování s optimalizací uvnitř odhadu parametrů založená na algoritmu Lo-RANSAC [19]

5.3.2 Definice kriteriální funkce

Nechť C je přiřazení, tedy množina dvojic $\{i, c(i)\}$, kde $c(i)$ splňuje podmínky 5.3. Ať α, β jsou parametry afinní funkce. Detekovaný žebříček je zadán jako seznam souřadnic $(y_i)_{i=1}^n$, referenční žebříček je zadán jako seznam souřadnic $(v_i)_{i=1}^m$. Pak se definuje kriteriální funkce $\mathcal{F}$ následovně:

$$\mathcal{F}(C, \alpha, \beta) = \sum_{\forall i: c(i) \neq -1} |y_i - \alpha - \beta v_{c(i)}|^p + \sum_{\forall i: c(i) = -1} \lambda, \tag{5.16}$$

kdy se suma provádí přes všechny značky detekovaného žebříčku, tj. $i \in \{0, 1, \ldots, n-1\}$.

Kriteriální funkce vychází z kritéria úlohy v rovnici 5.11. V případě, že je k detekované značce přiřazena značka referenční, je její přírůstek k hodnotě kritéria dán pomocí obecné p normy (respektive varianty p normy s vynechanou

p-tou odmocninou). Pokud k přiřazení nedojde, je kritérium penalizováno na základě parametru λ, tedy penalizace za vynechávání je zvolena jako lineární funkce počtu nepřiřazených značek. Velké hodnoty λ naopak silně penalizují vynechávání značek, a optimální přiřazení pak musí pro snížení kritéria používat i falešně positivní detekce. Takto formulované kritérium $\mathcal{F}$ je použito v optimalizačních úlohách dále.

5.3.3 Formulace optimalizační úlohy přes množinu přiřazení

Při optimalizaci se střídají dva kroky. Prvním krokem je vybrání optimálního přiřazení C^* na základě pevně zadaných parametrů α_p, β_p. Nechť množina všech přiřazení je $\mathcal{C}$. Pak je optimalizační úloha následující:

$$C^* = \arg\min\{\mathcal{F}(C, \alpha_p, \beta_p) \mid C \in \mathcal{C}\}, \tag{5.17}$$

Funkce $\mathcal{F}$ je definována v rovnici5.16.

5.3.4 Optimalizace přes množinu přiřazení pomocí metod lineárního programování

Optimalizační úloha popsaná v rovnici 5.17 je převoditelná na lineární program. Nechť je nalezeno n značek detekovaného žebříčku, a referenční žebříček má m značek. Pro pomocné iterační proměnné i, j platí, že $i \in \{0, 1, \ldots, n-1\}$, resp. $j \in \{0, 1, \ldots, m-1\}$. Pak optimalizační úloha je úlohou celočíselného programování, formulovaná následovně:

$$\begin{aligned}
&\text{argmin} && \textstyle\sum_{i,j} z_{i,j} \cdot |y_i - \alpha - v_j\beta|^p + \lambda \sum_i \gamma_i && \\
&\text{subject to} && z_{i,j} \in \{0;1\} && \forall i \forall j \\
& && \gamma_i \in \{0;1\} && \forall i \\
& && \delta_j \in \{0;1\} && \forall j \\
& && \textstyle\sum_{i=0}^{n-1} z_{i,j} + \delta_j = 1 && \forall j \\
& && \textstyle\sum_{j=0}^{m-1} z_{i,j} + \gamma_i = 1 && \forall i
\end{aligned} \tag{5.18}$$

Proměnné $z_{i,j}$ představují přiřazení i-té značky nalezeného žebříčku j-té značce žebříčku referenčního. Sumační podmínky zajišťují, že nejvýše jedna

nalezená značka je přiřazena každé jedné značce, a také, že každá nalezená značka je použita nejvýše jednou. Proměnné γ_i pak uchovávají počet nepoužitých nalezených značek, přispívajících k regulační části kriteriální funkce. Celkem se tedy jedná o úlohu s $m \cdot n + m + n$ proměnnými a $m + n$ omezeními typu rovnost. Protože se jedná o přiřazovací úlohu, lze problém převést LP relaxací na lineární program s garantovaným celočíselným řešením, viz [17].

$$\begin{aligned}
&\text{argmin} && \textstyle\sum_{i,j} z_{i,j} \cdot |y_i - \alpha - v_j\beta|^p + \lambda \sum_i \gamma_i && \\
&\text{subject to} && z_{i,j} \in \langle 0; 1\rangle && \forall i \forall j \\
& && \gamma_i \in \langle 0; 1\rangle && \forall i \\
& && \delta_j \in \langle 0; 1\rangle && \forall j \\
& && \textstyle\sum_{i=0}^{n-1} z_{i,j} + \delta_j = 1 && \forall j \\
& && \textstyle\sum_{j=0}^{m-1} z_{i,j} + \gamma_i = 1 && \forall i
\end{aligned} \tag{5.19}$$

K řešení lineárního programu bylo nejprve užito řešiče v knihovně *scipy*, viz [16]. Doba pro vyřešení úlohy v této implementaci však byla příliš dlouhá, a neumožňovala by vyřešení řešení úlohy v rozumném čase. Proto byl k řešení úloh lineárního programování nadále použit externí řešič Gurobi [20], který čas na vyřešení úlohy výrazně zmenšil. Na počítači s procesorem Intel i5, $2.3\,GHz$, trvalo řešení úlohy dle 5.19 pomocí řešiče Gurobi v řádu desetin milisekund až milisekund, narozdíl od původního řešiče, kdy se doba výpočtu, podle náročnosti a počtu proměnných, pohybovala v řádu desítek milisekund.

5.3.5 Optimalizace přes množinu přiřazení pomocí Maďarského algoritmu

Problém nalezení optimální přiřazení je řešitelný rovněž jako problém přiřazení. Obvyklá formulace problému je následující. Nechť existují 2 množiny S, T, v obvyklém případě tyto množiny představují pracovníky a práce, které se mají pracovníkům přířadit, viz [21], [22]. Nechť obě množiny obsahují stejné množství prvky, tedy se v základní formulaci úlohy řeší vyvážený problém přiřazení. Cílem je nalézt bijekci $\mathcal{R} : S \to T$, která minimalizuje, při zadané váhové funkci $\mathcal{V} : S \times T \to \mathbb{R}$, výraz:

$$\sum_{s \in S} \mathcal{V}(s, \mathcal{R}(s)). \tag{5.20}$$

Úlohu lze rovněž reprezentovat pomocí úplného bipartitního, jehož disjunktní množiny vrcholu jsou tvořeny množinami S, T. Ceny hran mezi vrcholy

obou množin jsou dány váhovou funkce $\mathcal{V}$. Cílem je pak vybrat maximální nezávislou množinu hran(tj. množinu o n hranách), která minimalizuje součet cen obsažených hran. Nezávislá množina hran je taková množina hran, kde žádné dvě hrany z množiny nesdílejí společný vrchol. Množina nezávislých hran je maximální, pokud už do množiny nelze přiřadit další hranu se zachováním předpokladu jejich nezávislosti.

Formalizace úlohy pro její řešení pomocí Maďarského algoritmu, jehož základní myšlenka je popsaná v [21], spočívá ve vytvoření matice, reprezentující hrany v bipartitním grafu. Nechť je nalezeno n značek, a vzorový žebříček má m položek. Pak pro daný odhad parametrů α, β zkonstruuji matici reprezentující úlohu jako bipartitní graf následovně:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \end{bmatrix}, \tag{5.21}$$

$$\mathbf{A}_1 \in \mathbb{R}^{n \times m},\ a_{i,j} = |y_i - \alpha - v_j\beta|^p, \tag{5.22}$$

$$\mathbf{A}_2 \in \mathbb{R}^{n \times n}, \tag{5.23}$$

$$a_{i,j} = \lambda,\ i = j, \tag{5.24}$$

$$a_{i,j} = \infty,\ i \neq j. \tag{5.25}$$

Položky $\mathbf{A}_1$ reprezentují příspěvky ke kritériu v případě, že vybrané detekované značce je přiřazena značka v referenčním žebříčku. Podmatice $\mathbf{A}_2$, jejíž složky jsou $a_{i,j}$ reprezentuje možnost, že daná značka je nepoužita (tj. pro dvojici $(i, c(i))$ z přiřazení C splňuje $c(i) = -1$). Řádky matice $\mathbf{A}$ odpovídají vrcholům grafu, náležejícím do množiny S, sloupce pak představují vrcholy z množiny T. Použitím podmatice $\mathbf{A}_2$ je množina T doplněna o virtuální vrcholy, kterým lze přiřadit vždy jen jeden vrchol z S (což je reprezentováno nekonečnou cenou hrany), a umožňuje nepřiřadit žádnou značku z referenčního žebříčku. Položky $a_{i,j}$ matice $\mathbf{A}$ definují váhovou funkci $\mathcal{V}(i, j)$, tedy i váhy hran příslušného bipartitního grafu. Z tohoto je patrné, že se skutečně jedná o ekvivalentní formulaci optimalizační úlohy dle rovnice 5.17.

Takto formalizovaný problém je již variantou nevyváženého problému přiřazení, který je řešitelný postupem popsaným v [22].

Pro řešení úlohy za použití Maďarského algoritmu se použila implementace ve *scipy* dle [16]. Doba výpočtu se pohybovala v okolí milisekund, stejně jako při použití řešiče LP v sekci 5.3.4, výsledky obou algoritmů ve smyslu optimálního přiřazení pak byly stejné. Pro vizualizaci délky výpočtu je vytvořena tabulka 5.1. Bylo vytvořeno 1000 žebříčků, a pro každý byl určen průměrný čas výpočtu. Generování syntetických žebříčků proběhlo postupem popsaným v sekci 5.3.9. Tabulka byla rozvrstvena podle rozdílu $s = n - m$, kde n je počet značek vygenerovaného žebříčku a m počet značek referenčního. Kvůli

s	-3	-2	-1	0	1	2	3
t_H [ms]	0.28	0.32	0.33	0.36	0.40	0.40	0.44
t_{LP} [ms]	0.55	0.55	0.61	0.67	0.70	0.76	0.81

Tabulka 5.1: Průměrné doby výpočtu optimálního přiřazení v závislosti na rozdílu značek detekovaného a referenčního žebříčku

způsobu generování však $s = 0$ neznamená, že by žebříček obsahoval pouze TP detekce. t_{LP} je průměrný čas pro výpočet pomocí LP se solverem Gurobi, t_H je průměrný čas pro výpočet pomocí Maďarského algoritmu. Z tabulky je patrné, že rozdíly v časech jsou minimální. Kvůli mírně lepším výsledkům Maďarského algoritmu byla nadále používána tato metoda optimalizace přes množinu přiřazení.

5.3.6 Formulace optimalizační úlohy přes množinu parametrů afinní funkce

Druhý krok optimalizace spočívá v hledání optimálních parametrů afinní funkce α^*, β^*, při pevném přiřazení C_p. Úkolem kroku je zpřesňovat parametry afinní funkce původně odhadnuté pomocí algoritmů RANSAC 1 nebo Lo-RANSAC 2. Zde je formulace úlohy téměř totožná rovnici 5.17:

$$(\alpha^*, \beta^*) = \arg\min\{\mathcal{F}(C_p, \alpha, \beta) \;\mid\; \alpha \in \mathbb{R},\ \beta \in \langle 0; \infty)\}. \tag{5.26}$$

Normalizační část rovnice 5.16 ($\sum \lambda$) je konstantní pro libovolné nastavení proměnných. V úloze je ponechána, aby bylo možné dobře porovnávat zlepšení kritéria mezi první a druhou fází optimalizační sekvence. Proměnná scale β je omezena na nezápornou část reálné osy, neboť se předpokládá stejná orientace žebříčku referenčního a nalezeného ve smyslu, že menší hodnota pozice značky v žebříčku odpovídá vyššímu počtu bází. Proměnná α není omezena.

5.3.7 Optimalizace přes množinu parametrů afinní funkce pomocí metod lineárního programování

Pokud je jako norma použita l_1, tj. platí $p = 1$ v rovnici 5.16, pak je druhý krok optimalizační sekvence řešen pomocí metod lineárního programování. Nechť $C_p = \{i, c(i)\}$ je vstupní přiřazení. Nechť má detekovaný žebříček n značek a $i \in \{0, 1, \ldots, n-1\}$. Nechť platí, že právě k značek detekovaného

žebříčku je přiřazeno k nějaké značce žebříčku referenčního. Pak úloha určená pro optimalizaci bias a scale je následující:

$$\begin{aligned} &\text{argmin} && \textstyle\sum_{\forall i:c(i)\neq -1} |y_i - \alpha - v_{c(i)}\beta| + (n-k)\cdot\lambda \\ &\text{subject to} && \alpha \in \mathrm{R} \\ & && \beta \geq 0 \end{aligned} \tag{5.27}$$

s proměnnými α, β. Je patrné, že vliv parametrů λ je při řešení úlohy přes množinu parametrů afinní funkce konstantní. Běžný tvar úlohy lineárního programování, získán převodem z úlohy 5.27, je následující:

$$\begin{aligned} &\text{argmin} && \textstyle\sum_{i=0}^{n-1} z_i \\ &\text{subject to} && -z_i \leq y_i - \alpha - v_j\beta \leq z_i \quad \forall i : c(i) \neq -1 \\ & && \alpha \in \mathbb{R} \\ & && \beta \geq 0 \\ & && z_i \geq 0 \qquad \forall i \end{aligned} \tag{5.28}$$

Úloha obsahuje $n+2$ proměnných a $2n$ omezení typu nerovnost. Proměnné α, β jsou hledané parametry afinní funkce, proměnné z_i slouží k převodu účelové funkce, obsahující absolutní hodnoty, na lineární. K výslednému kritériu úlohy v optimálním řešení je nutné přičíst konstantu $(n-k)\lambda$, aby byla hodnota kritéria ekvivalentní při použití funkce dle rovnice 5.16.

K řešení úlohy v implementaci byl stejně jako v případě lineárního programu pro řešení problému přiřazení 5.3.4 použit řešič Gurobi, viz [20].

5.3.8 Optimalizace přes množinu parametrů afinní funkce pomocí lineárních nejmenších čtverců

Při použití l_2 normy, tj. $p = 2$ v rovnici 5.16, je druhý krok optimalizace řešen pomocí metody lineárních nejmenších čtverců. Nejprve se zkonstruují matice $\mathbf{P}$ a vektor $\mathbf{q}$:

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_0^T \\ \vdots \\ \mathbf{p}_{n-1}^T \end{bmatrix}, \ \mathbf{q} = \begin{bmatrix} q_0 \\ \vdots \\ q_{n-1} \end{bmatrix}. \tag{5.29}$$

Kdy pro vektory $\mathbf{p}_i$, tvořící matici, platí:

$$\mathbf{p}_i = \begin{bmatrix} 1 \\ v_i \end{bmatrix} \iff c(i) \neq -1, \ \mathbf{p}_i = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \iff c(i) = -1 \tag{5.30}$$

, podobně pro q_i:

$$q_i = y_i \iff c(i) \neq -1,\ q_i = 0 \iff c(i) = -1. \tag{5.31}$$

Úkolem je pak nalézt řešení minimalizující l_2 reziduum přeurčené soustavy, tj.:

$$\mathbf{u}^* = \arg\min ||\mathbf{P}\mathbf{u} - \mathbf{q}||_2^2, \tag{5.32}$$

což je dle [17] úloha lineárních nejmenších čtverců, jejíž vektor optimálního řešení $\mathbf{u}^* = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$ je pak roven:

$$\mathbf{u}^* = \mathbf{P}^+\mathbf{q}, \tag{5.33}$$

kdy operátor $^+$ značí pseudoinverzi matice. Aby byla kriteriální funkce ekvivalentní funkci v rovnici 5.16, je třeba doplnit pevnou hodnotu danou nepoužívanými značkami. Nechť k je rovno počtu značek referenčního žebříčku, kterým byla přiřazena značka žebříčku referenčního. Pak je hodnota kritéria při optimálním řešení dána jako:

$$J = ||\mathbf{P}\mathbf{u}^* - \mathbf{q}||_2^2 + (n - k) \cdot \lambda. \tag{5.34}$$

5.3.9 Výběr parametrů fitování

Při ruční anotaci dat se označovalo pouze středy značek, počty bází, resp. struktura referenčního žebříčku anotovány nebyla. Z toho důvodu jsou pro nastavení vhodné hodnoty parametrů fitování použity syntetické žebříčky. Základem generování syntetických žebříčků jsou tři referenční žebříčky, odečtené z reálných obrázků. Referenční žebříčky jsou tvořeny pouze referenčními souřadnicemi v_i. Tyto jsou dále transformovány pomocí afinní funkce, s náhodně vybranými parametry bias a scale. Pro všechny výsledné transformované značky jsou pak provedeny následující operace. Nejprve je značka s pravděpodobností P_{FN} z transformovaného žebříčku vynechána, čímž se simuluje nezachycení značky při detekci. Zbylé značky jsou pak vychýleny od své optimální pozice, míra výchylky Δ je realizací náhodného výběru z normálního rozdělení se směrodatnou odchylkou σ a nulovou střední hodnotou. Nechť jsou náhodně zvoleny parametry afinní funkce α, β, a nechť v_i je pozice i-té značky referenčního žebříčku. Pak pozice y_i v syntetickém žebříčku je:

$$y_i = \alpha + \beta v_i + \beta\Delta. \tag{5.35}$$

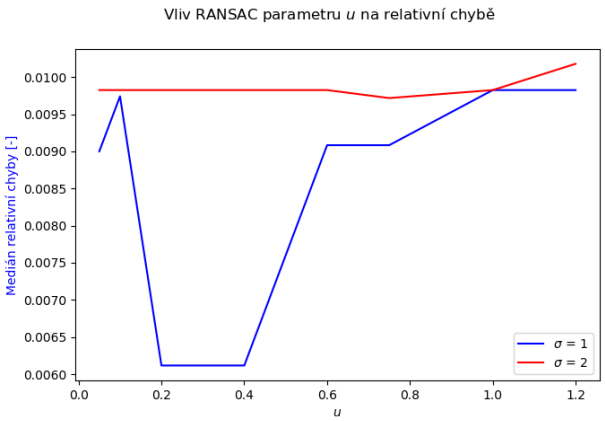
Poté jsou přidány značky reprezentující falešně positivní detekce.Každá z nejvýše N takových možných přídatných značek je přidána s pravděpodobností P_{FP}. Pokud je značka přidána, pak je její poloha vybrána z rovnoměrného náhodné rozdělení na intervalu $\langle x_{LOW} - \delta; x_{HIGH} + \delta\rangle$, kde limity x_{LOW}, x_{HIGH}

jsou minimální, resp. maximální hodnota všech poloh značek v syntetickém žebříčku, a konstanta δ umožňuje umisťovat značky i mimo syntetický žebříček. Nad těmito žebříčky pak byly prováděny testy, jejichž výsledky se vyhodnocovaly pomocí funkce:

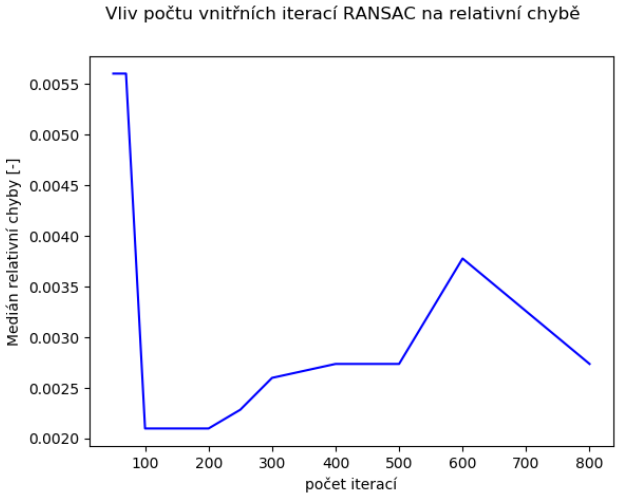
$$h(\beta, \beta^*) = \frac{|\beta - \beta^*|}{\beta^*}. \tag{5.36}$$

Tedy chyba je reprentována jako relativní chyba nalezené hodnoty parametru β afinní funkce od správné hodnoty β^*, která byla použita pro vygenerování vybraného syntetického žebříčku.

Nejprve byly vybrány parametry u z rovnice 5.15, určující náročnost toleranční podmínky při odhadu parametrů, a počet vnitřních iterací pro algoritmus 1. Test byl proveden při konstantním nastavení ostatních parametrů, a ukázalo se, že až na okrajové hodnoty na hodnotách těchto parametrů příliš nezávisí. Proto byla hodnota parametru nastaveny na $u = 0.3$ dle grafu 5.8, a počet iterací byl nastaven na 150, dle grafu 5.9.

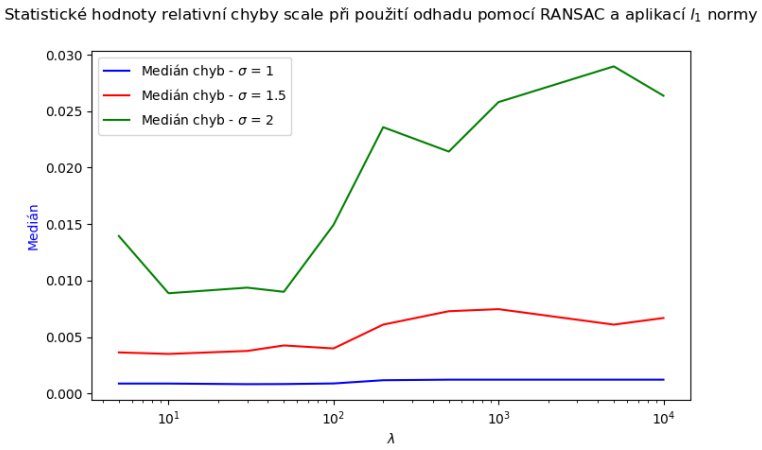


Obrázek 5.8: Vliv tolerančního parametru u na relativní chybu

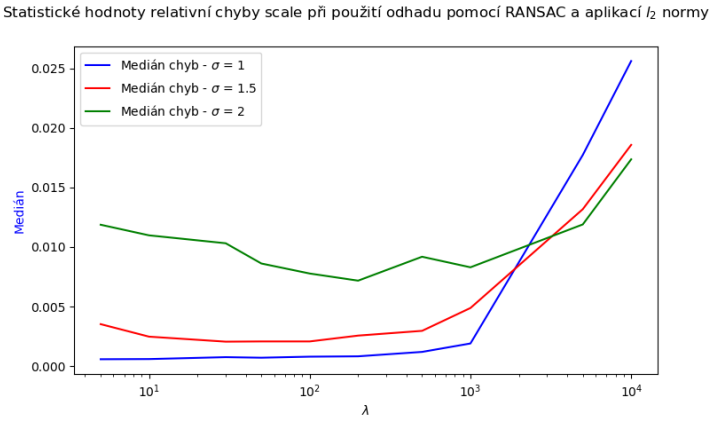


Obrázek 5.9: Vliv počtu iterací algoritmu 1 na relativní chybu

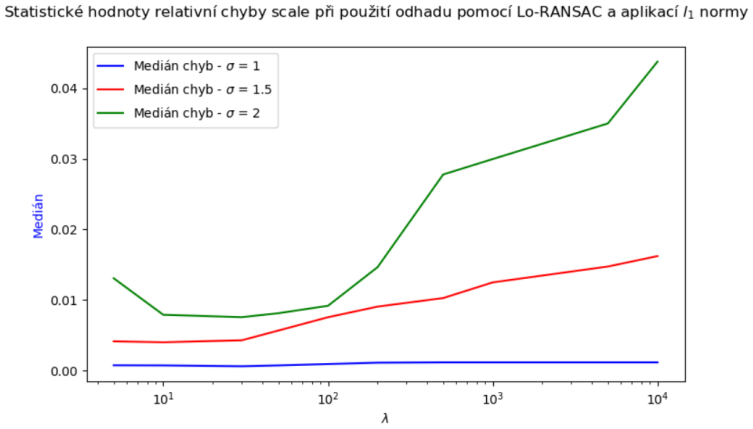
Pro každé λ jsou jako výsledky testování použity medián chybových funkcí nad danou množinou testovacích žebříčků. Testování proběhlo pro obě možnosti algoritmu 2, 1, a pro každý z algoritmů rovněž za použití l_1 i l_2 normy. Výsledky testování závislosti chyby na parametru λ jsou na grafech níže, pro osu x byla použita logaritmická stupnice. Testy byly provedeny rovněž pro více nastavení σ v algoritmu pro generování žebříčků. Test byl proveden tak, aby každý z algoritmů měl na vyřešení úlohy stejný čas, tím pádem bylo možné porovnat oba algoritmy za stejných podmínek pro výpočet. Celkem bylo pro každé z nastavení σ vygenerováno 80 žebříčků.



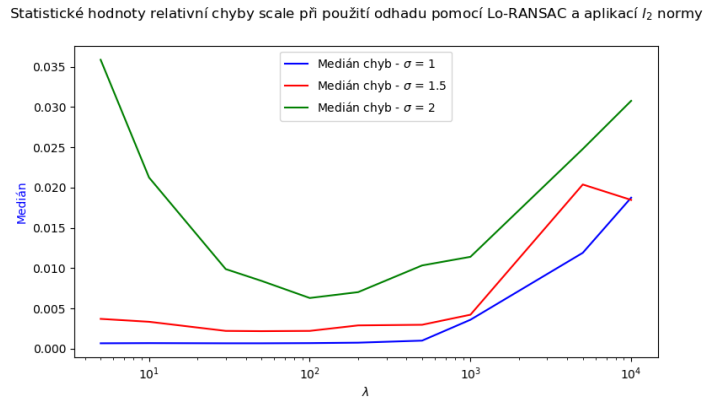
Obrázek 5.10: Vliv nastavení parametru λ na výsledky algoritmu RANSAC s použitím l_1 normy



Obrázek 5.11: Vliv nastavení parametru λ na výsledky algoritmu RANSAC s použitím l_2 normy

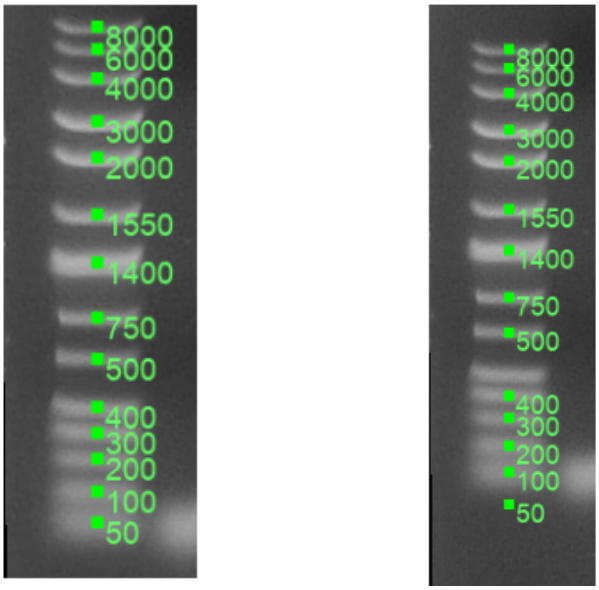


Obrázek 5.12: Vliv nastavení parametru λ na výsledky algoritmu Lo-RANSAC s použitím l_1 normy



Obrázek 5.13: Vliv nastavení parametru λ na výsledky algoritmu Lo-RANSAC s použitím l_2 normy

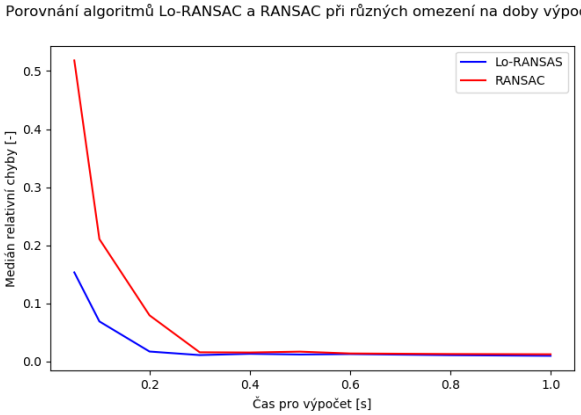
Rozdíly ve výsledcích jednotlivých variant algoritmů jsou pro příslušnou optimální volbu λ statisticky nevýznamné. Nakonec bylo přikročeno k využití l_1 normy, a to po analýze výsledků fitování v obrázcích. Ukázalo se, že někdy v případě použití l_2 normy docházelo k vynechávání značek, viz 5.14b.



(a) : Použití l_1 normy **(b) :** Použití l_2 normy

Obrázek 5.14: Porovnání fitování při použití různých norem, při vybrání optimální hodnoty λ z grafů 5.10 a 5.11

Pro přesnější porovnání obou algoritmů (nyní již pouze při použití l_1 normy) bylo vygenerováno 50 žebříčků s nastavením $\sigma = 1.5$. Následně byly zkoumány hodnoty chybové funkce obou algoritmů při postupném protahování času pro výpočet. Z grafu 5.15 lze ukázat, že implementace dle 2 rychleji snižuje relativní chybu, proto se nadále používala tato implementace. Dle grafu 5.12 byla nastavena hodnota λ na 10. Počet iterací hlavní smyčky algoritmu 2 se nastavil na 1200, což odpovídalo průměrné době výpočtu $0.4\,s$, kdy se již podle grafu 5.15 chyba nesnižovala. Počet iterací byl vybrán jako kompromis mezi funkčností algoritmu z hlediska časové náročnosti výpočtu a výsledků fitování. Výsledky fitování jsou popsány v sekci 6.3.



Obrázek 5.15: Vliv délky výpočtu na medián relativní chyby

5.4 Korekce chyb způsobených nehomogenitou proudu

Poté, co jsou nalezeny optimální hodnoty přiřazení a bias a scale, lze rekonstruovat chybějící značky v žebříčcích pomocí hodnot bias a scale. Nechť i-tá značka žebříčku chybí, pak je její poloha y_i na konci optimalizace nastavena pomocí značky referenčního žebříčku v_i na:

$$y_i = \alpha + \beta v_i. \tag{5.37}$$

Ačkoli je možné nahradit hodnoty souřadnice v i-té značky a_i stejným vzorcem i pro značky nalezené, je místo toho použita jejich skutečná detekovaná hodnota. Důvodem je zachování detekovaných pozic na středech značek v obrázku. Po tomto kroku je již zajištěno, že všechny žebříčky obsahují m značek, kde m je počet značek referenčního žebříčku. V obrázcích jsou obvykle přítomny 2-3 žebříčky. Kvůli nehomogenitě elektrického pole se však makromolekuly, které jsou na konci pokusy viditelné jako značky, pohybují různými rychlostmi v různých částech obrázku, proto jsou i žebříčky, které jsou tvořeny makromolekulami přesně dané velikosti, vůči sobě vychýleny.

Cílem korekce chyb způsobených nehomogenitou je nalézt transformaci polynomem $s(x, y)$, který je definován pro všechny pixely obrázku, a jehož výstupem je transformovaná hodnota v, která lze již přímo použít k odhadu počtu bází v sekci 5.5.1. Optimalizační úloha spočívá v nalezení takového polynomu $s(x, y)$ který minimalizuje kvadrát chyb od očekávané y-ové souřadnice v_j příslušné značky v referenčním žebříčku:

$$\arg\min\{\sum_i \sum_{j=1}^{m} (s(x_i, y_{i,j}) - v_j)^2 \;\middle|\; s(x, y) \in \mathcal{Q}\}, \tag{5.38}$$

kde x_i je x-ová souřadnice i-tého žebříčku, $y_{i,j}$ je y-ová souřadnice j-té značky v i-tém žebříčku a v_j jsou souřadnice značek v referenčním žebříčku, a $\mathcal{Q}$ je množina polynomů, která je definována pro různé počty žebříčků různým způsobem. Úloha lze řešit jako lineární nejmenší čtverce pomocí pseudoinverze. V případě jediného žebříčku je nemožné korigovat vliv nehomogenity pro různé souřadnice x, tedy množina polynomů je následující:

$$\mathcal{Q}_1 = \{a_0 + a_1 y \mid a_i \in \mathbb{R}\ \forall i\}. \tag{5.39}$$

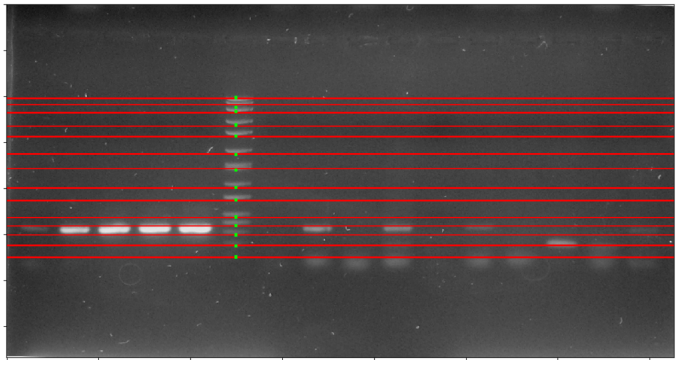
Je patrné, že v případě jediného žebříčku se jedná pouze o inverzní operaci k hledání scale a bias v sekci 5.3. Ačkoli jsou již koeficienty α, β nalezeny pomocí optimalizace s chybou ve smyslu l_1 normy, kvůli generalizaci úlohy pro větší počty žebříčků a použití nejmenších čtverců se opět určí nové koeficenty inverzní operace. Pokud jsou nalezeny dva žebříčky, pak je množina polynomů následující:

$$\mathcal{Q}_2 = \{a_0 + a_1 y + a_2 x + a_3 xy \mid a_i \in \mathbb{R}\ \forall i\}. \tag{5.40}$$

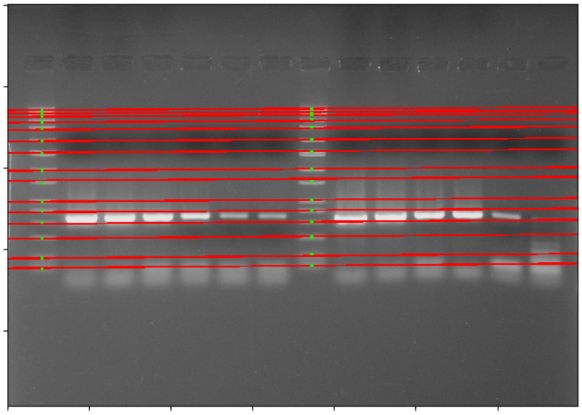
Použitím těchto koeficientů je umožněno lineární posunutí ve smyslu osy x mezi žebříčky. Pro více žebříčků je použita množina polynomů:

$$\mathcal{Q}_3 = \{a_0 + a_1 y + a_2 x + a_3 xy + a_4 x^2 \mid a_i \in \mathbb{R}\ \forall i\}, \tag{5.41}$$

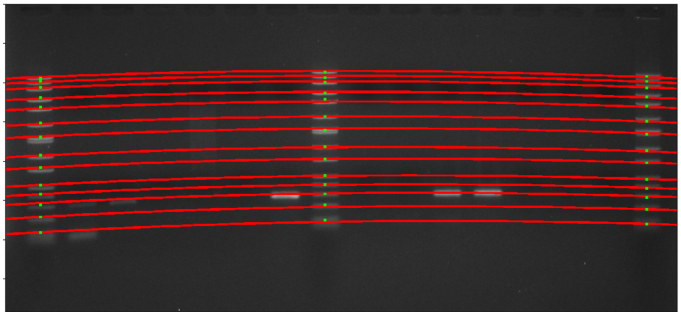
kdy je již možné provést parabolickou korekci. Charakter vrstevnic polynomu je x-ovou složkou polynomu. Funkčnost použití těchto polynomů se potvrdila na vzorcích obsahujících 1, 2 i 3 žebříčky, viz obrázky 5.16, 5.17, 5.18.



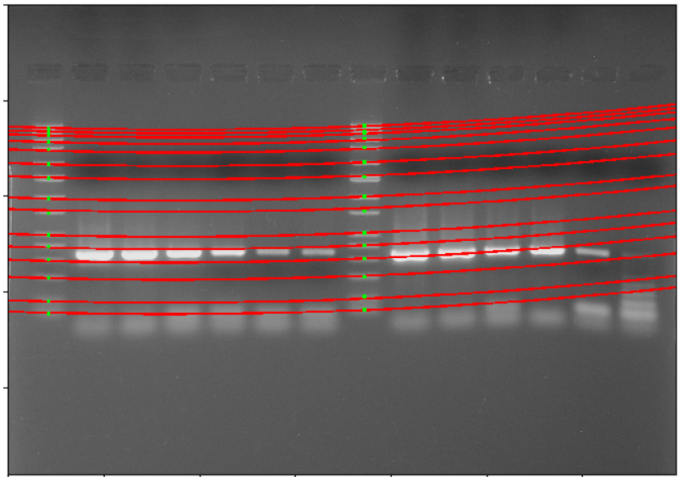
Obrázek 5.16: Vrstevnice korigujícího polynomu a značky žebříčku, jimiž mají vrstevnice procházet, pro 1 žebříček



Obrázek 5.17: Vrstevnice korigujícího polynomu a značky žebříčku, jimiž mají vrstevnice procházet, pro 2 žebříčky



Obrázek 5.18: Vrstevnice korigujícího polynomu a značky žebříčku, jimiž mají vrstevnice procházet, pro 3 žebříčky

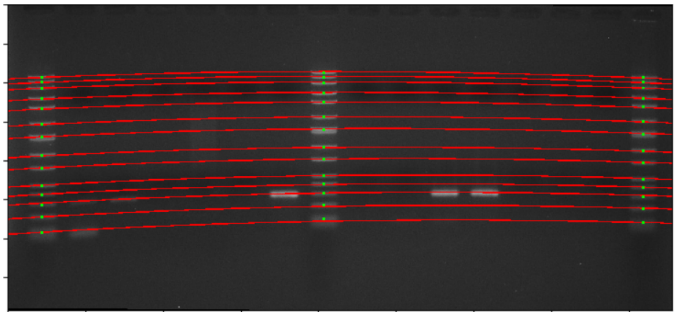


Obrázek 5.19: Příklad použití polynomu z $\mathcal{Q}_3$ pro obrázek s 2 žebříčky

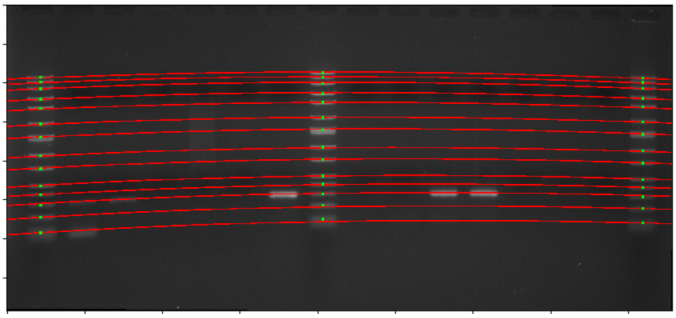
Na obrázku 5.19 je vidět problematické fitování vyšším polynomem pro

problém pouze se 2 žebříčky. Je patrné, že absence třetího žebříčku při použití polynomu z $\mathcal{Q}_3$ vede na přesnější fitování ve smyslu optimalizační úlohy v rovnici 5.38, ale zároveň aplikovaný parabolický charakter vrstevnic nemá bez třetího žebříčku oporu v použití.

Experimentálně bylo testováno přidání kvadratického členu y^2 do rovnice polynomů $s(x, y)$. Na obrázcích 5.20 a 5.21 je vidět, že význam přidání kvadratického členu je nepatrný, důvodem zjevně je způsob fitování polynomu, při kterém je předpokládána právě nejvýše lineární závislost mezi referenčním a detekovaným žebříčkem. Proto jsou nadále používány polynomu z množin $\mathcal{Q}_i$ dle rovnic 5.39, 5.40 a 5.41.



Obrázek 5.20: Vrstevnice polynomu při použití kvadratického členu y^2



Obrázek 5.21: Vrstevnice polynomu stejného obrázku při absenci y^2

5.5 Odhad počtu bází neznámých značek

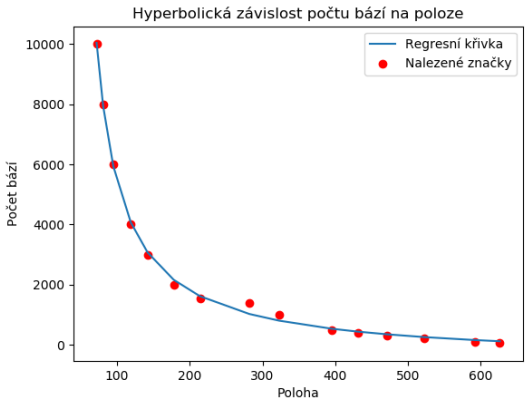
5.5.1 Odhad fitováním hyperboly

Dle [23] je závislost mezi polohou značky v žebříčku v, již po aplikaci korekce popsané v sekci 5.4, a příslušným počtem bází z hyperbolická. Tato teze byla

otestována na referenčních žebříčcích fitováním funkce:

$$z = h(v) = \frac{a}{v - b} + c, \tag{5.42}$$

pomocí metody nelineárních nejmenších čtverců. Výsledné fitování skutečně potvrzuje, že závislost lze popsat pomocí hyperboly:



Obrázek 5.22: Fitování hyperbolické funkce do dat expertního žebříčku

Nabízí se nalézt optimální hyperbolické proložení ve smyslu nejmenších čtverců pomocí funkce $h(v)$. Protože však měřené polohy značek jsou kvůli detekci a procesu dodefinování chybějících značek v rovnice 5.37 zatíženy chybou měření, a počty bází značek v žebříčku jsou přesně známy předem, je přirozenější optimalizovat inverzní funkci:

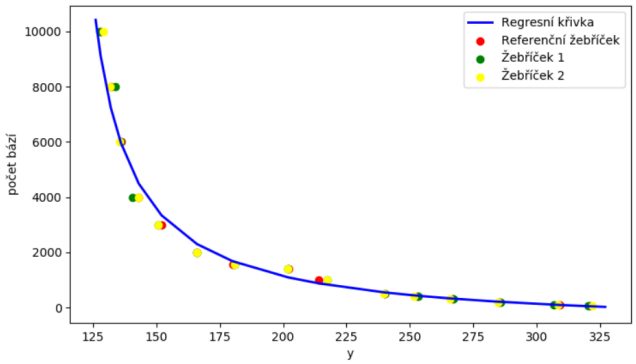
$$v = h^{-1}(z) = \frac{a}{z - c} + b, \tag{5.43}$$

kde použité parametry a, b, c jsou stejné jako v případě rovnice 5.42. Před samotnou optimalizací jsou pro všechny polohy (x, y) značek v žebříčcích pomocí polynomu $s(x, y)$, definovaném v sekci 5.4, získány normované y-ové souřadnice v. Každé z těchto normovaných souřadnic $v_{j,i}$, přes všechny nalezené žebříčky j včetně žebříčku referenčního, je na základě přiřazení, získaného optimalizační úlohou dle sekce 5.3, přiřazen počet bází z_i. Tedy v případě například 2 žebříčků je hyperbolická funkce fitována mezi zdvojené souřadnice v pro každý referenční počet bází. Parametry prokládané funkce se získají jako řešení úlohy:

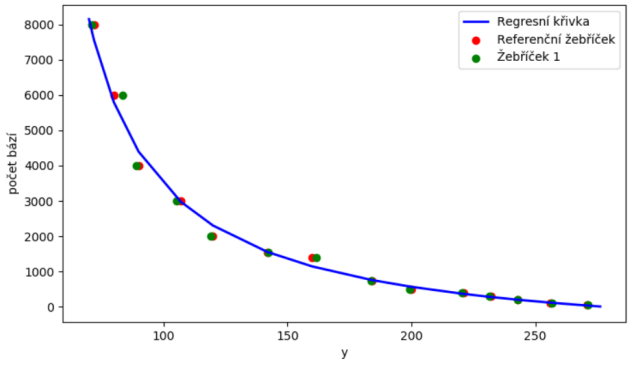
$$\arg\min\{\sum_i \sum_j (h^{-1}(z_i) - v_{j,i})^2 \;\Big|\; a \geq 0,\ b \leq \min_i v_{j,i},\ c \leq \min_i z_i\}, \tag{5.44}$$

kde funkce h^{-1} je definována v rovnici 5.43. Zadaná omezení na jednotlivé proměnné úlohy zajišťují, že dvojice $(z_i, v_{j,i})$ jsou prokládány „správným“ ramenem hyperboly, tj. zamezí se ukončení algoritmu v lokálním minimu

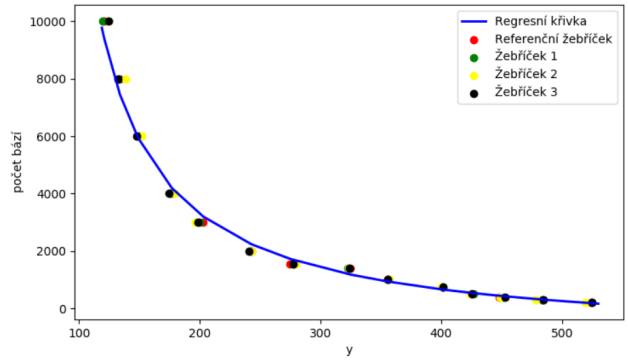
při přeskočení asymptot hyperboly. K řešení byla použita implementace nelineárních nejmenších čtverců v knihovně *scipy*. Protože parametry obou funkcí v rovnicích 5.42 a 5.43 jsou identické, lze k původně zkoumané funkci přejít okamžitě po optimalizaci. Výsledky prokládání jsou vizualizovány na grafech 5.23, 5.24, 5.25, za použití původní funkce z rovnice 5.42.



Obrázek 5.23: Příklad 1 - fitování hyperboly



Obrázek 5.24: Příklad 2 - fitování hyperboly

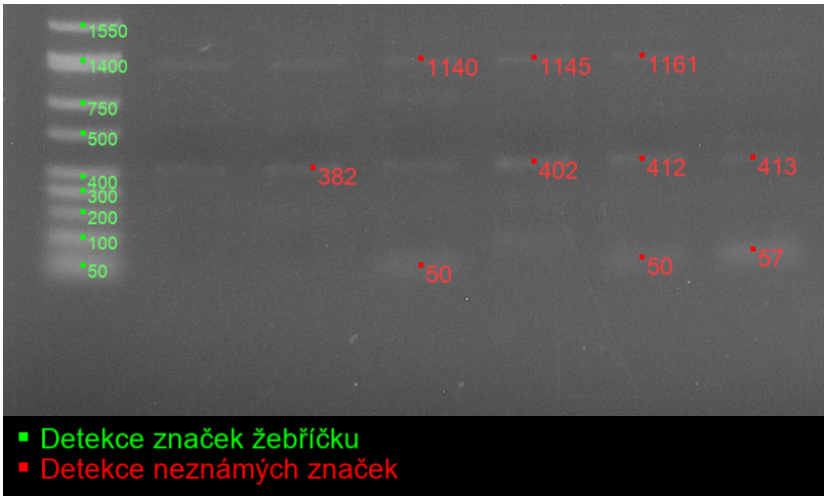


Obrázek 5.25: Příklad 3 - fitování hyperboly

Po získání koeficientů hyperboly pak odhad počtu bází neznámých značek probíhá ve dvou krocích. Nejprve jsou souřadnice neznámé značky (x, y) korigovány nalezeným polynomem $s(x, y)$ do nové souřadnice v. Následně je odhad počtu bází dán jako výstup funkce $f(v)$ dle rovnice 5.42 s naučenými parametry a, b, c.

5.5.2 Odhad počtu bází pomocí hyperboly v opačném smyslu a odhad lineární interpolací

Ačkoli v ideálním případě žebříčku je závislost počtu bází na souřadnici y skutečně hyperbolická, bohužel i při drobných chybách měření při dané variantě nelineárních nejmenších čtverců dochází k velkým chybám u značek s vyšším počtem bází, viz obrázek 5.26:



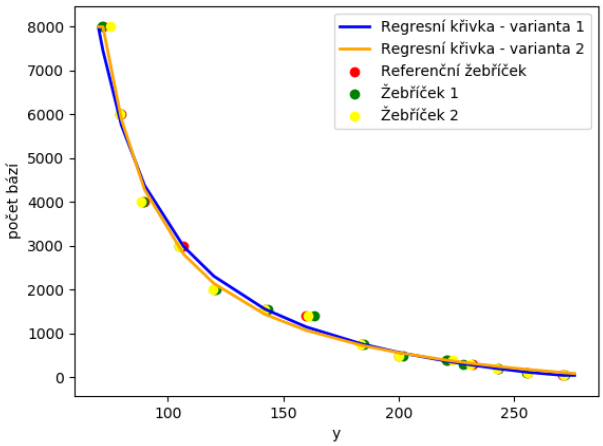
Obrázek 5.26: Chybný odhad fitovanou hyperbolou

Je patrné, že pro malé počty bází je odhad takto vybranou hyperbolickou funkcí dobrý, problém nastává u značky s nalezeným počtem bází ≈ 1100 leží na úrovi referenční značky s počtem bází 1400. Důvodem zřejmě je chybné nastavení referenčních dat pro žebříček a počty bází, neboť pro jiné varianty metoda fitování hyperbolickou funkcí fungovala. Eliminovat chybu značek s vyšším počtem bází bylo fitování v opačném smyslu, než popsaném v rovnici 5.43, tedy předpokládá se závislost počtu bází na pozici, kdy funkce h je definována v rovnici 5.42. Problematickou části přístupu je to, že přesná veličina (počet bází) je závislá na nepřesné měřené veličině (poloha značek). V této interpretaci jsou nalezeny parametry rovnice stejným způsobem, jako v případě sekce 5.5.1, pouze se od počátku prokládá funkce h, jejíž parametry

jsou řešením optimalizační úlohy:

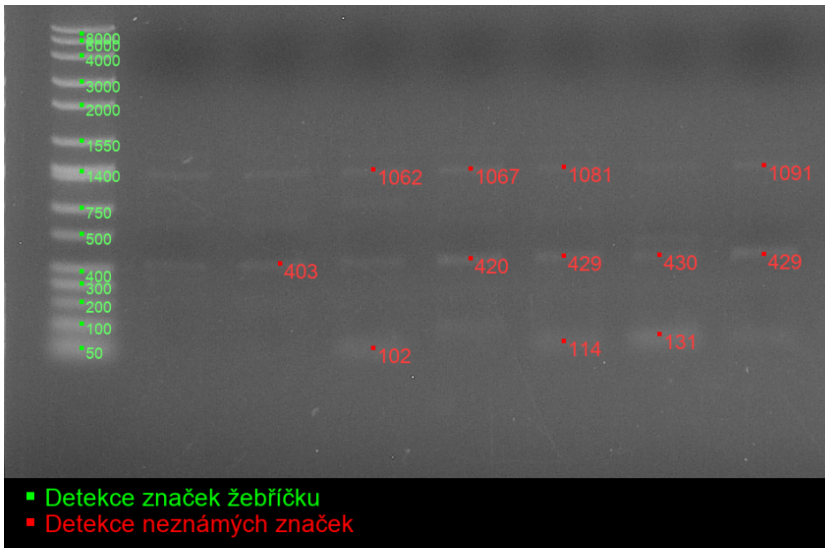
$$\arg\min\{\sum_i \sum_j (h(v_{j,i}) - z_i)^2 \;\Big|\; a \geq 0,\ b \leq \min_i y_{j,i},\ c \leq \min_i z_i\}. \qquad (5.45)$$

Porovnání hyperbolický křivek pro variantu 1 dle sekce 5.5.1, a variantu 2 dle rovnice výše, je v grafu:



Obrázek 5.27: Porovnání hyperbolických křivek pro různé varianty optimalizace

Rozdíly byly ale minimální, což se potvrdilo i v obrázcích, kde docházelo ke stejnému problému, jako na obrázku 5.26:



Obrázek 5.28: Výsledky odhadu počtu bází pomocí hyperbolické funkce dle rovnice 5.45

Z toho důvodu bylo nadále používáno fitování hyperbolické funkce dle úlohy 5.44. Pro případy, kdy je vstupní žebříček zatížen chybou, a jeho průběh neodpovídá hyperbolickému pro dostatečně přesné odhady počtu bází, byla přidána jednoduchá metoda lineární interpolací. Cílem lineární interpolace je zajištění, že počet bází neznámé značky je shora a zdola omezen počtem bází nejbližších úrovní v referenčním žebříčku. Nechť $s(x, y)$ je transformační polynom, a pro neznámou značku danou souřadnicemi (x, y) platí $w = s(x, y)$, kdy w je transformace do souřadnice referečního žebříčku. Nechť v_i jsou značky referenčního. Pak při odhadu počtu bází pomocí lineární interpolace se naleznou dvě nejbližší úrovně v_i, v_{i-1}, pro které platí:

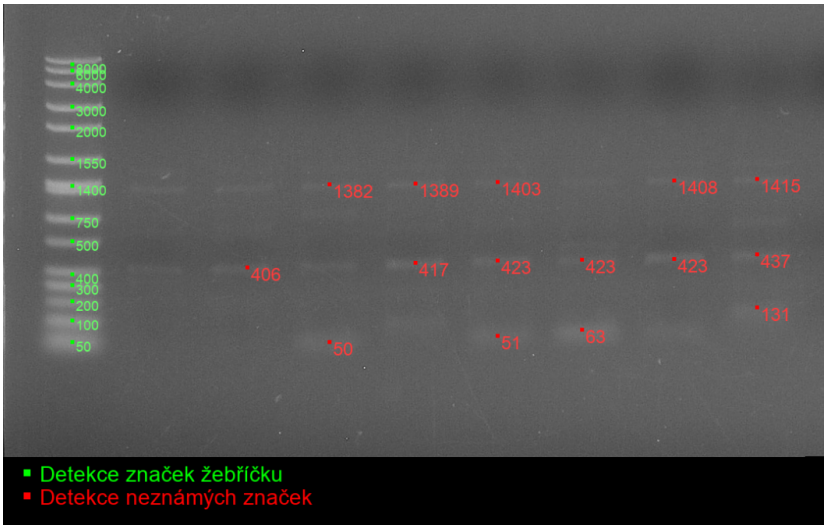
$$v_{i-1} \leq w \leq v_i. \tag{5.46}$$

Nechť z_i je počet bází i-té značky žebříčku. Pak pro odhad počtu bází neznámé značky, dané transformovanou souřadnicí w, tedy $h(w)$ platí:

$$h(w) = \frac{w - v_{i-1}}{v_i - v_{i-1}}(z_i - z_{i-1}) + z_{i-1}. \tag{5.47}$$

V případě, že se značka nenachází mezi dvěmi značkami referenčními, pak je $h(w)$ nastaveno na počet bází nejbližší referenční značky.

Užití lineární interpolace má za následek ztrátu hyperbolické závislosti funkce h pro odhad počtu bází. Na druhou stranu je však zajištěno, že odhady počtů bází značek jsou omezeny úrovněmi referenčních žebříčků, kdy případná výraznější nepřesnost je dána chybou korekčního polynomu $s(x, y)$. Výsledek je dokumentovaný na obrázku 5.29:



Obrázek 5.29: Ukázka odhadu počtu bází lineární interpolací

Kapitola 6

Výsledky

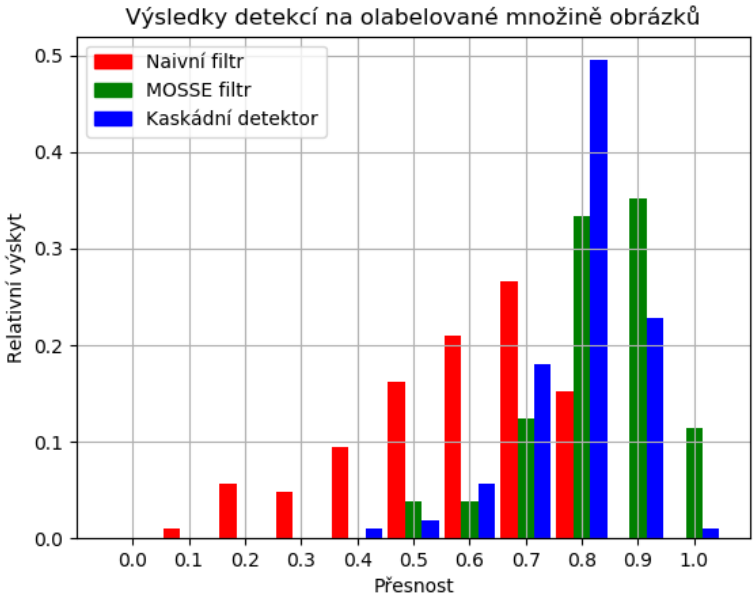
Zhodnocení výsledků algoritmů je rozděleno do několika sekcí. V první části 6.1 jsou hodnoceny výstupy detekčních algoritmů na skutečných anotovaných datech. V druhé části 6.2 jsou diskutovány výsledky detekčních metod na upravených datech, zatížených šumem, apod. Výsledky algoritmů druhé fáze zpracování jsou popsány v sekci 6.3, celkové zhodnocení výsledných algoritmů z hlediska reálné použitelnosti jsou diskutovány v sekci 6.4.

6.1 Hodnocení detekčních metod na reálných datech

Pro zhodnocení výsledku detekce byly provedeny výpočty hodnoty přesnosti dle rovnice 3.4. Pro jejich vizualizaci se předpokládala pevná hodnota parametru $w = 25$ v rovnici 3.2. Pro srovnání výsledků byl použit základní korelační filtr, využívající jako jádro filtru výstřižek jedné vzorové značky z vybraného obrázku. Filtr využíval stejnou strukturu prahování jako MOSSE filtr, a jeho parametry zpracování výstupu korelace byly optimalizovány stejným způsobem jako v případě MOSSE filtru. Bližší informace o implementaci jsou ve zprávě [24], výstup zpracování korelace tohoto filtru pak je zpracován stejným způsobem, jako výstup MOSSE filtru dle sekce 4.1.3. Oproti výsledkům naivního filtru v [24] byla v této práci množina obrázků rozšířena a mírně byl upraven způsob vyhodnocování. Vyhodnocování probíhalo nad 105 anotovanými obrázky, výsledek přesnosti acc byl zaokrouhlen k nejbližším 10 procentům. Data byla vynesena do společného histogramu pro detekci pomocí

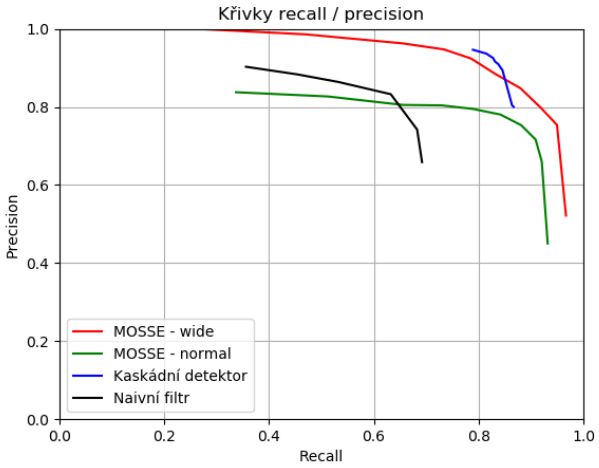
naivního korelačního filtru, MOSSE filtru a kaskádního filtru s MB-LBP deskriptory a post-processingem v obrázku 6.1. MOSSE filtr je v případě detekce v grafu 6.1 a grafu 6.3 užíval automatického rozlišování typu značky dle sekce 4.1.5, který ve všech případech vybral správnou heatmap pro zpracování výstupu korelace. Pouze na grafu s ROC křivkami 6.2 jsou vyneseny tréninkové křivky i pro rozdělené jednotlivé filtry MOSSE pro obě varianty značek.

Je patrné, že minimálně v porovnání s naivním korelačním filtrem jsou výsledky detekčních metod lepší. Potvrdila se teze, že naivní korelační filtr není dostatečně robustní. Ačkoli na obrázcích s tvarem značky, která byla použita jako vzor korelačního filtru, byly výsledky relativně dobré, na obrázcích s jinými typy značek pak výrazně klesala hodnota přesnosti. Pro komplexnější metody detekce se dá předpokládat, že s přesnější anotací značek, a v nejlepším případě při použití bounding-boxů, by byl trénink a rovněž celkové výsledky představených detekčních metod lepší.



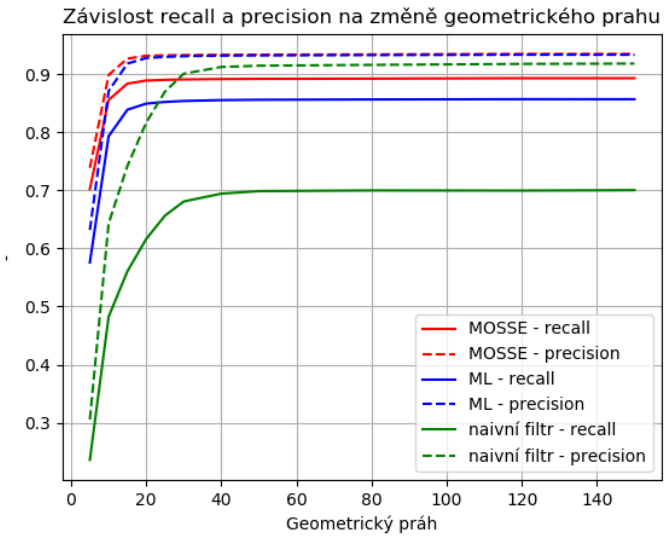
Obrázek 6.1: Histogram přesností pro různé metody detekce

Porovnání obou dvou metod detekce je možné například pomocí příslušných ROC křivek z kapitoly 4. Tréninkové ROC křivky pro jednotlivé filtry kombinovaného MOSSE filtru ze sekce 4.1, kdy obě varianty byly trénovány na příslušné množině obrázků (obrázky s širšími značkami pro MOSSE - wide, a ostatní pro MOSSE - normal) a kaskádního detektoru byly rovněž doplněny o tréninkovou křivku naivního korelačního filtru.



Obrázek 6.2: ROC křivky obou detektorů

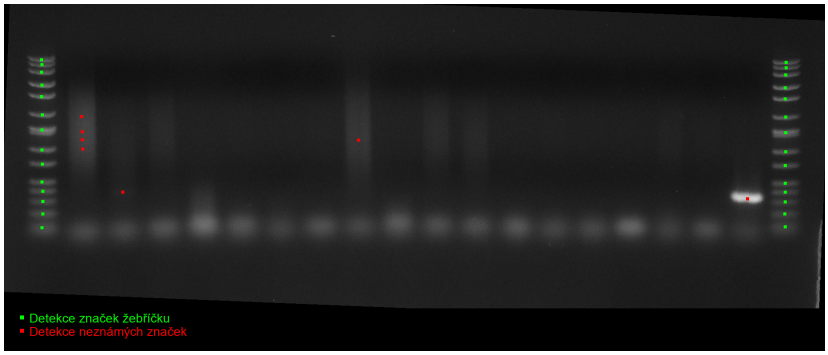
Výpověďní hodnota kombinovaného MOSSE filtru leží zejména v křivce pro MOSSE - normal, protože dataset byl vychýlen právě pro běžný (užší) typ značky. Rovněž byly ROC křivky vytvářeny pro omezené možiny obrázků pro vybrání vhodných hodnot prahování, proto se výsledky na celé množině obrázků následně mírně lišily. Jinak je patrné, že v optimálních bodech křivek, tj. v místech minimalizujících eukleidovskou vzdálenost k optimálnímu detektoru v bodě $(1, 1)$ je kaskádním detektorem lepší ve smyslu precision, zatímco MOSSE filtr ve smyslu recall. To se projevilo tak, že výsledky detekce kaskádním detektorem byly robustnější vůči FP detekcím na šmouhách (viz obrázky), MOSSE filtr má nižší množství FN případů, zejména pak lépe detekuje blízké značky, např. v žebříčcích. Podobný výsledek lze dokumentovat také na grafu 6.3. Při zkoumání průměrných hodnot recall a precision v závislosti na šířce geometrického prahu dle rovnice 3.2, tedy nastavení maximální možné vychýlky detekované značky od anotované referenční značky ve smyslu osy x. V tomto případě jsou již hodnoty recall a precision jsou určeny nad celou anotovanou množinou obázků (105 vzorků). Kromě potvrzení, že použitá původní šířky $w = 25$ dobře zohledňovala chyby při trénincích, jsou dobře dokumentované hodnoty recall a precision pro oba typy detekcí, navíc také pro detekci naivním korelačním filtrem. Zde již proběhlo vyhodnocení na všech anotovaných datech. Je patrné, že MOSSE filtr má skutečně lepší vlastnosti ve smyslu recall. Výhoda kaskádního detektoru ve smyslu precision není příliš významně viditelná, důvodem však může být menší počet TP pro kaskádní detektor, zejména na rozmazaných značkách.



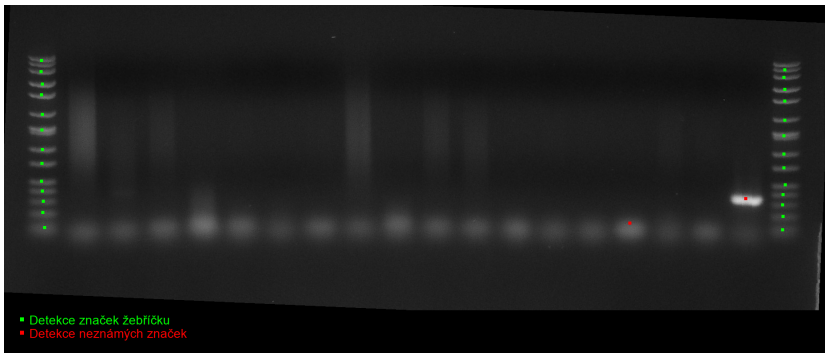
Obrázek 6.3: Závislost průměrných recall a precision na hodnotě geometrického prahu

V grafech je patrné, že i přes rostoucí šířku prahu (a tím i klesajícími nároky na přesnost detekce) nejdou hodnoty precision a recall k 1. Prvním důvodem může být konstantní výška geometrického prahu, která byla zvolena na počátku experimentu. Druhým důvodem je však zřejmě způsob výpočtu obou hodnot. Celkový počet detekcí algoritmu (tj. všechny detekce algoritmu před roztříděním) není závislý na hodnotě geometrického prahu. Tedy pokud platí, že počet detekcí se nerovná počtu anotací, alespoň jedna z hodnot recall a precision nebude rovna 1 ani při posunutí rozměrů geometrického prahu do nekonečna. Pokud totiž počet detekcí větší než počet anotací, pak nutně alespoň jedna z detekcí bude vyhodnocena jako FP, v opačném případě jedna anotace bude postrádat detekci, což bude vyhodnoceno jako FN. Průměrování recall a precision nad všemi obrázky pak ve výsledků způsobí, že recall ani precision nedosáhnou hodnoty 1.

Kaskádní klasifikátor se ukázal jako kvalitnější ve smyslu robustnosti vůči zašuměným vzorkům. Ačkoli s náhodným šumem si obvykle poradily obě metody, pro MOSSE filtr se ukázaly být problematické „šmouhy“ na některých obrázcích, které způsobovaly falešné detekce. Naproti tomu kaskádní detektor si těmto chybám dobře vyhýbal, protože byl trénován na vzorcích, které tyto šmouhy rovněž obsahovaly. Celkem bylo v množině obrázků nalezeno 10 takových vzorků, kde MOSSE detektor vytvořil FP detekce z důvodu „šmouh“, a kaskádní detektor ve všech těchto případech „šmouhy“ jako značky nedetekoval. Ukázka je na obrázcích 6.4, 6.5.

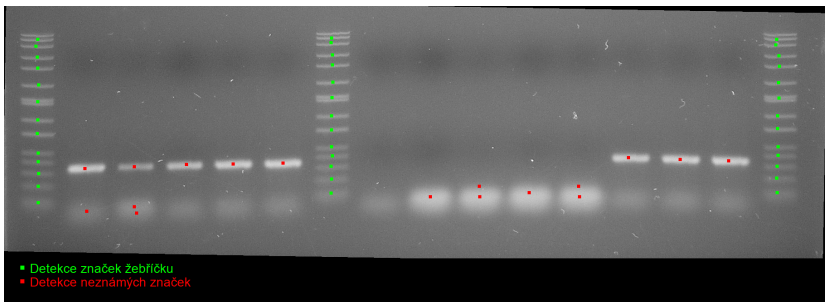


Obrázek 6.4: Detekce MOSSE filtrem v obrázku s problematickými „šmouhami“



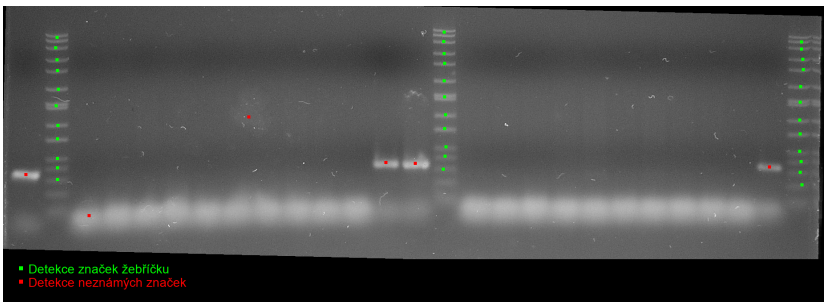
Obrázek 6.5: Detekce kaskádním detektorem v obrázku se „šmouhami“

Oproti tomu se v některých případech kaskádního detektoru stávalo, že rozmazané značky byly detekovány nadvakrát, viz obrázek 6.6. Důvodem pravděpodobně byl kompromisní způsob agregace positivních detekcí dle sekce 4.2.3. Zvětšením strukturního elementu nebo počtu iterací uzavření by i u těchto značek ve výsledku došlo ke splynutí všech positivních okének, v důsledku této úpravy by však došlo ke zhoršení detekcí jednotlivých značek žebříčků. Tento problém by zřejmě vyřešil kaskádní detektor dle sekce 4.2.4, který se však nepodařilo zprovoznit s dostatečně dobrými detekčními vlastnostmi.

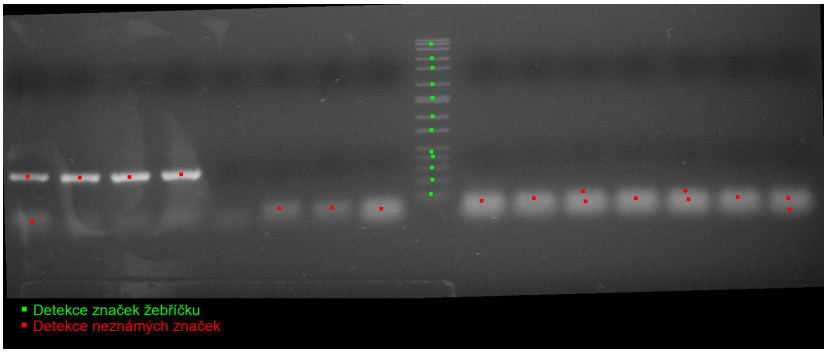


Obrázek 6.6: Problematické dvojité detekce kaskádního detektoru

Drobný šum, například na obrázku 6.7, obvykle nemá vliv na úspěšnost detekce pro obě metody. Pro MOSSE filtr se jako problematické projevují takové obrázky, na kterých jsou deformace v obrázku velikostně srovnatelné se značkami. Jedná se například o odlesky jako na obrázku 6.8. I přesto je většina značek správně detekována.



Obrázek 6.7: Příklad detekce MOSSE filtrem na obrázku zatíženém drobným šumem



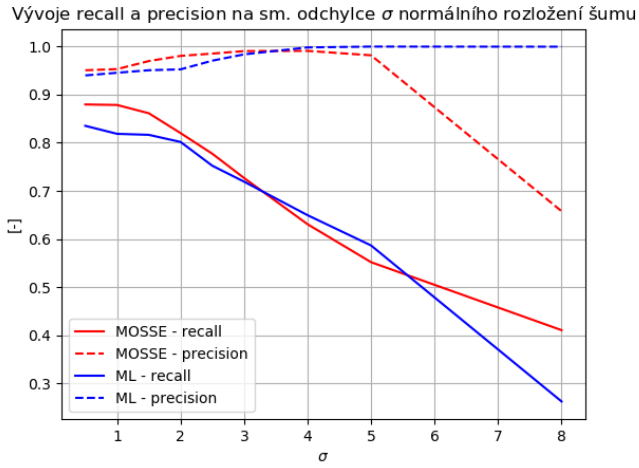
Obrázek 6.8: Detekce MOSSE filtrem na obrázku zatíženém většími chybami

6.2 Zhodnocení výsledků detekce na syntetických datech

Test na syntetických datech pro evaluaci vlivu šumu byl proveden na 30 anotovaných obrázcích. Každý vstupní obrázek f byl zatížen šumem X s normálním rozdělením o nulové střední hodnotě a směrodatné odchylce σ. Tj. syntetický obrázek f_{σ_i} byl dán následovně:

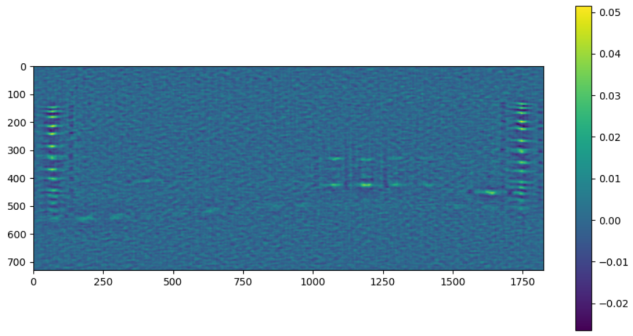
$$f_{\sigma_i}(x, y) = f(x, y) + X. \tag{6.1}$$

Každý obrázek byl takto zatížen šumem pro různé hodnoty σ a průměrné hodnoty recall a precision dle 3.6 byly vyneseny do grafu.



Obrázek 6.9: Vliv šumu na výsledky detekce

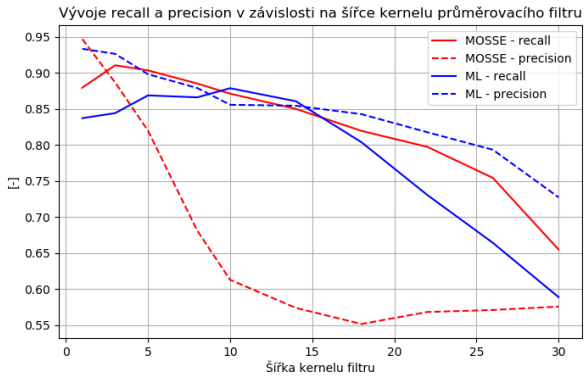
Lepších výsledků by se zřejmě dosáhlo při použití takto generovaných vzorků i v kroku tréninku, což platí zejména pro kaskádní detektor. U MOSSE filtru se zdálo, že značky zanechávají stále relativně velké špičky v heatmap, viz obr. 6.10. Problémem se ukázal způsob výpočtu prahu, který v případě zašuměných dat způsobil že mnoho špiček bylo pod hodnotou prahu, a nemohlo tím pádem být vyhodnoceno jako TP. Více k problematice prahování MOSSE filtru je v následujícím odstavci.



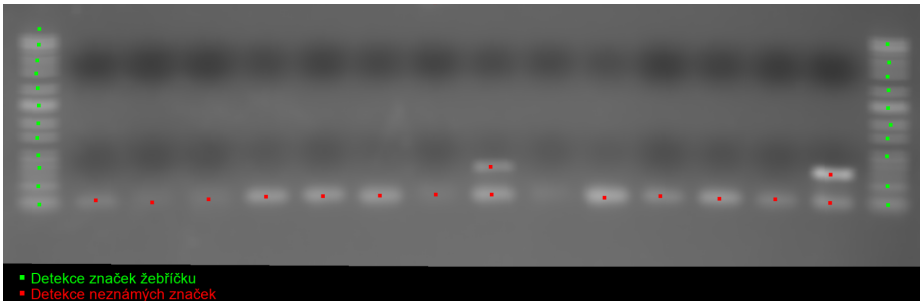
Obrázek 6.10: Heatmap MOSSE filtru při vstupním šumu se $\sigma = 5$

Vliv rozmazání na schopnost detekce je ukázán na grafu 6.11. Simulování rozmazání bylo prováděno konvolucí vstupního obrázku se čtvercovým filtrem dané šířky, jež je vynesena na vodorovné ose grafu. Z výsledků je patrné, že kaskádní detektor relativně dobře fungoval i při velkém rozmazání, jako na obrázku 6.12. MOSSE filtr trpí obdobným problémem volby prahování jako v případě šumu, avšak v opačném smyslu, kdy by bylo nutné zvýšit práh. Na obrázku 6.13 je heatmap korelovaného obrázku po prahování, na níž jsou

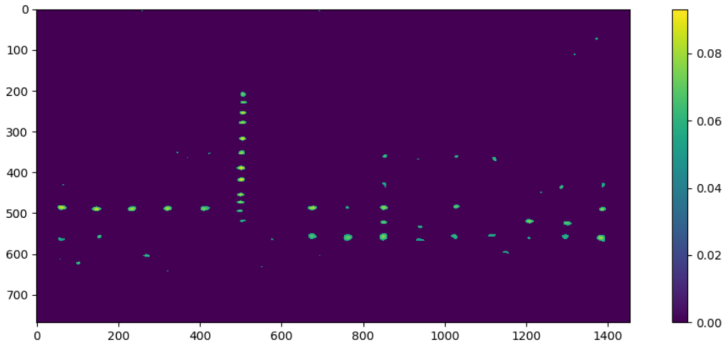
patrné drobné ostrůvky FP, které nebyly zahozeny prahováním. Výsledek detekce v tomto případě je na obrázku 6.14. Ručním zvýšením hodnoty prahu θ z rovnice 4.11 se pak dosáhlo výsledku na obrázku 6.15. Tedy hlavním zdrojem chyb v obou případech detekce na upravených datech je způsob vyhodnocování heatmap korelace. Pro dosažení lepších výsledků by bylo potřeba buď zcela upravit způsob vyhodnocování, nebo změnit způsob výpočtu prahu na robustnější. Možná úprava způsobu vyhodnocování by byla následující. Nejprve by byl vstupní obrázek oprahován základním prahem, který by byl natrénován tak, aby způsobil minimální množství FN případů. Výsledek by mohl být podobný jako na obrázku 6.13. Poté by se špičky zpracovaly stejným způsobem, jako nyní, pomocí prohledávání do šířky z každého lokálního maxima, a zároveň by se určil kumulativní součin hodnot navštívených položek heatmap. Pak by se pozice lokálního maxima prohlásila za značku jen tehdy, pokud by kumulativní součet překročil druhý natrénovaný práh. Minimálně na obrázku 6.13 by došlo k odstranění mnoha FP detekcí, které jsou reprezentovány pouze drobnými špičkami, které překonaly natrénovaný práh.



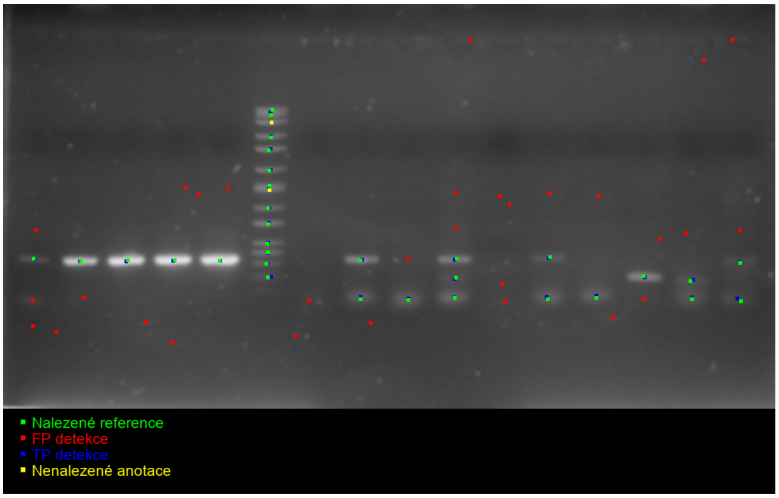
Obrázek 6.11: Vliv velikosti kernelu průměrovacího filtru na výsledky detekce



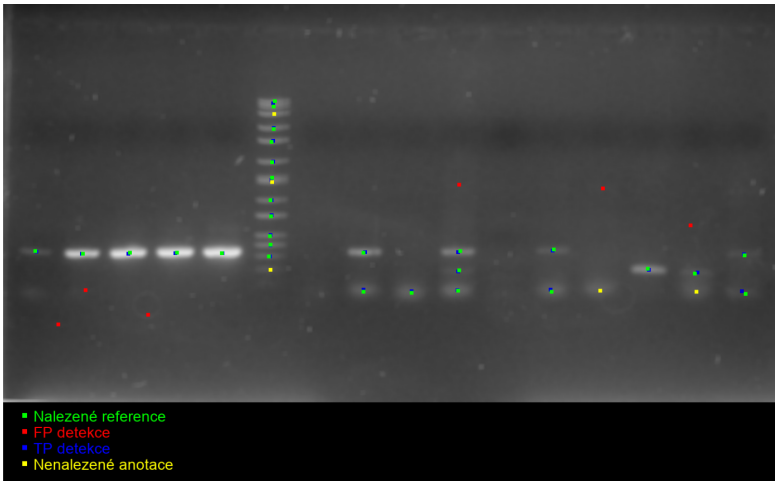
Obrázek 6.12: Výsledek detekce kaskádním detektorem při šířce kernelu průměrovacího filtru 20



Obrázek 6.13: Heatmap korelovaného obrázku po prahování



Obrázek 6.14: Výsledek detekce zašuměného obrázku s použitím původní hodnoty prahování



Obrázek 6.15: Výsledek detekce zašuměného obrázku s použitím nové hodnoty prahování

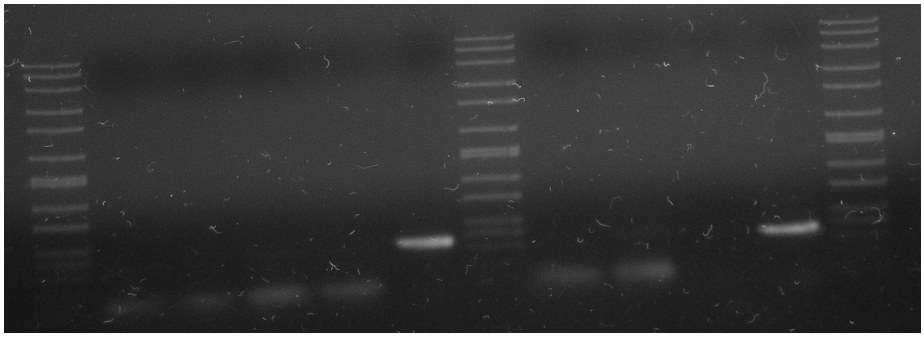
Drobné zvýšení precision v případě grafu 6.9 lze zdůvodnit klesáním celkového počtu detekcí. Růst recall v případě grafu 6.11 lze zejména v případě MOSSE filtr snadno vysvětlit špatnou volbou prahu. Nedostatečná hodnota prahu má za následek velký počet detekcí, což za následek strmý pád precision, ale část těchto detekcí se rovněž projeví na zvýšení počtu TP, resp. na zmenšení FN. Pro další zvyšování míry rozmázání má pak pro MOSSE filtr za důsledek, že spojené značky v žebříčcích již nejsou detekovány samostatně, ale pouze jako jeden „blob“, což výrazně snižuje hodnotu TP, a tím i recall.

Velikost obrázku má vliv zejména na detekci pomocí MOSSE filtru. Kaskádní detektor samotný v implementaci z OpenCV knihovny provádí multiscale detekci. MOSSE filtr v použité implementaci tuto možnost neposkytuje. Pro funkčnost nad různými, velmi rozdílnými velikostmi obrázků, by bylo nutné měnit vstupní rozměry obrázku tak, aby značky v nich velikostně odpovídaly naučenému filtru.

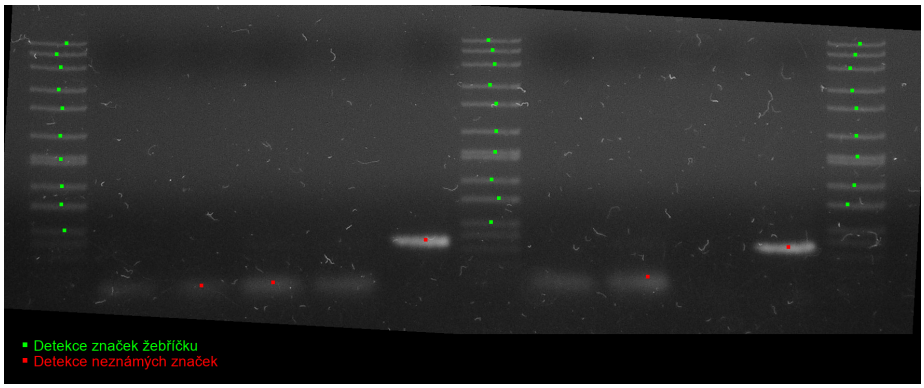
6.3 Zhodnocení výsledků algoritmů na zpracování detekovaných značek

Porovnávat výsledky ostatních algoritmů, používaných pro provádění korekcí, nalezení žebříčků a i závěrečné fitování, je obtížné kvůli absenci expertních ground-truth výsledků, a zhodnocení tedy vychází ze subjektivní vizuální kontroly získaných výsledků. K několika obrázkům byly k dispozici i počty bází neznámých značek, na těchto obrázcích je pak provedeno vyhodnocení pomocí těchto správných dat.

Po prozkoumání výsledků se ukázalo, že algoritmus na hledání žebříčků selhal pouze ve dvou případech (z celkových 250), kdy důvodem byla špatná detekce značek. Algoritmus pro korekci geometrického zkreslení provedl správnou opravnou rotaci podle nalezených žebříčků vždy správně, i v případech relativně velké rotace, jako na obrázcích 6.16, 6.17. Dále by proveden test, kdy obrázek vyrovnaný obrázek byl po načtení rotován, a následně byl výsledek vyhodnocován jako rozdíl úhlů použité rotace a úhlu odhadnuté algoritmem, viz graf 6.18.



Obrázek 6.16: Vstupní obrázek před rotací

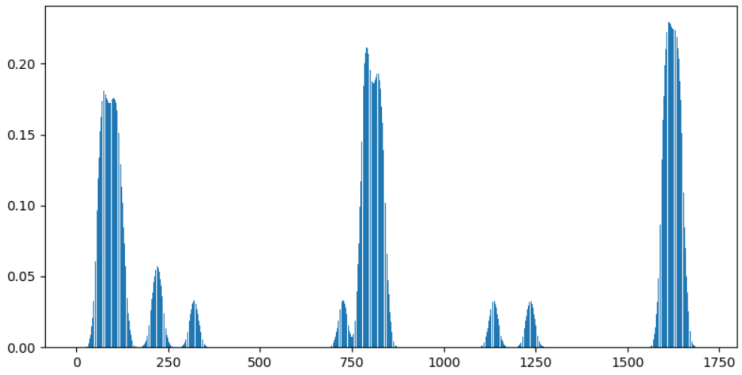


Obrázek 6.17: Obrázek po detekci a provedení rotace

Obrázek 6.18: Vizualizace přesnosti odhadu odchylky úhlu

Ukázalo se, že do odchylky ± 5° byl odhadnutý úhel velmi přesný. Pro

vyšší odchylky nastal problém s algoritmem na rozdělování značek do skupin. Protože je založen pouze na projekci do osy x, při větších výchylkách již nebylo možné nastavit vhodně parametr σ algoritmu dle sekce 5.1, tak, aby nedocházelo ke splývání různých skupin a zároveň došlo ke splynutí skupin jednotlivých, viz problematický histogram 6.19. Zjevně by k zajištění funkčnosti byl potřeba vytvořit robustnější postup než pouhé provedení projekce středů značek do osy x.

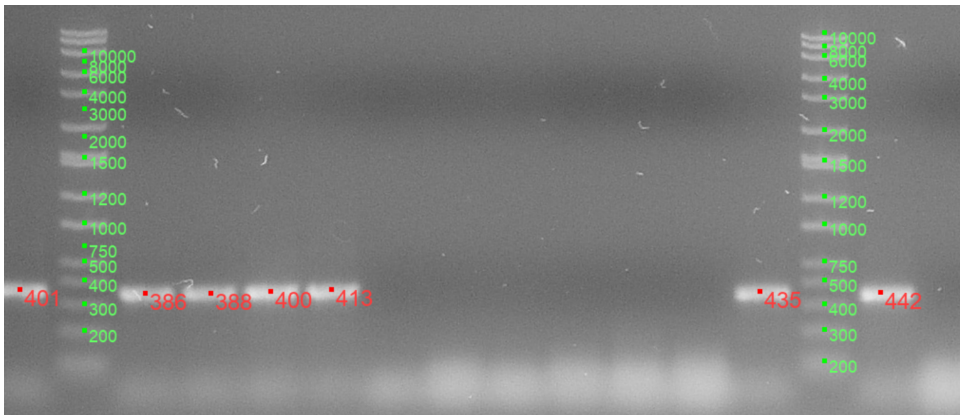


Obrázek 6.19: Nedostatečně vyhlazený histogram pro rozdělení značek

Po prozkoumání výsledků fitování žebříčků se zdá, že ve většině případů, kdy expertní žebříček odpovídal žebříčku na obrázku, byl algoritmus schopný fitovat žebříček i přes případné chybějící značky. Bylo prohlédnuto 80 obrázků s fitovanými žebříčky. Alespoň 60 bylo obrázků bylo subjektivně označeno za velmi přesné fitování, kdy vytvořený žebříček skutečně lícoval se značkami v obrázky, i přes případné FP detekce a chybějící detekce značek. Na dalších 15 obrázcích algoritmus zvládl dobře fitovat spodní část žebříčku, problémem se ukázala být horní část, kde se typicky objevuje několik velmi blízkých značek, které se špatně diferencují při detekci. Pak se při fitování může stát, že značce v horní části žebříčku jsou přiřazeny špatná značka referenční. Chybně přiřazená značka je však tak blízko oběma referenčním, že vliv na kritérium je minimální. Ve výsledku pak dochází ke špatné rekonstrukci chybějících značek, které se pak objevují mimo značky žebříčku. Posledních 10 případů byla fitování, která nebyla možná použít pro odhad počtu bází ostatních značek. Je však pravděpodobné, že výsledky těchto příkladů byly zkresleny faktem, že k fitování nebyl použit správný žebříček. Pro účely testování byly změřeny pouze 3 různé referenční žebříčky, a je tedy možné, že některé špatné případy obsahovaly neznámý typ žebříčků. Příklad chybného fitování do struktury je na obrázku 6.20.

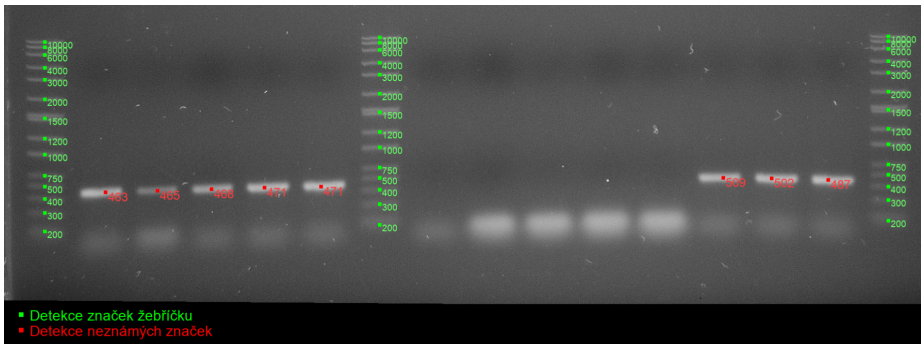
Problémem se ukázalo, že hodnota kritéria byla pro toto fitování velmi podobná případu správného fitování (pravý žebříček v obrázku 6.20). Rovněž se zdá, že problémem ve formulaci optimalizační úlohy dle rovnice 5.17 je fakt,

že nedochází k penalizaci při použití příliš malých hodnot β, resp. obecně při porovnání dvou kritérií by bylo vhodné penalizovat variantu užívající menší scale, nebo od počátku místo absolutní chyby užívat chyby relativní. V takovém případě by se však již nejednalo o úlohu optimalizace s lineární kriteriální funkcí.

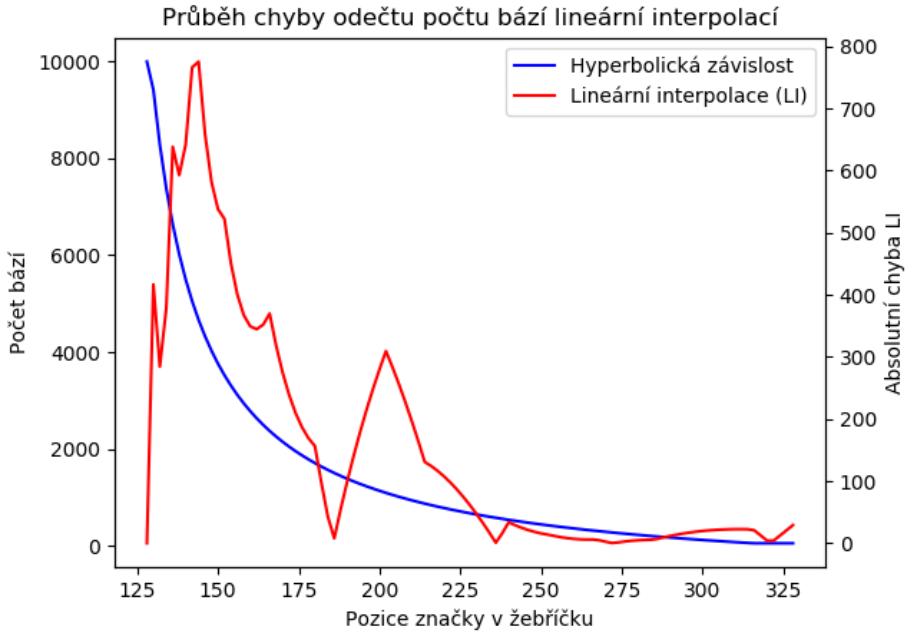


Obrázek 6.20: Chybné fitování na levém žebříčku

Testování algoritmů pro odhad počtu bází spočívalo v ručním odhadu počtu bází na základě porovnání s žebříčkem, a následném porovnání s programem generovanými daty. Až na problematický typ žebříčku, popsaný v sekci 5.5, který byl pravděpodobně špatně odečten, fungoval odhad počtu bází pomocí hyperbolické funkce dobře ve smyslu, že odhad počtu bází ležel mezi vrstevnicemi korigujícího polynomu vyznačeného značkami referenčních žebříčků, viz obr. 6.21. Použití lineární interpolace tuto vlastnost má od počátku. Při použití lineární interpolace pak lze vyjádřit absolutní chybu lineárního odhadu na grafu 6.22.

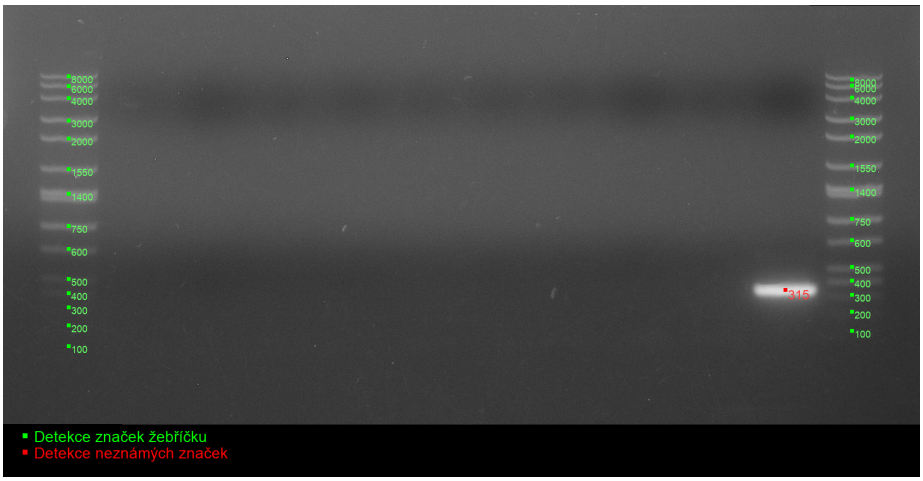


Obrázek 6.21: Odhad počtu bází hyperbolickou funkcí

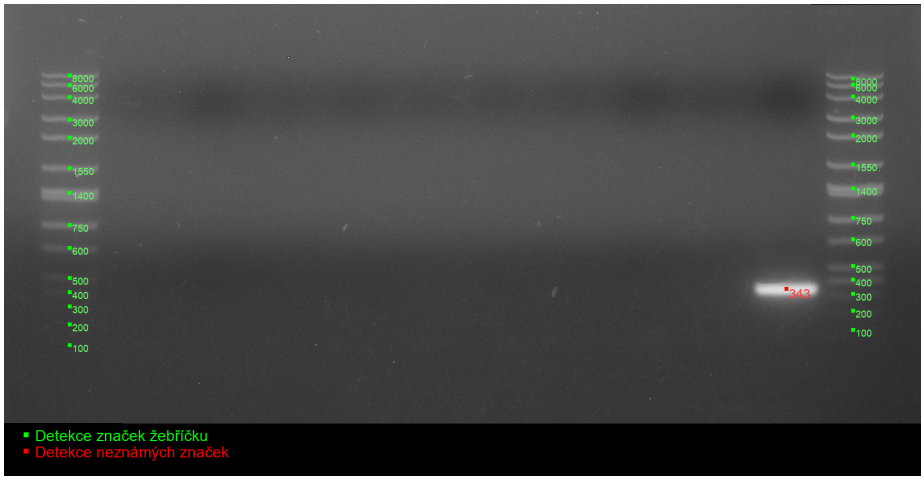


Obrázek 6.22: Absolutní chyba lineární interpolace v závislosti na poloze v žebříčku

K několika reálným obrázkům byla známá anotace značek. Pro experimenty s LD primery se očekávalo odhadnout počet bází na hodnotu 357 bp (base pairs) pro značky mimo žebříček. Odhad proběhl při použití hyperbolické funkce, i při použití lineární interpolace. Experimentálně bylo vyhodnoceno 26 značek na 12 obrázcích s LD primery. Výsledky se získaly jako absolutní rozdíl odhadu počtu bází příslušné značky od očekáváného počtu bází 357 bp. Statistické hodnoty jsou v tabulce 6.1.



Obrázek 6.23: Odhad počtu bází hyperbolickou funkcí - LD primery



Obrázek 6.24: Odhad počtu bází lineární interpolací - LD primery

	Medián chyb	Průměr chyb	Maximum chyb
Lineární interpolace	14.5	16.5	40
Hyperbolická závislost	22.5	21.9	50

Tabulka 6.1: Výsledky odhadů počtu bází

Výsledky ukazují ve prospěch lineární interpolace. Problémem pro hyperbolickou závislost se ukázal být fakt, že ve naprosté většině případů byl odhad počtu bází podhodnocen, tedy se zrejmě jednalo o problém ve fitování nepřesných značek, viz graf 5.23. Navíc by se zřejmě horší vlastnosti lineární interpolace projevily až při odhadech značek s vyšším počtem bází, viz graf 6.22.

6.4 Zhodnocení použitelnosti implementace

Detekční algoritmy předpokládají vstupní obrázky takové, kde značky reprezentují molekuly, pohybující se od shora dolů, až na chybu v rotaci. Rovněž je potřeba, aby se obrázek zobrazoval pouze z jedné provedené elektroforézy s jedním typem značky, jinak je potřeba rozdělit vstupní obrázek tak, aby byla podmínka splněna. Je to nutné z důvodu správného rozdělení značek do žebříčku a jejich fitování, a rovněž při použití více typů značky detekci pomocí MOSSE filtr nebude fungovat kvůli kombinaci více typů značek v jediné detekci, rovněž se předpokládá srovnatelná velikost značek elektroforézy vzhledem k značkám použitým k tréninku. Detekční metody by zřejmě

dosáhly lepších výsledků při kvalitnější úvodní anotaci dat, a případném rozšíření datasetu.

Praktické fungování algoritmů bylo testováno ručně v uživatelském rozhraní, popsaném v kapitole 7. Pokud byl poskytnut správný typ žebříčku spolu s obrázkem splňujícím předpoklady, pak jsou výsledky obvykle dobré. Pokud jsou značky ostré, tj. dobře viditelné, obvykle problém nenastává, při výskytu více rozmazaných značek je problém v existenci FP a FN detekcí, spojených právě s rozmazanými značkami.

Použitelnost výsledků z hlediska časové náročnosti ukázalo následující. Implementace OpenCV pro kaskádní detektor zajistila, že na běžném počítači s procesorem Intel i5 $2.3\,GHz$ byl čas potřebný pro aplikaci sliding window a zpracování positivních okének v řádu desetin sekund, nejvýše do 1 sekundy. Oproti tomu problémem MOSSE filtru byl výpočet Fourierovy transformace v obrázcích, který trval obvykle $1.5 - 1.8\,s$. FFT byla implementována pro demonstrační účely pomocí knihovny *scipy*, při použití optimalizovaných knihoven nebo výpočtu FFT pomocí GPU by se zřejmě dosáhlo lepších výsledků. S příhlédnutím k tomu, že se provádí výpočet FFT pro dva typy filtrů pro každý vstupní obrázek, trvá proces detekce i se zpracováním heatmap $3 - 4\,s$.

Rozdělení značek, oprava rotace i odhad počtu bází jsou z časového pohledu zanedbatelné. Dalším kritickým bodem je fitování žebříčků. V sekci 5.3 zvolený počet iterací algoritmu způsobil, že fitování jednoho žebříčku trvá obvykle $0.4\,s$. S rostoucím počtem žebříčku v obrázcích pak rovněž roste doba nutná k provedení fitování.

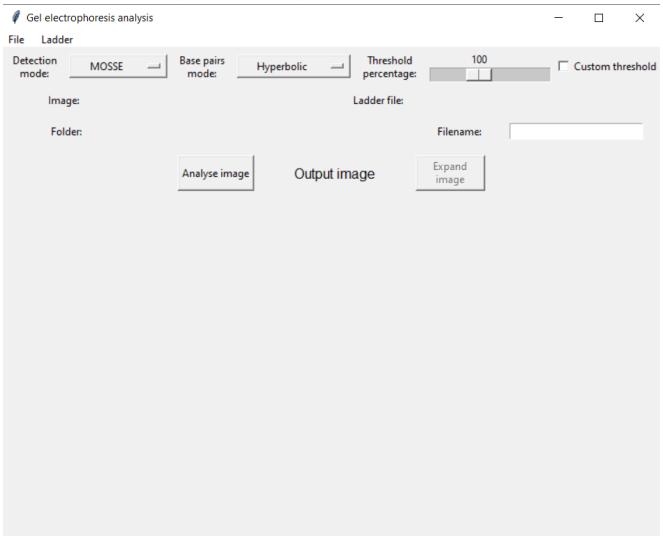
Příklady výsledků detekce a fitování jsou v příloze A.

Kapitola 7

Aplikace

7.1 Popis uživatelského rozhraní

Uživatelské rozhraní bylo vytvořeno za použití knihovny Tkinter. Úvodní okno rozhraní zajišťuje možnosti vybrání obrázku pro zpracování, nastavení cílového adresáře a jména souboru pro uložení, vybrání typu žebříčku určeného pro fitování, uložení souborů a spuštění analýzy. Rovněž v tomto okně lze vybrat variantu algoritmu pro odhad počtu bází a způsob detekce značek.



Obrázek 7.1: Úvodní okno programu

Po vybrání obrázku a typu žebříčku (pokud typ žebříčku není vybrán, je použito uložené nastavení) lze spustit analýzu obrázku. Analýza obrázku je závislá na vybraném módu. Výběr MOSSE provádí detekci pomocí MOSSE filtru, výběr ML provádí detekci kaskádním klasifikátorem s MB-LBP deskriptory a morfologickým post-processingem. Po spuštění analýzy je nejprve provedena detekce značek, následuje nalezení žebříčků, opravení rotace v obrázku a následně fitování žebříčku. V poslední fázi se určí počty bází neznámých značek s přihlednutím ke korigujícím polynomům, které řeší chyby způsobené nehomogenitou proudu. Způsob odhadu počtu bází značek se provádí dle volby přepínače Base pairs mode. Mód Hyperbolic užívá k odhadu hyperbolickou funkci s parametry naučenými metodou nelineárních čtverců, mód Linear Intrp provádí odhad pomocí prosté lineární interpolace. Rovněž lze pomocí posuvné stupnice vybrat hodnotu modifikující natrénované hodnoty prahování, která má vliv na počet detekovaných značek. Pokud před analýzou s novou hodnotou prahování byla provedena analýza stejným typem detektoru, odpadá nutnost počítat korelaci pro MOSSE filtr a provádět samotnou detekci pro kaskádní detektor, což zejména u MOSSE filtru výrazně urychluje tyto opakované analýzy.

Pro nastavení uživatelského žebříčku je možnost použít tlačítko Set ladder. Po stisknutí se otevře nové okno, viz 7.2. Po vybrání textového souboru, který obsahuje k řádků celých čísel, obsahujících počty bází v nově nastavovaném žebříčku, lze vybrat vzorový obrázek. Po jeho vybrání je pomocí MOSSE detektoru lokalizován žebříček v obrázku, jsou detekovány značky v něm a žebříček je z obrázku „vystřihnut“. Poté lze myší odebrat chybně lokalizované značky, přidat chybějící nebo schválit výstup automatického systému. Tedy poté, co je v obrázku zaneseno k značek, je možné žebříček uložit do složky s programem a později ho používat při analýzách. Alternativně lze pro vytvoření modelu okamžitě použít pouze referenční hodnoty počtů bází. Referenční souřadnice jsou v takovém případě získány použitím hyperbolické funkce dle sekce 5.5.1. V tomto případě je však výsledek následného fitování silně závislý na přesnosti hyperbolického rozložení značek žebříčku podle referenčních počtů bází v proběhlém experimentu, z toho důvodu je vhodnější vygenerovat ladder file. Aplikace předpokládá, že vstupní obrázky jsou orientovány tak, že značky se pohybují od horní strany obrázku ke spodní, a rovněž se předpokládá, že značky žebříčku s nejvyšším počtem bází se nacházejí nejblíže hornímu okraji obrázku.

Aplikace ukládá jak výsledný obrázek, obsahující fitované žebříčky a neznámé značky s doplněním počtu bází, tak soubor typu CSV, který obsahuje pro každou detekovanou značku mimo žebříčky index její lane, index značky v dané lane a odhad počtu bází.

Obrázek 7.2: Okno nastavování žebříčku

7.2 Popis souborů a spuštění aplikace

Soubory .py, použité v projektu, jsou následující:

- *application.py* - nastavuje okno uživatelské rozhraní, rovněž slouží ke spouštění programu uživatelem.
- *cascade_filter.py* - obsahuje funkce pro vytváření vzorků pro trénink kaskádního detektoru a funkce pro provádění detekcí kaskádním detektorem.
- *detection_evaluation.py* - obsahuje funkce pro výběr lokálních maxim pro MOSSE filtr, a dále funkce pro rozdělení značek do žebříčků, odhad vychýlení obrázku a korekci příslušného vychýlení.
- *evaluation.py* - obsahuje funkce pro fitování žebříčku, a rovněž třídu BandSolver pro odhad počtu bází značek a korekci chyb způsobených nehomogenitou proudu.
- *evaluation_nogrb.py* - obsahuje stejné funkcionality jako soubor výše, ale nepotřebuje pro použití externí solver Gurobi.
- *mosse.py* - obsahuje funkce pro trénink a detekci pomocí MOSSE filtru.
- *naive_detection.py* - obsahuje funkce pro detekci naivním korelačním filtrem.

- *picture.py* - obsahuje třídu Picture pro načítání a ukládání obrázků, jejich předzprácovánía generování vizualizací. Dále obsahuje funkce pro výpočet přesnosti, recall a precision, a generování ground-truth polí pro trénink MOSSE filtru.
- *support.py* - obsahuje podpůrné funkce pro vizualizace výsledků.

Kód byl psán v jazyce Python verze 3.7, za použití externích knihoven *numpy, scipy* pro zpracování dat, *Pillow* pro práci s obrázky, *matplotlib* pro vizualizaci grafů, *OpenCV* pro kaskádní detektor a *Tkinter* pro vytvoření uživatelského rozhraní. K řešení lineárních programů je použit externí solver *Gurobi* s jeho python interface *gurobipy*.

Kromě souborů výše jsou navíc dodány následující soubory:

- *mosse_filter.npy* - uložený MOSSE filtr pro běžný typ značek.
- *mosse_filter_wide.npy* - uložený MOSSE filtr pro širší typ značky.
- *cascade_lbp1.xml* - natrénovaný kaskádní detektor s MB-LBP deskriptory.
- *font.ttf* - bitmap s fontem Noto Mono, [25]

Rovněž je přiložena složka *samples* s několika testovací obrázky a vzorovým souborem pro generování ladder file.

Aplikace pro analýzu obrázků s uživatelským rozhraním se spouští zavoláním souboru *application.py* z terminálu nebo příkazové řádky. Pro spuštění je nutné zajistit, že Python interpreter je schopen importovat všechny knihovny popsané výše. Rovněž lze rozšířením volání o další přepínače omezit funkcionality aplikace, ale také zajistit spustitelnost aplikace na počítačích bez nainstalovaných knihoven Gurobi a OpenCV. Návod k použití je v příloze C, spolu s popisem vzorových souborů.

Kapitola 8

Závěr

Cílem bakalářské práce bylo navrhnout a otestovat postupy pro analýzu gelové elektroforézy. První částí práce byly metody pro detekci jednotlivých značek ze snímku gelové elektroforézy. Celkem byly vytvořeny dvě varianty detekce, v prvním případě se používalo korelace obrázku s natrénovaným MOSSE filtrem, který je popsán v sekci 4.1, v druhém případě se k vytvoření detektoru užilo metod strojového učení a výsledkem byl kaskádní klasifikátor, užívající deskriptory Multi-Block Local Binary Patterns, popsaný v sekci 4.2. Testy probíhaly na ručně anotované množině obrázků z gelové elektroforézy. Obě metody byly porovnány výpočtem přesnosti a vykreslením ROC křivek. Detekce na kvalitních obrázcích s nerozmazanými značkami probíhala dobře, problematickým vlivem při tréninku a zejména následném vyhodnocování byla absence expertní anotace, která byla nahrazena ruční anotací obrázků, výsledky jsou popsány v sekcích 6.1 a 6.2.

Následné zpracování nalezených značek se skládalo se z několika kroků. Pro vybrání žebříčků se užila projekce značek do souřadnicové osy x (viz 5.1), pro opravu rotace vstupního obrázku formě rotace byla vychýlka obrázku stanovena pomocí prokládání žebříčků afinním podprostorem a stanovením úhlu spektrálním rozkladem (viz 5.2). Úloha fitování žebříčku do referenční struktury byla řešena jak metodami lineárního programování, tak jako problém přiřazení Maďarským algoritmem, s využitím algoritmů inspirovaných algoritmem RANSAC pro počáteční odhad parametrů, popis řešení je v sekci 5.3. Problém zkreslení způsobeného nehomogenitou proudu byl řešen fitováním 2D polynomu v sekci 5.4. Závěrečný odhad počtu bází jednotlivých značek využívá buď jednoduché lineární interpolace, nebo hyperbolické funkce, které více odpovídá průběhu závislosti počtu bází na poloze značek v žebříčcích, je ale citlivější na chyby měření, viz 5.5.

Implementace byla vytvořena v jazyce Python, spolu s uživatelským rozhraním pro demonstraci výsledků. Rozhraní nevyžaduje, kromě několika ručních voleb ve formě přepínačů, žádné nastavování parametrů programu, jeho popis je v kapitole 7.

Cíle práce byly z většiny splněny. Byly otestovány metody detekce, které na zkoumané množině obrázků, za předpokladu nepříliš silného znehodnocení snímků, dovedly detekovat značky žebříčku i neznámé značky. Výsledky zpracování obrázků pak ve většině případů byly použitelné, to i z pohledu odhadu počtu bází. Použití uživatelské aplikace má význam zejména ve smyslu demonstrace výsledků a případných laických analýz. Problémem je zejména časová náročnost výpočtů fitování žebříčků, a v případě MOSSE filtru také provedení filtrace pomocí Fourierovy transformace. Pro automatické zpracování jednotlivých obrázků bez přísného požadavku na rychlost zpracování je pak aplikace použitelná, časová náročnost je diskutována v sekci 6.4.

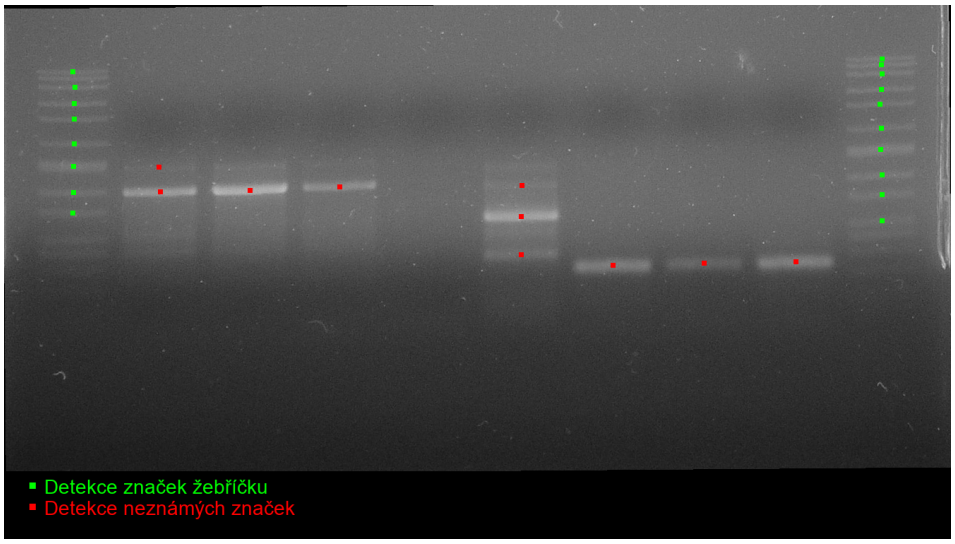
Při rozšiřování výsledků práce by bylo vhodné zaměřit se na efektivitu provedení detekce u MOSSE filtru, kdy je kritickým bodem výpočet Fourierovy transformace. Použitím specializovaných knihoven nebo převedením výpočtu na GPU by se dalo dosáhnout lepších výsledků z pohledu času. U MOSSE filtru by bylo rovněž vhodné upravit způsob zpracování výstupu korelace, možnosti úprav jsou navrženy v sekci 6.2, a rovněž navrhnout postup, jakým zprovoznit filtr pro různé velikosti obrázků. Problematickým algoritmem zpracování je rozdělení jednotlivých lanes se značkami na žebříčky. Použitá metoda je nedostatečně robustní vůči případným vyšším vstupním rotacím obrázků. Místo zkoumání projekcí značek do osy x by bylo vhodnější nalézt jednotlivé lanes prokládáním přímkami již v tomto kroku zpracování, nikoli až při odstraňování vstupní rotace v kroku následujícím. Dalším algoritmem, který by bylo vhodné vylepšit, je způsob odhadu počtu bází pomocí hyperbolické funkce, jehož problémy jsou popsány v sekci 5.5.1.

Příloha A

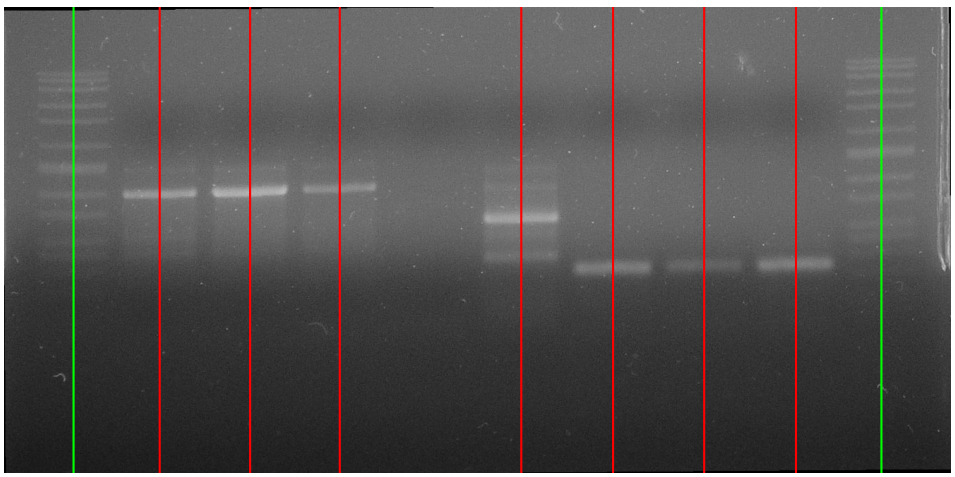
Ilustrace výsledků detekce značek a fitování žebříčků

Následující ilustrace výsledků sestávají z trojice obrázků. První obrázek ve trojici zobrazuje detekované značky a jejich rozdělení na žebříčky a neznámé značky. Druhý obrázek zobrazuje porovnání detekcí s anotací obrázku v případě, že anotace byla vytvořena, jinak obsahuje rozdělení značek do lanes. Třetí obrázek obsahuje fitované žebříčky, a neznámým značkám jsou přiřazeny odhadované počty bází. Nadpis sekce pak odpovídá názvu experimentu.

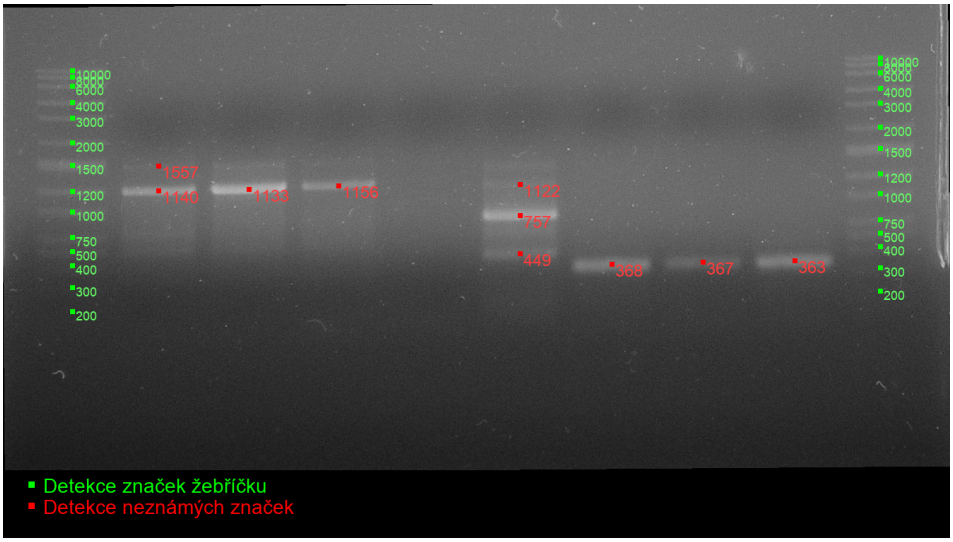
A.1 PCR-INTSPA-01-11-06



Obrázek A.1: Výsledky detekce - Př. 1

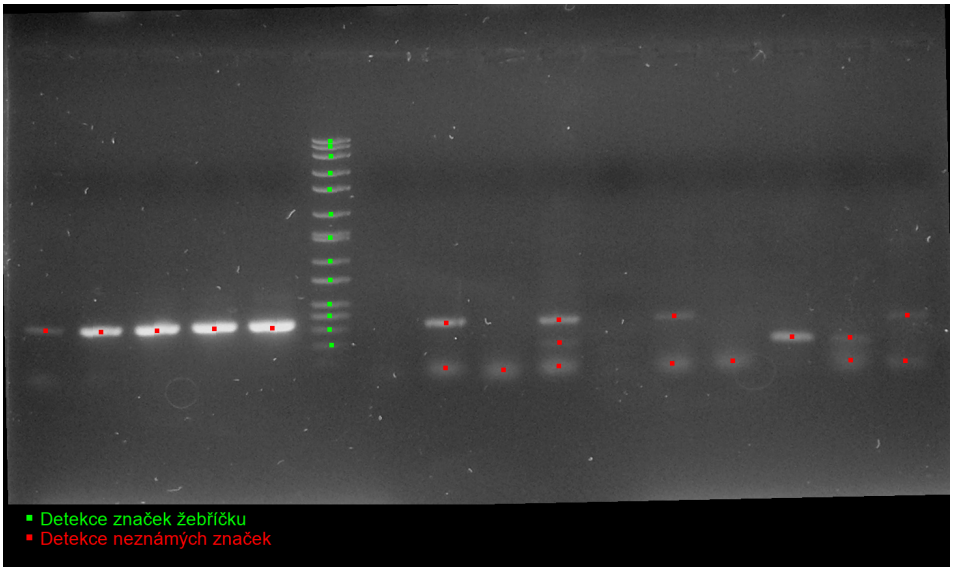


Obrázek A.2: Rozdělení do lanes - Př. 1

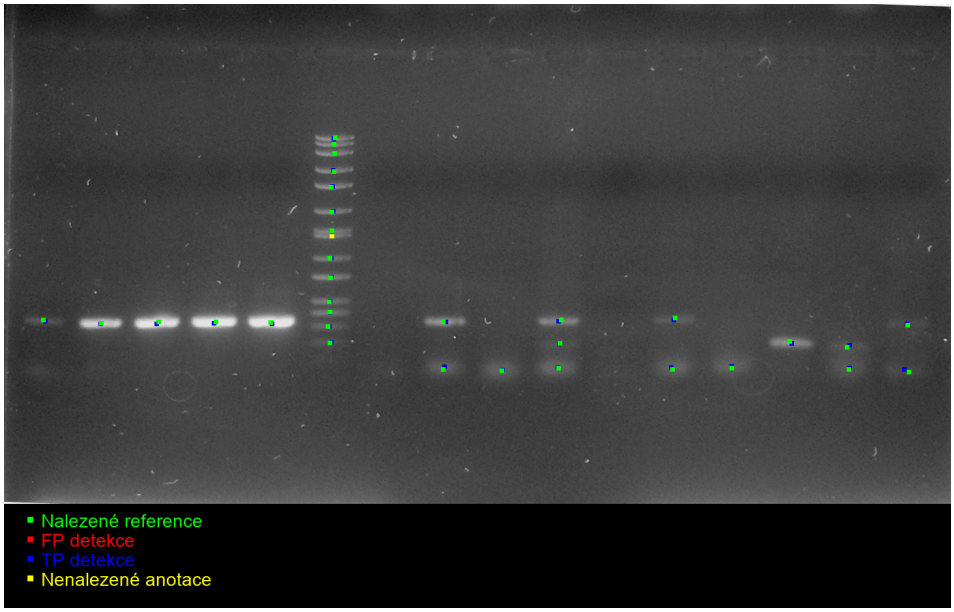


Obrázek A.3: Fitování žebříčku a odhady počtů bází - Př. 1

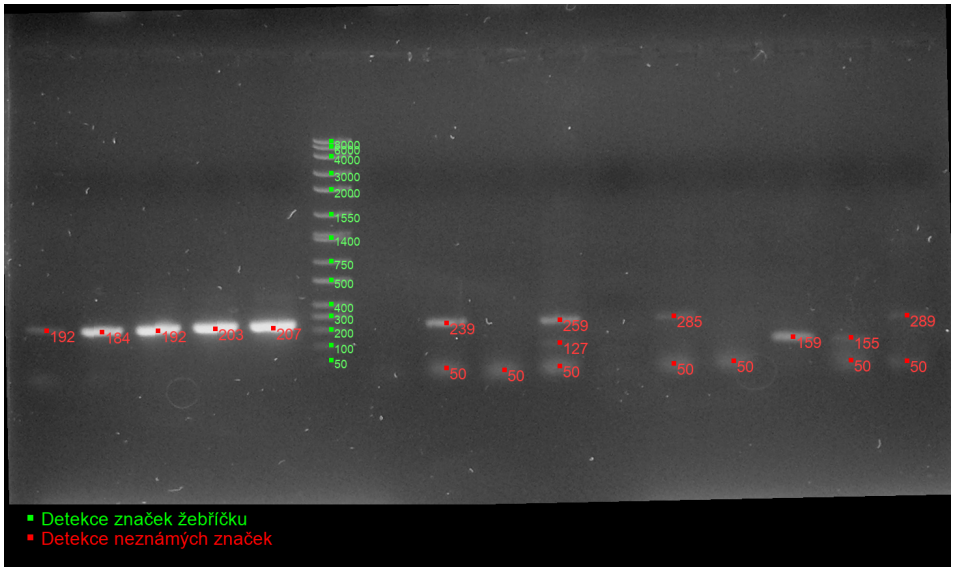
A.2 PCR-LD5-27-07-05



Obrázek A.4: Výsledky detekce - Př. 2

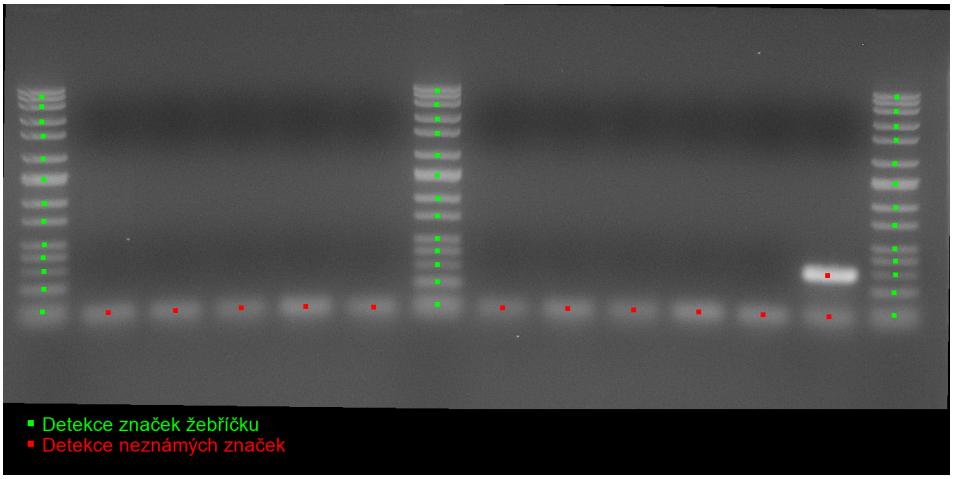


Obrázek A.5: Porovnání detekcí s anotacemi - Př. 2

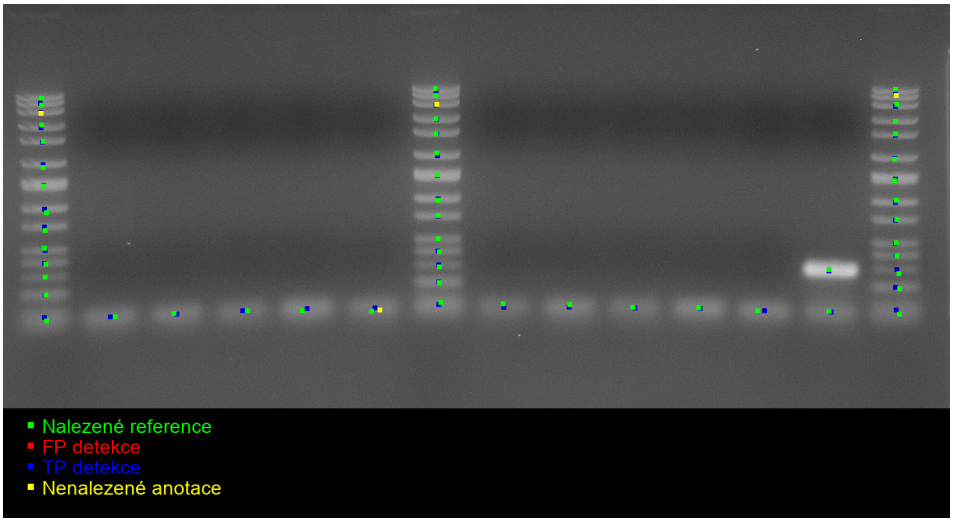


Obrázek A.6: Fitování žebříčku a odhady počtů bází - Př. 2

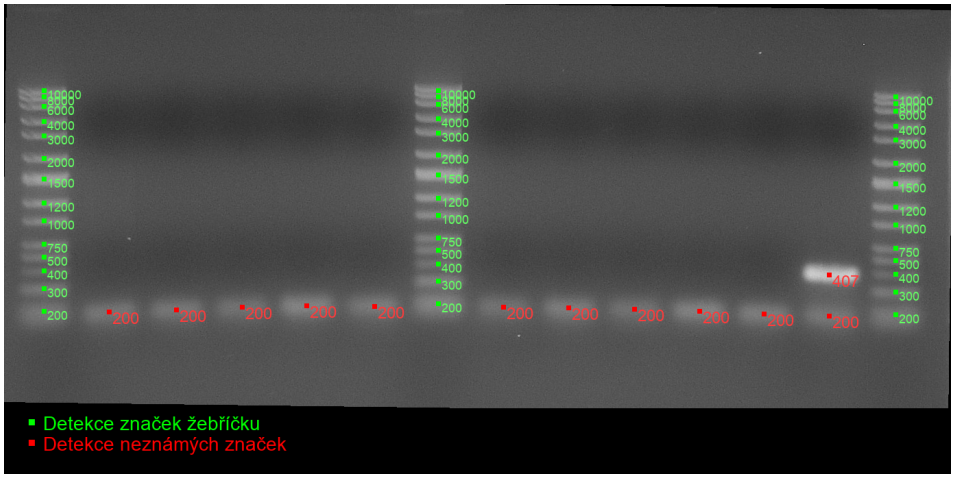
A.3 PCR-EHR709-KL-25-04-07



Obrázek A.7: Výsledky detekce - Př. 3



Obrázek A.8: Porovnání detekcí s anotacemi - Př. 3



Obrázek A.9: Fitování žebříčku a odhady počtů bází - Př. 3

Příloha B

Přílohy na CD

Přiložené CD a elektronická verze práce obsahují následující složky:

- /app - zdrojové kódy, s použitím interpreteru slouží ke spouštění aplikace
- /samples - vzorky testovacích obrázků a vzorový soubor pro generování ladder file
- /text - pdf soubor s bakalářskou prací

Příloha C

Návod k uživatelskému rozhraní

Aplikace předpokládá na vstupu obrázky gelové elektroforézy, kde pohyb značek probíhal ve vertikálním směru, a značky s nejvyšším počtem bází se nacházejí nejblíže hornímu okraji obrázku.

Aplikace se spouští pomocí souboru *application.py* z příkazové řádky. Je nutné zajistit, že se ve stejném adresáři nachází soubory popsané v 7.2 a jsou nainstalované dodatečné moduly (modul *gurobipy* pro práci s řešičem Gurobi, apod.). Jednotlivé *.py* soubory jsou umístěny na CD discích u výtisku bakalářské práce, a také v elektronické verzi na dspace, evenutálně na GitLab FEL na stránce:

- https://gitlab.fel.cvut.cz/grusjose/gel-electrophoresis-image-analysis

Součástí souborů je rovněž složka s testovacími obrázky a vzorovým souborem typu C.4a pro generování ladder file. Standardní způsob spuštění aplikace je příkazem:

- > python application.py

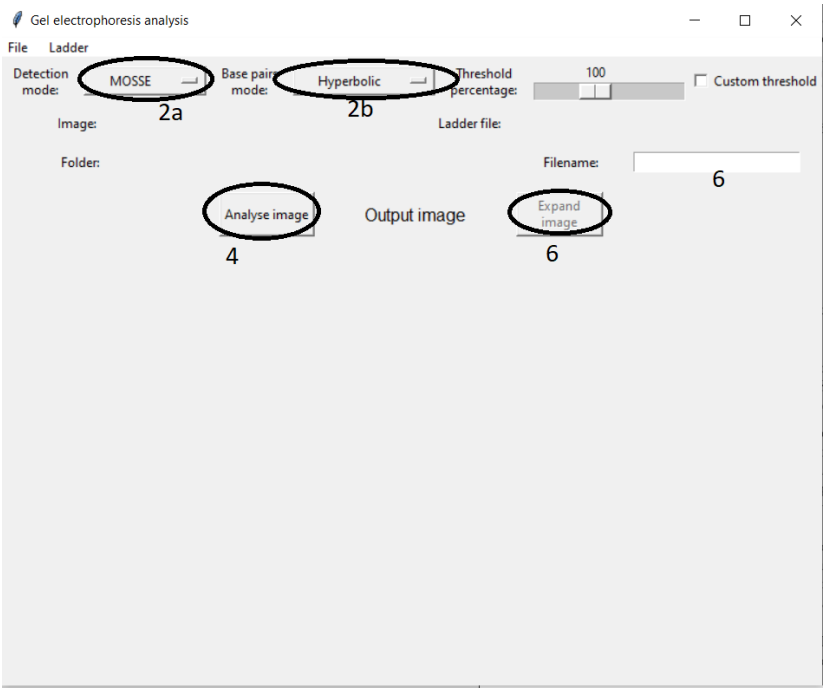
za předpokladu spouštění ze složky se soubory. Pomocí přepínače *-g* lze spouštět aplikace bez nutnosti importu Gurobi solveru, místo toho je použita implementace LP solveru v knihovně *scipy*. Důsledkem je prodloužení

doby výpočtu optimalizačních úloh. Rovněž lze eliminovat nutnost instalace knihovny *OpenCV* použitím přepínače *-c*. Výsledkem je vypnutí možností použití kaskádního detektoru. Použitím přepínače *-d* lze rovněž zapnout DEBUG-MODE, při kterém je status aplikace a případné chybové hlášky průběžně vypisován do konzole. Volání při použití všech přepínačů je následující:

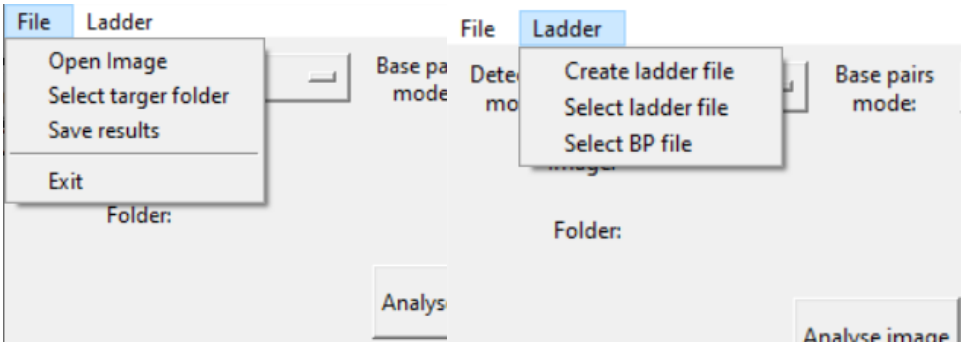
- > python application.py -g -c -d

Po spuštění programu je návod k provedení analýzy vybraného obrázku následující:

1. Vyberte obrázek pro analýzu pomocí volby Open image v menu File. Zkontrolujte obrázek v okně, v případě špatné volby proveďte volbu znovu.
2. a. Vyberte způsob analýzy obrázku. Volba MOSSE využívá pro detekci značek MOSSE filtr, volba ML používá k detekci kaskádní klasifikátor s MB-LBP deskriptory a morfologickým postprocessingem.
   b. Vyberte způsob odhadu počtu bází. Volba Hyperbolic užívá k odhadu hyperbolickou funkcí, odhad jejíž parametrů je však citlivý na kvalitní zadání vstupního žebříčku. Volba Linear Intrp používá lineární interpolaci, kdy je však odhad zatížen chybou nelinearity závislosti počtů bází na poloze značky.
3. Vyberte *.dat* soubor obsahující nastavení žebříčku pomocí volby Select ladder file v nabídce Ladder, soubor musí být ve formátu .dat, viz obr. C.4b. Příslušný soubor je možné vytvořit v rámci programu. Alternativně lze pomocí tlačítka Select BP file vybrat soubor obsahující pouze počty bází referenčního žebříčku jako v obrázku C.4a. Při této volbě je však výsledek analýzy citlivější na nedokonalé rozložení značek v žebříčcích. Pokud není vybrán žádný soubor s nastavením žebříčku, je k analýze použito základní nastavení žebříčku, uložené interně v aplikaci.
4. Spusťte analýzu tlačítkem Analyse image. Po skončení analýzy se ve spodní části okna objeví náhled výsledků.
5. Vyberte cílový adresář pomocí volby Select target folder v nabídce File a nastavte jméno souborů s výsledky v poli Filename.
6. Výsledky analýzy uložte volbou Save results v nabídce File. Kromě obrázku, jehož náhled je ve spodní části okna, se rovněž uloží textový dokument s pozicemi značek a odhady počtu bází ve formátu dle obr. C.5. Případně lze uvnitř aplikace zobrazit celý obrázek pomocí tlačítka Expand image.



Obrázek C.1: Zobrazení tlačítek



(a) : Nabídka File **(b) :** Nabídka Ladder

Obrázek C.2: Horní menu aplikace

Pro správné fitování žebříčků je nutné znát strukturu referenčního žebříčku předem, resp. její typický tvar, aby bylo možné odfiltrovat případné falešně pozitivní detekce, a zároveň aby bylo možné chybějící značky dodefinovat. Z toho důvodu je při vytváření tohoto referenčního modelu vzat referenční obrázek ze zkoumaného datasetu, a ten je ručně operátorem anotován. Pro lepší anotaci je vhodné z obrázku vyříznout pouze část s žebříčkem, viz obrázek C.3. Vytvořený model, tvořený dvojicemi počet bází a poloha značky, se ukládají ve formátu .dat do složky s aplikací. Při generování modelu se přihlíží pouze k vertikální poloze značek, proto je vhodné použít obrázek nezatížený rotací. Pokud pak zkoumané obrázky obsahují stejný typ žebříčku,

je jednoduché vybrat správný typ žebříčku z uložených .dat ladder file souborů. Pak již není nutné vytvářet model pro stejný typ žebříčku.

Návod pro vytvoření nového souboru s daty žebříčku je následující:

1. V hlavním okně stiskněte volbu **Create ladder file** v nabídce Ladder. Objeví se nové okno.

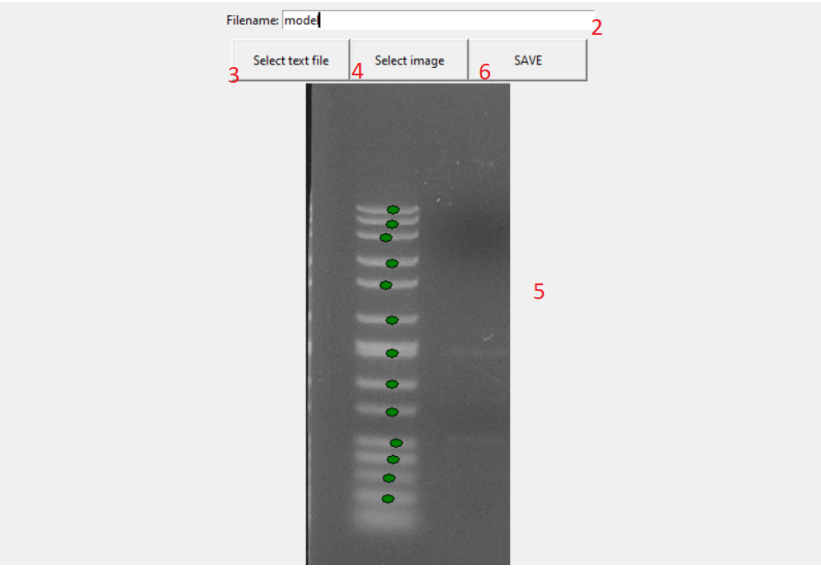
2. Nastavte jméno nového žebříčku.

3. Vyberte *.txt* soubor tlačítkem Select text file, obsahující na každém řádku počet bází značky v žebříčku, viz obr. C.4a. Počet řádků odpovídá počtu značek v žebříčku.

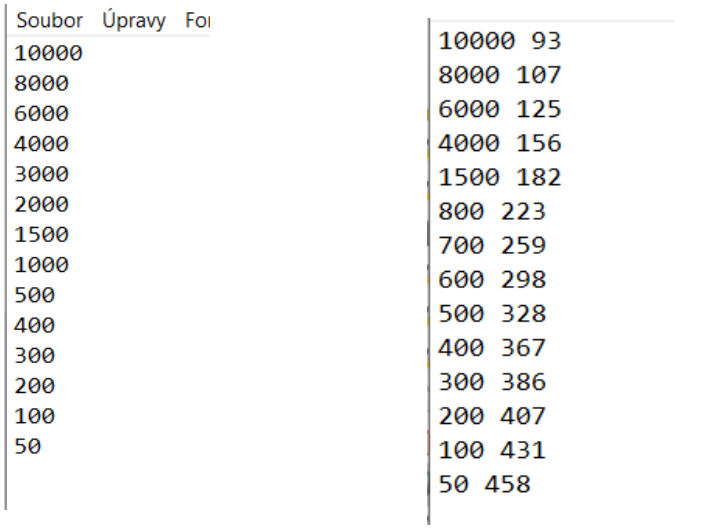
4. Vyberte obrázek obsahující daný typ žebříčku tlačítkem Select image. Po vybrání obrázku je provedena automatické detekce značek a nějaký nalezený žebříček je z obrázku vyříznut a nalezené značky jsou doplněny.

5. Myší doplňte chybějící značky anotace, případné chybně zvolené značky lze vyjmout kliknutím na ně. Po vybrání daného počtu značek, definovaného textovým souborem, se uvolní tlačítko SAVE. Přihlíží se pouze k vertikální poloze značek, není nutné zaměřovat jejich středy.

6. Uložte výsledky tlačítkem SAVE. Soubor se uloží do adresáře se souborem *application.py*. Formát uloženého souboru je na obrázku C.4b.



Obrázek C.3: Tlačítka nastavování nového žebříčku



(a) : Formát .txt souboru pro vytvoření žebříčku

(b) : Vygenerovaný ladder file

Obrázek C.4: Formáty textových souborů žebříčků

Výstupní soubor s daty je formátu CSV. Data jsou strukturovaná do tří sloupců. První sloupec kóduje lane nalezené značky zleva, druhý obsahuje index nalezené značky v lane, sestupně podle odhadnutého počtu bází. Poslední sloupec pak obsahuje samotný odhad počtu bází dané značky. Jako oddělovač je použita čárka ’,’.

	A	B	C
1	Line index	Band index	No. of Base pairs
2	1	1	1151
3	2	1	102
4	3	1	470
5	4	1	1234
6	4	2	484
7	4	3	115
8	5	1	128
9	6	1	1253
10	6	2	478
11	7	1	1225
12	7	2	211
13	8	1	326
14	9	1	295
15	9	2	119

Obrázek C.5: Formát výstupního souboru

Pokud nejsou výsledky automatické detekce uspokojivé, lze dosáhnout změny výsledků detekce manipulací s prahovací hodnotou jednotlivých algoritmů. K tomuto slouží posuvník v hlavním okně na obrázku C.1. Základní hodnota je 100 %. Nastavením jiné hodnoty se natrénovaná hodnota prahování změní na příslušnou procentuální část. Pro zapnutí změny prahování je nutno zaškrtnout pole Custom threshold. Pokud je provedena analýza stejným typem algoritmu po změně hodnoty prahování, není nutné vykonat úvodní kroky algoritmů (korelace u MOSSE, detekce u ML), což zejména ve variantě MOSSE výrazně zkrátí dobu potřebnou pro zpracování obrázku.

Příloha D

Bibliografie

1. KHAN ACADEMY. *Gel electrophoresis (article)* [online] [cit. 2020-05-01]. Dostupné z: `https://www.khanacademy.org/science/biology/biotech-dna-technology/dna-sequencing-pcr-electrophoresis/a/gel-electrophoresis`.
2. KUSIM, A.S.; ABDULLAH, Noor; RAHIM, A.A. An image processing enhancement approach to extract information in gel electrophoresis image. In: *2013 International Conference on Technology, Informatics, Management, Engineering and Environment* [online]. 2013, s. 9–16 [cit. 2020-05-01]. ISBN 978-1-4673-5730-2. Dostupné z DOI: `10.1109/TIME-E.2013.6611955`.
3. TSENG, Din-Chang; LEE, You-Ching. Automatic band detection on pulsed-field gel electrophoresis images. *Pattern Analysis and Applications* [online]. 2015, roč. 18, č. 1, s. 145–155 [cit. 2020-05-01]. ISSN 1433-755X. Dostupné z DOI: `10.1007/s10044-014-0424-4`.
4. KAŇKA, Jiří. *Automatické vyhodnocování obrazů z gelové elektroforézy.* Praha, 2008. Bakalářská práce. České vysoké učení technické v Praze, Fakulta elektrotechnická, Katedra kybernetiky.
5. BOLME, D. S.; DRAPER, B. A.; BEVERIDGE, J. R. Average of Synthetic Exact Filters. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition* [online]. 2009, s. 2105–2112 [cit. 2020-05-01]. ISSN 1063-6919. Dostupné z DOI: `10.1109/CVPR.2009.5206701`.
6. BOLME, D. S.; LUI, Y. M.; DRAPER, B. A.; BEVERIDGE, J. R. Simple real-time human detection using a single correlation filter. In: *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance* [online]. 2009, s. 1–8 [cit. 2020-05-01]. ISBN

978-1-4244-5504-1. Dostupné z DOI: 10.1109/PETS-WINTER.2009.5399555.

7. BOLME, D. S.; BEVERIDGE, J. R.; DRAPER, B. A.; LUI, Y. M. Visual object tracking using adaptive correlation filters. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* [online]. 2010, s. 2544–2550 [cit. 2020-05-01]. ISSN 1063-6919. Dostupné z DOI: 10.1109/CVPR.2010.5539960.

8. RUSS, John C. *The image processing handbook*. 6. vydání. Boca Raton: CRC Press, 2010. ISBN: 978-1-4398-4045-0.

9. ROSEBROCK, Adrian. *Intersection over Union (IoU) for object detection* [online]. 2020 [cit. 2020-05-01]. Dostupné z: https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/.

10. VIOLA, Paul; JONES, Michael J. Robust Real-Time Face Detection. *International Journal of Computer Vision* [online]. 2004, roč. 57, č. 2, s. 137–154 [cit. 2020-05-01]. ISSN 1573-1405. Dostupné z DOI: 10.1023/B:VISI.0000013087.49260.fb.

11. OPENCV DEV TEAM. *Cascade Classification* [online] [cit. 2020-05-01]. Dostupné z: https://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html.

12. AHONEN, T.; HADID, A.; PIETIKAINEN, M. Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* [online]. 2006, roč. 28, č. 12, s. 2037–2041 [cit. 2020-05-01]. ISSN 1939-3539. Dostupné z DOI: 10.1109/TPAMI.2006.244.

13. ZHANG, Lun; CHU, Rufeng; XIANG, Shiming; LIAO, Shengcai; LI, Stan Z. Face Detection Based on Multi-Block LBP Representation. In: LEE, Seong-Whan; LI, Stan Z. (ed.). *Advances in Biometrics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, s. 11–18. ISBN 978-3-540-74549-5.

14. OPENCV DEV TEAM. *Cascade Classifier Training* [online] [cit. 2020-05-01]. Dostupné z: https://docs.opencv.org/master/dc/d88/tutorial_traincascade.html.

15. ŠONKA, Milan; HLAVÁČ, Václav; BOYLE, Roger. *Image processing, analysis, and machine vision*. 4. vydání. Austrálie: Cengage Learning, 2015. ISBN: 978-1-133-59369-0.

16. JONES, Eric; OLIPHANT, Travis; PETERSON, Pearu et al. *SciPy: Open source scientific tools for Python*. 2001–. Dostupné také z: http://www.scipy.org/.

17. WERNER, Tomáš. *Elektronická skripta předmětu B0B33OPT* [online] [cit. 2020-05-01]. Dostupné z: https://cw.fel.cvut.cz/b191/_media/courses/b0b33opt/opt.pdf.

18. FISCHLER, Martin A.; BOLLES, Robert C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* [online]. 1981, roč. 24, č. 6, s. 381–395 [cit. 2020-05-01]. ISSN 0001-0782. Dostupné z DOI: 10.1145/358669.358692.

19. CHUM, Ondřej; MATAS, Jiří; KITTLER, Josef. Locally Optimized RANSAC. In: MICHAELIS, Bernd; KRELL, Gerald (ed.). *Pattern Recognition* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, s. 236–243 [cit. 2020-05-01]. ISBN 978-3-540-45243-0. Dostupné z: https://link.springer.com/chapter/10.1007/978-3-540-45243-0_31.

20. GUROBI OPTIMIZATION, LLC. *Gurobi Optimizer Reference Manual* [online]. 2020 [cit. 2020-05-01]. Dostupné z: http://www.gurobi.com.

21. MUNKRES, James. Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics* [online]. 1957, roč. 5, č. 1, s. 32–38 [cit. 2020-05-01]. Dostupné z DOI: 10.1137/0105003.

22. IAMPANG, A.; BOONJING, V.; CHANVARASUTH, P. A cost and space efficient method for unbalanced assignment problems. In: *2010 IEEE International Conference on Industrial Engineering and Engineering Management* [online]. 2010, s. 985–988 [cit. 2020-05-09]. ISSN 2157-3611. Dostupné z DOI: 10.1109/IEEM.2010.5674228.

23. RAGHAVA, Gajendra. DNAOPT: a computer program to aid optimization of DNA gel electrophoresis and SDS-PAGE. *BioTechniques* [online]. 1995, roč. 18, s. 274–8, 280 [cit. 2020-05-01]. ISSN 0736-6205. Dostupné z: https://www.researchgate.net/publication/15472318_DNAOPT_a_computer_program_to_aid_optimization_of_DNA_gel_electrophoresis_and_SDS-PAGE.

24. GRUS, Josef. *Gelová elektroforéza* [online]. Leden 2020 [cit. 2020-05-07]. Zpráva o práci na bakalářském projektu.

25. DUNHAM, Ethan. *Most Popular Fonts: Font Squirrel* [online] [cit. 2020-05-11]. Dostupné z: https://www.fontsquirrel.com/fonts/list/popular.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Grus** Jméno: **Josef** Osobní číslo: **474526**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra kybernetiky**

Studijní program: **Kybernetika a robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Analýza obrazu gelové elektroforézy

Název bakalářské práce anglicky:

Gel Electrophoresis Image Analysis

Pokyny pro vypracování:

Předmětem práce je zpracování obrázků z gelové elektroforézy.
Úkolem je odstranění šumu, detekce značek odpovídajících jednotlivým součástem směsi, automatické odečítání jejich poloh, a automatická kompenzace geometrického zkreslení vzniklého nehomogenitou proudu.
Výsledkem bude jednoduše ovladatelný samostatný program.

Seznam doporučené literatury:

[1] Russ, John C. – The image processing handbook – Londýn, 2010
[2] Šonka, Milan; Hlaváč, Václav – Image Processing, Analysis, and Machine Vision – USA, 2015
[3] Bolme, David S. – Visual object tracking using adaptive correlation filters – USA, 2010

Jméno a pracoviště vedoucí(ho) bakalářské práce:

prof. Dr. Ing. Jan Kybic, algoritmy pro biomedicínské zobrazování FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **09.01.2020** Termín odevzdání bakalářské práce: **22.05.2020**

Platnost zadání bakalářské práce: **30.09.2021**

prof. Dr. Ing. Jan Kybic	doc. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta