VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

ZPRACOVÁNÍ OBRAZU A AUTOMATICKÉ ŘEŠENÍ KŘÍŽOVEK
IMAGE PROCESSING AND AUTOMATIC SOLVING OF CROSSWORDS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Martin Kobath**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Ilona Janáková, Ph.D.**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření**
Ústav automatizace a měřicí techniky

Student: Bc. Martin Kobath ***ID:*** 158635
Ročník: 2 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Zpracování obrazu a automatické řešení křížovek

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je detekovat na snímcích mřížku a rozpoznat legendu švédských křížovek a napomoci luštiteli v jejich řešení s pomocí křížovkářského slovníku.

1. Proveďte rešerši dané problematiky.
2. Navrhněte robustní algoritmy detekce mřížky a legendy křížovek a výsledný algoritmus ověřte na dostatečně široké a pestré databázi snímků.
3. Aplikujte algoritmy automatického čtení textu (OCR) na legendu.
4. S pomocí importovaného či online křížovkářského slovníku vyberte k nalezeným legendám řešení.
5. Vytvořte uživatelskou aplikaci, která bude vedle pořízení snímku a prezentování všech známých vhodných řešení ze slovníku umožňovat i například nápovědu pouze pro konkrétní část křížovky.
6. Zhodnoťte výsledky všech jednotlivých kroků.

DOPORUČENÁ LITERATURA:

Hlaváč V., Šonka M.: Počítačové vidění, Grada, Praha, 1992, ISBN 80-85424-67-3.

Dorson, R., Golub, E. a Jacobs, D. CrosScan: The Crossword Scanning App [online]. [cit. 2016-09-08]. Dostupné z: http://honors.cs.umd.edu/uploads/thesis/file/178/Thesis.pdf

Termín zadání: 6.2.2017 ***Termín odevzdání:*** 15.5.2017

Vedoucí práce: Ing. Ilona Janáková, Ph.D.
Konzultant:

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

Abstrakt

Diplomová práce je zaměřena na vývoj aplikace pro Android, která je schopna rozpoznat na snímku mřížku a text legendy švédské křížovky a najít její řešení v křížovkářském slovníku. V práci je navržen postup segmentace políček hledáním významných bodů, text je rozpoznán pomocí OCR Tesseract a řešení je hledáno v lokální databázi. Vytvořená aplikace je testována na galerii snímků pořízených mobilním telefonem. Segmentace políček a hledání řešení podává dobré výsledky, zbytek zpracování podává neuspokojivé výsledky kvůli nepřesnému výstupu OCR.

Klíčová slova

Počítačové vidění, švédská křížovka, podobnost slov, Android aplikace.

Abstract

The Master's thesis is focused on development of Android application which is able to recognize grid and clue text of swedish crossword and find a solution for it in crossword dictionary. The thesis proposes a process of tile segmentation using corner detection, recognizes text using Tesseract OCR and searches for solutions in local database. Developed application is tested on a gallery of photos captured using a mobile phone. The tile segmentation and solution searching provide good results, the rest of the process provides unsatisfactory results thanks to inaccurate OCR output.

Keywords

Computer vision, swedish crossword, word similarity, Android application.

Bibliografická citace:

KOBATH, M. *Zpracování obrazu a automatické řešení křížovek.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2017. 67 s. Vedoucí diplomové práce Ing. Ilona Janáková, Ph.D.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Zpracování obrazu a automatické řešení křížovek jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.
Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **15. května 2017** ……
podpis autora

Poděkování

Děkuji vedoucí diplomové práce Ing. Iloně Janákové, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **15. května 2017** ……

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

1 ÚVOD

Tato práce se zabývá vývojem aplikace pro Android, která bude schopna rozpoznat švédskou křížovku na snímku pořízeném mobilním telefonem, včetně textu její legendy, a dokáže uživateli aplikace poskytnout řešení k vybrané legendě za pomocí slovníku. V práci je vysvětlena teorie nezbytná k řešení této úlohy, jsou zde diskutovány možné přístupy k jejímu řešení a také vysvětlen a zhodnocen zvolený způsob řešení.

Velká část této úlohy spadá do problematiky počítačového vidění. V teoretickém úvodu jsou vysvětleny metody, které byly použity k úpravě vstupního snímku, určení polohy jednotlivých políček křížovky a jejich obsahu. Teoretický úvod se ale kromě metod počítačového vidění zabývá také způsobem určení podobnosti mezi dvěma textovými řetězci, z kterého vychází řešení vyhledávání ve slovníku.

Rozbor úlohy se zabývá vlastnostmi švédských křížovek jako je jejich vzhled a pravidla pro jejich legendy tak, jak jsou definována ve směrnicích vypracovaných Českým svazem hádankářů a křížovkářů. Je zde také uvedeno několik existujících programů a aplikací, které poskytují podobné funkce jako aplikace vyvíjená v rámci této práce. Nakonec jsou diskutována možná řešení úlohy s ohledem na vlastnosti švédských křížovek a snímků pořízených kamerou mobilního telefonu.

Následující kapitola popisuje zvolený způsob řešení. Pro rozpoznání křížovky a jejích políček je navržen postup s využitím metod vysvětlených v teoretickém úvodu. K rozpoznání textu je v aplikaci použit open source systém optického rozpoznávání znaků Tesseract. Poslední částí úlohy je hledání řešení legendy ve slovníku, které bylo řešeno hledáním řešení v databázi hesel nacházející se v paměti mobilního telefonu.

Poslední část práce je věnována vyhodnocení vytvořené aplikace s pomocí galerie 87 snímků švédských křížovek, které byly pořízeny kamerami mobilních telefonů s různým rozlišením a za různých podmínek.

2 TEORETICKÝ ÚVOD

Tato kapitola vysvětluje teorii, nezbytně nutnou k pochopení ostatních částí této práce. Vzhledem k náplni práce je zaměřena hlavně na oblast počítačového vidění a vysvětluje metody použité v této práci. Kromě toho také vysvětluje základní princip určování podobnosti mezi textovými řetězci.

2.1 Předzpracování

Předzpracování je souhrnný název pro operace nad nezpracovaným vstupním snímkem, které mají za účel potlačit nežádoucí prvky, jako je šum nebo zkreslení, a vyzdvihnout některé žádoucí prvky v obrazu a tím zlepšit účinnost dalších operací.

2.1.1 Zmenšení obrazu

Výpočetní náročnost zpracování obrazu závisí mimo jiné na jeho velikosti. V některých případech můžeme mít snímek, který má vysoké rozlišení, ale vzhledem k povaze úlohy pro nás toto vysoké rozlišení nemá význam. Proto můžeme uvažovat o zmenšení obrazu, abychom zrychlili následující zpracování.

V případě změny velikosti obrazu dochází k jeho převzorkování a pixely nového obrazu vznikají interpolací okolních pixelů původního obrazu (1 str. 87).

Interpolace nejbližším sousedem – přiřadí pixelu výstupního obrazu hodnotu nejbližšího pixelu z původního obrazu. Tato metoda je jednoduchá, a tedy i rychlá, ale může vytvářet nežádoucí zkreslení (1 str. 88). Zachovává však ostré hrany.

Bilineární interpolace – využívá čtyři nejbližší pixely k určení hodnoty pixelu ve výstupním obrazu. Dává lepší výsledky než interpolace nejbližším sousedem, ale nezachovává ostrost hran. Hodnota pixelu *f(x, y)* v bodě *(x, y)* je rovna (1 str. 88):

$$f(x, y) = a_1 x + a_2 y + a_3 xy + a_0 \tag{2.1}$$

Koeficienty rovnice lze získat ze známých hodnot čtyř nejbližších pixelů $f(x_1, y_1)$, $f(x_1, y_2)$, $f(x_2, y_1)$ a $f(x_2, y_2)$ řešením čtyř rovnic o čtyřech neznámých:

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1 y_1 \\ 1 & x_1 & y_2 & x_1 y_2 \\ 1 & x_2 & y_1 & x_2 y_1 \\ 1 & x_2 & y_2 & x_2 y_2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} f(x_1, y_1) \\ f(x_1, y_2) \\ f(x_2, y_1) \\ f(x_2, y_2) \end{bmatrix} \tag{2.2}$$

Bikubická interpolace – počítá hodnotu pixelu ve výstupním obrazu z hodnot 16 nejbližších pixelů. Obecně zachovává více detailů než bilineární interpolace. Hodnota pixelu *f(x, y)* v bodě *(x, y)* je rovna (1 str. 88):

$$f(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j \qquad (2.3)$$

Koeficienty rovnice lze získat ze známých hodnot 16 nejbližších pixelů řešením soustavy rovnic, stejně jako v případě bilineární interpolace (1 str. 88).

Je samozřejmě možné počítat hodnotu neznámého pixelu i z více než 16 nejbližších pixelů. Kromě metod zde uvedených existují i složitější metody, které využívají spline křivek nebo vlnkových funkcí, a mohou podat lepší výsledky než uvedené metody interpolace (1 str. 90).

2.1.2 Převod na šedotónový obraz

Barevné obrazy s sebou nesou výhodu použití informace o barvě při zpracování obrazu. Avšak v případě, kdy chceme pracovat pouze s jasem obrazu, můžeme informaci o barvě odstranit a tím ušetřit místo v paměti a zjednodušit výpočty. Způsob, jakým převést barevný obraz na šedotónový, závisí na použitém barevném modelu a barevném prostoru.

Barevný model – slouží k číselnému reprezentaci barev pomocí několika složek. Sám o sobě je pouze matematickým modelem, který nevyjadřuje žádnou skutečnou barvu (2).

Barevný prostor – je specifickou implementací barevného modelu. Popisuje, jak mají být interpretovány jednotlivé složky modelu (2).

Barevných modelů a prostorů existuje mnoho, zde jsou uvedeny jen některé z nich.

2.1.2.1 Barevný model RGB

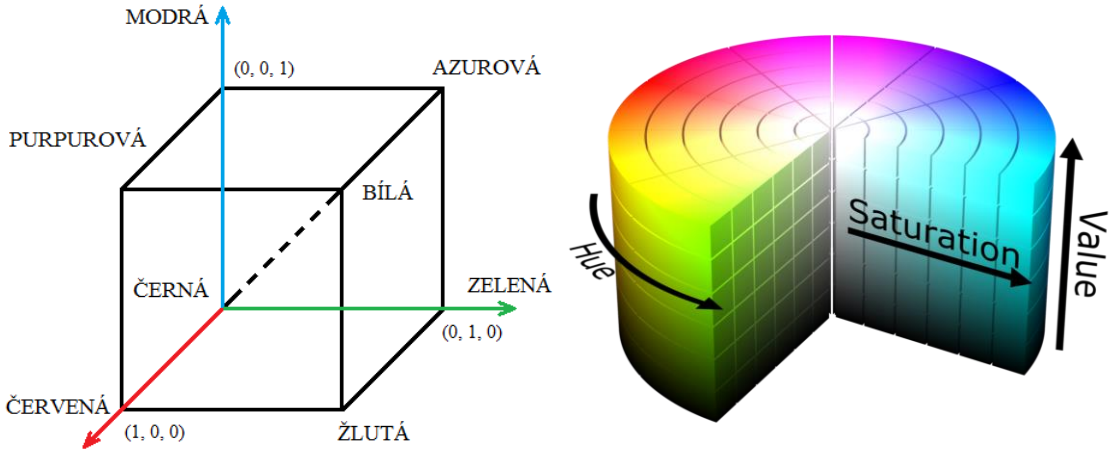
RGB model vytváří barvy sčítáním tří základních barev – červená, zelená a modrá. Je založen na pravoúhlém souřadnicovém systému a lze jej zobrazit jako jednotkovou kostku (viz Obrázek 2.1) (1 str. 424). Barevný model RGB je používán digitálními kamerami pro pořízení barevného obrazu a displeji pro zobrazení barevného obrazu. Barevné prostory, se kterými se lze často setkat jsou sRGB a Adobe RGB. Oba tyto prostory jsou používány při přenosu obrazů, přičemž sRGB je jedinou vhodnou volbou při nahrávání obrazů na webové stránky (2).

Opakem RGB modelu je CMYK model, který používá sekundární barvy RGB modelu (azurová, purpurová, žlutá) jako primární společně s barvou černou, a je využíván při tisku barev (1 str. 428).

2.1.2.2 Barevný model HSV

Zatímco barevný model RGB je vhodný pro implementaci do hardware, model HSV je vhodný pro popis barev způsobem, jakým to dělá člověk. Barevný model HSV popisuje barvy pomocí složek odstín barvy, sytost barvy a jas. Odstín barvy popisuje čistou barvu, sytost barvy popisuje, do jaké míry je tato čistá barva ředěna bílou barvou, a jas popisuje vnímanou intenzitu výsledné barvy (1 str. 429). Model je možné zobrazit jako prostor ve tvaru válce (viz Obrázek 2.1). Kromě HSV se často používá i podobný model HSL.

Jelikož je v HSV modelu jas samostatnou složkou, lze získat šedotónový obraz jednoduše odstraněním ostatních složek.



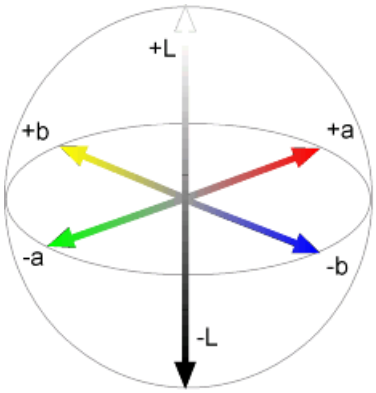
Obrázek 2.1: Barevný model RGB a HSV (3).

2.1.2.3 Barevný model CIELAB

Barevný model CIELAB, na rozdíl od modelů RGB, CMYK nebo HSV, definuje barvy nezávisle na použitém zařízení a je navržen tak, aby dobře napodoboval způsob, jakým člověk posuzuje barvu. Další vlastností modelu je percepční uniformita, díky které je v modelu mezi dvěma odlišnými barvami velká vzdálenost, a naopak mezi podobnými odstíny barev je vzdálenost krátká (4 str. 48).

Složkami modelu je světlost L* a složky a* a b* popisující dvojice opačných barev (zelená pro záporné a*, červená pro kladné a*, modrá pro záporné b* a žlutá pro kladné b*). Neutrální šedá barva se nachází v bodě a* = 0, b* = 0 (4 str. 48).

CIELAB má velký barevný prostor, který pokrývá všechny barvy vyjádřitelné menšími prostory jako je RGB. Je proto používán jako základní prostor pro převod barevných prostorů zařízení (4 str. 48).



Obrázek 2.2: Barevný model CIELAB (5).

2.1.2.4 Převod sRGB barevného obrazu na šedotónový

Máme-li obraz vyjádřený RGB modelem, je jeho jas rozložen na jas jednotlivých barevných složek. Šedotónový obraz se získá váženým součtem těchto složek, přičemž váhy se liší podle použitého barevného prostoru. Například pro jeden bod sRGB obrazu lze vypočítat jas Y použitím následujících vah (6):

$$Y = 0{,}213\ R + 0{,}715\ G + 0{,}072\ B \tag{2.4}$$

Kde R, G a B jsou jednotlivé složky bodu v sRGB obraze.

2.1.3 Potlačení šumu

Nedílnou součástí každého obrazu reálné scény je šum. Jedná se o nežádoucí informaci přidanou k užitečnému signálu. Výrazný šum negativně ovlivňuje práci s obrazem a je proto žádoucí šum potlačit. Šum v obraze vzniká během jeho pořízení a může vzniknout i během jeho přenosu. U obrazů pořízených digitální kamerou jsou zdrojem šumu např. podmínky prostředí, úroveň osvětlení, šum zesilovače nebo teplota snímače (1 str. 335). V případě barevných snímků může sum ovlivnit jednotlivé barevné složky různě silně. Existuje mnoho různých modelů šumu sloužících k popisu šumu z různých zdrojů. V této kapitole jsou popsány pouze významné druhy šumu, které se vyskytují v obrazech pořízených digitální kamerou.

2.1.3.1 Gaussovský šum

Zdrojem Gaussovského šumu v obraze je šum snímače, způsobený nedostatečným osvětlením nebo vysokou teplotou, a šum elektronického obvodu (1 str. 339). Obvykle postihuje celý obraz přičtením náhodné hodnoty z normálního rozložení ke každému pixelu. Gaussovský šum má nulovou střední hodnotu, určitý rozptyl a obvykle malou intenzitu (7 str. 36).

Při potlačení Gaussovského šumu se snažíme o snížení jeho rozptylu. K tomuto účelu se používají metody průměrování obrazu. Předpokládá se, že intenzita

blízkých bodů je velmi podobná až stejná a liší se pouze přičteným šumem, který má tendenci se sám kompenzovat díky nulové střední hodnotě (7 str. 38).

Průměrování lze provést konvolucí obrazu s konvoluční maskou, která má všechny prvky stejné a jejich součet je roven jedné:

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \tag{2.5}$$

Míru potlačení šumu je možné snížit poměrným zvětšením prostředního prvku masky vůči ostatním prvkům, nebo zvýšit zvětšením masky (7 str. 38). Jako konvoluční maska se často používá Gaussovský filtr.

Vedlejším efektem průměrování obrazu je rozmazání hran, jelikož neodpovídají předpokladu podobné intenzity blízkých bodů. Některé složitější metody vyhlazování jsou však schopné v obraze hrany zachovat (7 str. 38).

2.1.3.2 Impulzní šum

Impulzní šum bývá výsledkem chyb při přenosu signálu a selhání pixelů ve snímači obrazu (1 str. 339). V obrazu se projevuje tak, že pixely postižené tímto šumem jsou saturovány na maximální nebo nulovou intenzitu. Impulzní šum obvykle nepostihuje celý obraz, ale pouze některé pixely (7 str. 36).

Jelikož šum postihuje jen některé pixely v obraze, je možné tyto postižené pixely detekovat a nahradit je hodnotou některého z okolních nepostižených pixelů. Obvykle se však impulzní šum odstraňuje pomocí mediánového filtru. Filtr pro každý pixel obrazu vybere malou oblast pixelů okolo vyšetřovaného pixelu a seřadí je podle intenzity. Následně nahradí hodnotu vyšetřovaného pixelu mediánem z této řady. Protože mají pixely postižené impulzním šumem velmi vysoké nebo velmi nízké hodnoty intenzity, je nepravděpodobné, že by se staly mediánem a jsou proto nahrazeny hodnotou některého z nepostižených bodů (7 str. 41).

Impulzní šum je také možné potlačit průměrováním. Mediánový filtr však dává v případě impulzního šumu jednoznačně lepší výsledek a zachovává ostré hrany.

2.1.3.3 Šum typu moiré

Šum typu moiré vzniká v případě, kdy fotografovaná scéna nebo objekt obsahuje opakující se vzor, který je na použité rozlišení snímače příliš malý (8). Dochází zde k nedodržení Shannon-Nyquistova vzorkovacího teorému a vzniku aliasingu. Ten se na obraze pořízeném snímačem s Bayerovým filtrem projevuje vlnitými barevnými vzory (viz Obrázek 2.3).

Šum typu moiré je obtížné ze snímku odstranit a je lepší jeho vzniku předcházet již při pořizování snímku použitím snímače s vyšším rozlišením nebo anti-aliasing filtru. Na již pořízeném snímku je možné šum potlačit průměrováním a

některé profesionální softwarové nástroje pro práci s digitálními fotografiemi obsahují pokročilejší metody potlačení tohoto šumu (8).

Obrázek 2.3: Šum typu moiré (9).

2.1.3.4 Periodický šum

Periodický šum je typicky výsledkem elektrické nebo elektromechanické interference během pořizování snímku. Dojde ke zvýraznění některých spektrálních komponent, které se na obraze projeví proužkovým vzorem (1 str. 340).

Obrázek 2.4: Satelitní snímek postižený periodickým šumem (10).

Jelikož periodický šum postihuje pouze některé spektrální komponenty, je možné jej potlačit nulováním postižených spektrálních komponent použitím úzkopásmového filtru. Které komponenty byly zvýrazněny je možné zjistit z amplitudového spektra Fourierovy transformace obrazu, nejedná se však obecně o snadný úkol (7 str. 42).

2.1.3.5 Anizotropní a fixní šum

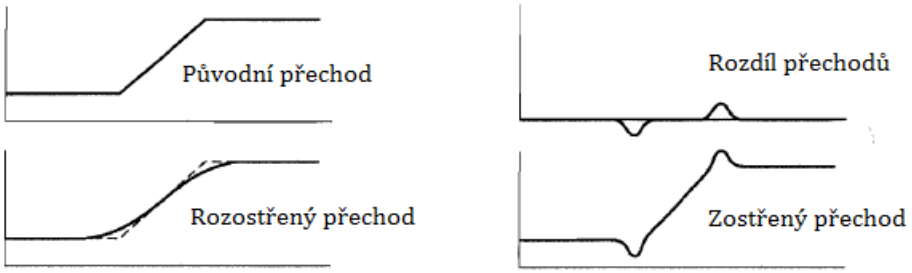
Anizotropní šum i fixní šum jsou silně závislé na použité kameře. Fixní šum vytváří na každém snímku stejný vzor a je patrný u snímků s dlouhou expozicí. Kvůli malým rozdílům v citlivosti jednotlivých pixelů snímače se různé pixely při stejném osvětlení jeví na pořízeném snímku jako jasnější nebo tmavší. Anizotropní šum se na snímku projevuje jako horizontální nebo vertikální pruhy, vzniká při čtení informace ze snímače a je zvýrazněn zvýšením citlivosti kamery (11).

Díky tomu, že fixní šum je mezi snímky neměnný, je možné ho v digitální kameře snadno kompenzovat. Oba typy šumu je možné potlačit průměrováním a některé profesionální programy poskytují specifické nástroje pro jejich odstranění.

2.1.4 Ostření obrazu

Účelem zostření obrazu je zvýraznění přechodů intenzity. Zostření zlepšuje vizuální vjem pozorovatele, ale také může zlepšit výsledky dalšího zpracování a analýzy obrazu.

Jedním z běžných způsobů zostření obrazu je přičtení rozdílu obrazu a jeho rozostřené kopie. Princip této metody ilustruje Obrázek 2.5. Rozostřením obrazu, například lokálním průměrováním pixelů, se sníží ostrost přechodů intenzity v obraze. Odečtením od původního obrazu získáme rozdíl tohoto snížení, který po přičtení k původnímu obrazu přechody naopak zvýrazní (1 str. 186).



Obrázek 2.5: Princip ostření obrazu pomocí rozostřené kopie obrazu (1 str. 186).

Tento postup můžeme vyjádřit za pomocí lokálních operátorů reprezentujících jednotlivé obrazy, jak je uvedeno v Rovnici 2.6. Matice X reprezentuje původní obraz a matice H je průměrující operátor zajišťující rozostření obrazu. Výsledný lokální operátor je možné konvolucí s obrazem použít přímo k ostření libovolného obrazu (7 str. 46).

$$X + (X - H) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right) = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad (2.6)$$

2.1.5 Detekce hran

Při detekci hran dochází k transformaci šedotónového obrazu na obraz binární (tzv. hranová reprezentace obrazu), ve kterém bílé pixely reprezentují polohu hran a černé pixely místa, kde se hrany nevyskytují (7 str. 54). Hranou rozumíme dostatečně rychlou změnu intenzity v obraze. Čím je změna rychlejší, tím je hrana zřetelnější. Jedná se tedy například o hranici mezi objektem a pozadím (7 str. 45).

Hranovou reprezentaci obrazu rozlišujeme na výslednou a hrubou. Výsledná se od hrubé odlišuje tím, že neobsahuje falešné hrany a pravé hrany jsou nepřerušené a mají tloušťku jednoho pixelu (7 str. 55). Z hranové reprezentace je možné zjistit hranice mezi objekty a slouží jako základ pro některé metody segmentace obrazu.

Detekce hran je negativně ovlivněna šumem v obraze, který způsobuje vznik falešných hran v jinak plochých částech obrazu. Z toho důvodu zahrnují pokročilé metody detekce hran, jako je např. Cannyho detektor, operace potlačující šum v obraze spolu se sledováním hran a potlačováním nevýrazných hran (7 str. 59). Tato podkapitola je zaměřena na vysvětlení základního principu detekce hran a nepopisuje žádné z pokročilých metod.

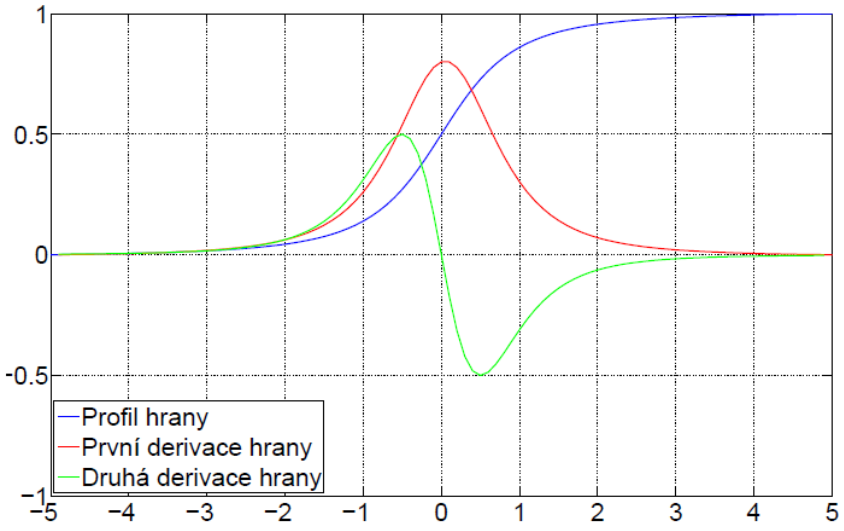
2.1.5.1 Lokální operátory aproximující derivaci hrany

Chceme-li nalézt polohu hrany v obraze, potřebujeme v obraze nalézt místa s rychlou změnou intenzity. K tomuto účelu lze využít derivací a hledat lokální extrémy v případě první derivace nebo průchody nulou v případě druhé derivace (viz Obrázek 2.6). Protože pracujeme s digitálním obrazem, musíme derivaci aproximovat pomocí diference, které lze realizovat pomocí lokálních operátorů.

Lokálních operátorů existuje velké množství variant a jejich společným znakem je nulový součet prvků masky. Jako příklad uvedeme operátor Prewittové pro aproximaci první derivace (7 str. 55):

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \qquad h_2 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.7}$$

Masky jsou směrové, detekují tedy hrany jen v určitém směru a v ostatním mají odezvu slabou nebo žádnou. Uvedená maska h_1 slouží k detekci horizontálních hran a maska h_2 naopak k detekci vertikálních hran. Otáčením masky lze získat detektor i pro šikmé hrany. Chceme-li detekovat hrany ve všech směrech, musíme provést konvoluci s jednotlivými směrovými maskami a výsledné obrazy složit dohromady do hrubé hranové reprezentace původního obrazu (7 str. 55).



Obrázek 2.6: Průběh první a druhé derivace hrany (7 str. 54).

Pro aproximaci druhé derivace se používají především operátor Laplacián (Rovnice 2.8) a kombinace Laplaciánu s Gaussovským filtrem tzv. Laplacián Gaussianu (7 str. 57).

$$L = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.8}$$

Pouhý operátor však nestačí pro určení správné polohy hrany. Ta se totiž nachází v místě, kde druhá derivace prochází nulou. Ve výsledku po konvoluci s maskou je proto nutné hledat průchody nulou s pomocí nelineárního operátoru (7 str. 57):

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & x & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.9}$$

Pixel x bude ve výstupním binárním obraze nastaven na logickou jedničku v případě, když jsou splněny všechny následující podmínky (7 str. 57):

- Alespoň jeden ze sousedů má hodnotu jiného znaménka než ostatní.
- Rozdíl extrémních hodnot sousedů převyšuje definovaný práh.
- Hodnota centrálního pixelu x leží mezi extrémními hodnotami.

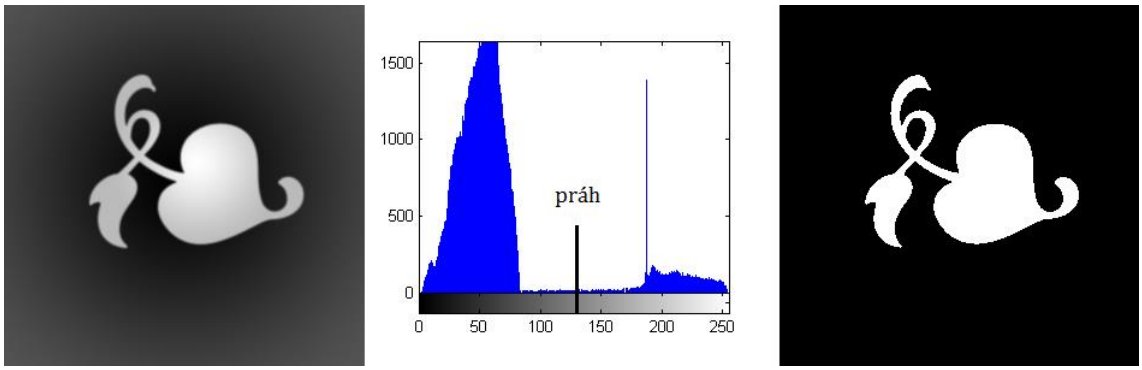
2.2 Segmentace

Při segmentaci dochází k rozdělení obrazu na nepřekrývající se oblasti, které mají v nějakém smyslu souvislost s věcným obsahem obrazu (7 str. 76). Segmentace obrazu je předpokladem pro to, abychom mohli v obraze rozpoznávat objekty a jejich vlastnosti.

2.2.1 Prahování

Prahování obrazu spočívá v rozdělení obrazu na oblasti porovnáním hodnot jednotlivých pixelů s definovaným prahem. Prahů může být více než jeden a prahování může být pouze částečné, kdy jsou pixely pod nebo nad prahem ponechány beze změny. Prahování se nejčastěji provádí podle jednoho parametru (např. jas v šedotónovém obraze), může ale využít i více parametrů (7 str. 76).

Základním problémem je tedy určení prahů, tedy nepřekrývajících se intervalů hodnot pro jednotlivé třídy, které odpovídají výsledným oblastem. V jednodušších případech, jako je například obraz světlého objektu na tmavém pozadí, je možné určit prahy za pomocí histogramu obrazu. V histogramu je možné rozpoznat dvě skupiny pixelů a vložit práh na pozici lokálního minima mezi těmito dvěma skupinami (viz Obrázek 2.7). V případě, kdy není určení polohy prahů jednoznačné, můžeme použít některé z metod výpočtu optimálního prahu jako je metoda Otsu, Kapur a jiné (7 str. 77). Pevně stanovený práh jasu pixelů však není schopen obraz správně segmentovat, pokud je nerovnoměrně osvětlen, a tedy objekty na obraze mají proměnou hodnotu jasu. Pokud je osvětlení neměnné, máme možnost ho kompenzovat. Pokud však nemáme nad osvětlením kontrolu můžeme tento problém obejít použitím adaptivního prahování.



Obrázek 2.7: Určení prahu podle histogramu obrazu.

Adaptivní prahování zahrnuje metody s proměnným prahem. Proměnný práh poskytuje oproti pevnému prahu schopnost sledovat plynulé změny jasu v obraze a vyhnout se tak chybám způsobených nerovnoměrným osvětlením a šumem (1 str.

778). Ve zbytku této kapitoly budou popsány metody výpočtu lokální hodnoty prahu.

2.2.1.1 Dělení obrazu

Nejjednodušší způsob určení proměnného prahu spočívá v rozdělení obrazu do nepřekrývajících se obdélníkových oblastí. Velikost oblastí je zvolena tak, aby jas v každém z nich byl přibližně uniformní. Pro každou oblast je poté určena hodnota prahu, která je použita při prahování pixelů uvnitř těchto oblastí (1 str. 778).

Tato metoda funguje dobře v případě, kdy lze obraz rozdělit na oblasti, ve kterých se vyskytují objekty i pozadí. Pokud se v oblasti vyskytuje jen objekt nebo jen pozadí, metoda obvykle selže. V těchto případech je lepší použít jednu z ostatních metod adaptivního prahování (1 str. 779).

2.2.1.2 Prahování na základě lokálních parametrů obrazu

Obecnější přístup, který počítá práh pro každý pixel obrazu na základě lokálních parametrů získaných z okolí vyšetřovaného pixelu. Vhodnými parametry k tomuto účelu jsou směrodatná odchylka a průměr, které popisují lokální kontrast a průměrný jas (1 str. 780).

Vezmeme-li pixel na pozici *(x, y)*, můžeme z lokální směrodatné odchylky σ_{xy} a lokálního průměru m_{xy} určit práh T_{xy} pro tento pixel podle následujícího vztahu (1 str. 780):

$$T_{xy} = a\sigma_{xy} + bm_{xy} \tag{2.10}$$

kde a a b jsou nezáporné konstanty. Pokud je pozadí obrazu přibližně konstantní, můžeme výsledek zlepšit tím, že lokální průměr m_{xy} nahradíme globálním průměrem vypočteným ze všech pixelů obrazu (1 str. 780).

K určení prahu je také možné použít predikátu a nastavovat hodnotu pixelu podle jeho pravdivosti, například (1 str. 781):

$$Q(\sigma_{xy}, m_{xy}) = \begin{cases} pravda & pokud\ f(x,y) > a\sigma_{xy} \cap f(x,y) > bm_{xy} \\ nepravda & jinak \end{cases} \tag{2.11}$$

Tímto způsobem je možné porovnat hodnotu pixelu s oběma parametry samostatně nebo nastavit jiný složitější způsob rozhodování.

2.2.2 Narůstání oblastí

Narůstání oblastí seskupuje pixely nebo oblasti do větších oblastí na základě předdefinovaných pravidel pro růst. Základní postup je začít od jednoho nebo více počátečních bodů (tzv. semínek) a vytvářet z nich oblast připojováním sousedních pixelů s podobnými parametry jako má semínko (7 str. 80).

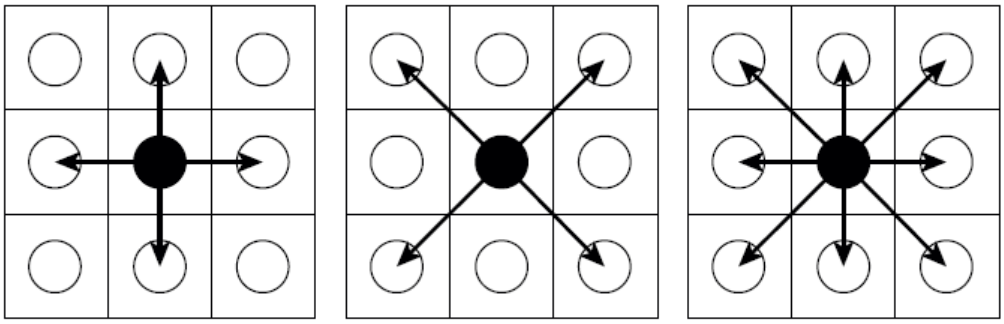
Počáteční pozici semínka můžeme zvolit ručně, na základě apriorní informace nebo můžeme vypočítat pro každý pixel parametry, podle kterých se bude růst oblastí řídit a pokud nalezneme shluky podobných hodnot, umístíme semínka do středu těchto shluků (1 str. 785).

Výběr vhodných parametrů závisí na úloze a dostupných obrazových datech. Kritérium pro přidání pixelu může být statické nebo dynamické. Statické kritérium porovnává hodnotu parametru p_j vyšetřovaného pixelu s hodnotou semínka p_s (7 str. 81):

$$\left|p_s - p_j\right| \leq T \tag{2.12}$$

V případě dynamického kritéria je parametr semínka p_s nahrazen parametrem naposledy přidaného pixelu nebo statistikou vypočítanou z pixelů, které se právě v oblasti nacházejí. Dynamicky proměnné kritérium umožňuje nárůst oblastí, jejichž pixely mají hodnotu zvoleného parametru pomalu proměnnou (7 str. 82).

Při rekurzivním narůstání oblastí se vyšetřují pixely sousedící s okrajovými pixely rostoucí oblasti. Tedy nejprve se vyšetřuje připojení pixelů sousedících se semínkem, poté se stejná operace opakuje pro nově připojené pixely a jejich sousedy atd. Sousedící pixely jsou obvykle vyšetřovány pro čtyřokolí nebo osmiokolí (viz Obrázek 2.8).



Obrázek 2.8: Okolí bodu: (a) horizontální a vertikální čtyřokolí, (b) diagonální čtyřokolí, (c) osmiokolí (7 str. 81).

Narůstání oblasti může být záplavové, kdy jsou pixely připojovány do oblasti na základě zvoleného kritéria nebo hraniční, kdy k sobě oblast připojuje všechny sousední pixely, dokud nenarazí na hranici pixelů určitých parametrů. Rozhodovací kritéria mohou být i složitější a brát v úvahu například rozměr a tvar rostoucí oblasti (1 str. 786).

2.2.3 Houghova transformace

Houghova transformace umožňuje hledat v binárním obraze tvary popsatelné rovnicí (přímky, kružnice apod.). Pro danou rovnici křivky *f(p) = 0* je hledán vektor parametrů *p* takový, aby křivka optimálně procházela pixely obrazu. Nalezené tvary je možné použít jako hranice objektů pro segmentaci obrazu. Princip Houghovy transformace je vysvětlen na příkladu hledání přímek (7 str. 86).

Pro ilustraci hledání přímek použijeme směrnicový tvar rovnice přímky (Rovnice 2.13). V praxi se však obvykle využívá normálový tvar přímky (Rovnice 2.14), protože směrnicový tvar nedokáže popsat přímku kolmou na osu x. Princip je však stejný.

$$y = kx + q \tag{2.13}$$

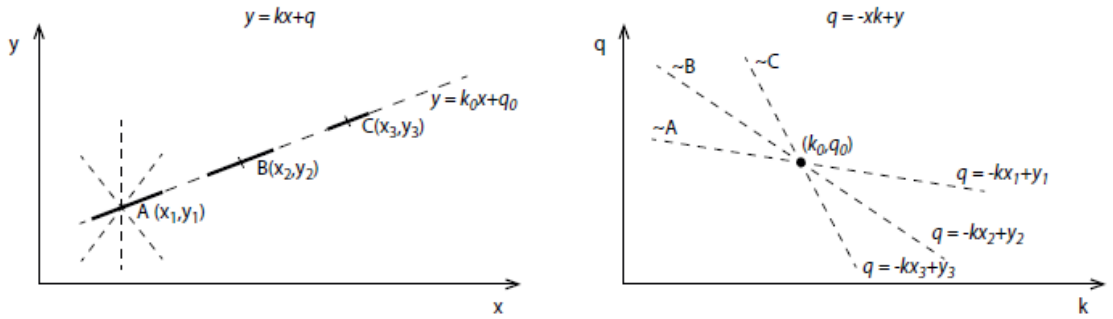
$$x\cos\theta + y\sin\theta - \rho = 0 \tag{2.14}$$

kde *(x, y)* jsou souřadnice pixelu v obraze, *k* je směrnice přímky a *q* je posunutí přímky ve směru osy y. Pro normálový tvar je ρ délka normály přímky od počátku souřadného systému a θ je úhel, který normála svírá s osou x.

Hledání přímek probíhá na binárním obraze, obvykle hrubé hranové reprezentaci, kde všechny hranové pixely mohou potenciálně tvořit přímku a všechny pixely pozadí jsou ignorovány. Parametry *k* a *q* tvoří dvojrozměrný parametrický prostor, který obsahuje všechny kombinace hodnot parametrů. Každým hranovým pixelem obrazu může procházet více přímek s různými hodnotami parametrů. Parametry těchto přímek se v prostoru parametrů transformují jako přímka (7 str. 87):

$$q = -xk + y \tag{2.15}$$

V prostoru parametrů se tedy pro každý hranový pixel vytvoří přímka. Průsečík těchto přímek pak značí přímku s takovou kombinací parametrů *k* a *q*, že prochází více než jedním hranovým pixelem (viz Obrázek 2.9).



Obrázek 2.9: Princip hledání přímek pomocí Houghovy transformace (7 str. 87).

Pokud bychom chtěli hledat jiné tvary než přímky, jako například kružnice, změní se pouze tvar použité rovnice a počet rozměrů parametrického prostoru, který je roven počtu parametrů. V případě kružnic máme pro obecnou rovnici

(Rovnice 2.16) tři parametry – poloměr *r*, a souřadnice středu *(x_0, y_0)*. Hledání parametrů kružnice probíhá opět hledáním průsečíků v parametrickém prostoru. Parametry se ale tentokrát transformují na kružnice různých poloměrů, místo přímek. Počítání parametrů pro každý poloměr kružnice by bylo zdlouhavé, proto pokud neznáme poloměr hledaných kružnic, měli bychom alespoň určit interval poloměrů hledaných kružnic (7 str. 90).

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \tag{2.16}$$

V případě digitálních obrazů je parametrický prostor Houghovy transformace diskrétní (tzv. akumulátor). Pro každý hranový pixel je akumulátor inkrementován na pozicích, které odpovídají kombinaci parametrů přímek procházejících hranovým pixelem. Místo hledání průsečíků přímek v parametrickém prostoru, hledáme v akumulátoru pozice s lokálně nejvyšší hodnotou (počtem hlasů) (7 str. 90).

Získat optimální hodnoty parametrů z akumulátoru lze více způsoby. Nejjednodušším způsobem je akceptovat všechny kombinace parametrů, které mají počet hlasů vyšší než stanovená mez. Lepším způsobem je hledání lokálního maxima nebo, v případě většího počtu velkých hodnot blízko u sebe, nalezení těžiště shluku pomocí shlukové analýzy.

2.3 Morfologické operace

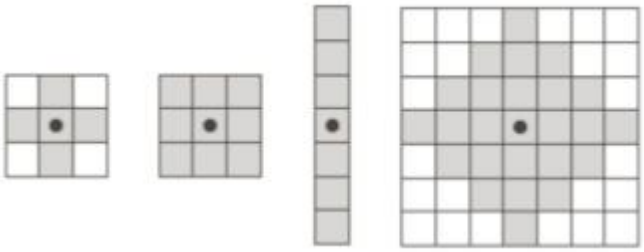
Morfologické operátory se používají k dodatečnému zpracování obrazů a analýze jeho objektů. Obvykle se používají při práci s binárními obrazy, ale byly zobecněny také na šedotónové obrazy. Tato podkapitola se však omezuje pouze na použití morfologických operátorů s binárními obrazy. Popisuje základní principy a metody ztenčování obrazů.

2.3.1 Základní morfologické operace

Morfologické operátory jsou lokální a používají masku, která se nazývá strukturní element. Strukturní element může být v podstatě libovolného tvaru, ale pro práci s obrazem je však strukturní element doplňován na obdélníkový tvar. U základních operací se ve strukturnímu elementu rozlišují aktivní prvky (obvykle značeno jedničkou nebo vybarveným políčkem) a neaktivní prvky (nuly, bílá políčka). V některých případech jsou ve strukturním elementu rozlišovány prvky, pod kterými se musí nacházet pixel náležící objektu, prvky, pod kterými se nesmí nacházet pixel náležící objektu, a prvky (obvykle značené křížkem) na kterých

nezáleží (1 str. 651). Volba strukturního elementu je důležitá a výrazně mění výsledek morfologických operací.

Strukturní element také obsahuje referenční bod (značený tečkou nebo kroužkem), pod který se do výstupního obrazu ukládá výsledek operace. Referenční bod se může nacházet kdekoliv ve strukturním elementu a jeho poloha ovlivňuje výsledek operace. Pokud není referenční bod vyznačen, lze předpokládat, že je referenčním bodem prostřední prvek (1 str. 651). Nejjednoduššími operacemi, které tvoří základ dalších složitějších operací, jsou dilatace a eroze.



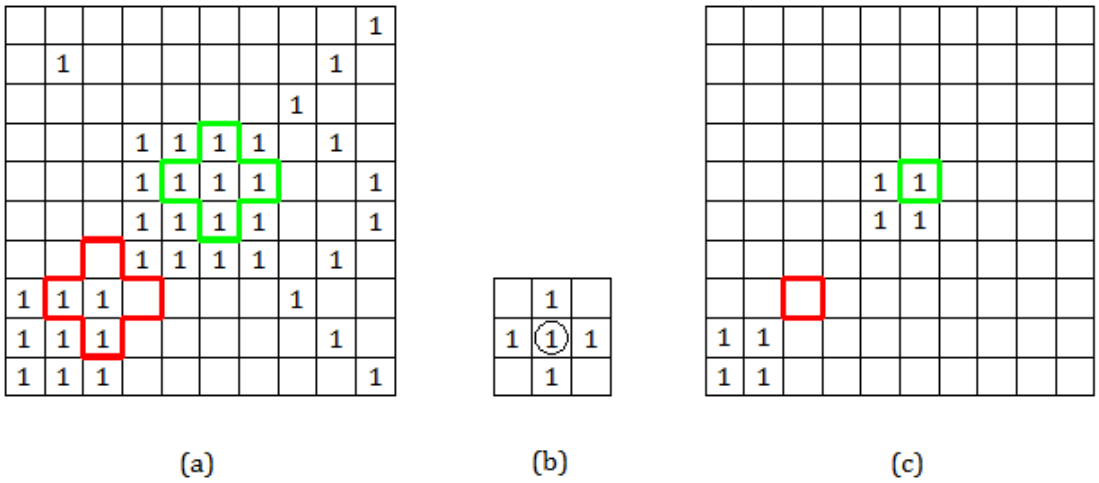
Obrázek 2.10: Příklad strukturních elementů (1 str. 651).

2.3.1.1 Eroze a dilatace

První základní operace, eroze, provedena nad binárním obrazem, kde hodnota jedna náleží objektům a nula pozadí, zapisuje do výstupního binárního obrazu na pozici referenčního bodu jedničku v případě, že jsou pod všemi aktivními prvky strukturního elementu v obraze jedničky. Tuto operaci lze vyjádřit jako (1 str. 653):

$$A \ominus B = \{z|(B)_z \subseteq A\} \tag{2.17}$$

Vyjádřeno slovy, obraz A erodovaný podle B je množina takových bodů $z = (z_1, z_2)$, že B, posunuté v obraze na souřadnice z, je podmnožinou obrazu A. Výsledek eroze obrazu znázorňuje Obrázek 2.11, kde jsou aktivní prvky strukturního elementu a pixely objektů obrazu vyznačeny číslem jedna. Na obrázku je zelenou barvou vyznačena pozice, pro kterou je podmínka eroze splněna a do výstupního obrazu je na tuto pozici zapsána jednička. Červenou barvou je naopak vyznačena jedna z pozic, kde v obraze nejsou body objektů pod všemi aktivními prvky strukturního elementu. Z výsledku eroze lze vypozorovat, že eroze objekty odstraňuje, zmenšuje a rozpojuje.

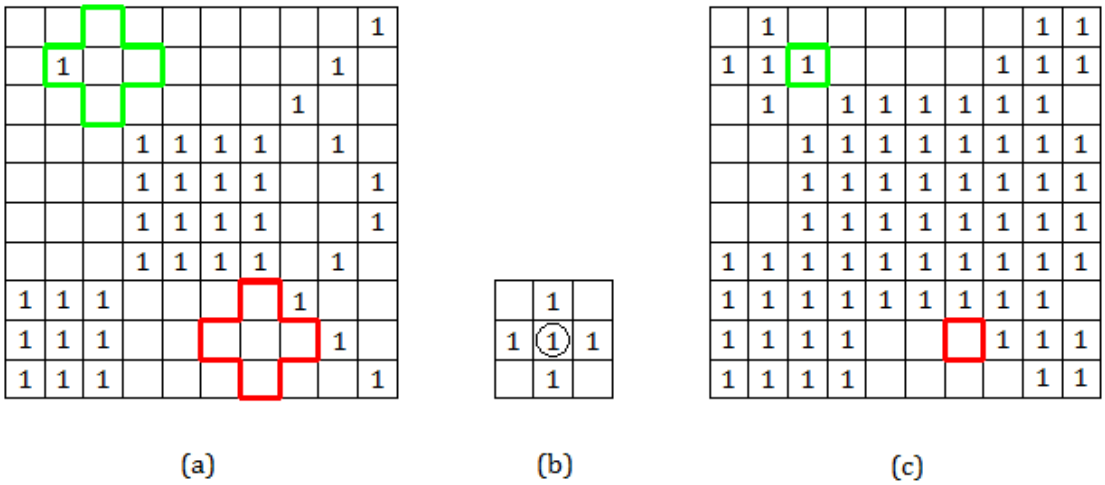


Obrázek 2.11: Znázornění binární eroze. (a) vstupní binární obraz s dvěma vyznačenými polohami strukturního elementu, (b) strukturní element, (c) výsledný erodovaný obraz.

Druhá základní operace, binární dilatace, do výstupního obrazu zapisuje jedničku v případě, že je v obraze alespoň pod jedním aktivním prvkem strukturního elementu bod objektu. Vyjádřeno matematicky (1 str. 655):

$$A \oplus B = \{z|(B)_z \cap A \neq \emptyset\} \tag{2.18}$$

Na rozdíl od eroze, dilatace zvětšuje a spojuje objekty na obrazu (viz Obrázek 2.12). Dilatace a eroze jsou duálními operacemi, ne inverzními. Pokud eroze odstraní z obrazu malý objekt, dilatace jej již nemůže vrátit zpět. Kombinaci eroze a dilatace se proto získají nové operátory morfologického otevření a uzavření (1 str. 657).



Obrázek 2.12: Znázornění binární dilatace. (a) vstupní binární obraz se dvěma vyznačenými polohami strukturního elementu, (b) strukturní element, (c) výsledný dilatovaný obraz.

2.3.1.2 Otevření a uzavření

Otevření a uzavření jsou morfologické operace, které vzniknou provedením operací dilatace a eroze za sebou. Operace otevření je provedena, pokud je obraz erodován a poté dilatován a operace uzavření je provedena, pokud je obraz dilatován a poté erodován. Pouze první použití těchto operací mění obraz, opakované použití nemá smysl (1 str. 658).

Při otevření jsou erozí všechny objekty zmenšeny a díry zvětšeny. Dojde k rozpojení objektů spojených tenkými čárami, odstranění velmi malých objektů a výstupků objektu, čímž se jeho povrch stane hladším. Dilatace následně většinu těchto změn vrátí zpět, nedokáže však obnovit zcela odstraněné objekty ani tenké spoje (1 str. 657).

Uzavření naopak při dilataci objekty zvětší, zaplní malé díry a propojí úzké mezery, čímž také dojde k vyhlazení povrchu objektů. Následná eroze pak vrátí objektům původní velikost, ale nedokáže už obnovit zcela zacelené díry nebo objekty rozpojit (1 str. 657).

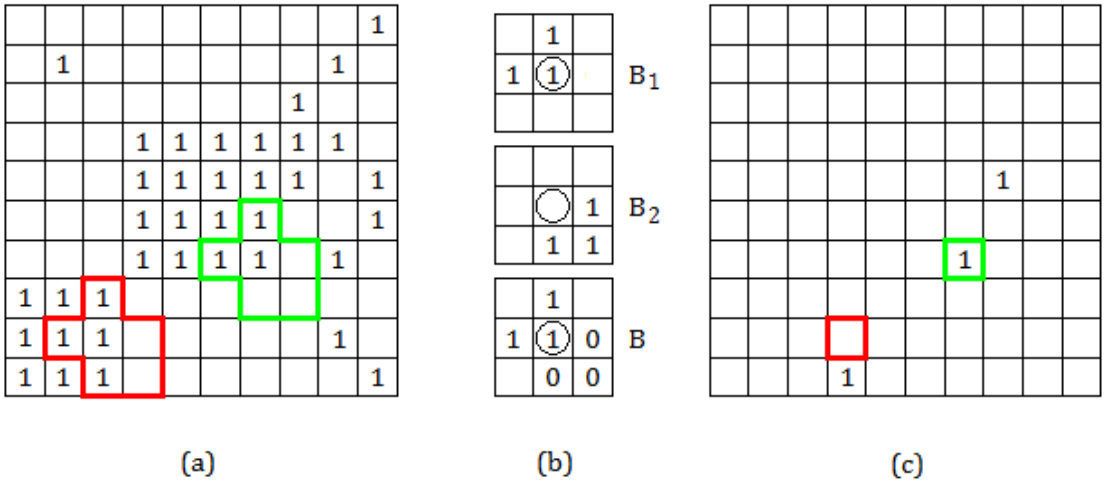
2.3.1.3 Operátor hit-or-miss

Operátor hit-or-miss je základním nástrojem pro detekci tvarů v obraze. Na rozdíl od eroze nebo dilatace pracuje s body objektů i s body pozadí obrazu. Strukturní element obsahuje prvky související s body objektů (jedničky), prvky související s body pozadí (nuly) a prvky, na kterých nezáleží. Do výstupního obrazu je pod referenční bod zapsána jednička v případě, že jsou v obraze body objektu pod všemi prvky strukturního elementu, které náleží objektu, a body pozadí pod všemi prvky náležící pozadí (1 str. 663).

Pokud rozdělíme strukturní element B na dva – B_1 a B_2, kde B_1 obsahuje pouze prvky B související s body objektů a B_2 pouze prvky B související s pozadím, můžeme operaci hit-or-miss zapsat jako (1 str. 663):

$$A \circledast B = (A \ominus B_1) \cap (A^C \ominus B_2) \tag{2.19}$$

kde A^C je množinový doplněk, tedy pozadí obrazu A. Obrázek 2.13 znázorňuje operaci hit-or-miss se strukturním elementem, který v obraze hledá pravé dolní rohy. Použitím vícero strukturních elementů nezávisle a složením výsledných obrazů dohromady lze dosáhnout operací jako je detekce rohů nebo ztenčování a v kombinaci s dalšími morfologickými operátory je možné získat například konvexní obálku objektu.



Obrázek 2.13: Znázornění operace hit-or-miss. (a) vstupní binární obraz se dvěma vyznačenými polohami strukturního elementu, (b) strukturní element a jeho části, (c) výsledný obraz.

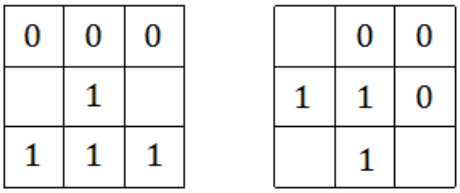
2.3.2 Morfologické ztenčování

Morfologické ztenčování odstraňuje pixely objektů z binárního obrazu, dokud nejsou všechny čáry tloušťky jednoho pixelu. Ztenčování je jedním ze způsobů, jak lze získat skelet objektu. Operaci ztenčování je možné vyjádřit s pomocí operátoru hit-or-miss (1 str. 671) jako:

$$A \otimes B = A - (A \circledast B) \tag{2.20}$$

K morfologickému ztenčování se používá několik strukturních elementů (viz Obrázek 2.14) a všechny čtyři jejich rotace. Jedna iterace ztenčování spočívá v postupné aplikaci jednotlivých strukturních elementů. Výsledek operace s prvním strukturním elementem je použit jako vstup do operace s druhým strukturním elementem atd. Iterace pokračují, dokud v obraze probíhají změny (1 str. 671).

Algoritmy implementující operaci ztenčování (např. Zhang-Suen nebo Guo-Hall) mají použitý strukturní element obvykle zapsaný přímo v kódu. Algoritmy se liší v rychlosti, ale také ve tvaru vytvořených skeletů a v tom, jestli jsou pixely spojeny v osmi směrech nebo čtyřech směrech.



Obrázek 2.14: Příklad strukturních elementů pro morfologické ztenčování.

2.4 Podobnost textových řetězců

Kdykoliv je nutné porovnat dvě slova podrobnějším způsobem než jen na základě totožnosti, dostáváme se do problematiky určení podobnosti textových řetězců. Určení podobnosti dvou slov nebo vět je základem pro detekci plagiátorství a je užitečné při vyhledávání ve slovnících, zvláště pak, pokud vyhledávaná slova mohou obsahovat chyby a překlepy.

Abychom mohli určit podobnost dvou řetězců, je potřeba zvolit metodu měření vzdálenosti mezi řetězci a metriku pro určení podobnosti ze zjištěné vzdálenosti.

2.4.1 Měření vzdálenosti textových řetězců

Běžným způsobem určení vzdálenosti dvou textových řetězců je zjištění počtu změn nutných k přeměně jednoho řetězce ve druhý. Pro tento účel existuje vícero vzdáleností, lišící se v typech změn, které dovolují. Dále existují i zobecnění jako například pro měření vzdáleností mezi řetězci DNA. Princip měření vzdálenosti na základě počtu změn je v následujícím textu vysvětlen s pomocí často používané Levenshteinovy vzdálenosti (12 str. 109).

Levenshteinova vzdálenost povoluje jako změny v textu operace záměny znaku, odstranění znaku a přidání znaku. Například, Levenshteinova vzdálenost mezi slovy Levenshtein a Frankenshtein je:

$$Levenshtein \rightarrow \boldsymbol{F}evenshtein \rightarrow F\boldsymbol{r}venshtein \rightarrow Fr\boldsymbol{a}enshtein \rightarrow Fra\boldsymbol{n}enshtein \rightarrow Fran\boldsymbol{k}enshtein$$

rovna 5 (3 záměny, 2 přidání). Levenshteinva vzdálenost tedy nalezne minimální počet změn k transformaci jednoho řetězce na druhý. Platí také, že není nikdy menší než rozdíl délek obou řetězců a není nikdy větší než délka delšího z řetězců.

Pokročilejší varianta Levenshteinovy vzdálenosti umožňuje navíc nastavit váhu jednotlivým operacím. Pokud nás zajímá více podobnost významů řetězců, můžeme je před měřením vzdálenosti zpracovat lematizací (převod slova do základního tvaru) nebo odstraněním slov nesoucí pouze malou informaci (spojky, předložky apod.) (12 str. 109).

Jelikož Levenshteinova vzdálenost pracuje s řetězcem jako s celkem, není vhodná k zjištění vzdálenosti mezi větami, protože dvě věty mohou obsahovat stejná slova v jiném pořadí, ale jejich vzdálenost by byla určena jako velká. Větu je ovšem možné rozdělit na jednotlivá slova a vzájemně je porovnat. Pro určování podobnosti rozsáhlých textů však existují lepší alternativy (12 str. 109).

2.4.2 Metrika podobnosti textových řetězců

Pro určení podobnosti dvou řetězců není dostačující pouze znalost vzdálenosti mezi nimi. Vzdálenost dvou úprav má pro řetězce o délce tří znaků jiný význam než pro řetězce o délce deseti znaků. Pro určení podobnosti mezi daty existuje mnoho metrik, každá vhodná pro použití na jiných datech. Nejjednodušší metrikou pro určení míry podobnosti mezi řetězci podle jejich vzdálenosti je prostý poměr mezi zjištěnou vzdáleností *dist*(*A, B*) a maximální možnou vzdáleností:

$$sim(A,B) = \frac{dist(A,B)}{max(|A|,|B|)} \tag{2.21}$$

Tím bychom získali výsledek v rozmezí hodnot od nuly do jedné, kde nula odpovídá totožnosti mezi *A* a *B*. Jedná se tedy o míru nepodobnosti a míru podobnosti lze získat jako *1 – sim*(*A, B*). Další pokročilejší metrikou vhodnou pro použití na vzdálenost mezi textovými řetězci je Sørensen–Dice koeficient.

Koeficient Sørensen–Dice je určen jako poměr počtu společných prvků sdílených dvěma množinami ku součtu velikostí obou množin (13 str. 4):

$$DICE(A,B) = \frac{2|A \cap B|}{|A| + |B|} \tag{2.22}$$

Výsledkem je míra podobnosti mezi množinami *A* a *B*. Avšak vzdálenost řetězce, jako je Levenshteinova vzdálenost, vyjadřuje nepodobnost mezi dvěma řetězci. Pro použití s Levenshteinovou vzdáleností by tedy bylo nutné koeficient Sørensen–Dice upravit do následujícího tvaru:

$$DICE(A,B) = 1 - \frac{2 \cdot dist(A,B)}{|A| + |B|} \tag{2.23}$$

kde *dist*(*A, B*) je Levenshteinova nebo jiná metrika vzdálenosti. V tomto tvaru koeficient vyjadřuje podobnost mezi řetězci *A* a *B*, které jsou totožné pro hodnotu rovnu jedné a zcela odlišné pro hodnotu rovnu nule. Vrátíme-li se k příkladu pro Levenshteinovu vzdálenost, byla by míra podobnosti mezi slovy Levenshtein a Frankenshtein rovna:

$$DICE(A,B) = 1 - \frac{2 \cdot 5}{11 + 13} = 1 - 0{,}41\overline{6} = 0{,}58\overline{3} \tag{2.24}$$

3 ROZBOR ÚLOHY

V této kapitole je proveden podrobný rozbor úlohy a jsou diskutovány možnosti jejího řešení, které navazují na teorii vysvětlenou v předchozí kapitole.

3.1 Švédská křížovka

V úloze se zabýváme zpracováním švédské křížovky, která je pouze jedna z mnoha druhů křížovek. Švédská křížovka je však pro nás tou nejdůležitější, jelikož je nejpopulárnějším typem křížovky ve střední Evropě, a tudíž i v České republice (14).

Švédská křížovka se vyznačuje tím, že její legenda je vepsána do obrazce. Má také vyznačenou tajenku, která může být tvořena řadou vyznačených polí ve sloupci či řádku nebo jednotlivými očíslovanými poli rozprostřenými po křížovce. Následuje ukázka typické švédské křížovky.

1	PŘIČLENĚNÍ PODNIKU K DRUHÉMU	**1. DÍL TAJENKY**	KÓD ITÁLIE	ČIN	SHROMAŽDIŠTĚ V KONCENTRÁKU	**POMŮCKA:** Avatea, Lui, NTU, ša.	SLOUČENINY KYSELIN A ZÁSAD	TERMINÁL (ANGLICKÁ ZKRATKA)	DOSTÁVAT ZPRÁVU	VOUS
ZÍDKA NAD HLAVNÍ ŘÍMSOU						ST. NAKLADATELSTVÍ DĚT. KNIHY / KAL				
DRUH PROTONU										
ZNAČKA INDIA			**3. DÍL TAJENKY** / SRBSKÁ ČEPICE							
ULEKNUTÍ				ŘEKA V ZAMBII / TEXTILNÍ ROSTLINA				SEVEŘAN (SLOVEN.) / BÝT DEŠTIVO		
SMĚNEČNÝ DLUŽNÍK							ZNAČKA PALLADIA / KARETNÍ VÝRAZ			POUČKA
ŠPIČKA						HOSPODÁŘSKÝ DOBYTEK / ÚKAZ				
MUŽSKÉ JMÉNO					**2. DÍL TAJENKY** / ČÍNSKY „PÍSEK“					
				ŠUSTOT						
				POLYNÉSKÝ MĚSÍČNÍ BŮH						

Obrázek 3.1: Švédská křížovka z obálky časopisu Švédské křížovky září–listopad 2015 (15)

3.1.1 Vlastnosti

Tvorba a řešení křížovek se řídí Směrnicemi pro tvorbu křížovek, které byly vypracovány Odbornou křížovkářskou komisí Českého svazu hádankářů a křížovkářů. Jejich platná verze je z roku 1994 ve znění dodatků z let 1998, 2003, 2004 a 2006 (16).

Na základě těchto směrnic a vlastního pozorování můžeme určit vždy, nebo alespoň obecně, platné vlastnosti švédských křížovek, o které se můžeme opřít při zpracování obrazu křížovky a následném automatickém luštění.

3.1.1.1 Vlastnosti obrazu

Obrazec švédské křížovky je tvořen čtvercovou základní sítí. Síť má nejčastěji obdélníkový nebo trojúhelníkový tvar. Většina políček má velikost jednoho čtverce, ale jsou povolena i políčka vytvořená spojením několika sousedních čtverců. Políčka můžeme rozdělit na políčka s legendou, políčka slepá, políčka tajenková a políčka vpisovací (17 stránky 9-13).

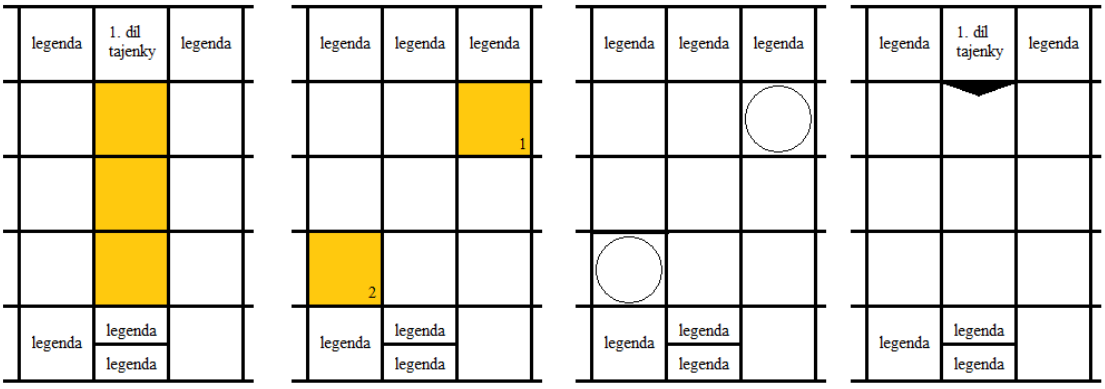
Políčka s legendou obsahují text a vždy sousedí s políčkem tajenkovým anebo s políčkem vpisovacím. Může být vodorovně rozpůleno na dvě části. Horní část obsahuje legendu pro řádek a spodní část obsahuje legendu pro sloupec. Pokud políčko není rozděleno, obsahuje legendu pro sloupec nebo řádek podle toho, z které strany sousedí s vpisovacím políčkem. Políčka s legendou se zpravidla nachází na levé straně řádku a na horní straně sloupce.

Slepá políčka obvykle nesousedí s políčkem tajenkovým ani vpisovacím a nikdy neobsahují legendu. Používají se pro vyplnění mezer v křížovce, aby se udržel požadovaný tvar křížovky. Často obsahují pomůcku – vpisované výrazy obsažné v křížovce, které autor považoval za obtížné na vyřešení. Mohou také obsahovat grafické prvky nebo může dojít ke sloučení více políček, které pak obsahují větší obrázek.

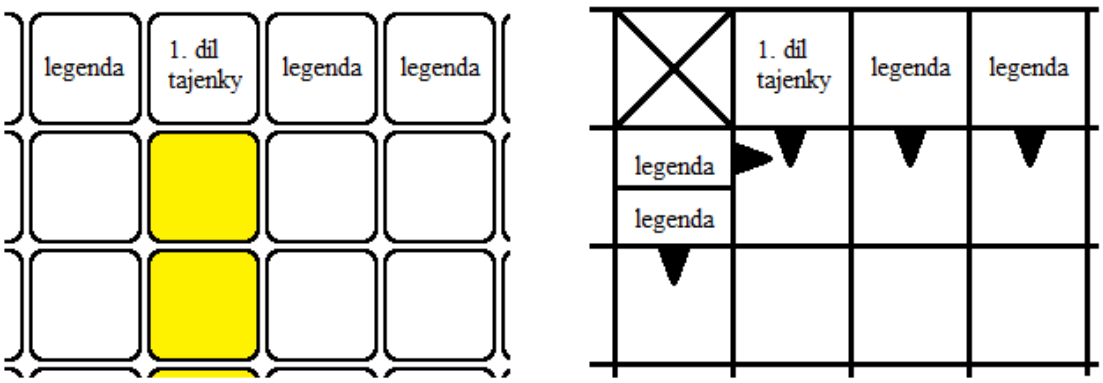
Tajenková políčka jsou od vpisovacích odlišena nejčastěji barvou políčka, v případě rozprostřené tajenky mohou políčka také obsahovat malou číslici označující pořadí písmen v tajence. V případě tajenky tvořené celým řádkem nebo sloupcem může být tajenka pouze označena legendou a směrovou šipkou, tajenkou jsou tak políčka od šipky po konec řádku či sloupce (viz. Obrázek 3.2).

Mřížka křížovky bývá obvykle jednoduchá a černá, ale může být i jiné barvy. Barva mřížky je obvykle odlišná od pozadí všech políček, ale nemusí to být pravidlem (např. pokud se v křížovce střídá více barev pozadí).

Výjimečně je možné narazit na další varianty grafického ztvárnění křížovky. Například křížovka bez klasické mřížky, která je místo toho tvořena jednotlivými políčky – buňkami, poskládanými vedle sebe, nebo také znázornění směru vpisování slova do křížovky pomocí malé šipky (viz. Obrázek 3.3).



Obrázek 3.2: Příklad způsobů značení tajenky.



Obrázek 3.3: Neobvyklé grafické úpravy křížovky.

3.1.1.2 Vlastnosti textu

U křížovky se rozlišují 3 druhy výrazů – legendový, vpisovaný a tajenkový. Vpisovaný výraz může být slovo, část slova, číslo, zkratka a značka, spojení výrazů nebo název. Slovo musí být v základním tvaru (1. pád jednotného nebo množného čísla, infinitiv), částmi slova se rozumí předpony, přípony (s výjimkou koncovek) a části složeného slova (17 stránky 2-7).

V křížovce může být každý vpisovaný výraz použit pouze jednou bez ohledu na věcný význam. Vpisovaný výraz musí být také vepsán alespoň do dvou políček křížovky. U švédských křížovek se výraz vpisuje po jednotlivých písmenech, nerozlišujeme malá a velká písmena a za písmeno není považován doplněk znaku (tedy odsuvník, index a exponent). Písmenem je i mezislovní mezera, která se ale obvykle nepoužívá (17 stránky 8, 15-18).

K vpisovaným výrazům se také vztahuje křižování, které je založené na shodě znaků v příslušném políčku, kde se protínají výrazy zapisované ve svislém a vodorovném směru (17 str. 20).

Legendový výraz slouží jako vodítko pro nalezení správného vpisovaného výrazu. Směrnice definuje legendový výraz jako výstižnou a aktuální

charakteristiku, synonymum nebo definici vpisovaného výrazu. Pokud je vpisovaný výraz cizojazyčný, je u legendového výrazu uvedeno, o jaký jazyk se jedná (17 str. 24). Tajenkové výrazy tvoří tajenku, která je součástí každé křížovky (17 str. 35).

3.2 Existující nástroje pro práci s křížovkami

Myšlenka převodu křížovky z papíru do digitální podoby nebo využití software k jejímu řešení není nová a už jen krátkým průzkumem internetu je možné nalézt různé aplikace pro Android i jiné systémy. Aplikace, které jsem nalezl, lze rozdělit do následujících skupin na základě jejich funkce.

Editory křížovek

Příklad aplikací: CwdStudio, Headache!, EclipseCrossword

Tyto nástroje umožňují tvořit či přímo generovat křížovky ze slovníku. Vytvořené křížovky lze kromě tisku distribuovat i ve formě souboru. Digitální křížovky nemají žádný standardní formát. Dlouhou dobu byl nejpoužívanějším formátem PUZ, který má však určitá omezení a byl zpřístupněn neoficiálně zpětným inženýrstvím formátu (18). Proto postupem času vznikly další formáty a některé z nich jsou i otevřené.

Aplikace pro převod křížovky do digitální podoby

Příklad aplikací: Conspire, Puzzle Monkey, CrosswordScanner

Tyto aplikace po předložení snímku s křížovkou vytvoří digitální křížovku, kterou může uživatel luštit zadáváním znaků přes klávesnici. Aplikace, které jsem nalezl, využívají v různé míře pomoc od uživatele při převodu křížovky a rozeznávají políčka oddělená mřížkou, ale už ne jejich obsah.

Aplikace umožňující luštění křížovek

Příklad aplikací: Across Lite, onlinekrizovky.cz

Tyto aplikace umožňují uživateli dekódovat soubory křížovek a následně je řešit. Mohou také poskytnout přístup k zabudovanému křížovkářskému slovníku nebo nápovědu ze známého řešení, které je zakódováno v souboru křížovky. Kromě aplikací na Android a další systémy existují i online služby poskytující křížovky k řešení.

Křížovkářské slovníky

Příklad aplikací: krizovkarsky-slovnik.cz, Crossword Solver

Nabízí možné odpovědi na hledaný výraz. Pokročilejší slovníky umožňují zúžit rozsah možných odpovědí zadáním počtu znaků odpovědi nebo také zadáním již známých znaků odpovědi. Většina slovníků je přístupná pouze online, ale existují i aplikace, které fungují jako offline slovníky (např. Crossword Solver).

Aplikace řešící křížovky samostatně

Příklad aplikací: Crossword Maestro

Zatímco ostatní aplikace nabízí nanejvýš nápovědu ze známého řešení, Crossword Maestro je dle výrobce schopen sám křížovku vyřešit, a to i v případě, že je legenda tvořená anagramy či jinými kryptickými vodítky. K tomu mu dopomáhá rozsáhlý slovník, který je ovšem v anglickém jazyce.

Přestože všechny podstatné funkce, které by měla cílová aplikace obsahovat, jsou již vyřešeny v některé z existujících aplikací, nebyl jsem schopen nalézt aplikaci, která by dokázala převést křížovku z fotografie do digitální podoby, umožnit uživateli křížovku luštit a také ji sama vyřešit. Navíc většina z výše uvedených aplikací pochází z cizích zemí, a proto buď nepodporuje švédský typ křížovky nebo alespoň nepodporuje český jazyk.

3.3 Segmentace švédské křížovky

Abychom mohli s křížovkou pracovat, potřebujeme ji nejprve na snímku rozpoznat. Chceme tedy určit polohu a rozměry jednotlivých políček a rekonstruovat je do mřížky, která bude totožná s mřížkou křížovky, které byl snímek pořízen. K tomu potřebujeme vybrat vhodnou metodu segmentace mřížky.

3.3.1 Výběr metody segmentace

Výběr metody segmentace se bude odvíjet od informací, které známe o vzhledu švédských křížovek a vlastnostech snímků, které tvoří naše vstupní data. Nesmíme také zapomenout na cílovou platformu, kterou je mobilní zařízení s operačním systémem android, což omezuje možnosti dostupného hardware i software.

Přestože mají autoři švédských křížovek velkou volnost ve volbě jejich vzhledu, křížovka bude vždy složena ze sítě stejně velkých čtvercových polí. Takovou křížovku by tedy bylo možné pokládat za silnou, pravidelnou, hrubou

texturu a bylo by možné zde uplatnit texturní analýzu. Ovšem křížovka v sobě obsahuje také text a obrazy, které nejsou rozmístěny pravidelně a tuto texturu narušují. Navíc je naší snahou nalézt polohu jednotlivých políček, zatímco za pomocí textury jsme schopni pouze segmentovat oblast kde se nachází křížovka.

Další možností je hranově orientovaná segmentace. Tedy vytvoření hranové reprezentace obrazu. Jelikož mřížka křížovky je výrazně odlišná od pozadí, nachází se na rozhraní mřížky a pozadí silná hrana. V ideálním případě by hranová reprezentace obsahovala oddělené oblasti odpovídající políčkům a zbývalo by jen rozlišit, které oblasti jsou políčka a které ne. Toho je možné dosáhnout kontrolou velikosti a popřípadě polohy oblastí, jelikož políčka by měla mít podobnou velikost a budou mít ve své blízkosti další políčka (tedy oblasti podobné velikosti a tvaru). Nesmíme ale také zapomenout na fakt, že políčka, která obsahují legendu, nebudou segmentovány zcela, a tudíž budou mít mimo jiné odlišný obsah než prázdná políčka. Hrubá hranová reprezentace však může obsahovat přerušené hrany a díky tomu propojené oblasti políček. Tento postup proto vyžaduje provést propojení přerušených hran nebo oblasti oddělit s pomocí morfologických operací, čímž se zvyšuje náročnost výpočtu.

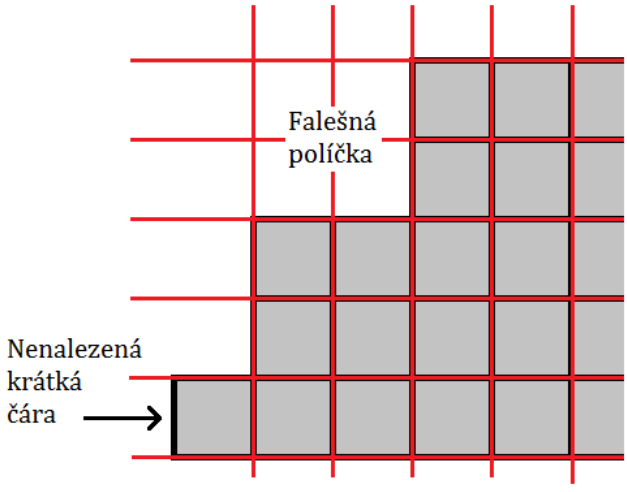
Podobné hranově orientované segmentaci jsou v této úloze metody založené na regionech. Narůstání oblastí s dynamickým kritériem bude v případě, kdy se kritérium řídí podle jasu obrazu, opět vytvářet oblasti oddělené na hranách mřížky. Jednou z negativních vlastností narůstání oblastí je možnost odlišného výsledku v závislosti na pořadí zpracování pixelů a startovního pixelu. V závislosti na způsobu volby pozice semínka a implementace algoritmu růstu může být výsledek segmentace pokaždé jiný. Podobný výsledek získáme i metodou dělení a slučování oblastí, jen s tím rozdílem, že oblasti, vzniklé rekurzivním dělením na kvadranty a opětovným slučováním podobných oblastí, tíhnou k pravoúhlosti. Což by neměl být problém vzhledem k tomu, že křížovka samotná je také pravoúhlá, ale situace se změní je-li křížovka na snímku pootočena. A stejně jako u hranově orientované segmentaci i zde se můžeme setkat s propojenými oblastmi vlivem podmínek pořizování snímku (např. nerovnoměrné osvětlení), nad kterými nemáme kontrolu.

Další z metod segmentace, metoda rozvodí, nese oproti předchozím metodám tu výhodu, že vytváří automaticky spojité a uzavřené hranice oblastí. Je tedy méně pravděpodobné že by došlo ke sloučení dvou sousedících políček vlivem slabší hrany. Na druhou stranu má tendenci k přesegmentování obrazu, a proto potřebuje dodatečně spojit malé podobné oblasti do větších. Rozlišení samotných políček pak probíhá obdobným způsobem jako v předchozích případech.

Další možností segmentace políček je prahování. Vzhledem k možnosti špatného osvětlení křížovky na snímku se neobejdeme bez prahování adaptivního. Výsledek adaptivního prahování je srovnatelný s hranově orientovanou segmentací, jsou tu však určité rozdíly. Mřížka křížovky má dvě hrany, a proto po detekci hran vzniknou v obraze oblasti políček oddělené dvojitou hranou. Adaptivní prahování

naproti tomu vytváří při volbě dostatečně velkého okolí jednu silnou hranu. Na druhou stranu, detekce hran podává stejný výsledek pro tmavou mřížku se světlým pozadím jako na světlou mřížku s tmavým pozadím. Výsledek adaptivního prahování by se mezi těmito dvěma případy lišilo a bylo by potřeba obraz invertovat.

Z pokročilejších metod máme k dispozici Houghovu transformaci, a to konkrétně pro přímky. Získáme-li hranovou reprezentaci obrazu, můžeme nalézt přímky tvořící mřížku křížovky a na základě jejich průsečíků určit rohy jednotlivých políček. Houghova transformace s sebou nese určité výhody jako je například nízká citlivost na přerušené hrany a také není potřeba brát ohled na obsah políček. Nese však i určité nevýhody. Křížovka obvykle není na papíře sama, ale má ve svém okolí text a případně i obrazy. Tyto objekty mohou způsobit vytvoření falešných přímek, pokud předtím nevyčleníme z obrazu oblast křížovky. Ale i v případě, že máme obraz, který obsahuje pouze křížovku, je třeba počítat i s tvarovými variacemi křížovek. Například, v případě poměrně časté trojúhelníkové varianty křížovky, musíme řešit problém fiktivních políček, které nám vznikají vlivem křížení přímek vně křížovky, a problém krátkých čar na rozích křížovky, které mohou být vlivem nízkého počtu hlasů v akumulátoru vyhodnoceny jako slabé. Průsečíky přímek také nemusí příliš přesně odpovídat skutečné poloze políček kvůli možným deformacím mřížky křížovky.



Obrázek 3.4: Problémy segmentace podle přímek Houghovy Transformace.

Další z pokročilých metod segmentace jsou metody pružných a aktivních kontur. Tyto metody deformují prvotní konturu oblasti na základě obsahu obrazu a jsou schopné přesně segmentovat i objekty složitých tvarů. Pro tuto úlohu však nejsou aktivní kontury příliš vhodné, protože by bylo nutné opakovaně tvarovat konturu pro každé políčko. Vzhledem k tomu, že políčka nemají složitý tvar a implementace řešení má pracovat na mobilním zařízení, jsou aktivní kontury zbytečně náročným řešením, jelikož můžeme dosáhnout podobně dobrého řešení z pomocí jiných jednodušších metod.

Kromě segmentace podle oblastí políček a podle úseček tvořících mřížku můžeme segmentovat políčka také na základě rohů a průsečíků mřížky. K tomu lze využít algoritmy detekující významné body, což jsou body, které ve svém okolí vykazují vysoký gradient jasové funkce, tedy vrcholy, rohy a hranice. Tyto algoritmy (Moravcův operátor, detektor Shi&Tomasi, FAST aj.) detekují na obraze všechny průsečíky, rohy a vrcholy, což zahrnuje průsečíky a rohy mřížky křížovky, ale také body na písmenech a v obrazech uvnitř i vně křížovky. Problém tedy spočívá v rozlišení významných bodů mřížky od významných bodů okolních objektů. Při rozlišení těchto bodů využijeme znalostí o vlastnostech mřížky křížovky. Víme například, že průsečíky mřížky jsou od sebe vzdáleny v násobcích délky strany políčka a jsou uspořádány do pravoúhlé mřížky. Mřížka kromě toho obsahuje velké množství průsečíků. Průsečíky obsahuje také text ve znacích “t“ a “f“, text je však odlišitelný tím, že má obecně velkou koncentraci významných bodů. Průsečíky a rohy je možné také detekovat v hranové reprezentaci obrazu pomocí morfologických operátorů nebo konvolucí s vhodnými maskami (např. tvaru kříže).

Nakonec je třeba ještě zmínit segmentaci založenou na barvě. Rozlišovat oblasti podle barvy není pro tuto úlohu vhodné kvůli neznalosti barev používaných křížovkou na snímku. Navíc většina políček má shodnou barvu se zbytkem papíru v okolí křížovky.

Máme tedy k dispozici vícero metod, které jsou schopné políčka segmentovat, a každá má své výhody a nevýhody. Z toho vyplývá, že k dosažení robustního řešení bude vhodné zkombinovat několik metod a vyvážit tak tím jejich nedostatky.

K řešení problému segmentace políček jsem tedy zvolil kombinaci Houghovy transformace pro hledání přímek a detekce průsečíků v binárním obraze hran pomocí křížové masky. Detekce průsečíků umožňuje nalézt rohy jednotlivých políček bez ohledu na tvar křížovky a je méně citlivá na přerušené hrany a deformace mřížky. Přímky nalezené Houghovou transformací mi umožňují získat odhad velikosti políček křížovky, podle kterého je možné filtrovat falešné rohy a průsečíky v obraze. Dále je možné z nich získat informaci o rotaci křížovky v obraze a snímek křížovky zarovnat se souřadnicovými osami obrazu. Díky tomu není potřeba při hledání významných bodů otáčet křížovou masku. Pro získání hranové reprezentace obrazu jsem zvolil metodu adaptivního prahování, která poskytuje silné nezdvojené hrany, které poskytují Houghově transformaci určitou toleranci na deformace mřížky.

3.3.2 Předzpracování vstupních dat

Máme-li zvolenou metodu segmentace políček, můžeme přistoupit k předzpracování obrazu, které bude odpovídat potřebám vybrané metody. Jde především o potlačení šumu a normalizaci rozlišení, které jsou důležité pro mnoho z výše uvedených metod.

3.3.2.1 Potlačení šumu

V úloze pracujeme se snímky pořízenými kamerou mobilního telefonu. Jedná se o digitální kameru, v drtivé většině se snímačem CMOS. Zatímco některé druhy šumu se ve snímcích vyskytují vždy, s jinými se na těchto snímcích s velkou pravděpodobností nesetkáme. Redukce šumu je důležitá pro většinu metod segmentace. Šum může například způsobit výskyt falešných hran při detekci hran nebo prahování.

První druh šumu, který můžeme vyloučit, je šum typu moiré. Mřížka křížovky ani text neobsahují dostatečně malé detaily na to, aby způsobili tento druh šumu. Pokud by přesto v obraze tento šum vyskytoval, bude se vyskytovat pouze v okolí křížovky nebo v ilustraci uvnitř křížovky, a tedy neovlivní proces segmentace políček křížovky.

Výskyt impulzního šumu a periodického šumu je také velmi nepravděpodobný. Ve většině případů budou aplikaci předloženy fotografie pořízené stejným zařízením, na kterém se aplikace nachází. Nedochází tedy k žádnému přenosu dat, při kterém by mohlo dojít ke ztrátě informace způsobující impulzní šum nebo interferenci způsobující periodický šum. Více pravděpodobný je jen impulzní šum vlivem selhání jednoho nebo více pixelů ve snímači kamery mobilního telefonu.

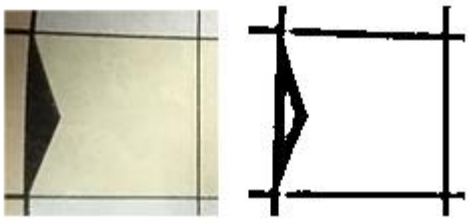
Fixní šum je přítomný vždy, je neměnný, a proto lehce kompenzovatelný. I v případě, že by nebyl kamerou kompenzován, projevuje se hlavně při dlouhé expozici snímků, která není nutná pro pořízení snímku křížovky na papíře.

Anizotropní šum je podobný Gaussovskému šumu, jelikož se oba v různé míře vyskytují ve snímku vždy a jejich efekt je silnější při špatném osvětlení, kdy dojde ke snížení odstupu signálu od šumu. Oba také postihují celý obraz a je možné je zredukovat průměrováním.

Na vstupní snímky proto ve svém řešení používám průměrování kvůli potlačení Gaussovského a anizotropního šumu. Průměrování je navíc schopné do určité míry potlačit i jiné druhy šumu v případě, že by přece jen došlo k jejich výskytu.

3.3.2.2 Normalizace rozlišení

Výsledek některých metod je také do značné míry ovlivněn rozlišením snímku. Zatímco některé metody mohou na snímku jiného rozlišení, než pro které byly nastaveny, podávat horší výsledky, jiné mohou i kompletně selhat. Například adaptivní prahování, pokud pracuje s okolím větším, než je tloušťka mřížky křížovky, vytváří v místech, kde je na obraze mřížka, jednu silnou hranu. V případě většího rozlišení obrazu, než pro které bylo okolí nastaveno, se prostředek úsečky mřížky jeví jako plochá oblast a dynamický práh adaptivního prahování klesne na úroveň srovnatelnou s jasem pixelů mřížky. Výsledkem je tak zdvojení mřížky křížovky (viz Obrázek 3.5).



Obrázek 3.5: Efekt tloušťky čáry na adaptivní prahování.

Dalším problémem je také čas zpracování a dostupná paměť zařízení. Zpracování snímků s vysokým rozlišením trvá podstatně déle a vyžaduje více paměti. Obojí může být problém, pokud chceme snímek zpracovávat na mobilním zařízení.

Z těchto důvodů je výhodné vstupní snímek zmenšit na určité rozlišení a nastavit parametry algoritmů zpracovávajících obraz s ohledem na toto rozlišení.

3.3.2.3 Další zpracování

Dle potřeb vybraných metod může být nutné nebo výhodné další předzpracování obrazu. Vstupní snímek bude s velkou pravděpodobností barevný, ale velká část metod segmentace informaci o barvě nepotřebuje. Proto je vhodné převést barevný obraz na šedotónový, čímž se sníží množství potřebné paměti a zrychlí operace s obrazem.

Některé metody, jako je metoda rozvodí, pracují lépe s parametrickým obrazem, jako je výsledek konvoluce s operátory aproximující první nebo druhou derivaci obrazu a profitují z redukce množství úrovní jasových hodnot (7 stránky 83, 85).

Další předzpracování tedy závisí čistě na zvolených metodách. V případě mého řešení má smysl převést obraz na šedotónový, jelikož je informace o barvě pro segmentaci neznámé křížovky těžko využitelná. Obraz je navíc invertován, pokud je detekována křížovka se světlou mřížkou na tmavém pozadí. Adaptivní prahování lze

v mém případě považovat za předzpracování, jelikož segmentaci provádí až Houghova transformace a detekce významných bodů křížovou maskou.

3.4 Rozpoznání typu políček švédské křížovky

Známe-li polohu políček křížovky, zbývá rozpoznat jejich obsah. Jednotlivé druhy políček je možné rozeznat na základě vlastností jejich obsahu, které jsou pro ně specifické. Těmito vlastnostmi jsou barva, množství hranových pixelů (text, obrazy) a poloha v mřížce křížovky.

3.4.1 Klasifikace políček podle barvy

Barva je pro rozpoznání druhu políčka užitečná, přestože předem neznáme, jaké barvy křížovka používá. Pokud provedeme analýzu barvy pozadí políček a rozdělíme je do skupin na základě podobnosti barvy, můžeme podle velikosti jednotlivých skupin tyto skupiny zařadit k určitým typům políček.

Většina políček křížovky jsou vpisovací políčka, ta největší skupina tedy pravděpodobně obsahuje vpisovací políčka. Zbylé skupiny tedy mohou obsahovat políčka legendy, políčka tajenky anebo slepá políčka. Políčka legendy mají občas stejnou barvu pozadí jako vpisovací políčka, na jiných křížovkách zase sdílí barvu s tajenkou nebo mají svoji vlastní barvu políčka. Tajenka má často svoji vlastní barvu, ale může sdílet barvu s vpisovacími políčky a být odlišena jiným způsobem (zakroužkování apod.). Barva slepých políček je stabilní nejméně, tato políčka často sdílí barvu s legendou, ale také často obsahuje obrázky, a tedy nemá jednu dominantní barvu pozadí.

Pro určení barvy pozadí políčka je vhodné převést analyzovaný výřez obrazu do barevného modelu, který umožňuje lépe separovat barevnou informaci od jasové informace než model RGB. Takovým modelem je například barevný model HSV nebo CIELAB, které mají vyhrazený samostatný kanál pro jas. To nám umožní potlačit vliv případného nerovnoměrného osvětlení na analýzu barev.

V některých případech však není možné rozpoznat políčka na základě barvy, protože je křížovka tištěna pouze v odstínech šedi a tajenka je odlišena od ostatních políček jiným odstínem šedi, než jaký mají ostatní políčka. V tomto případě je nutné brát v úvahu informaci o jasu, která je však nespolehlivá a může snadno způsobit špatnou klasifikaci políčka.

Ve svém řešení používám ke klasifikaci políček informaci o barvě získanou v barevném modelu CIELAB, který jsem zvolil kvůli jeho zaměření na napodobení lidského posuzování barvy a percepční uniformitě. Tyto vlastnosti jsou výhodné vzhledem k tomu, že barevné odlišení políček křížovek je voleno s ohledem na

snadné rozlišení člověkem. V řešení je použita kromě barvy také informace o jasu políček v případě, kdy analýza barev rozliší pouze jednu velkou barevnou skupinu, což naznačuje křížovku tištěnou v odstínech šedi.

3.4.2 Klasifikace políček podle hranové reprezentace

Dalším způsobem, jak odlišit typy políček je podle objektů, které obsahují. Ať už se bude jednat o text nebo o obrázek, obojí se projeví v hranové reprezentaci políčka. Takto je možné odlišit políčka legendy a slepá políčka od políček vpisovacích a tajenkových.

Samozřejmě, vpisovací a tajenková políčka nemusí být vždy prázdná. V případě, kdy by byla křížovka již vyplněná, nastává problém s tím, že vpisovací políčka mohou být klasifikována jako políčka legendy. Je tedy potřeba odlišit jedno ručně psané písmeno od většího množství tištěných písmen. Možností existuje více, jedno písmeno je obvykle vyšší než širší, zatímco legenda je naopak širší než vyšší. Jedno velké písmeno má také menší hustotu pixelů než celé slovo a detektor významných bodů by na jednom písmenu nalezl podstatně méně bodů než na jednom či více slovech.

Tajenkové políčko bývá také občas odlišeno zakroužkováním nebo silnou šipkou. Obojí je možné vcelku snadno detekovat, například Houghovou transformací pro kružnice a segmentováním oblasti a analýzou tvaru. Je ale také možné detekovat kružnice nebo trojúhelníky ve slepých políčkách, pokud obsahují obrazec s těmito prvky. Proto je potřeba nespoléhat se na výsledek detekce a kontrolovat ho.

V případě rozlišení políčka legendy od slepého políčka obsahujícího obraz nebo část obrazu je potřeba přistoupit k rozpoznání znaků. Jelikož je součástí práce také rozpoznání textu z obrazu a jeho použití pro hledání řešení křížovky, používám ve svém řešení k rozpoznání políček legendy přímo výstup ze systému optického rozpoznávání znaků (OCR z anglického optical character recognition). Pokud je v políčku rozpoznán dostatečně velký počet znaků, je pravděpodobné, že se jedná o políčko legendy.

3.4.3 Klasifikace políček podle polohy v mřížce

Posledním způsobem, jak rozpoznat typ políčka křížovky je podle jejich polohy v mřížce. Pro určité typy políček mají smysl jen určité polohy v křížovce. Díky tomu můžeme některé typy políček v určitých částech křížovky vyloučit a tím si zjednodušit rozhodování o správném typu.

O políčkách legendy víme, že se jejich legenda vztahuje pouze pro vpisovací políčka vpravo od políčka legendy, popřípadě vpravo a dolů pokud je políčko legendy dvojité. Proto je nesmysl detekovat v nejspodnějším řádku křížovky dvě

políčka legendy vedle sebe nebo v nejpravějším sloupci dvě políčka legendy nad sebou. V obou případech je jedno z těchto políček zbytečné, jelikož nenavazuje na žádná vpisovací políčka.

Obdobně vpisovací políčka musí být vždy alespoň dvě, jelikož dle směrnice má mít řešení legendy minimální délku dvou znaků. Tajenka, pokud tvoří řadu, by neměla být přerušena vpisovacím políčkem. A políčko, které je obklopeno pouze políčky legendy anebo slepými políčky je slepým políčkem.

3.5 Nápověda řešení švédské křížovky

Poslední částí úlohy je nalézt řešení legendy křížovky. K tomu je zapotřebí text legendy přečíst za pomocí systému OCR a nalézt k němu nejbližší záznam v křížovkářském slovníku.

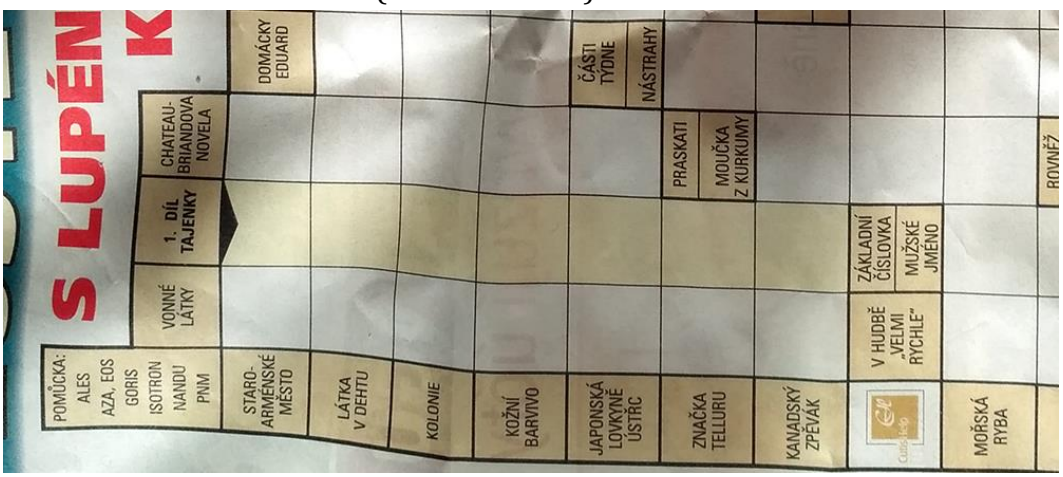
K rozpoznání textu legendy napomáhá skutečnost, že jde o tištěné písmo. Přestože není směrnicí preferován jeden určitý typ písma, vzhled písma mezi různými křížovkami se příliš neliší. Od OCR je tedy vyžadováno, aby dokázalo zpracovat tištěný text v češtině a bylo použitelné na zařízení s operačním systémem Android. V ideálním případě by OCR mělo být schopno rozpoznat i ručně psaná písmena u již vyplněných křížovek.

Křížovkářský slovník obsahuje záznamy textů legendy a jejich řešení. Máme na výběr dvě možnosti, buď hledat řešení v databázi nacházející se na zařízení, nebo použít databázi na serveru. Použití databáze na serveru vyžaduje připojení k internetu a zpracování dotazu může trvat delší nebo kratší dobu než u lokální databáze, v závislosti na rozsáhlosti databáze a způsobu vyhledávání. Rozdíl je také v nutnosti databázi distribuovat spolu s aplikací.

Protože výstup OCR systému nemusí být stoprocentně přesný a hesla v databázi nemusí přesně odpovídat legendě křížovky, je lepší hledat heslo, které je podobné hledané legendě než hledat kompletní shodu. Počítat míru podobnosti mezi každým heslem v databázi a legendou by však bylo příliš časově náročné. Rozsah možných odpovědí je však možné omezit výběrem pouze těch hesel, jejichž řešení má tolik znaků, kolik má křížovka pro danou legendu políček. Nesmíme také zapomenout na to, že v českém jazyce se písmeno “ch“ považuje za jeden znak, zatímco v kódu programu se pracuje se dvěma znaky.

4 ŘEŠENÍ ÚLOHY

Tato kapitola popisuje můj způsob řešení zadané úlohy, jehož výsledkem je aplikace pro operační systém Android. Struktura popisu řešení sleduje cestu vstupního snímku programem od předzpracování až po zobrazení digitální rekonstruované křížovky a hledání nápovědy v databázi. Jsou zde popsány použité metody i mnou vytvořené algoritmy. Pro ilustraci jednotlivých částí segmentace je použit výřez jednoho z testovacích snímků (viz Obrázek 4.1).



Obrázek 4.1: Výřez snímku trojúhelníkové křížovky.

4.1 Specifikace aplikace

Aplikace byla vytvořena pro operační systém Android verze 4.0.3 a novější (minimální verze SDK 15). Aplikace v sobě obsahuje knihovnu OpenCV 3.2.0 pro Android, knihovnu Simmetrics verze 1.6.2 a projekt tess-two verze 6.2.0, který obsahuje OCR Tesseract verze 3.04.01 a knihovny Leptonica verze 1.74.1, libjpeg verze 9b a libpng verze 1.6.25. Aplikace ke své činnosti vyžaduje externí trénovací data "ces.traineddata" pro OCR Tesseract. Aplikace vyhledává řešení k legendě v externí SQLite databázi "kriz_slov.sqlite3", pokud existuje, obsahující hesla, jejich řešení a počet znaků řešení. Za účelem otestování aplikace byla vytvořena databáze s využitím slovníku obsaženém v programu VKS 4.0 (19).

Aplikace je napsána v jazyce Java a používá nástroje Android NDK k volání nativních funkcí v jazyce C++, které provádí většinu zpracování obrazu. Java část aplikace se stará o komunikaci s uživatelem, obsluhu OCR a hledání výrazů v databázi.

4.2 Předzpracování

Aplikace umožňuje uživateli prohlédnout si obsah paměti zařízení a vybrat soubor ke zpracování. Podporované formáty souboru obrazu jsou JPEG, PNG a BMP. Vybraný barevný obraz je načten do paměti knihovnou OpenCV a předán nativní části programu, která nad načteným obrazem provede předzpracování za účelem získání hranové reprezentace obrazu.

Jako první dojde k normalizaci rozlišení obrazu na maximální velikost 4 Mpx. Program vypočte předpokládaný počet sloupců obrazu jako

$$expectedSizeCols = 4\,000\,000/počet\ řádků\ obrazu \tag{4.1}$$

Pokud je tento předpokládaný počet sloupců nižší než skutečný počet sloupců obrazu, je obraz zmenšen na násobek své původní velikosti OpenCV funkcí **resize** s interpolací INTER_AREA, která je vhodná pro zmenšování obrazů a zabraňuje vzniku šumu typu moiré (20). Měřítko *scale* pro funkci **resize** je vypočteno jako

$$scale = \sqrt{expectedSizeCols/počet\ sloupců\ obrazu} \tag{4.2}$$

Odmocnina v rovnici 4.2 je nutná, protože *scale* vstupuje do výpočtu dvakrát, jednou pro zmenšení výšky obrazu a podruhé pro zmenšení šířky obrazu. Tím je počet pixelů obrazu zmenšen na $scale^2$ původního počtu pixelů. Rozlišení 4 Mpx bylo zvoleno na základě zkušenosti za účelem zrychlení zpracování obrazu a vyřešení problému se závislostí některých metod na rozlišení obrazu.

Normalizovaný barevný obraz je následně převeden na šedotónový pomocí OpenCV funkce **cvtColor**. Přestože **cvtColor** používá pro výpočet jasu součinitele odpovídající barevnému prostoru NTSC RGB (20) a výpočet je tedy pro obraz v jiném barevném prostoru nesprávný, je hlavní rozdíl v celkovém jasu obrazu a relativní rozdíl mezi sousedními hodnotami jasu zůstává téměř stejný (viz Tabulka 4.1). Vliv na adaptivní prahování je tak zanedbatelný, a proto jsem se rozhodl tento výpočet ponechat za účelem rychlejšího zpracování obrazu.

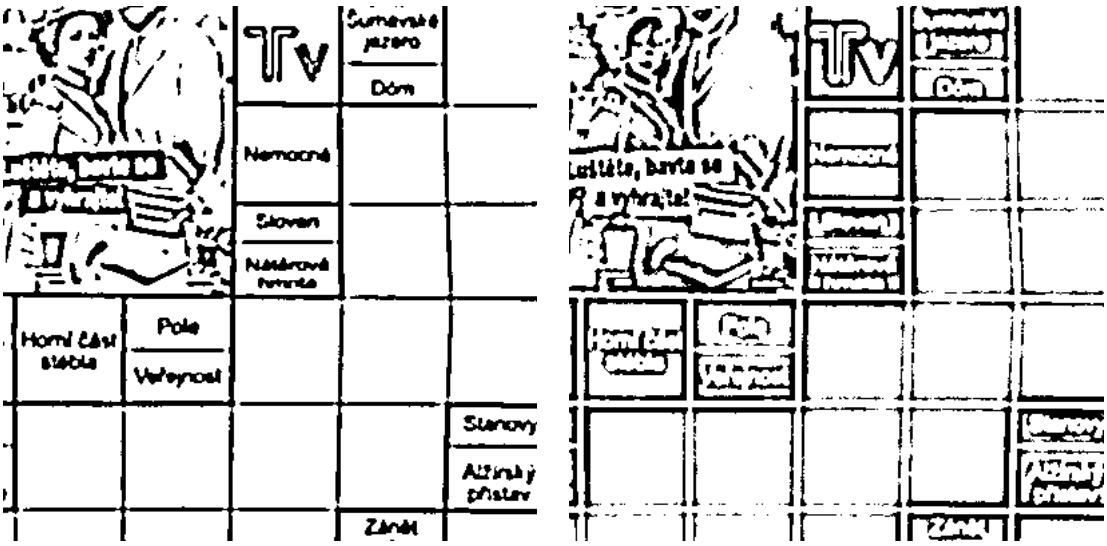
Tabulka 4.1: Intenzita pixelů v profilu hrany sRGB obrazu.

R	164	157	123	79	64	62	90	129	148	149
G	164	157	123	79	64	62	90	129	148	148
B	154	147	111	67	52	50	80	119	140	143
NTSC	162,9	155,9	121,6	77,6	62,6	60,6	88,9	127,9	147,1	147,7
sRGB	163,3	156,3	122,1	78,1	63,1	61,1	89,3	128,3	147,4	147,9
Adobe	163,2	156,2	122,1	78,1	63,1	61,1	89,2	128,2	147,4	147,9

Šum v obraze je potlačen pomocí Gaussovského rozostření. Velikost masky byla zvolena na základě zkušenosti za předpokladu maximálního rozlišení 4 Mpx.

Rozostření kromě potlačování šumu také rozmazává hrany, proto má příliš velká maska negativní vliv na výslednou hranovou reprezentaci. Některé metody, jako je bilaterální filtr, zachovávají ostré hrany. Oproti běžnému Gaussovskému rozostření se však jedná jen o malé zlepšení za cenu podstatně náročnějšího výpočtu. Dále jsem zjistil, že potlačení šumu před převodem na šedotónový obraz a před zmenšením obrazu dává mírně lepší výsledek, ale stejně jako v případě bilaterálního filtru má toto zlepšení za cenu podstatně delší zpracování snímku.

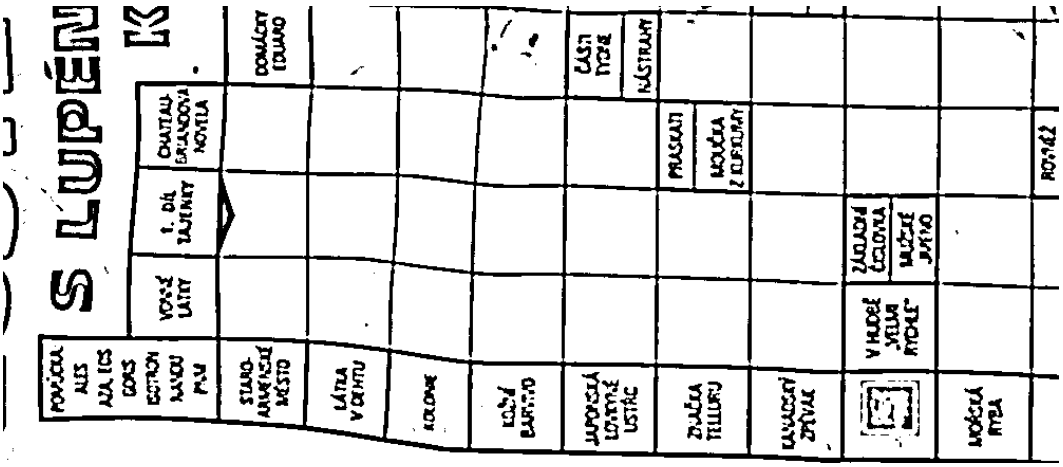
Adaptivní prahování obrazu je nastaveno pro nejčastější případ, kterým je tmavá mřížka křížovky na světlém pozadí. V případě světlé mřížky na tmavém pozadí by takto nastavené adaptivní prahování vyprodukovalo dvojitou hranu místo jedné silné hrany (viz Obrázek 4.2). Této skutečnosti je využito k detekci potřeby invertovat obraz před adaptivním prahováním. Z prostředku obrazu je vyříznut čtverec o straně 400 pixelů, který je převeden na binární obraz adaptivním prahováním, původní výřez je invertován a převeden na binární obraz znovu. Jelikož adaptivní prahování vytváří na světlé mřížce dvojitou hranu, má výřez se světlou mřížkou větší počet nenulových pixelů než výřez s tmavou mřížkou. Pokud má tedy neinvertovaný výřez větší počet nenulových pixelů, je nutné obraz invertovat.



Obrázek 4.2: Výsledek prahování správně a nesprávně invertovaného obrazu.

Nakonec je třeba šedotónový obraz převést na binární adaptivním prahováním. OpenCV funkce **adaptiveThreshold** používá k určení prahu vzájemnou korelaci s Gaussovým oknem (20) a zbylé parametry jsou nastaveny na základě zkušenosti.

Výsledkem předzpracování je hrubá hranová reprezentace vstupního obrazu a koeficient *scale*, který značí o kolik je hrubá hranová reprezentace zmenšena oproti původnímu obrazu.



Obrázek 4.3: Výsledek adaptivního prahování výřezu.

4.3 Segmentace políček

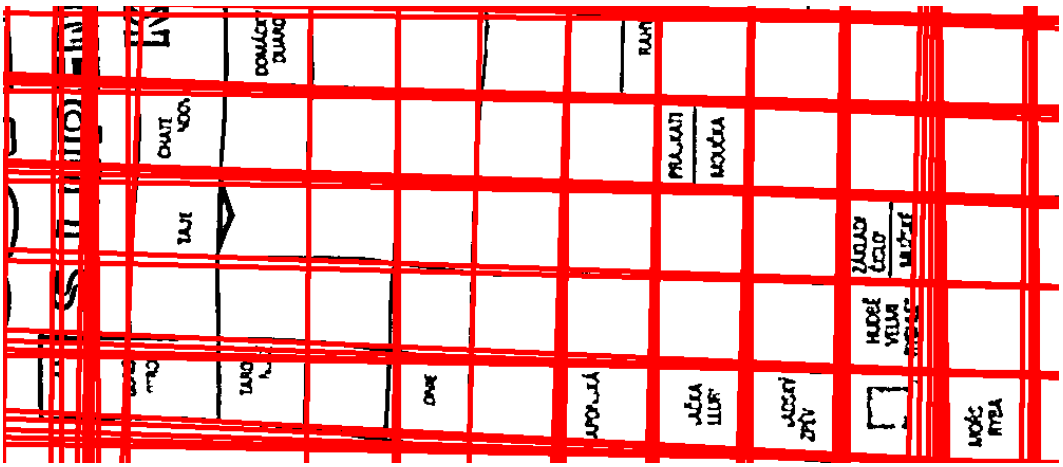
Segmentační část programu má za úkol nalézt polohu jednotlivých políček nalezením polohy jejich rohů v obraze. Výstupem je množina bodů v obraze a pole, ve kterém jsou tyto body uspořádány do mřížky odpovídající skutečné mřížce křížovky.

4.3.1 Hledání přímek v obraze

Aby se snížilo množství falešných přímek, které by Houghova transformace nalezla, jsou z obrazu nejprve odstraněny všechny malé objekty, jako jsou písmena textu nebo šum. Toho je dosaženo získáním minimálního opsaného obdélníku objektu pomocí záplavového narůstání oblastí (OpenCV funkce **floodFill**). Pokud jsou obě strany obdélníku menší než definovaný práh, je objekt odstraněn. Hodnota prahu byla určena podle rozměrů textu na testovacích snímcích.

Na vyčištěném obraze jsou nalezeny přímky pomocí Houghovy transformace. Použitá Houghova transformace je mírně upravená verze OpenCV funkce **HoughLines**. Upravená verze pracuje totožně, pouze vrací navíc informaci o počtu hlasů, které přímky dostaly v parametrickém prostoru. Funkce vrací všechny přímky, jejichž počet hlasů je vyšší než stanovený práh. Práh byl stanoven odhadem na 500 hlasů. Přestože přímky obdélníkové křížovky mohou snadno dosáhnout i trojnásobného počtu hlasů, kdyby byl práh vyšší, došlo by k problémům u křížovky trojúhelníkového typu, kde by chyběly přímky u rohů křížovky.

V případě, že Houghova transformace nenalezne žádné přímky, je zpracování obrazu ukončeno.



Obrázek 4.4: Přímky nalezené ve vyčištěném výřezu pomocí Houghovy transformace.

Z nalezených přímek je dále určen jejich dominantní úhel. Ten je určen rozdělením přímek do skupin na základě jejich úhlu. Skupina je určena průměrným úhlem vypočteným z úhlů členů skupiny, kteří mají všichni přibližně stejný úhel. Nejpočetnější skupina je poté vybrána jako dominantní úhel. Kvůli nízkému prahu Houghovy transformace je mezi přímkami mnoho přímek falešných, proto se při hledání dominantního úhlu pracuje jen s 10 procenty přímek, a to těch s největším počtem hlasů z Houghovy transformace.

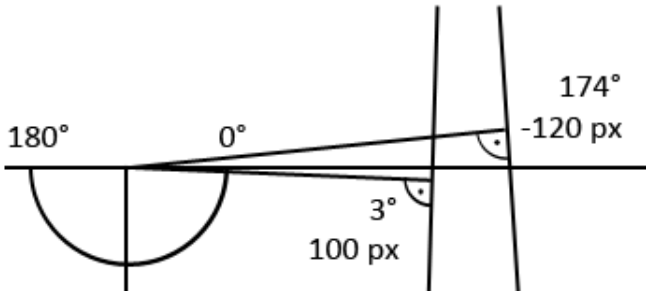
4.3.2 Filtrace přímek

Dominantní úhel přímek odpovídá jednomu ze dvou úhlů, které mají čáry tvořící mřížku křížovky na obrazu, jelikož byly k jeho určení použity ty nejvýraznější přímky, které s velkou pravděpodobností odpovídají mřížce. Prvním krokem filtrace tedy bude odstranění přímek, které dominantnímu úhlu, nebo úhlu na něj kolmému, neodpovídají.

Z vektoru přímek jsou tedy odstraněny přímky, jejichž úhel se od obou požadovaných úhlů liší více než daná tolerance. Je tady však problém, který vychází ze způsobu, jak jsou přímky z Houghovy transformace vyjádřeny. Tyto přímky používají polární souřadnice s počátkem v levém horním rohu obrazu a úhlem od nuly do 180 stupňů. Tedy dvě skoro kolmé přímky mohou mít v polárních souřadnicích úhly 1 a 179 stupňů a jsou tedy daleko od sebe. Pokud by byl dominantní úhel 0 stupňů, přímka s úhlem 179 stupňů by byla odstraněna, přestože by měla správně být zachována. Souřadnice vzdálenosti od počátku navíc může být u přímek s úhlem vyšším než 90 stupňů záporná, což je problém při hledání přímek, které leží blízko u sebe (viz Obrázek 4.5).

Proto pokud je dominantní úhel nebo úhel na něj kolmý blízko této nespojitosti, jsou přímky, které odpovídají tomuto úhlu a mají úhel vyšší než 90

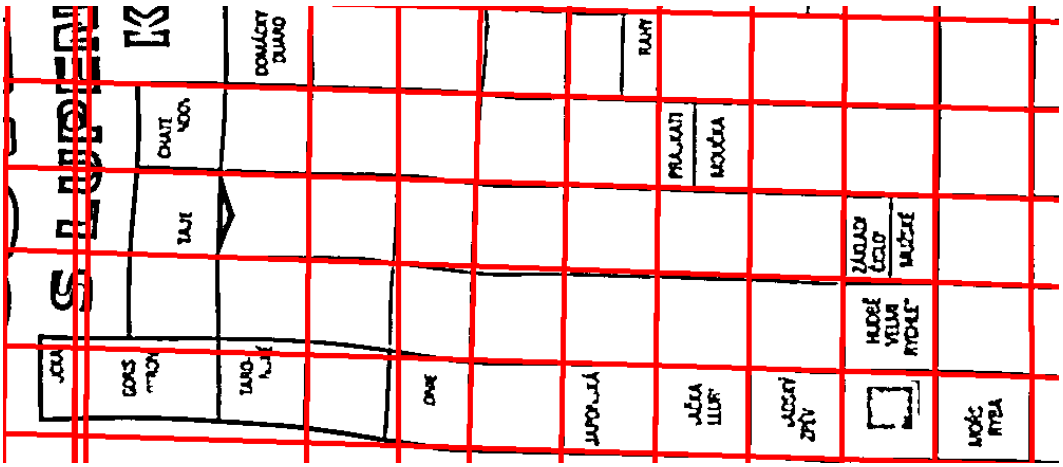
stupňů, normalizovány negací souřadnice vzdálenosti a odečtením 180 od souřadnice úhlu.



Obrázek 4.5: Nespojitost v souřadnicovém systému přímek Houghovy transformace.

Filtrací tedy projdou dvě skupiny přímek. Pokud jedna ze skupin neobsahuje žádné přímky, znamená to, že mřížka nebyla nalezena a segmentace je ukončena. Pro obě skupiny proběhne samostatně slučování přímek, které se nachází blízko u sebe.

Při slučování jsou postupně vybírány nesloučené přímky od přímky s nejvíce hlasy až po tu s nejméně hlasy. Pro každou vybranou přímku jsou nalezeny přímky s podobnou vzdáleností od počátku a jsou sloučeny do jedné přímky. Tato přímka má souřadnice rovny váženému průměru sloučených přímek, kde vahou je počet hlasů, a její počet hlasů je roven nejvyššímu počtu hlasů mezi sloučenými přímkami.



Obrázek 4.6: Přímky zbývající ve výřezu po filtraci.

Tento odhad se provádí samostatně pro obě skupiny na sebe kolmých přímek. Získáme tedy odhad šířky a výšky políčka, které by měly být přibližně stejně velké. V případě, že je mezi oběma odhady příliš velký rozdíl, znamená to, že jeden z těchto odhadů je nesprávný. V tom případě se provede výpočet velikosti políčka znovu, tentokrát s oběma skupinami přímek najednou a touto hodnotou je nahrazen chybný odhad.

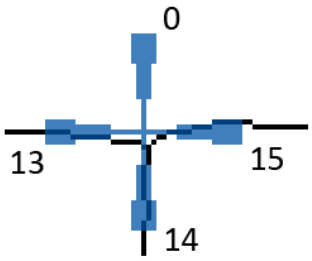
4.3.3 Hledání významných bodů v obraze

Nalezení rohů políček křížovky je provedeno hledáním významných bodů v původním obraze hran (před odstraněním malých objektů) pomocí křížové masky, kterou jsou detekovány rohy a průsečíky. Aby nebylo nutné masku otáčet je mřížka zarovnána se souřadnicovými osami obrazu otočením obrazu o úhel 90 stupňů mínus dominantní úhel. Protože se jedná o binární obraz, je při otáčení použita interpolace nejbližším sousedem.

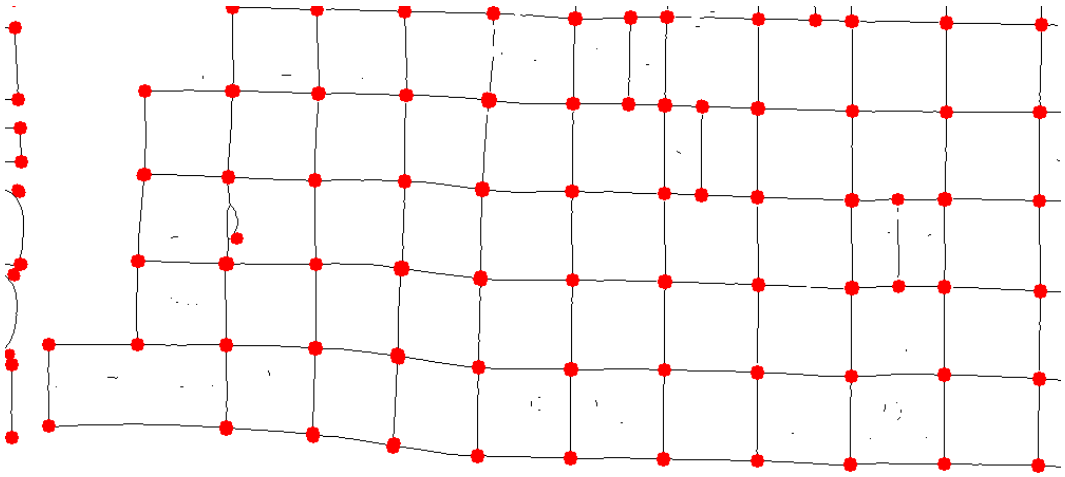
Z otočeného obrazu jsou znovu odstraněny malé objekty na základě rozměrů jejich opsaného obdélníku. Jsou odstraněny všechny objekty menší, než je velikost políčka a zároveň větší než 6 pixelů. Přidáním spodní hranice se zachovají dlouhé, ale přitom úzké objekty jako jsou například přerušené části mřížky.

Na výsledný obraz je dále použita operace morfologického otevření k odstranění velmi malých objektů vzniklých šumem, které byly při odstraňování malých objektů ignorovány. Nakonec jsou všechny čáry v obraze ztenčeny na tloušťku 1 pixelu operací morfologického ztenčování metodou Guo-Hall. V programu je použita mírně upravená implementace této metody v jazyce C++, jejímž autorem je Nash (21).

Jakmile je obraz připraven, vyhledají se v něm významné body. K hledání významných bodů je využita křížová maska (viz Obrázek 4.7). Ramena křížové masky se od středu rozšiřují, aby částečně kompenzovaly deformaci mřížky. Délka ramene je určena jako pětina velikosti políčka a maximální šířka jako čtvrtina délky. Protože je maska i hranová reprezentace obrazu binární, konvoluce ramen masky s obrazem spočítá v každé možné poloze počet hranových bodů pod ramenem masky. Pro každý bod obrazu je tedy známo množství hranových bodů v jeho okolí. Jako významné body jsou vybrány všechny body, které mají alespoň ve dvou ramenech masky větší hodnotu, než je šířka ramena masky. Pokud má bod jen dvě ramena masky s velkou hodnotou, ramena nesmí být protilehlá. Tímto způsobem získáme body odpovídající dostatečně výrazným rohům a průsečíkům v obraze.



Obrázek 4.7: Konvoluce s křížovou maskou.

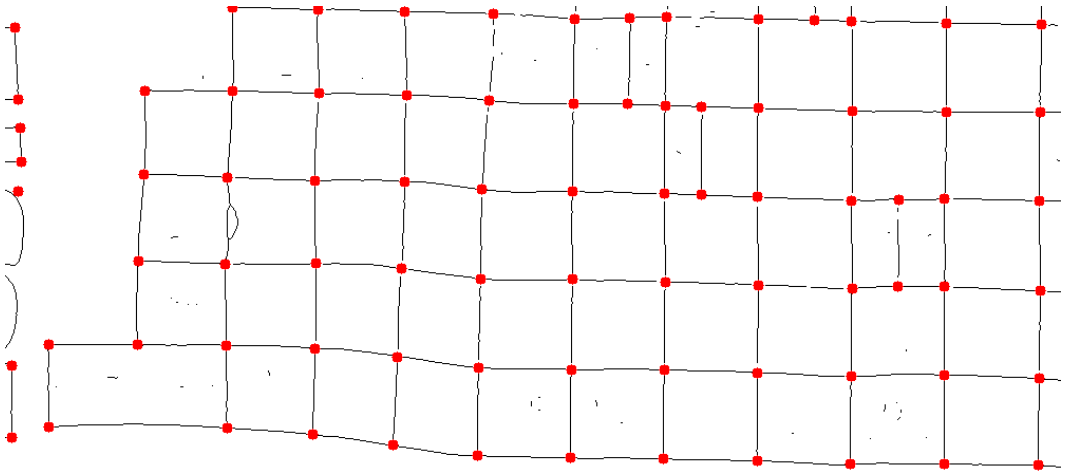


Obrázek 4.8: Významné body na výřezu ztenčeném metodou Guo-Hall.

4.3.4 Filtrace významných bodů

Mezi nalezenými body jsou kromě hledaných rohů políček také body odpovídající rohům na různých objektech v okolí křížovky. Tyto nepotřebné body je potřeba rozpoznat a odstranit je.

Nejprve dojde ke snížení počtu nalezených bodů jejich slučováním s body ve svém okolí. Slučování se opakuje, dokud se mění počet významných bodů. Slučování probíhá výběrem dosud nesloučených bodů podle součtu hodnot ramen jejich masky, počínaje bodem s největším součtem, a jejich sloučením s body v okolí. Při slučování jsou z okolí vybrány body se stejnou hodnotou součtu ramen masky jako má bod, pro který bylo slučování spuštěno, a je nalezen střed výpočtem průměru souřadnic vybraných bodů (včetně spouštěcího bodu). Z těchto bodů je pak vybrán bod, který je nejblíže vypočtenému středu. Ten nahradí všechny body v okolí.



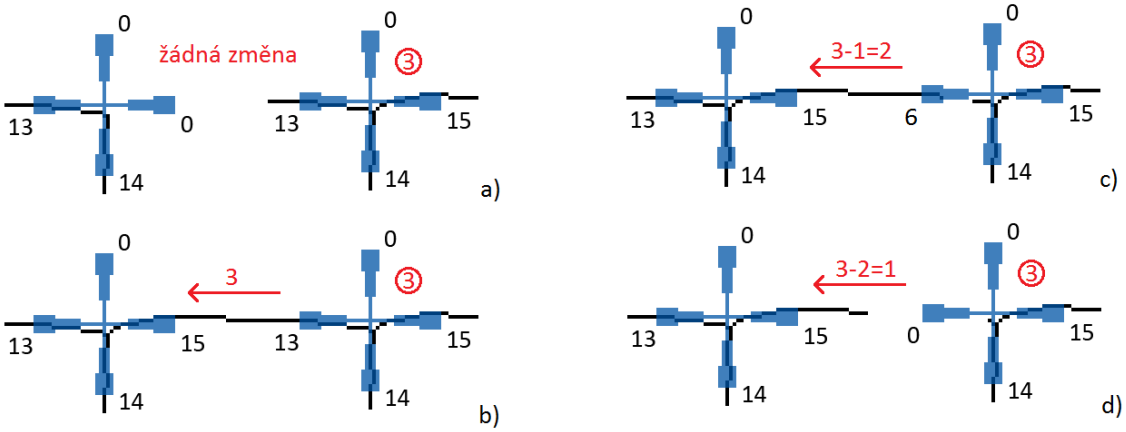
Obrázek 4.9: Významné body zbývající po slučování.

Ze sloučených bodů se následně sestaví mřížka. Jako počáteční bod je vybrán bod s největším součtem ramen masky a ve vzdálenosti přibližně odpovídající velikosti políčka, která byla zjištěna z přímek Houghovy transformace, jsou hledány body nad, pod, vlevo a vpravo od vybraného bodu. Nalezené body jsou přidány do 2D pole představujícího mřížku a slouží jako počáteční body pro další iteraci algoritmu. Skládání bodů do mřížky také zajišťuje, že algoritmus nenajde více než jeden bod pro každý z průsečíků mřížky. Na základě vybraných bodů je upřesňován odhad velikosti políčka. Všechny ostatní body, které nebyly vybrány při vytváření mřížky, jsou následně odstraněny.

U takto filtrovaných bodů není ještě zaručeno, že obsahují pouze body odpovídající mřížce křížovky. Mohou se zde vyskytovat falešné body, které ke mřížce nepatří, ale jsou jen ve vhodné vzdálenosti od ostatních. Proto je zvolen bod, u kterého se bude předpokládat, že je součástí mřížky, a bude se od něj kontrolovat vazba ostatních bodů. Zvoleným bodem je ten, který má největších součet ramen své masky a masek čtyř svých sousedních bodů. Tomuto bodu je přiřazeno nejvyšší skóre 3 a zbylé body začínají s nulovým skóre. Algoritmus přiřazuje sousedům počátečního bodu skóre v závislosti na síle vazby mezi počátečním bodem a daným sousedem. Algoritmus se poté opakuje pro zpracované sousední body a jejich sousedy a tak dále. Aktivní bod přiřazuje sousednímu bodu skóre podle následujících pravidel:

1. Pokud je hodnota ramene masky sousedního bodu směrem k aktivnímu bodu menší než šířka ramene masky, skóre se nemění (viz Obrázek 4.10a).
2. Pokud je hodnota ramene masky aktivního bodu směrem k sousedovi větší než polovina délky ramene masky, skóre souseda je rovno skóre aktivního bodu (viz Obrázek 4.10b).
3. Pokud je hodnota ramene masky aktivního bodu směrem k sousedovi větší než šířka ramene masky a skóre souseda je nižší než skóre aktivního bodu mínus 1, je skóre souseda rovno skóre aktivního bodu mínus 1 (viz Obrázek 4.10c).
4. Pokud je skóre souseda nižší než skóre aktivního bodu mínus 2, je skóre souseda rovno skóre aktivního bodu mínus 2 (viz Obrázek 4.10d).

Podmínky těchto pravidel se testují postupně od prvního po čtvrtý. Je-li některé z pravidel splněno, zbylá pravidla se už netestují. Z mřížky jsou poté odstraněny všechny body, které mají skóre nižší než 2 a také všechny body, které mají pouze jednoho souseda. Tím se odstraní body se slabou návazností na body mřížky a body, které z mřížky vyčnívají.



Obrázek 4.10: Šíření skóre mezi významnými body podle křížové masky.

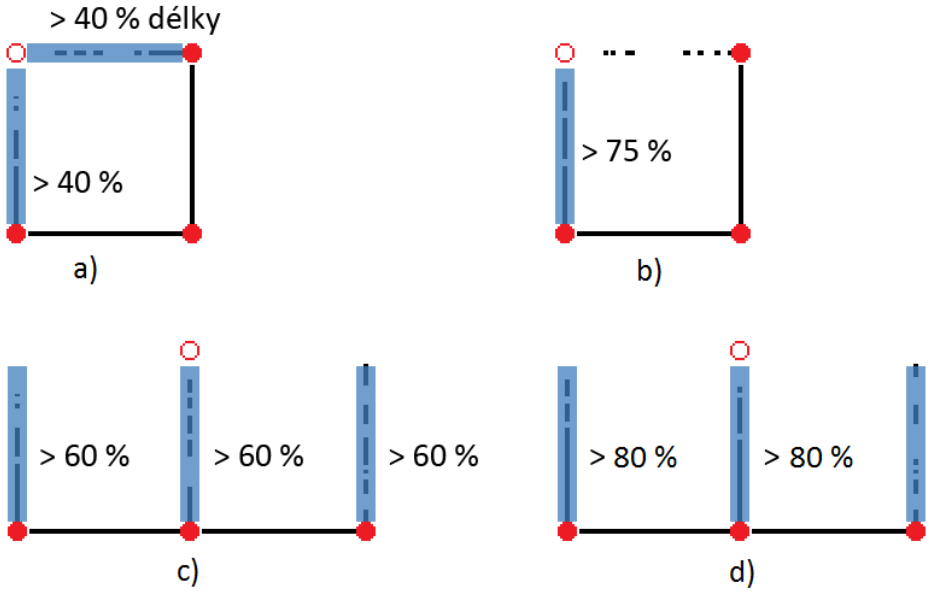
Tímto odstraňováním na základě informací z křížové masky však mohlo dojít k odstranění bodů, které do mřížky patří, ale mají ve svém blízkém okolí nevýrazné hrany. Proto se dodatečně kontrolují prázdná místa uvnitř a v okolí mřížky, jestli není potřeba na prázdné místo doplnit bod. Při této kontrole se bere v úvahu celý prostor mezi bodem mřížky a předpokládanou polohou nového bodu, nejen blízké okolí bodu.

Algoritmus opakovaně prochází všechny body mřížky, dokud dochází ke změnám mřížky. Ignorují se všechny body, které neleží na rozhraní mřížky a prázdného místa v mřížce. Zkoumá-li algoritmus prázdné místo, spočítá se počet hranových pixelů mezi prázdným místem a sousedními body mřížky konvolucí s obdélníkovou maskou délky odpovídající velikosti políčka a šířky 9 pixelů. Síla hrany mezi prázdným místem a bodem mřížky je určena poměrem počtu nalezených hranových pixelů a délky masky. Pokud k prázdnému místu vedou alespoň dvě hrany o síle větší než 40 % (viz Obrázek 4.11a), je na prázdné místo přidán bod mřížky. Pokud k prázdnému místu vede jedna hrana o síle větší než 75 % (viz Obrázek 4.11b), je také na prázdné místo přidán bod mřížky. Dochází tedy k zaplňování děr v mřížce.

Pokud algoritmus zkoumá bod mřížky sousedící s prázdným místem, spočítá hranové pixely mezi bodem mřížky a prázdným místem, ale spočítá také hranové pixely ve stejném směru pro levého a pravého souseda zkoumaného bodu mřížky. Pokud je síla hrany u všech tří bodů mřížky větší než 60 % (viz Obrázek 4.11c) je na prázdné místo přidán nový bod mřížky. Totéž se stane, pokud mají dva body mřížky sílu hrany větší než 80 % (viz Obrázek 4.11d). Tím dojde k expanzi mřížky v případě, kdy byla její část odstraněna.

Při vkládání nového bodů mřížky na prázdné místo je získán výřez hranové reprezentace na předpokládané pozici nového bodu (určené podle polohy sousedních bodů a velikosti políčka). Je provedena konvoluce výřezu s křížovou maskou a mezi body s nejlepším součtem hodnot ramen masky je hledán bod, který

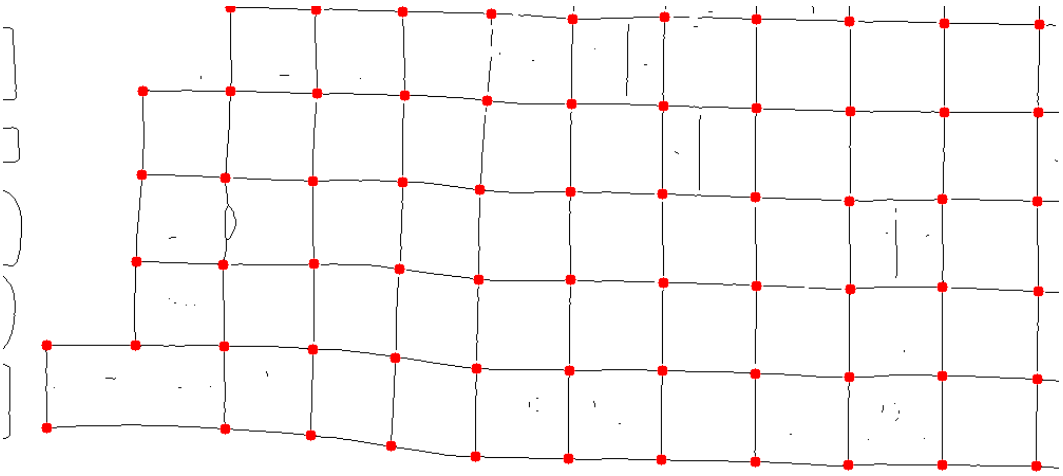
je nejblíž předpokládané pozici. Pokud není nalezen žádný vhodný bod, je nový bod mřížky umístěn přímo na předpokládanou pozici.



Obrázek 4.11: Doplnění chybějících bodů mřížky křížovky.

Nakonec jsou do mřížky přidány body na prázdná místa, která nejsou vzdálena od okraje mřížky dále než o čtyři místa. Je to z toho důvodu, že legenda musí mít řešení, které má alespoň dva znaky. Pokud je tedy na okraji křížovky políčko legendy, musí po něm následovat další dvě políčka, což odpovídá celkem čtyřem bodům, které jsou součástí těchto tří políček.

V poli bodů, které reprezentuje mřížku, zůstaly už jen body patřící k mřížce. Teoreticky nejmenší funkční křížovka má tři krát tři pole. Pokud je nalezená mřížka menší, rozpoznání mřížky selhalo. V opačném případě je sestaveno 2D pole, které místo významných bodů uspořádaných do mřížky obsahuje struktury představující políčka, uspořádané do mřížky.



Obrázek 4.12: Výsledné rohy políček křížovky.

4.4 Rozpoznání typu políček

K rozpoznání typu políček je použit původní barevný nezmenšený obraz. Poloha políček zjištěná ze segmentace křížovky však odpovídá zmenšené a otočené verzi obrazu. Proto je původní obraz otočen s využitím kubické interpolace a souřadnice nalezených bodů jsou poděleny měřítkem *scale* (Rov. 4.2). Následně je analyzováno každé nalezené políčko křížovky.

4.4.1 Analýza políček

Při analýze je z obrazu získán výřez políčka jako minimální opsaný obdélník obsahující všechny čtyři rohové body políčka. Kvůli deformacím mřížky neodpovídá opsaný obdélník zcela segmentovanému políčku a na okrajích výřezu se nachází nežádané části obrazu. Proto je vytvořena maska, která skryje část výřezu mimo čtyřúhelník tvořený rohy políčka. Z výřezu je vytvořena kopie, která se převede do odstínů šedi a filtruje se bilaterálním filtrem, který potlačí šum, ale zachová ostrost hran. Z šedotónového výřezu je vytvořena další kopie, která je převedena na binární obraz adaptivním prahováním. Vzhledem k tomu, že není předem známo rozlišení obrazu, se velikost použitého okolí u adaptivního prahování určí jako jedna dvanáctina délky strany výřezu.

Pokud je v binárním obraze nalezen dostatečně velké množství pixelů objektů, předpokládá se, že se v políčku nachází text a políčko je zpracováno OCR. Do OCR je předán šedotónová verze výřezu políčka, který je navíc zostřen odečtením rozostřeného obrazu, má odstraněny okraje v místech určených maskou a je zvětšen lineární interpolací na výšku 400 pixelů. Zvětšení je provedeno za účelem zlepšení výstupu OCR a velikost políčka byla zvolena tak, aby text u všech testovaných snímků dosáhl velikosti alespoň 10 bodů při rozlišení 300 DPI (obrazových bodů na palec) (22). OCR používá trénovací data pro český jazyk a je omezeno pouze na písmena a interpunkční znaménka běžně používané v českém jazyce.

První výřez s potenciálním textem je zpracován OCR čtyřikrát, jednou pro každou 90 stupňovou orientaci. Nejdelší výstup OCR je vybrán jako správné otočení křížovky, podle kterého jsou otočeny všechny následující výřezy vstupující do OCR a po rozpoznání políček i celá křížovka. U ostatních výřezů políček je také kontrolováno, jestli není políčko zdvojené. Ve vyprahovaném políčku je hledána čára, která rozděluje dvě legendy, podle součtu objektových bodů v řádku. Pokud je nalezen dostatečně velký součet, je v tomto místě políčko rozděleno a OCR tyto dvě části zpracovává samostatně.

Kromě rozpoznání textu jsou také vyhledány barvy v pozadí políčka. Při analýze barev jsou brány v úvahu pouze políčka, která nejsou pokryta vyprahovaným políčkem, které slouží jako maska. Tímto způsobem jsou zanedbány

okraje a případný text v políčku. Barevný výřez políčka je zprůměrován a převeden do barevného systému CIELAB. V políčku jsou vyhledávány barvy podle složek a* a b* jako průměr pixelů s podobnou barvou.

Nakonec pokud OCR nalezlo v políčku méně než 4 znaky, znamená to, že toto políčko může být tajenkovým políčkem, a proto je provedena detekce nebarevného označení. Takovým označením může být zakroužkování políčka, proto je provedena detekce kružnic Houghovou transformací. Druhým častým typem označení je velká trojúhelníková šipka. Ta je detekována hledáním kontur OpenCV funkcí **findContours**. Nalezené kontur jsou aproximovány funkcí **approxPolyDP**, čímž se minimalizuje množství bodů, které konturu tvoří. Tyto body se testují, jestli jsou pouze tři a jsou vzájemně ve správné poloze, aby tvořili trojúhelník požadovaného tvaru.

4.4.2 Rozpoznání políček

Rozpoznání typu políčka stojí na zjištěných informacích o barvě, textu a dalších objektech nacházejících se v jednotlivých políčcích. Pro rozpoznání políček legendy se algoritmus opírá hlavně o výstup OCR. Při rozpoznání tajenky se nejprve pracuje s barvou políček a pokud nelze tímto způsobem tajenku nalézt, jsou políčka tajenky rozpoznávána podle jasu nebo zvláštních značení.

Nejprve jsou políčka rozdělena do skupin na základě barvy a je rozpoznána nejpočetnější skupina. Políčka jsou přiřazována do skupin podle své dominantní barvy. Pokud je výsledkem pouze jedna skupina, znamená to, že všechna políčka mají stejnou barvu a musí být tedy rozdělena znovu, ale tentokrát podle jasu.

Jako první jsou rozpoznána vpisovací políčka a políčka legendy. Pokud políčko obsahuje text delší než 3 znaky, je pokládáno za políčko legendy. Naopak pokud políčko obsahuje málo objektových bodů (a tedy žádný text ani ilustraci) a jeho barva je shodná s dominantní barvou, je označeno jako políčko vpisovací.

V případě, že byla rozpoznána více než jedna barevná skupina, je možné, že jedna z těchto barev připadá na políčka tajenky. V případě, že po odstranění dominantní barvy a barev, které používají políčka legendy, zbude jen jedna barva, je jasné, že tato barva patří políčkům tajenky a všechna políčka této barvy jsou proto označena jako políčka tajenky.

Pokud nebyla políčka tajenky rozpoznána na základě barvy, pokusí se je program rozpoznat podle jiného značení. Podle toho, jestli políčko obsahuje kruh, je tajenka rozpoznána v případě, že bylo detekováno dostatečné množství políček s kruhem. Tím je zabráněno falešnému rozpoznání tajenky v případě, kdy slepá políčka obsahují ilustraci, která má pozitivní odezvu na detekci kruhů. V případě, kdy není tajenka rozpoznána na základě přítomnosti kruhů, program se ji pokusí rozpoznat podle šipek. Pokud nalezne políčko s šipkou, která směřuje dolů nebo

doprava, bude sledovat směr šipky a označovat jako tajenku všechna políčka, která přejde, až dokud nenarazí na slepé políčko, políčko legendy nebo okraj křížovky.

Pokud i tento způsob rozpoznání selže, zbývá jediná možnost a to ta, že tajenka je rozlišena pouze číslem v políčku. Proto je zkontrolován text ve všech nerozhodnutých políčcích, jestli obsahuje čísla. Pokud ano, je políčko označeno za políčko tajenky.

Zbývající nerozhodnutá políčka jsou označena jako políčka legendy, pokud obsahují více než jeden znak a mají stejnou barvu jako ostatní políčka legendy. Políčka, která zůstanou neoznačena jsou označena jako slepá, pokud obsahují velké množství barev nebo velké množství objektových pixelů, což je obojí znakem toho, že se v políčku pravděpodobně nachází obrázek. Všechna ostatní neoznačená políčka označena jako vpisovací. Nakonec dojde ke kontrole prázdných políček, jestli není některé z nich obklopeno pouze slepými políčky nebo políčky legendy, pokud ano je nahrazeno slepým políčkem, protože osamocené vpisovací políčko nemá v křížovce význam.

4.5 Nápověda k řešení křížovky

Poté, co je křížovka úspěšně rozpoznána, je vykreslena její kopie podle rozpoznaných typů políček. Pro zobrazení textu políček legendy není použit výstup OCR, ale výřezy z původního snímku. Tím je zabráněno zobrazení nesmyslného textu v případě jeho špatného rozpoznání systémem OCR. Vykreslenou křížovku může uživatel přiblížit nebo oddálit a posouvat po obrazovce. Může také zapisovat do políček a vyžádat si nápovědu k vybrané legendě.

Při hledání řešení legendy je použit text rozpoznaný pomocí OCR, který je očištěn od některých možných chyb jako jsou tečky nebo jiná interpunkční znaménka na začátku textu nebo v nadbytečném množství uvnitř textu. Řešení je hledání v SQLite databázi, která obsahuje hesla odpovídající legendám a jejich možná řešení.

Jakmile je vybráno políčko legendy, jsou napřed získána písmena jeho řešení, která byla do křížovky již zadána. V databázi jsou hledány všechny záznamy, které mají řešení s takovým počtem znaků, kolik je k dané legendě přidruženo políček. Výběr se dále zúží zanedbáním záznamů s heslem kratším, než je text legendy nebo s heslem, které neobsahuje písmeno, které se vyskytuje v textu legendy nejčastěji.

Mezi nalezenými záznamy je potřeba najít ty nejpodobnější textu legendy. Pro tento účel je určena míra podobnosti mezi legendou a heslem záznamu jako

$$sim(A, B) = 1 - \frac{dist(A, B)}{max(A, B)} \tag{4.3}$$

kde *dist*(*A*, *B*) je Levenshteinova vzdálenost mezi legendou a heslem. Míra podobnosti *sim*(*A*, *B*) nabývá hodnot <0, 1>, kde 1 značí, že A je totožné s B, a 0 značí, že A je zcela odlišné od B. Při určování podobnosti je text legendy a heslo záznamu rozděleno na jednotlivá slova a je určována podobnost mezi všemi kombinacemi slov. Výsledná podobnost je rovna součtu podobnosti všech kombinací. Určení podobnosti pro každý záznam je však vzhledem k velkému množství záznamů zdlouhavé, proto jsou ignorovány záznamy, které neobsahují ani jedno ze slov textu legendy. Aby se předešlo zanedbání hledaného záznamu v důsledku rozdílného skloňování slov, jsou při kontrole obsahu slov použity slova legendy bez koncovek a předpon. Levenshteinova vzdálenost je však počítána z úplného tvaru slov.

Z nalezených záznamů je vybráno alespoň pět s největší hodnotou podobnosti a jsou seřazeny sestupně podle podobnosti. Následně jsou uživateli zobrazena nalezená řešení i s jejich hesly v databázi. Uživateli jsou zobrazena jen řešení, která odpovídají již zadaným písmenům v křížovce. Řešení jsou pro každou legendu hledána pouze jednou a nalezená řešení jsou uložena.

Obrázek 4.13: Výsledek zpracování snímku křížovky.

5 VYHODNOCENÍ

Vyhodnocení řešení úlohy je rozděleno do několika samostatných částí, které jsou součástí vytvořené aplikace. Těmito částmi jsou segmentace políček, rozpoznání typu políčka, rozpoznání textu a vyhledání řešení ve slovníku.

Pro testování aplikace byla použita galerie 87 snímků křížovek s různým rozlišením a podmínkami prostředí. Snímky je možné rozdělit na snímky umělé, vytvořené mnou za účelem simulace různých problémových situací, a na snímky přirozené, které jsou pořízeny jinými lidmi a simulují tak běžné použití aplikace.

Umělé snímky byly pořízeny třemi různými telefony s rozlišením fotoaparátu 3,2 Mpx (8 snímků), 5 Mpx (16 snímků) a 13 Mpx (13 snímků). Snímky obsahují křížovky různých tvarů a barev, které jsou na snímcích úmyslně otočeny, deformovány nebo nerovnoměrně osvětleny.

Přirozené snímky byly všechny foceny v rozlišení 13 Mpx (50 snímků). Snímky mají různou kvalitu a křížovky jsou různou měrou zdeformované přeložením papíru.

Pro provoz aplikace byl použit mobilní telefon Sony Xperia Tipo ST21i s operačním systémem Android 4.0.4. Soubory potřebné pro provoz aplikace (trénovací data pro OCR, databáze slovníku) byly uloženy v paměti SD karty. Hardwarové specifikace zařízení:

- Procesor: Qualcomm MSM7225A
- Frekvence procesoru: 800 MHz
- Počet jader: 1
- Paměť RAM: 512 MB
- Paměť SD karty: 2.1 GB

5.1 Segmentace políček křížovky

Přesnost segmentace políček křížovky byla vyhodnocena pro všechny snímky v galerii jako:

$$ACC = \frac{TP}{TP + FP + FN} \tag{5.1}$$

kde *TP* je počet správně rozpoznaných rohů políček, *FP* je počet falešných rohů políček a *FN* je počet nerozpoznaných rohů políček. Přesnost *ACC* nabývá hodnot od nuly do jedné, kde jedna odpovídá zcela správně segmentované křížovce. Výsledky pro jednotlivé snímky jsou uvedeny v příloze číslo 2, včetně časové náročnosti zpracování snímku na výše uvedeném zařízení. V případech, kdy nebyla křížovka nalezena, byla přesnost ohodnocena hodnotou nula.

Z výsledků vyplývá, že se výsledek segmentace políček křížovky lepší spolu s rostoucím rozlišením obrazu a u obrazů s rozlišením 5 Mpx a vyšším dosahuje velmi dobré přesnosti – v průměru 0,92 (5 Mpx), 0,98 (13 Mpx umělé snímky) a 0,95 (13 Mpx přirozené snímky). V těchto případech podává segmentace špatný výsledek pouze pro špatně osvětlené umělé obrazy, kde odlesk světla skrývá část mřížky, a pro snímky křížovky se silně prohnutým papírem, kde segmentace ignoruje jednu z částí křížovky. Méně příznivé výsledky poskytuje segmentace pro snímky s rozlišením 3,2 Mpx, kde je průměrná přesnost jen 0,74 díky dvěma případům selhání křížovku v obraze nalézt. Příčinou těchto dvou selhání byla špatná inverze snímku křížovky.

Celkově mohu říci, že zvolený postup segmentace políček křížovky funguje dobře. Je schopen správně segmentovat křížovku různých tvarů nezávisle na otočení a je do určité míry odolný na špatné osvětlení a deformace mřížky. Doba nutná k segmentaci křížovky na použitém zařízení se v průměru pohybuje okolo 40 sekund. Doba zpracování se u jednotlivých snímků liší v závislosti na počtu políček křížovky, ale ne na rozlišení obrazu.

5.2 Rozpoznání typu políčka

Přesnost rozpoznání typů políček u segmentované křížovky byla určena podle následujícího vztahu:

$$ACC = \frac{P}{P + N} \tag{5.2}$$

kde P je počet políček, u kterých byl správně identifikován jejich typ, a N je počet políček, u kterým byl přiřazen nesprávný typ. Přesnost ACC nabývá hodnot od nuly do jedné, kde jedna odpovídá správnému rozpoznání všech políček. Výsledky pro jednotlivé snímky jsou uvedeny v příloze číslo 2, včetně časové náročnosti zpracování snímku na uvedeném zařízení. V případech, kdy bylo špatně určena orientace křížovky, byla přesnost ohodnocena hodnotou nula.

Díky závislosti aplikace na systému OCR pro určení orientace křížovky, jsou výsledky pro obrazy s nízkým rozlišením velmi špatné, kvůli nesprávnému otočení celé křížovky, což nelze ohodnotit jinak než jako nesprávné určení typu všech políček. S rostoucím rozlišením se však situace výrazně lepší, od průměrné přesnosti 0,15 pro 3,2 Mpx snímky se rozpoznávací algoritmus dostává na průměr 0,72 (5 Mpx a 13 Mpx umělé snímky). V případě přirozených snímků je průměrná přesnost horší, pouze 0,52 vlivem velkého množství chybných otočení křížovky, způsobených rozostřeným textem legendy na okrajích křížovky.

Kromě problému s určením správné orientace křížovky má rozpoznání políček také potíže s vyplněnými křížovkami. Ručně vepsaná písmena jsou systémem OCR

často rozeznána jako několik znaků, což způsobuje nesprávné rozpoznání políčka jako políčko legendy. V galerii se také nachází několik křížovek tisknutých v odstínech šedi, u kterých mají políčka tajenky jemnou texturu, která na vyprahovaném políčku vytvoří šum. V tomto šumu je poté OCR schopno nalézt mnoho znaků, což opět vede ke špatnému rozpoznání typu políčka a také k velmi dlouhému času zpracování, zvláště u vyšších rozlišeních snímku.

Přestože je průměrná přesnost rozpoznání políček špatná, v případech, kdy je správně rozpoznána orientace křížovky, je množství chyb v rozpoznání typu políček daleko menší, v mnoha případech jsou rozpoznána všechna políčka. Průměrná přesnost rozpoznání políček je po odstranění špatně otočených snímků rovna 0,93 (3,2 Mpx), 0,80 (5 Mpx), 0,85 (13 Mpx umělé snímky) a 0,91 (13 Mpx přirozené snímky).

Rozpoznání políček tedy funguje uspokojivě pouze v případě správného určení orientace křížovky, které je největším nedostatkem této části zpracování. Ovšem i pouze několik špatně rozpoznaných políček může znehodnotit celou křížovku. Doba zpracování se pohybuje v průměru od 60 do 90 sekund v závislosti na rozlišení a vzhledu křížovky. U některých křížovek však doba zpracování dosahuje velmi vysokých hodnot, řádově ve stovkách sekund.

5.3 Rozpoznání textu

Přesnost rozpoznání textu závisí zcela na systému OCR a je úzce spjato s rozpoznáváním políček legendy. Podle výstupu z OCR jsou také vyhledávány řešení k legendě ve slovníku. Přesnost rozpoznání textu je určena jako:

$$ACC = \frac{P}{P + N} \tag{5.3}$$

kde P je počet políček legendy se správně rozpoznaným textem a N je počet políček legendy s nesprávně rozpoznaným textem. Jako správně rozpoznaný text je považován takový výstupní text OCR, který má v porovnání s původním textem pro danou legendu maximálně jednu chybu v každém slově. Jednou chybou je zde myšleno jedno chybějící, přebývající nebo zaměněné písmeno. Hodnoceny byly pouze uměle vytvořené snímky, u kterých byla správně určena orientace. Přesnost ACC nabývá hodnot od nuly do jedné, kde jedna odpovídá správnému rozpoznání textu ve všech políčcích legendy. Výsledky pro jednotlivé snímky jsou uvedeny v příloze číslo 2.

Podle dokumentace potřebuje Tesseract OCR pro dobré výsledky alespoň písmo velikosti 10 bodů při rozlišení 300 DPI (20). Toho je obtížné dosáhnout kamerou mobilního telefonu, pokud chceme mít na snímku křížovku celou. Z těchto důvodů se sice přesnost rozpoznání textu s rostoucím rozlišením snímku zlepšuje,

ale ve většině případů nedosáhne uspokojivých hodnot ani pro snímky s rozlišením 13 Mpx. V případě 3,2 Mpx snímků nerozezná OCR správně prakticky žádný text. U 5 Mpx snímků je situace podstatně lepší, přesnost se pohybuje mezi různými snímky od 0,171 do 0,875 a průměrná přesnost je 0,46. Pro snímky s rozlišením 13 Mpx se minimální přesnost zvedá na 0,429 a průměrná přesnost rozpoznání textu je 0,607.

Protože je text legendy zobrazen pomocí výřezů ze snímku místo výstupu z OCR, má špatné rozpoznání textu na zobrazení křížovky jen malý vliv. Z výstupu OCR však vychází hledání řešení k legendě ve slovníku. Hledání řešení podle podobnosti legendy s heslem ve slovníku dokáže najít správné řešení i pro text legendy, který obsahuje chyby, ale čím více chyb se v textu nachází, tím méně pravděpodobné nalezení správného řešení je.

Rozpoznání textu, tak je v aplikaci implementováno, tedy nepřináší uspokojivé výsledky při rozpoznávání textu legendy. Písmena ručně vepsaná do prázdných políček není možné rozpoznat vůbec, protože jsou v drtivé většině případů rozpoznána jako několik znaků. Tyto skutečnosti mají neblahý vliv na fázi rozpoznání políček i fázi hledání řešení křížovky.

5.4 Vyhledání řešení ve slovníku

Za účelem vyhodnocení úspěšnosti nalezení řešení ve slovníku bylo hledáno řešení pro 62 různých políček legendy ve třech různých křížovkách. Aby se zabránilo zkreslení výsledků vlivem chybovosti OCR, bylo hledáno řešení pouze pro políčka legendy, jejichž text byl rozpoznán zcela bez chyb. Řešení bylo vyhledáváno za podmínek zcela nevyplněné křížovky. Pokud aplikace řešení nenašla, byla ručně prohledána databáze, jestli se v ní hledané řešení nachází a jestli je k němu přiřazeno heslo, které přibližně odpovídá textu legendy.

Z 62 legend se v databázi nachází řešení pro 51 z nich. Z těchto 51 řešení v databázi aplikace nalezla 36 řešení. Úspěšnost vyhledání řešení legendy je tedy přibližně 71 %. Neúspěchy ve vyhledání řešení jsou zaviněny z části způsobem filtrace odpovědí z databáze a z části tvarem, ve kterém jsou hesla k řešení zapsána v databázi. Heslo v databázi může používat jiná slova než text legendy nebo stejná slova, ale jinak skloňovaná, což sníží míru podobnosti s textem legendy a do popředí se dostanou podobnější hesla s nesprávným řešením.

Doba nutná k vyhledání řešení se velmi liší v závislosti na počtu znaků legendy a řešení. Řešení s počtem čtyř až šesti znaků jsou velmi běžná, a proto i často zastoupená v databázi. Krátkému textu legendy také odpovídá velké množství hesel v databázi. V obou případech musí algoritmus zkontrolovat velké množství hesel a doba prohledávání je tak prodloužena. Při testování se doba prohledávání měnila od tří sekund do čtyř minut, v průměru je řešení nalezeno do jedné minuty. Výsledky testu jsou součástí přílohy číslo 2.

6 ZÁVĚR

Po bližším prozkoumání zadané úlohy vyjde najevo její obtížnost, zapříčiněna především velkou volností, kterou autoři švédských křížovek mají. Rozmanitost vzhledu švédských křížovek a neurčitost podmínek pořízení jejich snímků vyžadují robustní metody, které jsou schopny obsáhnout všechny možné situace. Programy a aplikace řešící úlohu podobnou zadání této práce již existují, ale nebyl jsem schopen nalézt žádnou, která by spojovala rozpoznání křížovky na snímku s hledáním řešení ve slovníku a zároveň podporovala švédské křížovky v češtině tak, jak je požadováno v zadání práce. Vytvořil jsem tedy vlastní aplikaci pro Android, která tyto požadavky splňuje a vyhodnotil ji s pomocí galerie snímků švédských křížovek.

Na základě provedených testů aplikace mohu říci, že navržený způsob detekce mřížky křížovky a segmentace jejích políček pracuje navzdory již zmíněné obtížnosti úlohy dobře a u mnoha snímků dosáhl zcela správné segmentace mřížky.

Horší výsledky podává aplikace při rozpoznání typu políček, kde je častým problémem špatně rozpoznaná orientace křížovky. Zde se aplikace spoléhá na implementovaný systém OCR, který v mnoha případech nerozezná text v křížovce správně, zvláště u snímků s nižším rozlišením. Dalším problémem jsou také ručně psaná písmena, která OCR chybně rozpozná jako několik písmen.

Rozpoznání textu legendy pomocí OCR také nepřináší příliš dobré výsledky. V rozpoznaném textu se často objevují chyby, které mohou negativně ovlivnit úspěšnost hledání řešení ve slovníku. I ve snímcích s nejvyšším použitým rozlišením (13 Mpx) se vyskytuje nezanedbatelné množství chyb v přibližně polovině políček legendy.

Na druhou stranu algoritmus vyhledávající řešení ve slovníku, pokud rozpoznaný text legendy neobsahuje chyby, dokázal nalézt správné řešení ve 71 % případů bez jakékoliv informace o písmenech řešení. Tato úspěšnost není příliš velká, ale na úspěšné nalezení řešení má také velký vliv způsob zapsání hesel ve slovníku a úspěšnost by stoupla při použití slovníku vytvořeného aplikaci na míru.

Nejslabším článkem vytvořené aplikace je použitý systém OCR, který na testovacích snímcích vykazoval nedostatečnou přesnost, což negativně ovlivňuje celkový výsledek detekce křížovky, protože je na základě rozpoznaného textu určována orientace křížovky a typ políček. Aplikaci by proto výrazně pomohlo zlepšení přesnosti OCR nebo alespoň použití jiného způsobu určení orientace křížovky. Prostor ke zlepšení poskytují také snímky již vyplněných křížovek, které aplikace ve své současné podobě neřeší.

Seznam použitých zdrojů

1. **Gonzalez, Rafael C. a Woods, Richard E.** *Digital Image Processing.* 3. New Jersey : Pearson Prentice Hall, 2007. 978-0131687288.

2. **Anderson, Richard a Krogh, Peter.** Color Space and Color Profiles. In: American Society of Media Photographers. *dpBestflow.* [Online] Philadelphia: American Society of Media Photographers, Inc., 22. září 2015. [Citace: 27. února 2017.] http://www.dpbestflow.org/color/color-space-and-color-profiles.

3. **SharkD.** *HSV color solid cylinder alpha lowgamma.* [Online] 22. března 2010. [Citace: 27. února 2017.] https://en.wikipedia.org/wiki/File:HSV_color_solid_cylinder_alpha_lowgamma.png.

4. **Hlaváč, Václav.** *Barva, barevné obrazy a správa barev*. [Online] Praha: České vysoké učení technické, 17. Dubna 2017. [Citace: 26. Dubna 2017.] people.ciirc.cvut.cz/~hlavac/TeachPresCz/11DigZprObr/04ColorImagCz.pdf.

5. **Appalachian State University.** *cielab2*. [Online] 2. Července 2012. [Citace: 7. Května 2017.] http://www1.appstate.edu/~kms/classes/psy3215/ColorModels/cielab2.gif.

6. **Lindbloom, Bruce Justin.** RGB Working Space Information. *Bruce Lindbloom.com.* [Online] 31. ledna 2014. [Citace: 27. února 2017.] http://www.brucelindbloom.com/index.html?WorkingSpaceInfo.html.

7. **Walek, Petr, Lamoš, Martin a Jan, Jiří.** *Analýza biomedicinských obrazů počítačová cvičení.* Brno : FEKT VUT v Brně, Ústav biomedicinského inženýrství, 2015. 978-80-214-4792-9.

8. **Mansurov, Nasim.** What is Moiré? In: Photography Life. *photographylife.* [Online] 20. Září 2014. [Citace: 27. Dubna 2017.] https://photographylife.com/what-is-moire/.

9. **Pinterest.** *6930fe60556d0ba244cad9772be85e35*. [Online] [21. století] [Citace: 28. Dubna 2017.] https://s-media-cache-ak0.pinimg.com/originals/69/30/fe/6930fe60556d0ba244cad9772be85e35.jpg.

10. **Cattin, Philippe.** *Image Restoration: Introduction to Signal and Image Processing* [Online]. Basel: MIAC, University of Basel, 26. Dubna 2016. [Citace: 26. Dubna 2017.] https://miac.unibas.ch/SIP/06-Restoration.html.

11. **McHough, Sean.** Digital Camera Image Noise: Concept and Types. *Cambridge in Colour - Photography Tutorials & Learning Community.* [Online] Cambridge: Cambridge in Colour, © 2005-2017. [Citace: 28. Dubna 2017.] http://www.cambridgeincolour.com/tutorials/image-noise.htm.

12. **Přibil, Jiří.** *Efektivní metody detekce plagiátů v rozsáhlých dokumentových skladech.* Jindřichův Hradec, 2010. Doktorská dizertační práce. Fakulta managementu v Jindřichově Hradci. Vedoucí práce Prof. Radim Jiroušek.

13. **Kondrak, Grzegorz.** *N-Gram Similarity and Distance* [Online]. Edmonton : Department of Computing Science, University of Alberta, Canada, 2005 [Citace: 6. Května 2017]. https://pdfs.semanticscholar.org/5cfb/028595fff3e550c9a5fd2a51e4f23b4b58b6.pdf

14. **Skalický, Ivo.** O programu. *CwdStudio.* [Online] Pardubice: ITPro, ©2003-2017. [Citace: 25. února 2017.] http://www.cwdstudio.com/cs/about/.

15. **TURPRESS.** *ŠK-s-přílohou_obálka*. [Online] 13. října 2016. [Citace: 25. února 2017.] http://www.turpress.eu/wp-content/uploads/2016/10/ŠK-s-přílohou_obálka.jpg.

16. **SČHAK.** Směrnice. *Český svaz hádankářů a křížovkářů, z. s.* [Online] Praha: ČSHAK, z. s., © 2011-2013. [Citace: 25. února 2017.] http://www.cshak.cz/smernice.

17. **SČHAK**. *Směrnice pro tvorbu křížovek* [Online]. Praha : ČSHAK, 1994-2006. [Citace: 25. února 2017.] www.cshak.cz/sites/default/files/Sm%C4%9Brnice_0_0.doc

18. **Raiter, Brian.** Cryptic Crossword: Amateur Crypto and Reverse Engineering. *Muppetlabs, Inc.* [Online] [2013]. [Citace: 26. února 2017.] http://www.muppetlabs.com/~breadbox/txt/acre.html.

19. **Cvajniga, Vladimír a Čálek, Karel.** Velký křížovkářský slovník. [Online] 4.0, 16. Července 2014. [Citace: 8. Května 2017.] http://www.nvks.cz/demoverze.

20. **opencv dev team.** OpenCV API Reference - OpenCV 3.0.0-dev documentation. *OpenCV documentation.* [Online] 10. Listopadu 2014. [Citace: 3. Května 2017.] http://docs.opencv.org/3.0-beta/modules/refman.html.

21. **Nash.** Implementation of Guo-Hall thinning algorithm. *OpenCV Code.* [Online] 12. Ledna 2013. [Citace: 4. Května 2017.] https://web.archive.org/web/20160314104646/http://opencv-code.com/quick-tips/implementation-of-guo-hall-thinning-algorithm/.

22. **Muccigrosso, John.** FAQ tesseract-ocr/tesseract Wiki. *GitHub.* [Online] 6. Srpna 2016. [Citace: 5. Května 2017.] https://github.com/tesseract-ocr/tesseract/wiki/FAQ#is-there-a-minimum-text-size-it-wont-read-screen-text.

23. **Lindbloom, Bruce Justin.** RGB to XYZ. *Bruce Lindbloom.com.* [Online] 30. dubna 2012. [Citace: 27. února 2017.] http://www.brucelindbloom.com/index.html?Eqn_RGB_to_XYZ.html.

Seznam použitých zkratek a symbolů

OCR	optical character recognition, optické rozpoznávání znaků
DPI	dots per inch, počet obrazových bodů na jeden palec délky
px	zkráceně pixel (obrazový bod)

Seznam příloh

Příloha 1:	Uživatelský manuál k aplikaci.
Příloha 2:	Tabulky s výsledky testů aplikace.
Příloha 3:	DVD s elektronickou verzí písemné zprávy práce, projektem pro vývojové prostředí Android Studio, instalačním balíčkem aplikace včetně trénovacích dat a slovníku, dokumentací zdrojového kódu a galerií použitých snímků.