VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZPOZNÁVÁNÍ ZOUBKOVÁNÍ POŠTOVNÍCH ZNÁMEK

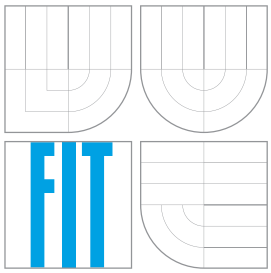
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. VLADIMÍR KONÍČEK

BRNO 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZPOZNÁVÁNÍ ZOUBKOVÁNÍ POŠTOVNÍCH ZNÁMEK

POST STAMP PERFORATION RECOGNITION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. VLADIMÍR KONÍČEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL ŠPANĚL,

BRNO 2010

Abstrakt

Rozpoznávání zoubkování poštovních známek je důležitým faktorem při posuzování pravosti poštovní známky. Typ a rozměr zoubkování mají výrazný vliv na cenu poštovní známky. Tato práce se zabývá navrhem detektoru zoubkování poštovních známek. Cílem práce je vytvořit aplikaci, která z fotografie určí zoubkování zobrazené poštovní známky. Aplikace pro práci s obrazy využívá knihovnu OpenCV.

Abstract

Post stamp perforation recognition is important factor in authentication of post stamps. Type and perforation size have major influence on price of post stamps. This Masters thesis is handling suggestion of detector of post stamp perforation. Goal of this work is to create application, which sill recognize perforation size from photography of post stamp. Application is using OpenCV library.

Klíčová slova

Poštovní známka, zoubkování, detekce.

Keywords

Post stamp, perforation, detection.

Citace

Vladimír Koníček: Rozpoznávání zoubkování poštovních známek, diplomová práce, Brno, FIT VUT v Brně, 2010

Rozpoznávání zoubkování poštovních známek

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Michala Španěla.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Vladimír Koníček
25. května 2010

Poděkování

Děkuji vedoucímu diplomové práce Ing. Michalu Španělovi za pomoc, rady a připomínky při vedení této práce.

Obsah

Kapitola 1

Úvod

Rozpoznání zoubkování známek je důležitým faktorem při posuzování pravosti poštovní známky, typ a rozměr zoubkování má výrazný vliv na cenu známky. Například u první vydané edice známek samostatného Československa – Hradčan se může cena nepoužité známky v závislosti na typu a rozměru zoubkování pohybovat od desetikoruny po desítky tisíc korun.

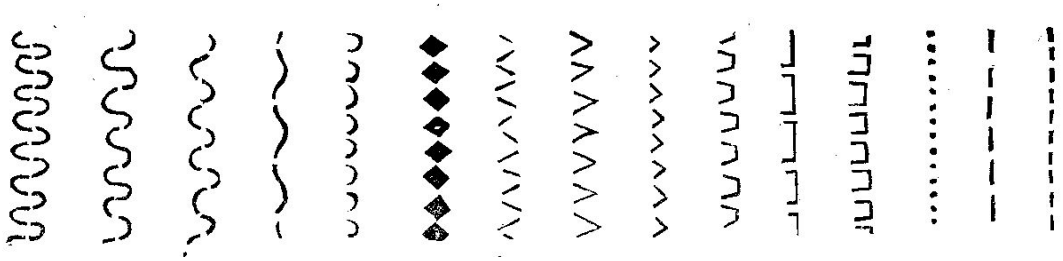
Úkolem diplomové práce bylo vytvořit program, který je z obrazu známky schopen zjistit rozměr zoubkování známky, popřípadě jeho typ. Úvodní kapitola práce se zabývá historií zoubkování známek, popisuje jednotlivé druhy zoubkování známek, rozměry a nepravidelnosti zoubkování. Následuje popis algoritmů použitých při zpracování obrazu známky, návrh detektoru a popis implementace. Práce je pokračováním mé bakalářské práce [**?**].

Kapitola 2

Zoubkování poštovních známek

2.1 Historie zoubkování

Známky jsou tištěny od počátku svého použití v arších, arch obsahuje nejčastěji 10 x 10, tedy 100 známek. První metodou oddělování známek bylo stříhání, kdy jednotlivé známky byly z archu oddělovány ustřižením. Mezery mezi známkami byly zpočátku velmi malé, např. jen 0,5 mm, stříhání vyžadovalo od poštovních úředníků zručnost a manipulace s archem byla zdlouhavá. Proto byla snaha o usnadnění oddělování jednotlivých známek z archu. Už od roku 1841 byly známky opatřovány průsekem - jednotlivé známky jsou oddělovány zářezy, tak aby nenastalo vypadávání prosekaných míst. Podle tvaru proseknutí rozlišujeme průsek čárkovitý, stříškovitý, pilovitý, obloučkovitý, vlnkovitý. Známky oddělené průsekem nemají na okrajích pravidelné zoubky, jak je známe z dnešních známek. Používané typy průseků jsou zobrazeny na obrázku 2.1.



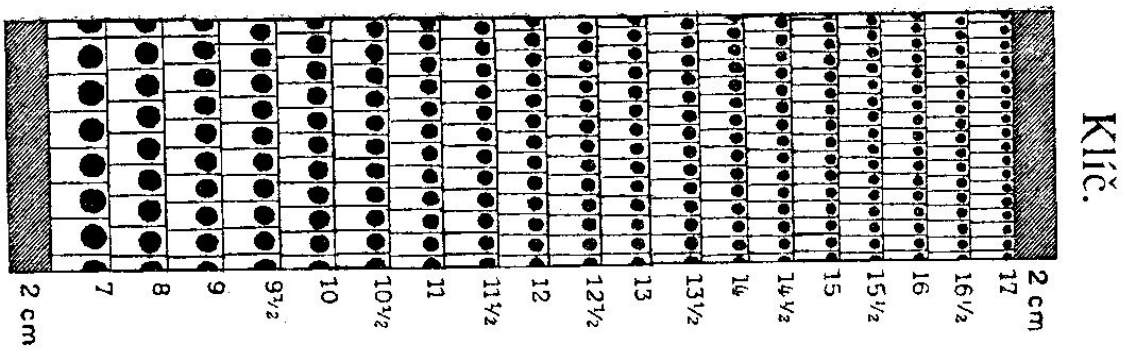
Obrázek 2.1: Používané typy průseků, převzato z [?]

Některá vydání známek nebyla opatřována průsekem, ale tzv. průpichem, kdy je prostor mezi známkami propichován tenkými jehlami, někdy se k tomu používal i obyčejný šicí stroj. Většinou se jednalo o svépomocné, poštmistrovské zoubkování, přesto se nazývá soukromé zoubkování.

Průsek ani průpich nebyly dokonalými způsoby oddělování známek a proto již roku 1847 byly pokusně oddělovány známky v Anglii strojem perforačním. Ten je tvořen řadou dutých jehel se zabroušeným okrajem a v archu jsou tak vysekávány malé kruhové otvory. První perforační stroje měly pouze jednu lištu s řadou jehel, později byly uspořádány jehly do tvaru hřebenu, popřípadě umístěny na rotující válec. Podle způsobu tvorby perforace rozeznáváme několik druhů zoubkování.

Neméně důležitým údajem o zoubkování je jeho rozměr. Na důležitost měření, počítání

zoubků na známkách upozornil poprvé roku 1866 v časopise *Timbre-Poste* Dr. J. A. Legrand. Aby si počítání zoubků usnadnil, vyřízl z černého kartonu proužek přesně 2 cm dlouhý a počítal kolik zoubků se vejde na tento proužek. Později si na karton nakreslil řady bodů odpovídající počtu zoubků a vynalezl tak klíč, pomocí kterého je možno zjistit počet zoubků na 2 cm délky okraje. Ukázka takového klíče je na obrázku 2.2.



Obrázek 2.2: Ukázka klíče, převzato z [?]

První perforační stroj byl tvořen lištou s řadou dutých jehel, pod které se vkládal přepážkový arch a postupně se v jednotlivých řadách vyrážely otvory – vzniklo řádkové zoubkování. Pro zefektivnění práce se později upravila lišta do tvaru hřebenu a snížil se počet úderů lišty s jehlami. Nakonec byly vytvořeny perforační stroje, které jsou schopné operforovat celý arch najednou. Rozeznáváme historicky tři druhy zoubkování:

1. řádkové, dříve nazývané liniové
2. hřebenové
3. rámcové

Kombinací těchto základních druhů může vzniknout zoubkování sdružené, kombinací rozměrů pak zoubkování smíšené.

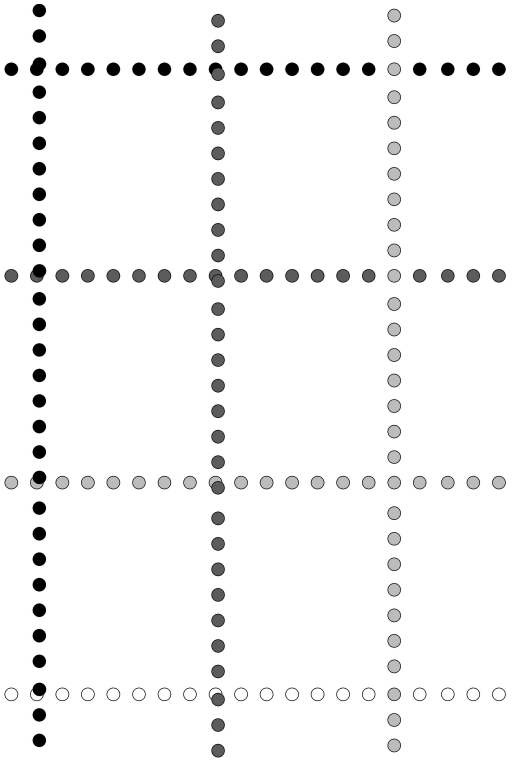
Problematiku zoubkování zmiňuje kterákoli kniha o filatelii, já jsem se při zpacování této kapitoly opíral o publikace [?], [?], [?], [?].

2.2 Řádkové zoubkování

Název tohoto zoubkování je odvozen od obrazu, který vytvoří jehly po prvním úderu perforačního stroje do přepážkového archu. K ozoubkování stokusového archu (10 x 10 známek) je potřeba ve vodorovném směru 11 úderů perforačky, poté je arch otočen o 90° a operforován dalšími jedenácti údery.

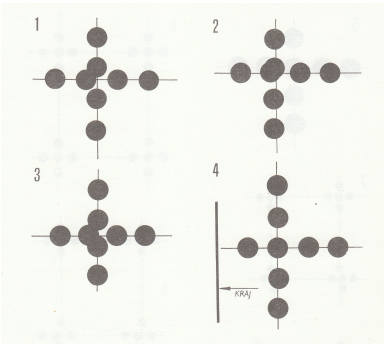
Schéma je na obrázku 2.3, kde je pořadí jednotlivých úderů rozlišeno stále světlejšími barvami. Na obrázku jsou vidět charakteristické znaky řádového zoubkování:

1. Při pohledu na celý arch je vidět, že řady perforačních otvorů procházejí skoro až do okrajů archu a to na všech stranách archu. Na všech stranách archu jsou vytvořeny tzv. kupóny, tj. operforovaná místa bez obrazu známky.



Obrázek 2.3: Schéma řádkového zoubkování

2. Perforační otvory se na průniku svislých a vodorovných řad setkávají zcela nahodile, jak je vidět na obrázku 2.3 (levá a střední svislá řada). Jen opravdu vyjímečně se stává, že na průniku svislých a vodorovných řad, tedy přesně ve stejném místě, došlo k průniku jehly při perforaci ve vodorovném i svislém směru (třetí svislá řada na obr. 2.3). Ještě méně častá je možnost, že by k takové situaci došlo ve všech čtyřech rozích známky. Příklady usporádání perforačních otvorů v rohu známky jsou na obr 2.4.



Obrázek 2.4: Varianty umístění perforačních otvorů v rohu známky, převzato z [?]

Ukázka reálného řádkového zoubkování je na obrázku 2.5, v každém z rohů známky jsou opravdu perforační otvory jinak uspořádány.

Obrázek 2.5: Ukázka řádkového zoubkování

K řádkovému zoubkování řadíme také tzv. rotační zoubkování. Místo jedné lišty jsou jehly umístěny na válci v jedenácti řadách a operforují přepážkový arch na dva průchody přes válce. Na obrázku 2.6 je detail takového válce s osazenými jehlami. Proti řadě jehel je lišta s dírkami, takže při vytváření dírky je okraj perforačního otvoru zahnut dolů.

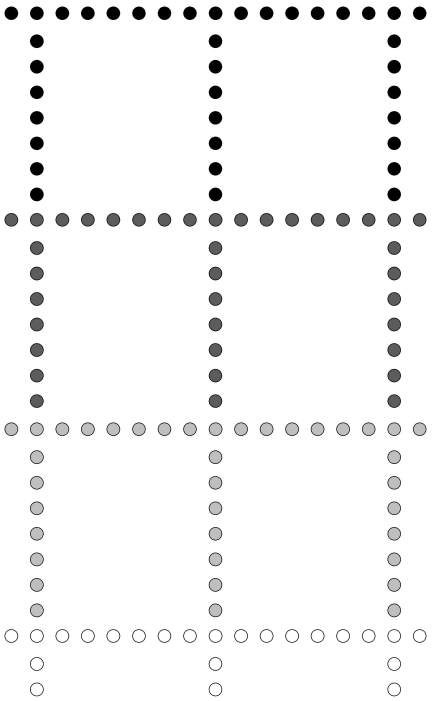
Obrázek 2.6: Válec pro rotační perforaci, převzato z [?]

2.3 Hřebenové zoubkování

Hřebenové zoubkování bylo zavedeno kvůli zrychlení práce při zoubkování známek. Název je opět odvozen podle obrazu, který vytvoří soustava jehel po prvním úderu. Prvním úderem jsou ozoubkovány známky na horním okraji první řady a dále je provedeno svislé zoubkování první řady. Spodní okraj první řady známek je ozoubkován až při druhém úderu.

Příklad postupu hřebenového zoubkování je na obrázku 2.7. K perforaci celého archu je potřeba pouze jedenáct úderů perforačního stroje a odpadá otáčení archu. Na obrázku jsou také vidět charakteristické znaky hřebenového zoubkování:

1. Při pohledu na celý arch je vidět, že řady perforačních otvorů procházejí až do okrajů archu pouze na spodní straně archu, ne na horním, levém nebo pravém okraji. Obvykle, ne však vždy, přesahují vodorovné řady o jeden perforační otvor. To platí, postupuje-li



Obrázek 2.7: Schéma hřebenového zoubkování

hřebenové zoubkování shora dolů. Vzácně mohlo dojít k postupu zleva doprava, potom prochází perforace až k okrajům na straně a perforační otvor *navíc* je nahoře a dole. Takové zoubkování se nazývá *ležmý hřeben*.

2. Na průniku svislých a vodorovných řad perforačních otvorů je centrovaný otvor. Ten také vymezuje charakteristický obraz rohu známky. To platí u hřebenového zoubkování postupujícího shora dolů pro levý a pravý horní roh známky. U dolního levého a pravého rohu je otvor přesně na průniku svislých a vodorovných os jen tehdy, pracuje-li perforační stroj dokonale.

Obrázek 2.8: Ukázka hřebenového zoubkování

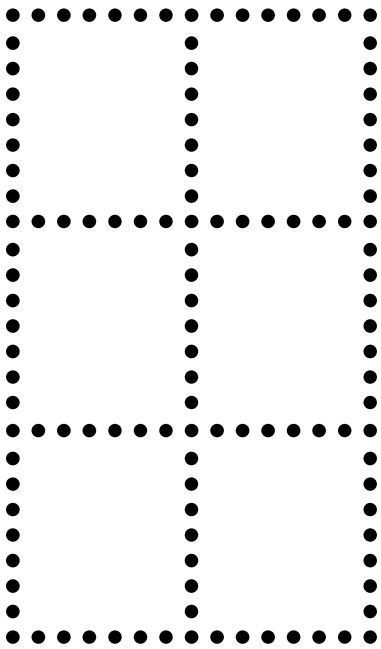
Ukázka reálného hřebenového zoubkování je na obrázku 2.8.

Nejčastější postupuje hřebenové zoubkování shora dolů, popř. zdola nahoru. Méně často zleva doprava, popř. zprava doleva. Potom se jedná o ležmé hřebenové zoubkování, které se vyskytuje např. u známek z emise Holubice a Osvobozená republika. Rozpoznat lze na celém archu, popř. bloku známek, určení tohoto typu zoubkování u jednotlivé známky je obtížné.

Poměrně málo se vyskytuje tzv. *dvojité hřebenové zoubkování*. Jehly jsou na liště rozmístěny do tvaru písmene H a jedním úderem je ozoubkována vodorovná strana a polovina svislých stran u horní a dolní řady známek. Perforuje se opět jedenácti údery a kupóny jsou na horním i dolním okraji archu.

2.4 Rámcové zoubkování

Rámcové zoubkování je postup, při kterém, jak název sám napovídá, je perforačními otvory opatřen celý arch najednou, jediným úderem perforačního stroje. Použitím tohoto způsobu zoubkování je možno dosáhnout pravidelně uspořádaných otvorů po celém obvodu známek, lepší kvality zoubkování a také zrychlení výroby. Používalo se hlavně při zoubkování aršíků a tiskových archů s malým počtem známek. Postup je naznačen na obrázku 2.9.



Obrázek 2.9: Schéma rámcového zoubkování

Je vidět, že otvory nejsou na žádném okraji archu, při správně pracujícím perforačním stroji nelze u jednotlivé známky rozlišit rámcové a hřebenové zoubkování. To lze jen při pohledu na celý přepážkový arch nebo při nepravidelnosti práce perforačního stroje.

Ukázka reálného rámcového zoubkování je na obrázku 2.10.

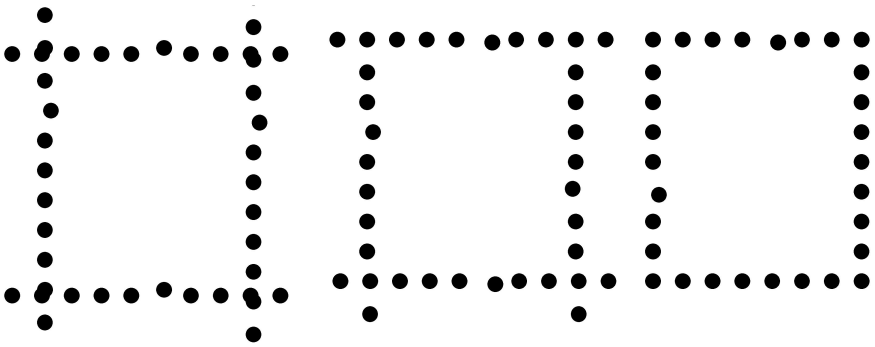
Obrázek 2.10: Ukázka rámcového zoubkování

V minulosti bylo rámcové zoubkování používáno k zoubkování příležitostných aršíků, popř. přepážkových archů s malým počtem známek. V současné době jsou tímto zoubkováním zoubkovány všechny známky vydávané v České republice.

2.5 Nepravidelnosti zoubkování

Vychýlená jehla

Jehly nejsou v perforační liště umístěny přesně v přímce, v průběhu práce dochází k jejich vychýlení. Tyto odchylky můžeme využít ke stanovení typu zoubkování.



Obrázek 2.11: Vychýlené perforační jehly u řádkového, hřebenového a rámcového zoubkování

U řádkového zoukování se musí vyskytovat vychýlení na obou protilehlých stranách, protože je známka peforována pořád stejnou řadou jehel.

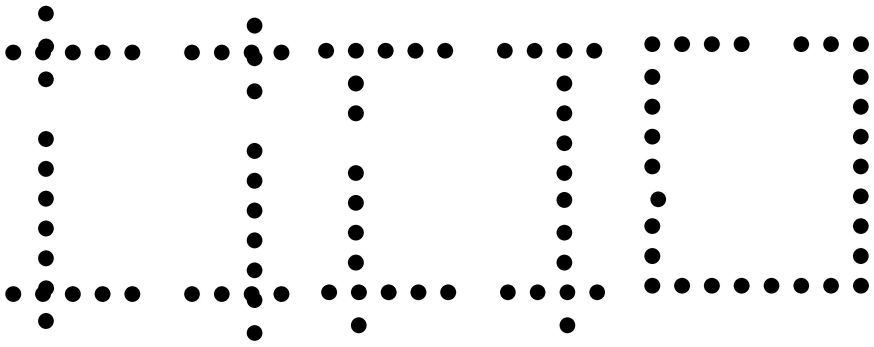
U hřebenového zoubkování platí, je-li vychýlení ve vodorovné řadě, musí být na horním i dolním okraji (opět jako u řádkového je horní i dolní okraj perforován stejnou řadou jehel).

Je-li ve svislé řadě, je pouze v jedné z nich (pravý a levý okraj je perforován jinou řadou jehel). To platí, postupuje-li hřebenové zoubkování zhora dolů. U ležmého hřebene naopak musí být shoda na levém a pravém okraji a odlišnost u horního a spodního okraje.

U rámcového zoubkování se vychýlení projeví pouze na jednom okraji známky (každý okraj je perforován jinou řadou jehel).

Vynechaný perforační otvor

V průběhu perforace se může některá jehla úplně poškodit a perforační otvor není vůbec vyražen. Pro jeho umístění a pravidla stanovení typu zoubkování platí co bylo napsáno výše u vychýlení jehly.



Obrázek 2.12: Vynechaný perforační otvor u řádkového, hřebenového a rámcového zoubkování

Obrázek 2.13: Ukázka nepravidelnosti u hřebenového zoubkování

Nepravidelnost v práci perforačního stroje

Nepravidelnost v práci perforačního stroje se projeví na velikosti známky. U řádkového zoubkování to nemá vliv na pravidelnost zoubkování, u hřebenového se projeví nepravidelnost dvojím způsobem. Pokud je posun perforačního stroje menší než správný, dochází ke

zkrácení známky. Pokud je posun větší než správný, dochází k protažení velikosti známky a může to vypadat, jako by byl vynechán perforační otvor.

Ukázka nepravidelnosti v práci perforačního stroje je na obrázku 2.13, vlevo zkrácená známka, vpravo prodloužená verze.

2.6 Rozměry zoubkování

Perforační stroje byly osazovány jehlami o průměru od 0,8 mm do 1,25 mm, takže na různých známkách se můžeme setkat s různou velikostí perforačních otvorů. Pravděpodobnost, že budou různé otvory na téže známce je velmi malá, ikdyž se vyskytují známky s tzv. sdruženým zoubkováním, kdy při hřebenovém zoubkování nebyl oražen spodní okraj (jedenáctý úder perforačky) a dodatečně byla dodělána řada otvorů na liniové perforačce, někdy s jiným rozměrem zoubkování, popřípadě s jiným průměrem perforačních jehel. U řádkového zoubkování může nastat situace, že vodorovně byly známky operforovány na jednom perforačním stroji a svisle na jiném s odlišným průměrem jehel.

Rozměr zoubkování se udává jako číslo, které znamená počet otvorů na vzdálenosti 20 mm a přesností na čtvrtinu otvoru. V historii byly použity rozměry zoubkování od $4\frac{3}{4}$ do 18 [?] . Při rozměrech zoubkování menších než 10 je těžké oddělovat jednotlivé známky, hrozí spíše, že se utrhne některý zoubek. Naopak pro zoubkování větší než 15 dochází k samovolnému oddělování známek již při manipulaci s tiskovým archem, protože zoubky jsou již moc jemné. Nejrozšířenější rozměry zoubkování jsou 11 - 14. Nějčastějšími rozměry u současných českých známek jsou $11\frac{1}{4}$, $11\frac{1}{2}$ a $11\frac{3}{4}$.

Přehled úředně zoubkovaných známek

			A Hz 13¾ : 13½	B Hz 11¾	C Řz 13¾	D Řz 11½	E Řz 11½ : 10¾	F Řz 13¾ : 11½	G Řz 11½ : 13¾	H Řz 13¾ : 10¾
1	1 h	hnědá	-	-	●	-	-	-	-	-
3	5 h	zelená	-	-	-	●	●	-	-	-
4	5 h	modrozel.	●	●	●	●	●	●	-	-
5	10 h	červená	-	-	-	●	-	-	-	-
6	10 h	zelená	●	●	●	-	-	-	-	-
7	15 h	červená	●	●	●	●	●	●	●	●
8	20 h	modrozel.	●	-	●	●	-	-	-	-
9	20 h	karmínová	●	-	●	-	-	-	-	-
10	25 h	modrá	●	-	●	●	-	-	-	-
11	25 h	fialová	●	●	-	●	●	●	-	●
13	30 h	fialová	●	●	●	●	-	-	●	-
17	60 h	oranžová	●	●	-	-	-	-	-	-
21	120 h	šedá	-	-	-	●	-	-	-	-
22	200 h	ultramar.	●	-	-	-	-	-	-	-

Přehled zahrnuje jen základní varianty zoubkování a známek bez rozlišení barevných odstínů.

Obrázek 2.14: Přehled úředních zoubkování Hradčan, převzato z [?]

U řádkového zoubkování se mohou vyskytovat jiné rozměry ve vodorovném a jiné ve svislém směru, to v případě, kdy byl přepážkový arch po jedenácti úderech ve vodorovném směru přenesen na druhý peforační stroj s lištou s jiným rozměrem zoubkování.

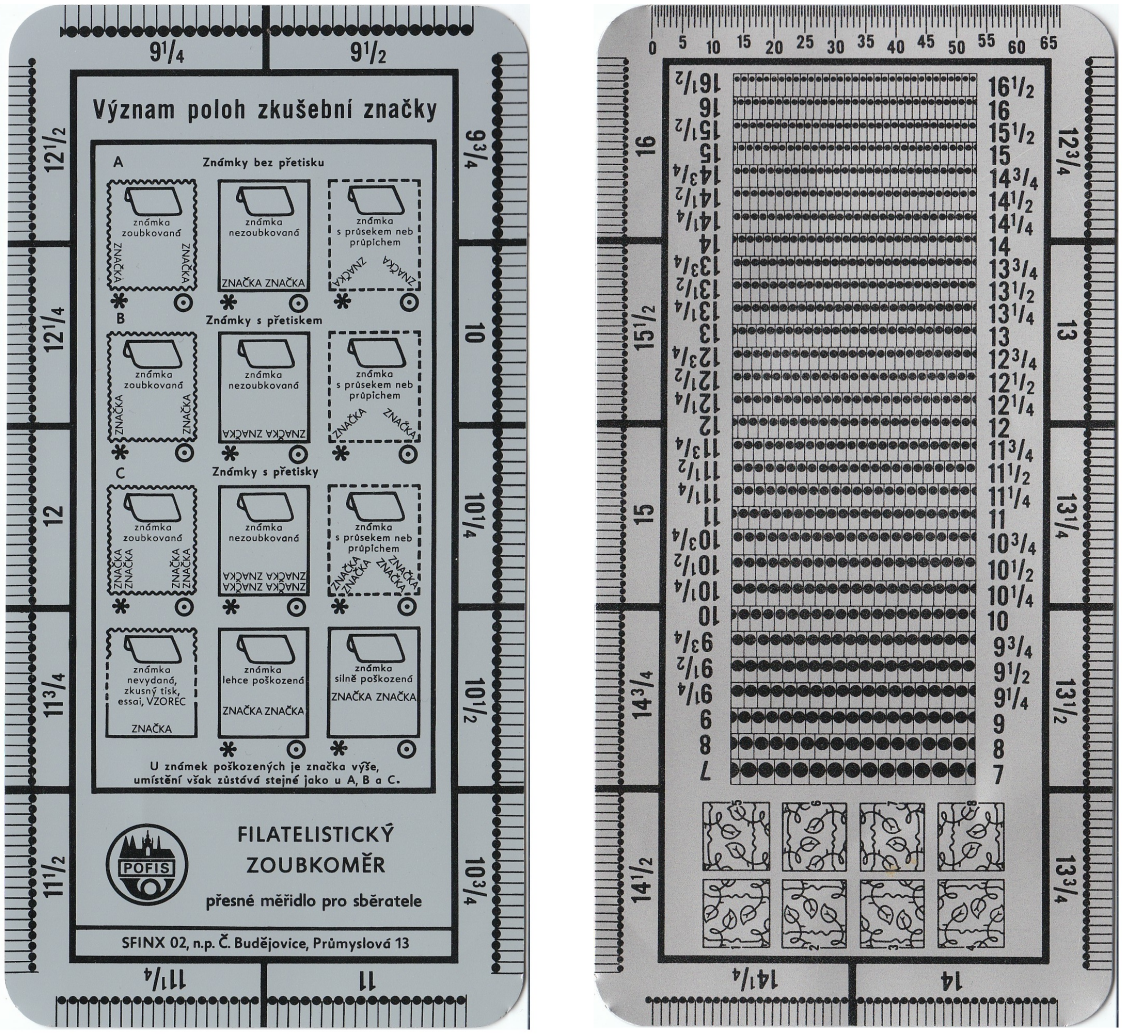
U hřebenového zoubkování je možné, že spodní okraj poslední řady známek byl operforován perforační řadou s jiným rozměrem zoubkování, pak je takové zoubkování nazýváno

sdružené (hřebenové + řádkové zoubkování). Rozměr zoubkování i u hřebenového bývá většinou na svislém a vodorovném okraji odlišný.

V současné době jsou známky perforovány tak, že rozměr zoubkování známky je většinou jeden, z období první republiky se ale vyskytují známky, které mají až deset možných rozměrů zoubkování (viz. tabulka na obrázku 2.14, kde je uvedeno pro Hradčany 8 úředních druhů zoubkování a existují další soukromá zoubkování). Podle četnosti výskytu jednotlivých typů jsou také známky oceňovány.

Obrázek 2.15: Elektronický přístroj Perfotronic, převzato z [?]

K určení rozměrů zoubkování se používá zoubkoměr (obrázek 2.16), popřípadě elektronický přístroj Perfotronic (obrázek 2.15). Zoubkoměr může být z papíru, plastu nebo kovu, je na něm předtištěno množství rozměrů zoubkování ve formě bodů a čar, stejně jako před sto padesáti lety. Určení rozměru zoubkování se provádí postupným přikládáním okrajů známek k předtištěným vzorům a subjektivním posouzením, zda vzor souhlasí s určovanou známkou. V elektronickém přístroji pro určování rozměrů zoubkování je umístěn lineární snímač podobně jako ve skeneru a přístroj vyhodnotí rozměr zoubkování. Stejně jako u zoubkoměru je potřeba vložit známku do přístroje příslušnou měřenou stranou, protože známky mohou mít rozměr zoubkování na každé straně známky odlišné. V katalozích je rozměr zobkování uváděn číslem, např. $13\frac{3}{4}$, pokud je uvedeno např. $14 : 13\frac{1}{2}$, znamená to, že vodorovné strany mají zoubkování 14, svislé strany $13\frac{1}{2}$. Pro typy zoubkování se požívají v katalozích zkratky Řz - řádkové zoubkování, Hz - hřebenové zoubkování, Rz - rámcové zoubkování.



Obrázek 2.16: Přední a zadní strana zoubkoměru

Kapitola 3

Zpracování obrazu

Tato kapitola obsahuje úvod do problematiky zpracování obrazu. Jsou zde uvedeny způsoby, jak je možno detekovat hrany nebo jednoduché objekty v obraze. Některé z technik popsaných v této kapitole jsem dále využil při implementaci samotného programu.

3.1 Digitální obraz

Převzato z [**?**].

Než začneme obraz zpracovávat, je potřeba jej převést do digitální podoby například pomocí skeneru nebo digitálního fotoaparátu. Digitalizace obrazu je převod analogového signálu do diskrétního tvaru.

Vstupní signál je popsán funkcí $f(x, y)$ dvou proměnných, které udávají souřadnice bodu v obraze. Funkční hodnota odpovídá například jasu nebo hodnotám spektrálních složek signálu při barevném snímání. Vstupní signál je kvantován a vzorkován. Výsledkem těchto operací je matice čísel popisující obraz - *digitální obraz*. Jeden prvek matice představuje jeden obrazový element, který se nazývá *pixel*.

Vzhledem k tomu, že budeme potřebovat nalézt hrany v obraze, bude nás zajímat hlavně černobílý obraz - informace o barvě proto nebude důležitá. Proto je i barevný vstupní obraz na začátku zpracovávání zbaven barev. K tomu lze použít hned několik různých technik.

3.2 Detekce hran

Převzato z [**?**]

Lidské vnímání je založeno na rozpoznávání hran. Jednou z možností, jak zvýraznit nějaký obraz, abychom ho vnímali jako ostřejší, je zvýraznit v něm hrany.

Hranu (edge) v diskrétním obraze vnímáme tam, kde dochází k výrazné změně sousedních pixelů. Hrana je vysokofrekvenční informace, a proto je její zvýraznění inverzní operací k odstranění šumu. Hrana je určena *gradientem*, tj. velikostí a směrem. Směr lze popsat vektorovým operátorem nabla ∇

$$\nabla f(x, y) = (\frac{\partial f(x, y)}{\partial x}, \frac{\partial f(x, y)}{\partial y}) \tag{3.1}$$

a velikost gradientu je tedy určena jako délka vektoru:

$$|\nabla f(x,y)| \quad = \quad \sqrt{(\frac{\partial f(x,y)}{\partial x})^2 + (\frac{\partial f(x,y)}{\partial y}^2)} \tag{3.2}$$

Výše uvedené funkce platí pro spojité funkce. V diskrétním obraze gradient odhadujeme.

Ostření obrazu je pak založeno na následujícím postupu. Označíme si $s(i,j)$ jako funkci, která reprezentuje velikost gradientu obrazu f v bodě $[i,j]$. Výsledný obraz $g(i,j)$ získáme z obrazu f(i,j) ostřením pomocí koeficientu c.

$$g(i,j) \quad = \quad f(i,j) + c.s(i,j) \tag{3.3}$$

Funkce s(i,j) vrací velikost gradientu a o její patřičný násobek se zvýší intenzita pixelu v odpovídajícím bodě. Pro určení gradientu se používají postupy založené na analýze okolí pixelu s použitím konvolučních nebo jiných operátorů.

Robertsův operátor

Pravděpodobně nejjednodušší metodou určení velikosti gradientu pixelu je použití takzvaného *Robertsova operátoru*. Tento operátor není založen na konvoluci a jeho implementace je snadná. Robertsův operátor používá k výpočtu pixel a tři sousední pixely tohoto pixelu a má tvar:

$$|\nabla f(i,j)| \quad = \quad |f(i,j) - f(i+1,j+1)| + |f(i,j+1) - f(i+1,j)| \tag{3.4}$$

Velikost gradientu touto metodou se tedy určí jako součet absolutních hodnot změn ve směru hlavní a vedlejší diagonály obrázku. Robertsův operátor se používá především pro detekci hran se sklonem 45°.

Robertsův operátor můžeme rozdělit na dvě složky, z nichž každá detekuje hrany v jednom ze dvou na sebe kolmých směrů.

Sobelův operátor

Na podobném principu jako Robertsův operátor je založen Sobelův operátor (obr. 3.1). Tento operátor je směrově orientovaný, aproximuje první derivaci a pracuje s okolními pixely právě zkoumaného pixelu podle použité dvojice komplementárních konvolučních masek. Vždy je složen z dvojice komplementárních konvolučních masek označených jako h a $\bar{h}$. Komplementární maska se získá z původní masky rotací o devadesát stupňů kolem středu. Protože se v dalším výpočtu vypočítá druhá mocnina nebo absolutní hodnota, je nepodstatné, zda masku otáčíme doleva či doprava.

Jako komplementární konvoluční masky můžeme použít například tyto matice:

$$h = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \bar{h} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix},$$

$$h = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}, \bar{h} = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}.$$

Absolutní velikost gradientu je potom získána dvojnásobnou aplikací konvoluce, nejprve pro h a poté pro $\bar{h}$, a součtem:

$$|G| = \sqrt{h^2 + \bar{h}^2}. \tag{3.5}$$

Součet je tedy možno zjednodušit na:

$$|G| = |h| + |\bar{h}|. \tag{3.6}$$

Obrázek 3.1: Lena – Sobelův operátor, převzato z [**?**]

Laplaceův operátor

Další možnou metodou pro detekci hran je *Laplaceův operátor*. Výpočet pomocí tohoto operátoru je na rozdíl od předešlých případů založen na konvoluci. Laplaceův operátor se označuje jako Δ a pro výpočet ze čtyřokolí pixelu ve směru kolmém na souřadnicové osy má jeho konvoluční jádro tvar

$$h = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Varianta, která k výpočtu velikosti gradientu využívá hodnot z osmiokolí pixelu, má tvar

$$h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Laplaceův operátor je invariantní k otáčení o násobky 45°.

Cannyho hranový detektor

Další možností pro detekci hran v dvourozměrném obraze je Cannyho hranový detektor [**?**] (obr. 3.2). Cannyho hranový detektor je obecně znám jako optimální hranový detektor. Je navržen tak, aby splňovat tři základní požadavky, které detektory zmíněné dříve nezaručují. Tyto požadavky jsou:

1. minimální počet chyb - musí být detekovány všechny hrany a nesmí být žádná odezva na místa, která hranami nejsou
2. přesnost - poloha detekované hrany musí být určena co nejpřesněji
3. jednoznačnost - každá hrana může být detekována pouze jednou

Postup detekce hran v obraze pomocí tohoto detektoru by se měl skládat z následujících částí:

1. eliminace šumu Gaussovým filtrem
2. určení gradientu(první derivace)
3. nalezení lokálních maxim
4. eliminace nevýznamných hran

Obrázek 3.2: Lena – Cannyho hranový detektor, vytvořeno použitím programu IrfanView

Eliminace šumu Gaussovým filtrem

Dvourozměrná varianta Gausova normálního rozložení je dána vztahem

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.7}$$

kde x, y jsou souřadnice pixelu v obraze a σ je standardní odchylka rozdělení (běžně $\sigma = 1 - 1.4$).

Výpočet je vhodné realizovat pomocí konvoluce. Tímto vzorcem se vypočítá pouze konvoluční maska, která se pak aplikuje na celý obraz.

Velikost a směr gradientu

V této části algoritmu je nejvhodnější použít Sobelův operátor 3.2. Sobelův operátor totiž vrací nejen velikost gradientu hrany, ale také její směr. Směr hrany totiž potřebujeme v dalších krocích.

Nalezení lokálních maxim (thining)

Úkolem této části je vybrat z hodnot gradientů (stanovených v předchozím kroku) jen lokální maxima. Respektive odebrat body, které nejsou maximem. Tím zajistíme, že hrana bude detekována v místě největšího gradientu.

Toto znamená najít pixely, jejichž okolí je ve směru a proti směru gradientu nižší. Máme-li například pixel, jímž prochází svislá hrana, musí být jeho levý a pravý soused nižší hodnoty (hodnota jeho gradientu) aby byl určen jako skutečná hrana. Pokud podmínku nesplňuje, není označen za hranu. Které dva okolní pixely zahrnout do porovnávání je dáno směrem gradientu (určeno v předchozím kroku).

Eliminace nevýznamných hran (prahování)

V předchozím kroku jsme určili kde přesně leží hrany, ale doposud jsme se nezabývali významem hran. V tuto chvíli jsou označeny i ty nejmenší hrany, protože i ty mají své lokální maximum. Není vhodné určit jeden práh nad kterým budeme gradient považovat za významný, protože hodnota může kolísat například vlivem šumu.

Zvolíme si tedy minimální (T1) a maximální (T2) hodnotu (prahy) mezi kterými může gradient kolísat. Pokud hodnota gradientu daného pixelu leží nad vyšším prahem T2 je přímo označen jako hranový. Pokud posuzujeme bod, jehož hodnota leží mezi T1 a T2 pak je jako hrana označen jedině pokud sousedí s bodem který už byl jako hrana označen dříve.

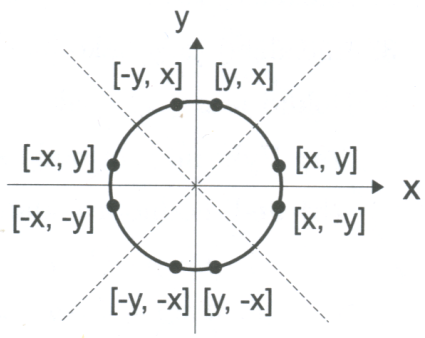
Cannyho hranový detektor je například pro detekci hran použit v programu IrfanView.

Operátory, které zvýrazňují hrany, zvýrazňují bez rozdílu všechny vysoké frekvence a tedy i šum. Operátory, které pracují s větší konvoluční maskou (s větším okolí pixelu), zvýrazňují šum méně.

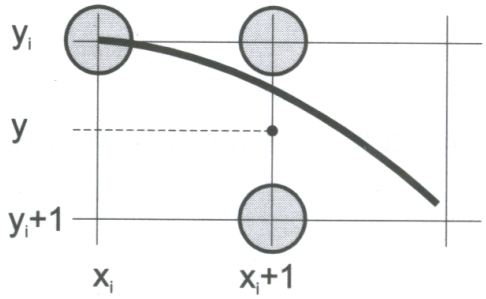
Součet hodnot konvolučních masek pro detekci hran musí být roven nule. Toto zaručí, že v oblastech s konstantní hodnotou bude i odezva konvoluce nulová.

3.3 Bresenhamův algoritmus pro kresbu kružnice

Též známý jako Midpoint algoritmus, popsaný např. v [?] je algoritmus pro kresbu kružnice, který při rasterizaci nachází body ležící nejblíže skutečné kružnici pouze pomocí celočíselné aritmetiky. V případě vykreslování kružnice je možno s výhodou využít skutečnosti, že kružnice je středově symetrická (obr. 3.3). Z jediného vypočítaného bodu kružnice lze tedy odvodit dalších sedm bodů pouhou záměnou souřadnic a změnou jejich znaménka. Pro vykreslení celé kružnice tedy stačí vypočítat hodnoty souřadnic bodů ležících v jednom oktantu, například v úseku od $x = 0$ do $x = y$.



Obrázek 3.3: Symetrické body na kružnici, převzato z [?]



Obrázek 3.4: Část kružnice v rastru, převzato z [?]

Pro popis postupu výpočtu jednotlivých bodů kružnice využiji obrázek 3.4, kde je zobrazena část rastru, který leží v jednom oktantu kružnice. V tomto oktantu se souřadnice x liší od sousedních bodů vykreslované kružnice právě o jeden pixel. Krok v ose x je tedy konstantní, vedlejší osou je osa y. Algoritmus začíná v bodě $[0, r]$ a končí v průsečíku kružnice s hlavní diagonálou, kdy $x = y$.

Pro body na kružnici platí implicitní rovnice $x^2 + y^2 - r^2 = 0$, kterou zapíšeme jako funkci:

$$F(x, y) : x^2 + y^2 - r^2 \quad = \quad 0 \tag{3.8}$$

Znaménko funkce určuje polohu bodu $[x, y]$ vůči kružnici. Funkční hodnota pro body uvnitř kružnice je záporná, pro body vně je kladná. Funkce F je proto vhodným kritériem při zavedení rozhodovacího členu. Předpokládejme, že bod $[x_i, y_i]$ byl určen jako bod nejbližší skutečné kružnici. Následující bod může tedy mít buď souřadnice $[x_i+1, y_i]$, nebo $[x_i+1, y_i-1]$. Na obrázku můžeme dále vidět naznačený bod ležící v polovině mezi dvěma uvedenými kandidáty (takzvaný midpoint). Dosadíme-li jeho souřadnice $[x_i + 1, y_i - \frac{1}{2}]$ do funkce 3.8, znaménko výsledku určí, zda tento bod leží vně nebo uvnitř kružnice. Výslednou hodnotu budeme opět nazývat rozhodovacím členem p_i:

1. Inicializuj pomocné proměnné: $dvex = 3$, $dvey = 2r - 2$
2. Inicializuj rozhodovací člen p na hodnotu $1 - r$
3. Inicializuj $[x, y]$ jako $[0, r]$
4. Dokud je $x \leq y$, opakuj:
   (a) Vykresli osm bodů symetrických s bodem $[x, y]$
   (b) Je-li hodnota p kladná, pak
      i. $p = p - dvey$
      ii. $dvey = dvey - 2$
      iii. $y = y - 1$
   (c) $p = p + dvex$
   (d) $dvex = dvex + 2$
   (e) $x = x + 1$

Algoritmus 3.1: Bresenhamův algoritmus pro kresbu kružnice

$$\begin{aligned} p_i &= F(x_i + 1, y_i - \frac{1}{2}) \\ &= (x_i + 1)^2 + (y_i - \frac{1}{2})^2 - r^2 \end{aligned} \tag{3.9}$$

Je-li znaménko p_i záporné, bude pro další kresbu vybrán bod se stejnou souřadnicí y_i, jinak bude vykreslen bod ležící o jeden pixel níže.

Hodnotu iteračního členu budeme určovat během iteračního výpočtu z předcházející hodnoty. Po úpravě tedy získáme výsledný vzorec:

$$p_{i+1} = p_i + 2x_i + 3 + (y_i - \frac{1}{2})^2 + (y_{i+1} - \frac{1}{2})^2 \tag{3.10}$$

Vzorec 3.10 můžeme tedy vyčíslit podle znaménka p_i:

$p_i \leq 0$	$p_{i+1} = p_i + 2x_i + 3$
$p_i > 0$	$p_{i+1} = p_i + 2x_i + 5 - 2y_i$

I přesto, že vzorce pro aktualizaci rozhodovacího členu obsahují násobení, lze je implementovat jen pomocí sčítání a odečítání. Pro členy $2x_i$ a $2y_i$ zavedeme pomocné proměnné $dvex$ a $dvey$, obsahující dvojnásobek příslušné souřadnice. Když se hodnota x, respektive y změní o jedničku, aktualizujeme příslušnou pomocnou proměnnou přičtením, respektive odečtením dvojky. Tyto proměnné inicializujeme tak, abychom se ve výše uvedené tabulce zbavili konstant tři a pět.

3.4 Algoritmus RANSAC

Převzato z [**?**].

RANSAC je zkratka z *RANdom SAmple Consensus*. Je to iterativní metoda určení parametrů matematického modelu z dodaných dat. Je to nedeterministický algoritmus, což znamená, že požadované výsledky pomocí něj získáme jen s určitou pravděpodobností. Tato pravděpodobnost se zvyšuje s počtem iterací, které algoritmu umožníme.

Základní předpoklad pro tento algoritmus je, že data lze rozdělit na takzvané *inliers*, což jsou data, která vyhovují hledanému modelu a *outliers*, což jsou naopak data, která hledanému modelu nevyhovují. Tento algoritmus je hodně závislý na okolním šumu v obraze. Algoritmus RANSAC také obsahuje funkci, která ověřuje, zda jsme nalezli dostatečný počet *inliers* a lze tak s určitou jistotou určit, zda jsme nalezli opravdu hledaný objekt.

Vstupem algoritmu RANSAC jsou data, ve kterých hledáme objekt, parametrický popis hledaného objektu a parametry, pomocí kterých lze určit, že jsme již hledaný objekt nalezli. RANSAC dosahuje svého cíle iterativním vybíráním náhodné podmnožiny dat. Tyto data označí jako *hypotetické inliery* a provede s nimi následující testy:

1. Pokusí se zrekonstruovat model tak, že všechny nalezené *hypotetické inliery* umístí tak, aby co nejlépe odpovídaly hledanému modelu.

2. Všechny ostatní data jsou poté také otestována a pokud je lze umístit do hledaného modelu, jsou také zařazeny do skupiny *hypotetických inlierů*.

3. Nalezený model můžeme označit za dobrý, pokud bylo dostatečně mnoho bodů klasifikováno jako *hypotetické inliery*.

4. Algoritmus se následně pokusí zrekonstruovat model pomocí všech nalezených *hypotetických inlierů*, protože model byl rekonstruován pouze pomocí počáteční skupiny *hypotetických inlierů*.

5. Nakonec je model porovnán s parametry, které určují, zda jsme skutečně nalezli hledaný model a pokud nalezený model vyhoví těmto parametrům, je označený za skutečný.

Tyto kroky se opakují určitý předem daný počet iterací. Z každé iterace dostaneme jako výsledek buď nově nalezený model, nebo nedostaneme nic. Pokud dostaneme nový model, porovnáme ho s nejlepším zatím nalezeným modelem a pokud má lepší vlastnosti, než předchozí model, uchováme si tento nově nalezený model.

Určení parametrů

Hodnoty parametrů t a d musí být určeny podle druhu dat, ve kterých hledáme model a podle speciálních požadavků, které na tento hledaný model klademe. Parametr k (počet iterací) může být zjištěn pomocí teoretických výpočtů. Nechť p je pravděpodobnost, že algoritmus v některé iteraci vybere ze vstupních dat jenom n inlierů, ze kterých může být model sestaven. Pokud se toto stane, model bude pravděpodobně považován za dobrý, takže p udává pravděpodobnost, že algoritmus v jednom kroku nalezne užitečný výsledek. Nechť w je pravděpodobnost, že pokaždé, kdy se vybere bod z dat, bude tento bod patřit mezi *inliery*.

$$w = \frac{pocet\ inlieru\ v\ datech}{celkovy\ pocet\ bodu\ v\ datech} \tag{3.11}$$

Častým případem je, že w není dopředu známo, ale můžeme určit alespoň hrubý odhad této hodnoty. Pokud předpokládáme, že n bodů potřebných pro sestavení modelu je vybráno samostatně, w^n je pravděpodobnost, že všechny nalezené body patří mezi *inliery* a $1-w^n$ je pravděpodobnost, že alespoň jeden z n nalezených bodů je *outlier*, tedy že dojde k nalezení špatného modelu. Pravděpodobnost $(1-w^n)^k$ je pravděpodobnost, že algoritmus nikdy nevybere sadu n bodů, ve kterých jsou všechny body označeny jako *inliery*. Tato hodnota musí být stejná jako $1-p$, proto lze napsat

$$1-p = (1-w^n)^k \tag{3.12}$$

což po zlogaritmování obou stran vede na vzorec

$$k = (log(1-p))/(log(1-w^n)) \tag{3.13}$$

Je ale potřeba zdůraznit, že tento vzorec platí pouze v případě, pokud n bodů z dat bylo vybráno nezávisle na sobě, což znamená, že pokud byl vybraný bod vyřazen jako nevyhovující, je možné ho ve stejné iteraci opět vybrat jako vyhovující. Tento výpočet počtu iterací je ale zdlouhavý, proto se častěji počet iterací odvozuje od způsobu, jakým chceme model v datech hledat.

Výhody a nevýhody

Výhoda algoritmu RANSAC je robustní odhad parametrů modelu, z čehož vyplývá, že algoritmus je schopen najít model i pokud data obsahují větší množství *outlierů*. Nevýhodou tohoto algoritmu je neexistence horní hranice, jak dlouho má toto hledání probíhat. Pokud horní hranici (počet iterací) nastavíme, pak je možné, že neobdržíme optimální výsledek, v nejhorším případě to může být výsledek, který se vůbec v datech nenachází. Dobrý model může být tímto algoritmem vrácen pouze s určitou pravděpodobností. Tato pravděpodobnost nalezení správného modelu se zvyšuje se zvyšujícím se počtem iterací algoritmu. Další nevýhodou tohoto algoritmu je potřeba nastavení jednotlivých prahů podle hledaného modelu.

Algoritmus RANSAC může být použit pouze pokud se v prohledávaných datech nalézá jeden jediný model, který chceme najít. Pokud se v datech nachází dva a více modelů, je pravděpodobné, že algoritmus nenalezne ani jeden z těchto modelů.

Algoritmus RANSAC pro hledání kružnice

Algoritmus RANSAC lze s omezeními popsanými výše využít například pro vyhledávání kružnic v obraze. Tuto verzi algoritmu si popíšeme pro jednu iteraci. Ještě před započetím práce algoritmu se barevný obraz převede na černobílý a nadetekují se v něm hrany. Pixely v obraze označující hrany se setřídí do jednoho pole.

Z tohoto pole bílých pixelů se vyberou náhodně tři pixely, o kterých budeme předpokládat, že jsou to pixely na obvodu hledané kružnice. Vypočítáme si z nich tedy střed a poloměr pomyslné kružnice pomocí následujících vzorců:

$$
\begin{aligned}
bax &= x_2 - x_1 && (3.14)\\
bay &= y_2 - y_1 && (3.15)\\
cax &= x_3 - x_1 && (3.16)\\
cay &= z_3 - z_1 && (3.17)\\
E &= bax \cdot (x_2 + x_1) + bay \cdot (y_1 + y_2) && (3.18)\\
F &= cax \cdot (x_1 + x_3) + cay \cdot (y_1 + y_3) && (3.19)\\
G &= 2 \cdot (bax \cdot (y_3 - y_2) - bay \cdot (x_3 - x_2)) && (3.20)\\
cx &= \frac{cay \cdot E - bay \cdot F}{G} && (3.21)\\
cy &= \frac{bax \cdot F - cax \cdot E}{G} && (3.22)\\
dx &= cx - x_1 && (3.23)\\
dy &= cy - y_1 && (3.24)\\
rad &= \sqrt{dx^2 + dy^2} && (3.25)
\end{aligned}
$$

kde

- x_1, x_2, x_3, y_1, y_2 a y_3 jsou souřadnice tří náhodně vybraných bodů
- cx a cy jsou souřadnice středu detekované kružnice
- rad je poloměr detekované kružnice

Poté procházíme obraz a zkoušíme, které pixely označující hrany jsou ve vzdálenosti poloměru předpokládané kružnice. Pokud těchto pixelů nalezneme dostatečný počet větší než určený práh, střed a poloměr této kružnice si zapamatujeme, dokud neukončíme algoritmus po určitém počtu kroků nebo dokud v některém dalším kroku nenalezneme kružnici s parametry, které nám vyhovují lépe.

3.5 Houghova transformace

Převzato z [**?**].

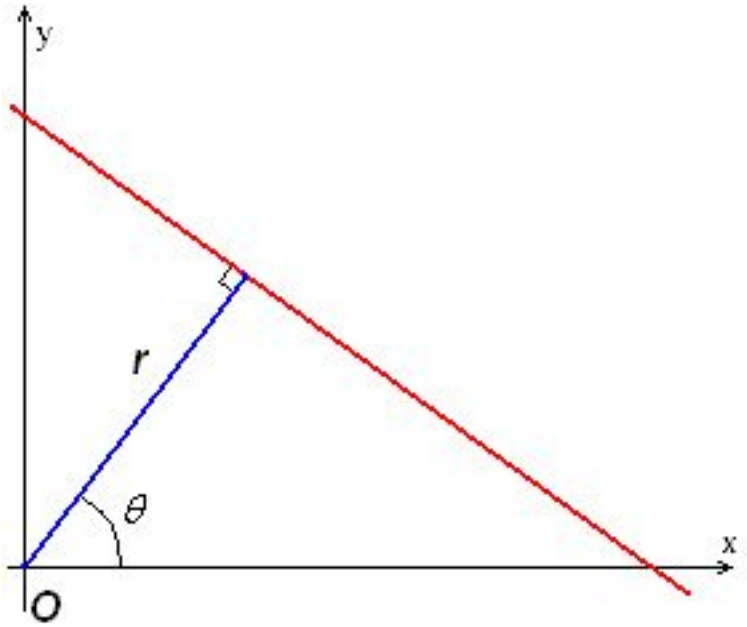
Houghova transformace je metoda pro nalezení parametrického popisu objektů v obraze. Při implementaci je třeba znát analytický popis tvaru hledaného objektu. Tato metoda je proto používána pro detekci jednoduchých objektů v obraze, jako jsou přímky, kružnice, elipsy, atd. Výhodou této metody je její robustnost. Tato metoda totiž není příliš citlivá na šum.

Nejjednodušší případ Houghovy transformace je lineární transformace pro hledání přímek. Přímka může být popsána pomocí následující funkce

$$y = mx + b \qquad (3.26)$$

a může být jednoduše nakreslena do obrazu za použití obrazových bodů (x, y). Hlavní idea Houghovy transformace je nebrat v úvahu přímku jako posloupnost jednotlivých obrazových bodů, ale místo toho brát v úvahu parametry této přímky. V uvedené funkci jsou to

parametry m pro sklon přímky a b pro posun přímky. Tento přístup je založen na faktu, kdy může být přímka $y = mx + b$ reprezentována jako bod (b, m) v parametrickém prostoru. Ale protože parametry m a b nejsou u svislých přímek nijak omezeny, zavádí se další dva parametry r a Θ.



Obrázek 3.5: Parametrické vyjádření přímky, převzato z [?]

Parametr r reprezentuje vzdálenost mezi přímkou a počátkem souřadné soustavy, zatímco parametr Θ reprezentuje úhel, který svírá vektor směřující z počátku souřadné soustavy k nejbližšímu dobu přímky.

Parametricky lze přímku vyjádřit následujícím vzorcem.

$$y = (-\frac{\cos(\Theta)}{\sin(\Theta)}) \cdot x + \frac{r}{\sin(\Theta)} \tag{3.27}$$

který lze přepsat na vzorec

$$r = x \cdot \cos(\Theta) + y \cdot \sin(\Theta) \tag{3.28}$$

Proto je možno každé přímce v obrazu přiřadit dvojici (r, Θ), která je jednoznačná, pokud $\Theta \in [0, \pi)$ a $r \in R$, nebo pokud $\Theta \in [0, 2\pi)$ a $r \geq 0$. Rovina (r, Θ) se nazývá *Houghův prostor*.

Skrz jeden bod roviny je možno vést nekonečně mnoho přímek. Pokud je tento bod v obraze reprezentován body (x_0, y_0), pak pro všechny přímky procházející tímto bodem platí

$$r(\Theta) = x_0 \cdot \cos(\Theta) + y_0 \cdot \sin(\Theta) \tag{3.29}$$

Toto odpovídá sinusoidě v rovině (r, Θ), která je pro tento bod jedinečná. Pokud v Houghově prostoru zobrazíme dvě sinusoidy, které odpovídají dvěma různým bodům, pak místo v Houghově prostoru, kde se protnou, odpovídá přímce v původním obrazu. Obecně tedy lze říci, že sada bodů, které tvoří přímku, vytvoří sinusoidy, které se všechny protnou v bodě, který odpovídá parametrům hledané přímky v obrazové rovině.

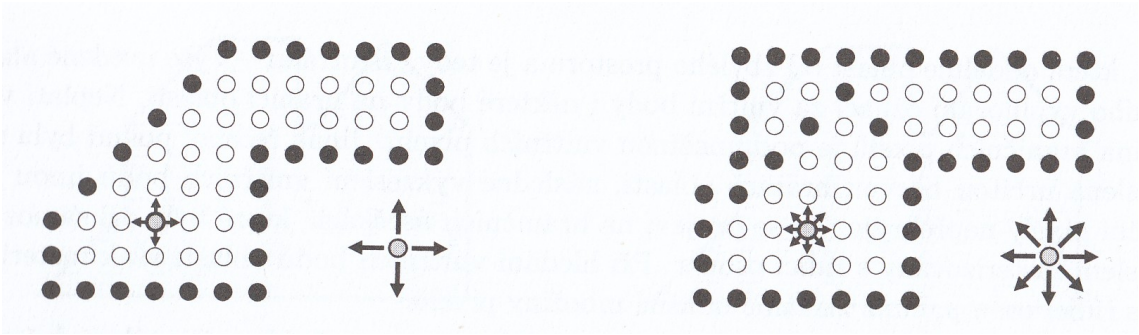
3.6 Vyplňování hranice nakreslené v rastru

Převzato z [?].

Metody pro vyplňování již nakreslené hranic v rastru jsou obecně nazývány jako *semínkové vyplňování* (*seed fill*), neboť jejich nezbytným parametrem jsou souřadnice *semínka* - vybraného vnitřního bodu oblasti. Vzhledem k tomu, že hranice oblasti není předem geometricky jasně definována, všechny informace o oblasti se získávají čtením z obrazové (rastrové) paměti. Od semínka se postupně rozšiřuje prohledávání obrazové paměti a nalezeným vnitřním bodům se nastaví nová barva.

Směr prohledávání a vyplňování úzce souvisí s charakterem hranice, respektive s tím, co považujeme za vnitřní část oblasti. Rozlišujeme *4spojité* (*4souvislé*) a *8spojité* (*8souvislé*) oblasti podle toho, zda mezi jejich libovolnými body existuje cesta složená ze *4spojitých*, respektive *8spojitých* spojnic.

Jak lze vidět na obrázku 3.6, každému typu oblasti odpovídá jiný typ hranice. Například Bresenhamův algoritmus 3.3 vytváří *8spojité* posloupnosti pixelů, které ohraničují *4spojité* oblasti. Pro ohraničení *8spojité* oblasti musíme použít *4spojitou* hranici.



Obrázek 3.6: 4spojitá oblast (vlevo) ohraničená 8spojitou hranicí a 8spojitá oblast (vpravo) ohraničená 4spojitou hranicí, (semínko zvýrazněno), převzato z [?]

Při semínkovém vyplňování tedy postupujeme od zadaného semínka a zkoumáme, zda jeho sousední body patří k vnitřní části oblasti. O příslušnosti bodu k oblasti rozhodujeme podle některé testované vlastnosti, například podle barevné intenzity. Existují dvě základní varianty semínkového vyplňování:

1. Hraniční semínkové vyplňování - testovaný bod je vnitřní, pokud má testovanou vlastnost odlišnou od vlastnosti hranice, například má-li jinou barvu.

2. Záplavové semínkové vyplňování - testovaný bod je vnitřní, pokud má testovanou vlastnost shodnou jako zadané semínko. Tato metoda se také nazývá lavinové vyplňování či přebarvování.

1. Umísti semínko (x, y)
2. Pokud je bod $[x, y]$ vnitřním bodem oblasti a dosud nebyl obarven, pak
   (a) Obarvi bod $[x, y]$ požadovanou barvou
   (b) Umísti semínko $(x + 1, y)$
   (c) Umísti semínko $(x - 1, y)$
   (d) Umísti semínko $(x, y + 1)$
   (e) Umísti semínko $(x, y - 1)$

Algoritmus 3.2: Semínkové vyplňování rekurzivním postupem, převzato z [**?**]

Test porovnání vlastností může být složitější, než jen ověření shody dvou hodnot. Můžeme například požadovat, aby byla testovaná vlastnost v určitém rozsahu hodnot, například v intervalu jasu či barevného odstínu. Tato verze testu je vhodná v okamžiku, kdy byla hranice oblasti vyhlazena a získala nestejný odstín.

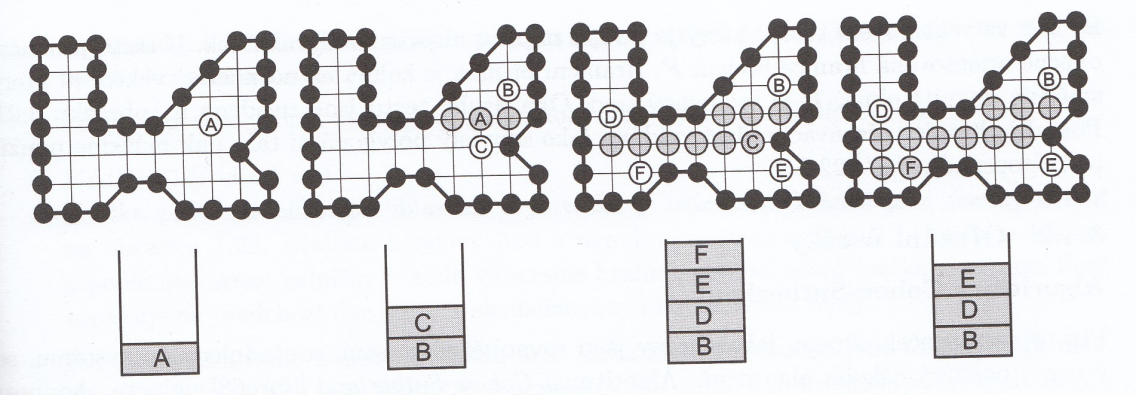
Nejjednodušší metoda je popsána rekurzivním algoritmem 3.2. Na první pohled elegantní, přehledný zápis je ve skutečnosti téměř nepoužitelný. Šíření semínek do všech směrů má totiž za následek , že každý vnitřní pixel je testován několikrát i poté, co již byl obarven. Čtení z obrazové paměti přitom patří mezi pomalé operace.

Řádkové semínkové vyplňování

Metoda, která snižuje počet přístupů do rastrové paměti,se nazývá *řádkové semínkové vyplňování* (*scan-line seed fill*). Je uvedena v algoritmu 3.3. Princip algoritmu spočívá ve vyplňování souvislých vodorovných úseků a prohledávání intervalů nad a pod těmito úseky. Každá vodorovná řada vnitřních bodů nad, respektive pod daným úsekem tvoří nový úsek, který je rekurzivně zpracován. Algoritmus je třeba inicializovat nalezením prvního úseku, který obsahuje zadané semínko.

Na obrázku yrefseminka2 je nakreslen stav vyplňované oblasti při postupném provádění algoritmu. Starovním semínkem je bod A. Při inicializaci je nalezen úsek obsahující tento bod a končící v hranicích oblasti. Poté je možno zavolat algoritmus 3.3.

Po vyplnění úseku v okolí bodu A jsou testovány intervaly nad a pod tímto úsekem. Výsledkemje volání algoritmu pro úseky s bodem B (z horní oblasti) a C (z dolní oblasti). Po vyplnění úseku s bodem C jsou nalezeny volné úseky s body D, E a F. Postupně je vyplněna oblast pod úsekem E, bad úsekem D a nakonec nad úsekem B.



Obrázek 3.7: Postup při řádkovém semínkovém vyplňování, převzato z [?]

- VyplňÚsek(x, x_L, x_R)

  1. vyplň pixely v úseku od $[x_L, y]$ do $[x_R, y]$
  2. v (horním) intervalu mezi $[x_L, y-1]$ a $[x_R, y-1]$ hledejsouvislé vnitřní úseky. Pro každý i-tý úsek proveď:

     VyplňÚsek$(y-1, x_{Li}, x_{Ri})$
  3. v (dolním) intervalu mezi $[x_L, y+1]$ a $[x_R, y+1]$ hledej souvislé vnitřní úseky. Pro každý j-tý úsek proveď:

     VyplňÚsek$(y+1, x_{Lj}, x_{Rj})$

Algoritmus 3.3: Řádkové semínkové vyplňování, převzato z [?]

Kapitola 4

Návrh detektoru zoubkování známek

Tato kapitola obsahuje obecný popis problémů řešených v aplikaci a nastiňuje možnosti jejich řešení.

Program, který se v této práci pokusím navrhnout, bude na vstupu požadovat obrázek poštovní známky. Výstupem bude hodnota zoubkování na jednotlivých stranách rozpoznané poštovní známky.

Aplikace bude pracovat v několika krocích:

1. načtení vstupního obrázku obsahujícího poštovní známku pro určení jejího zoubkování
2. prahování - převede vstupní obrázek na černobílý
3. vyčištění okolí známky
4. detekce poštovní známky v obraze
5. detekce středů perforací
6. určení zoubkování

Blokové schéma na obrázku 4.1 znázorňuje princip práce programu.

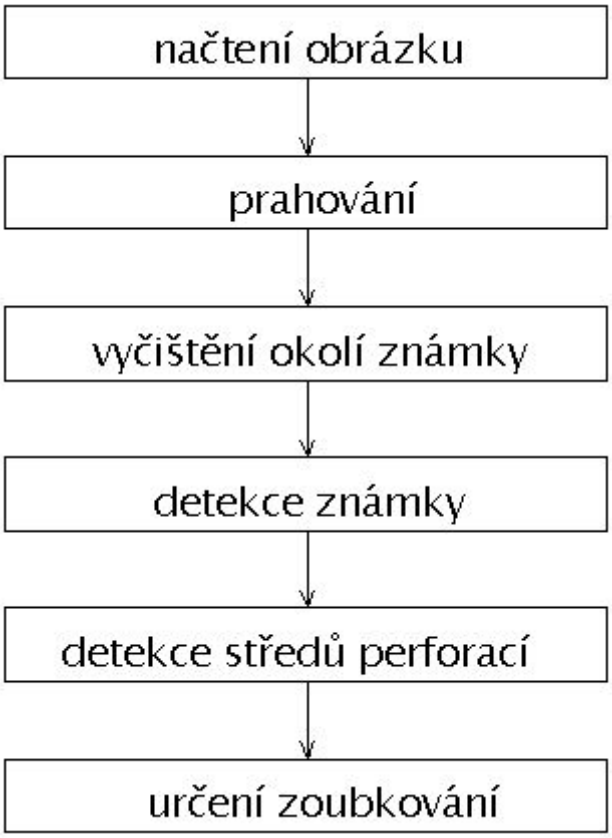
Implementační detaily jednotlivých kroků jsou popsány dále v kapitole 5. Nyní bude následovat hrubý náčrt principu těchto kroků.

4.1 Načtení vstupního obrazu

Načtení obrazu je nutné před každým zpracováním obrazu. Obraz je načten podle jména souboru obsahujícího zkoumanou známku.

4.2 Prahování

Prahování zbaví obraz barevné informace, která je pro další kroky zbytečná a vstupní obraz převede na černobílý. Prahování funguje na jednoduchém principu, kdy hodnota jasu jednotlivých bodů je srovnána se zvolenou hodnotou prahu. V případě, že je hodnota jasu menší než hodnota prahu, je tento bod označen jako černý, v opačném případě jako bílý. Pokud je vstupní obraz barevný, provede se nejdříve jeho převedení na stupně šedi pomocí rovnice $y = 0,299 \cdot R + 0,587 \cdot G + 0,114 \cdot B$, kde R, G a B jsou jednotlivé barevné složky (červená, zelená a modrá) a y je výsledná hodnota jasu. Prahování slouží k segmantaci obrazu - rozděluje obraz na části, které jsou známka a které jsou pozadí.



Obrázek 4.1: Blokové schéma návrhu detektoru zoubkování poštovních známek

4.3 Vyčištění okolí známky

Dalším krokem programu je odstranění nežádoucích objektů z okolí poštovní známky. Tato operace je důležitá, protože pokud bychom ji neprovedli a známka by nebyla vyfocena na nějakém konstantním pozadí, mohlo by toto pozadí zanést chybu do následujících funkcí programu. Toto vyčištění okolí lze provádět například opakovanou aplikací filtrů, které slouží pro rozostřování obrazu s následnou opětovnou detekcí hran.

4.4 Detekce poštovní známky

Poté, co jsme z obrazu odstranili okolní šum, je potřeba, abychom nějakým způsobem zjistili, kde přesně se v obraze hledaná známka nachází, abychom v dalších částech programu nemuseli procházet celý obraz, ale abychom procházeli pouze jeho užitečnou část. Pro tuto část programu se hodí například algoritmus RANSAC upravený pro vyhledávání obdélníků. Kromě algoritmu RANSAC je možno například poštovní známku detekovat pomocí histogramů.

4.5 Detekce středů perforací

Po detekci hran poštovní známky je dále potřeba určit středy perforací na jednotlivých stranách detekované poštovní známky. K tomuto je možno například využít algoritmus RANSAC upravený pro vyhledávání kružnice, nebo lze také využít Houghovu transformaci upravenou pro vyhledávání kružnic.

Vzhledem k tomu, že existují i poštovní známky, které mají na každé straně jiné zoubkování, je nutné tuto detekci středů perforací provést pro každou stranu poštovní známky zvlášť.

V této části programu se musíme nějakým způsobem vypořádat hlavně s rozlišením obrazu, protože i pro stejné obrázky, jenom v jiném rozlišení nám budou námi detekované poloměry středů perforačních otvorů vycházet jako velmi rozdílná čísla.

Nepotřebujeme zde zjišťovat přesné rozlišení obrazu, jenom v této části musíme toto rozlišení nějakým způsobem zohlednit.

4.6 Určení zoubkování

Poslední část programu by se měla věnovat již konečnému určování zoubkování.

V této části programu již přesné rozlišení obrazu zjistit potřebujeme, abychom mohli poté přepočítat nalezené počty středů na vzdálenost 2 cm, ve které se zoubkování poštovních známek udává.

Vzhledem k tomu, že knihovna OpenCV neobsahuje ve struktuře načítaných obrázků jejich rozlišení, musíme v této části využít některého jiného způsobu určování rozlišení obrázku.

Tuto část by bylo možno řešit detekcí nějakého předem známého objektu, který by byl vyfocen společně s poštovní známkou, například detekcí kruhu o předem známém poloměru. Tuto detekci je nutno dělat ještě před detekcí samotné poštovní známky a tento detekovaný objekt je nutno odstranit, aby nedocházelo ke zkreslení detekovaných hran poštovní známky. Detekci kruhu by bylo možno provádět například pomocí algoritmu RANSAC. Tento algoritmus však z důvodu vybírání náhodných bodů nemusí vykazovat uspokojivé výsledky.

Vzhledem k faktu, že používané průměry perforačních jehel se od sebe příliš neliší, je také možno při zjišťování rozlišení obrazu vycházet z poloměrů nadetekovaných perforačních otvorů.

Následující tabulka udává přibližné rozlišení obrázku vzhledem k průměrům detekovaných perforačních otvorů.

rozlišení obrazu	průměr perforačních otvorů
1200 DPI	> 40
600 DPI	$20 - 40$
300 DPI	$10 - 20$
150 DPI	< 10

Pomocí této tabulky tak lze odhadnout rozlišení obrazu. Toto rozlišení určuji pro každou stranu poštovní známky zvlášť. Pokud určím stejné rozlišení na všech čtyřech stranách známky, pak lze předpokládat, že vstupní obraz má toto rozlišení.

Po zjištění rozlišení obrazu již tedy pouze přepočítáme počty nalezených středů perforačních otvorů.

Stejně jako v předchozím případě detekci středů perforací, je nutno i tuto klasifikaci provádět pro každou stranu poštovní známky zvlášť.

Kapitola 5

Implementace

V této kapitole je podrobněji popsána vlastní implementace programu. Budou zde vysvětleny použité postupy i algoritmy.

Program pracuje s obrázky ve formátu bmp a jpg, získanými pomocí digitálního fotoaparátu nebo skeneru. Obrázky poštovních známek není potřeba před zpracováním pomocí programu nijak upravovat, ale je vhodné, aby byly známky vyfoceny na kontrastním pozadí a aby byly otočeny zadní stranou vzhůru. Algoritmy, které jsem v programu použil totiž na nepříliš kontrastním pozadí nevykazují dobré výsledky a vyfocení známky přední stranou vzhůru také zanáší do algoritmů chybu.

Program jsem psal v programovacím jazyce C++, v programu Dev-cpp, pod operačním systémem Windows 7. V programu využívám knihovnu OpenCV ve verzi 1.1. Aby bylo vysvětlování mnou vytvořených algoritmů názornější, budu je doplňovat ukázkovými obrázky.

5.1 Knihovna OpenCV

OpenCV (Open Computer Vision Library) [?] je volně dostupná grafická knihovna původně vyvíjená společností Intel. Má bohatou zásobu funkcí zaměřených převážně na zpracování obrazu, ale i na tvorbu uživatelského rozhraní, práci s grafickými formáty a mnoho dalších.

V programu ji využívám právě kvůli funkcím pro zpracování obrazu. Konkrétně z ní využívám Cannyho hranový detektor, prahování a práci se vstupním formátem obrazu.

5.2 Parametry programu

Parametry programu

```
projekt filename [-o] [-d] [-dpi rozliseni]
```

Program pro své spuštění požaduje minimálně jeden parametr, kterým je název souboru včetně přípony obsahujícího obraz známky. Knihovna OpenCV podporuje soubory typu bmp a jpg.

Název souboru musí být uveden jako první parametr.

Několik následujících parametrů je nepovinných

- -o - tento parametr udává, zda se bude zobrazovat výstupní obrázek

- -dpi rozliseni - tímto parametrem je možno zadat rozlišení vstupního obrazu. Pokud je tento parametr zadán, rozlišení se již v programu neurčuje
- -d - program bude kromě názvu vstupního obrazu a zoubkování na jednotlivých stranách vypisovat také podrobnější kontrolní výpisy.

Příklad spuštění programu

```
projekt test.bmp -o -dpi 300
```

Program vypisuje na standardní výstup výsledky v následujícím formátu

```
filename ZoubkovaniVlevo ZoubkovaniVpravo ZoubkovaniNahore ZoubkovaniDole
```

Program je primárně určen pro dávkové zpracování. Nepovinné parametry slouží pouze pro ladění, případně ke kontrole výsledků.

5.3 Načtení vstupního obrazu

V této části využívám knihovnu OpenCV, respektive její funkci *cvLoadImage*. Touto funkcí načtu vstupní obraz a převedu jej na tříkanálový. Název zpracovávaného obrazu očekávám jako první parametr zadávaný při spuštění programu. Ukázka části obrazu známky načteného pomocí funkce *cvLoadImage* je na obrázku 5.1.

```
IplImage* cvLoadImage( const char* filename, int flags=CV_LOAD_IMAGE_COLOR );
```

Prvním parametrem funkce cvLoadImage je název otevíraného souboru, druhým parametrem je barevnost a bitová šířka obrázku.

5.4 Prahování

Obrázek, který jsem v předchozím kroku načetl je tříkanálový. V tomto kroku provedu jeho převod na jednokanálový černobílý obraz, se kterým budu v programu dále pracovat. Tento převod provádím opět pomocí funkcí OpenCV ve dvou krocích. V prvním kroku provedu převedení tříkanálového obrazu na jednokanálový pomocí funkce *cvCvtColor*, která barevný obraz převede na obraz v odstínech šedi. Obraz v odstínech šedi ale také ještě není pro tento program vhodný, proto v následujícím kroku takto upravený obraz dále upravím pomocí prahování. K tomuto prahování využívám Cannyho hranový detektor obsažený v knihovně OpenCV, tedy funkci *cvCanny*. Obraz upravený pomocí Cannyho hranového detektoru dále využívám v následujících částech programu.

```
void cvCvtColor( const CvArr* src, CvArr* dst, int code );
```

Prvním parametrem této funkce je vstupní obraz, druhým parametrem je výstupní obraz, třetím parametrem je kód udávající převodní vztah mezi obrázky.

Obrázek 5.1: Výchozí obraz známky

```
void cvCanny( const CvArr* image, CvArr* edges, double threshold1,
              double threshold2, int aperture_size=3 );
```

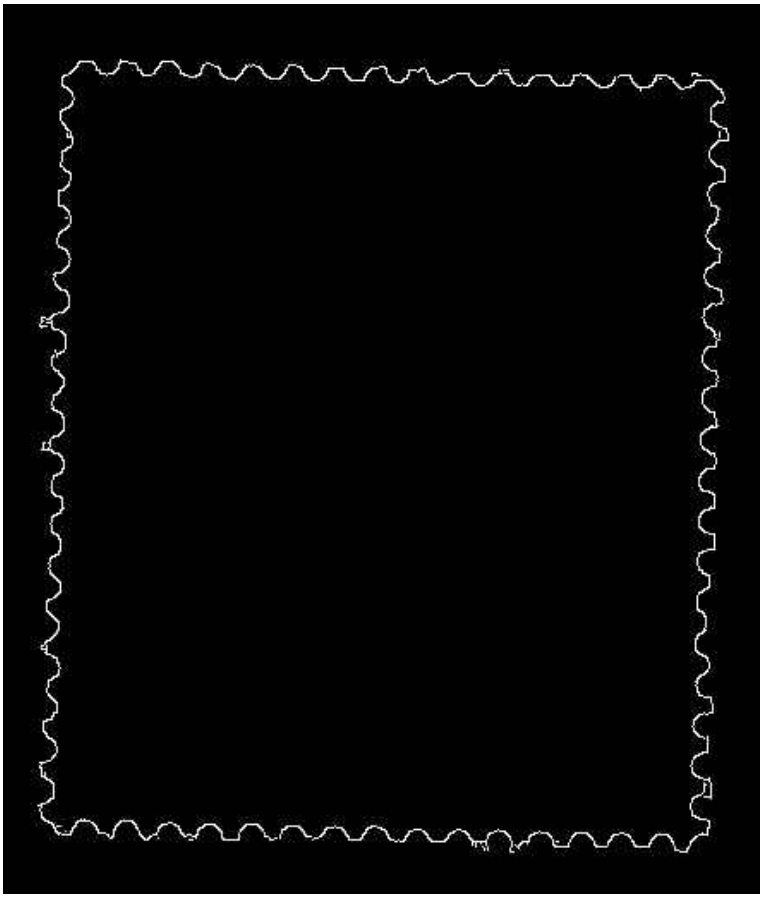
Prvním parametrem této funkce je vstupní obraz, druhým parametrem je výstupní obraz, třetím parametrem je první threshold, čtvrtým parametrem je druhý threshold. Posledním parametrem je velikost kernelu pro Sobelův operátor.

5.5 Vyčištění okolí známky

Tuto část programu jsem implementoval opět pomocí funkcí OpenCV, konkrétně s využizím funkce *cvSmooth*. Pomocí této funkce provedu rozmazání obrazu pomocí filtru *Gaussian blur*. Po použití funkce *cvSmooth* dojde k rozmazání obrazu. Následně provádím opětovné prahování obrazu pomocí Cannyho hranového detektoru - pomocí OpenCV funkce *cvCanny*. Tuto akci - rozmazání a následné prahování provádím celkem dvakrát, aby došlo k vyčištění co největší části okolí známky.

Případné další vícenásobné použití funkce *cvSmooth* a Cannyho hranového detektoru nemá již na vstupní obraz velký vliv, proto tyto funkce provádím pouze dvakrát.

obraz po použití uvedených funkcí je na obrázku 5.2.



Obrázek 5.2: Obraz známky po vyčištění okolí známky

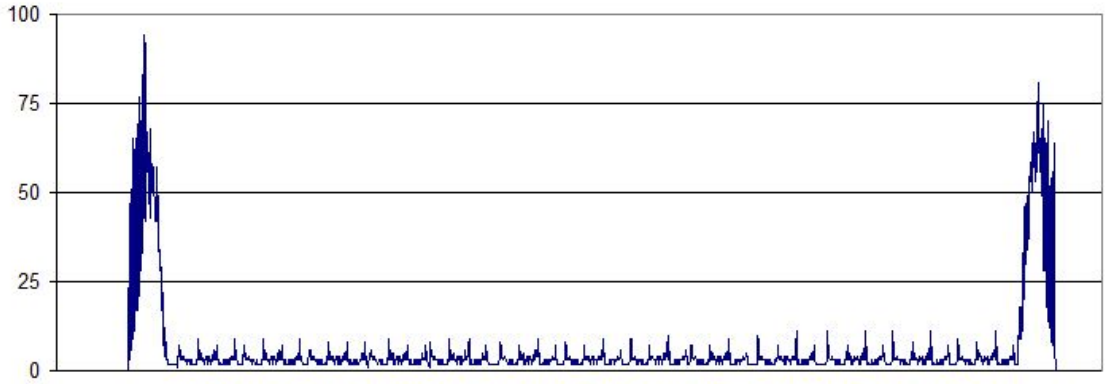
```
void cvSmooth( const CvArr* src, CvArr* dst,
               int smoothtype=CV_GAUSSIAN,
               int param1=3, int param2=0, double param3=0, double param4=0 );
```

Prvním parametrem této funkce vstupní obraz, druhým parametrem je výstupní obraz. Třetí parametr udává typ filtru, který se pro rozmazání použije. Následující parametry - param1, param2, param3 a param4 udávají informace o použitém filtru.

5.6 Detekce poštovní známky

V této části programu procházím obrázek upravený Cannyho hranovým detektorem a zbavený nežádoucího šumu postupně po jednotlivých řádcích a poté po jednotlivých sloupcích. Při těchto průchodech si zaznamenávám počty bílých pixelů a poté z nich vytvořím dva histogramy. Jeden histogram pro řádky (výšku) obrázku a druhý histogram pro sloupce (šířku) obrázku. Ukázkový histogram je zobrazen na obrázku 5.3. x-ová osa označuje pozice jednotlivých pixelů, na y-ové ose jsou zobrazeny počty nalezených bílých pixelů.

Pomocí těchto histogramů poté programem určím pozici poštovní známky v obraze a dále určím rozsahy v okolí jednotlivých hran známky, ve kterých budu poté provádět vyhledávání



Obrázek 5.3: Ukázka histogramu pro detekci hran známky

jednotlivých středů perforací.

Zjišťování rozsahů z histogramů provádím v následujících krocích.

1. Naleznu nejvyšší bod v levé části histogramu
2. Od tohoto bodu postupuji v histogramu směrem doleva a hledám místo, kde jsou hodnoty pod 10% maximální hodnoty. Toto místo si zapamatuji.
3. Od bodu nalezeného v prvním kroku nyní v histogramu postupuji směrem doprava a opět hledám místo, kdy je dosaženo hodnoty pod 10% maximální hodnoty. Toto místo si opět zapamatuji.
4. Nyní opět naleznu nejvyšší bod v histogramu, ale tentokráte v jeho pravé části a opakuji doby 2. a 3.

Uvedeným postupem získám celkem osm míst. Z těchto nalezených míst vytvořím celkem osm bodů, které tvoří okolí hrany poštovní známky na obrázku. Čtyři body označující vnější hranici a zbylé čtyři body označující vnitřní hranici.

Hranice poštovní známky, které byly detekovány tímto způsobem jsou zobrazeny na obrázku 5.4.

Následující část programu provádím pouze v takto vymezené oblasti, aby se detekce středů perforací urychlila.

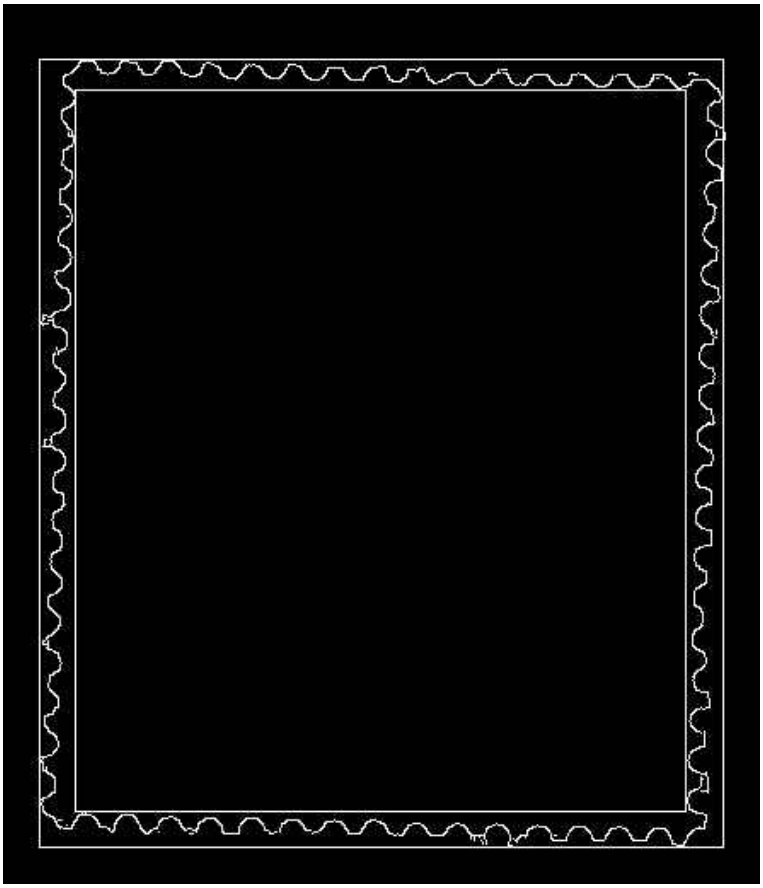
5.7 Detekce středů perforací

Tato část programu se zabývá detekcí středů perforací v obrázku.

V této části programu využívám hlavně semínkové vyplňování a Houghovu transformaci. U semínkového vyplňování využívám verzi hraničního semínkového vyplňování a další semínka generuji do čtyřokolí, protože pokud bych semínka generoval do osmiokolí, mohl by mi algoritmus nalézt pouze jeden střed na každé straně známky.

Tuto část programu jsem pro větší přehlednost rozdělil do více částí:

1. Přibližné určení poloměru středů perforací



Obrázek 5.4: Detekované ohraničení známky

2. Zjištění počtu perforací pro každý zkoumaný poloměr
3. Výběr nejlepších středů perforací

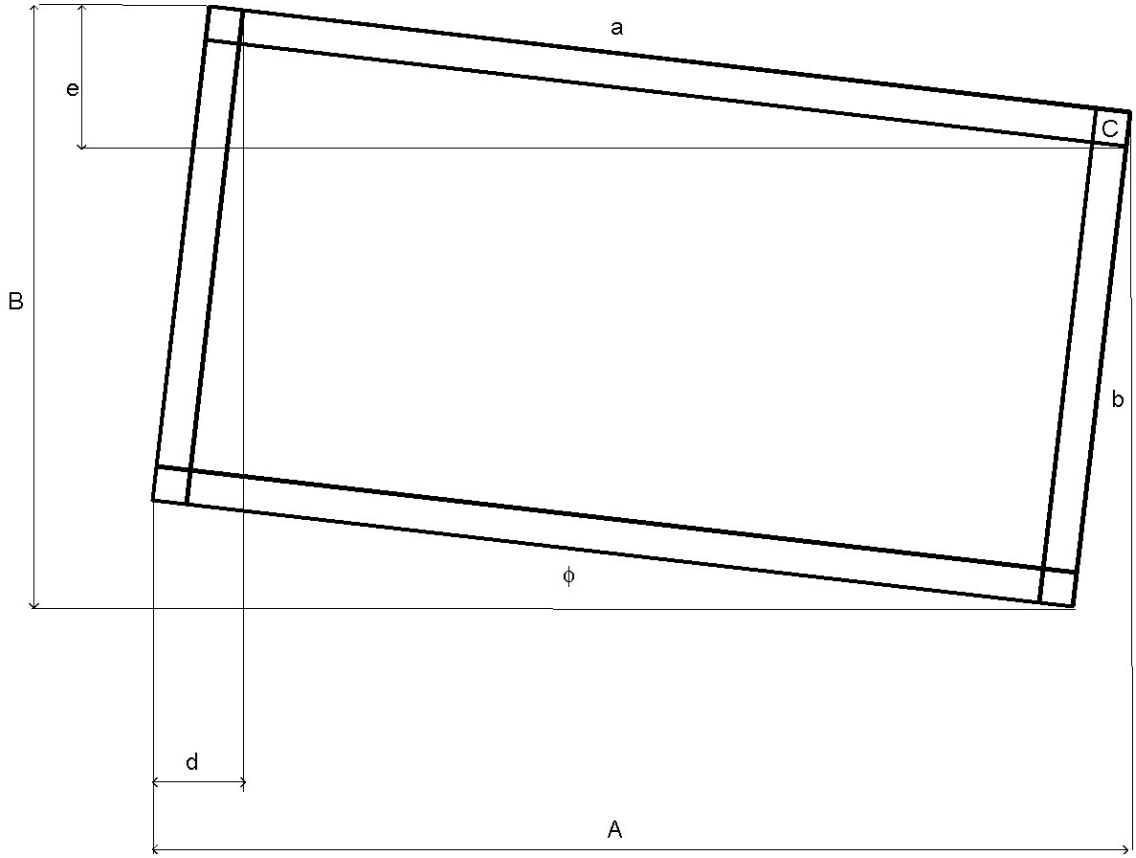
Jednotlivé části dále podrobněji popíší.

Přibližné určení poloměru perforací

V předchozí části jsme prováděli detekci poštovní známky v obraze. Tato detekce nám omezila rozsah průměrů (poloměrů), kterých mohou středy perforací nabývat. Tento rozsah je však stále docela velký a proto v této podčásti programu provádím další omezování velikostí, kterých poloměry perforací mohou nabývat.

Toto omezování provádím pomocí následujících vzorců. Rozměry, se kterými výpočty začínám jsou zobrazeny v obrázku 5.5.

$$d = b \cdot F + c \cdot G \tag{5.1}$$
$$e = a \cdot F + c \cdot G \tag{5.2}$$
$$A = a \cdot G + b \cdot F \tag{5.3}$$
$$B = b \cdot G + a \cdot F \tag{5.4}$$



Obrázek 5.5: Detekované ohraničení známky

Pomocí těchto rovnic sestavím rovnici

$$\cot(\phi) = \frac{A - B + e - d}{c - d} \tag{5.5}$$

S jejíž pomocí vypočítám velikost natočení poštovní známky (úhel ϕ).
Následně si vypočítám hodnoty proměnných F a G.

$$F = \sin(\phi) \tag{5.6}$$
$$G = \cos(\phi) \tag{5.7}$$

Dále si vypočítám hodnoty proměnných a, b, c a C.

$$a = \frac{A \cdot G - B \cdot F}{G^2 - F^2} \tag{5.8}$$
$$b = \frac{A \cdot F - B \cdot G}{F^2 - G^2} \tag{5.9}$$
$$c_1 = \frac{d - b \cdot F}{G} \tag{5.10}$$
$$c_2 = \frac{e - a \cdot F}{G} \tag{5.11}$$

Hodnoty proměnných c_1 a c_2 následně zprůměruji a dostanu tak přibližnou hodnotu poloměru perforačních otvorů, kterou využiji v následující podčásti.

Následující podčásti programu provádím zvlášť pro každou stranu poštovní známky.

Zjištění počtu perforací

Pokud již tedy známe přibližný poloměr perforačních otvorů, postupujeme podle následujícího postupu.

1. Houghova transformace pro poloměr r
2. Detekce středů perforací pomocí semínkového vyplňování

Nejdříve je nutno určit si poloměry, pro které budu aplikovat Houghovu transformaci. Experimentálně jsem určil, že naprosto postačují rozsahy poloměrů ± 4 od vypočítaného poloměru.

Části uvedené o něco výše provádím pro každý tento poloměr zvlášť. Pro jednoduchost zde popíši postup pro jediný poloměr.

Nejprve je tedy potřeba vytvořit si kopii původního obrázku, kde budou všechny pixely obarveny na černo. Tato kopie nám bude sloužit jako Houghův prostor. Houghovu transformaci, modifikovanou pro hledání kružnic provádím následujícím postupem:

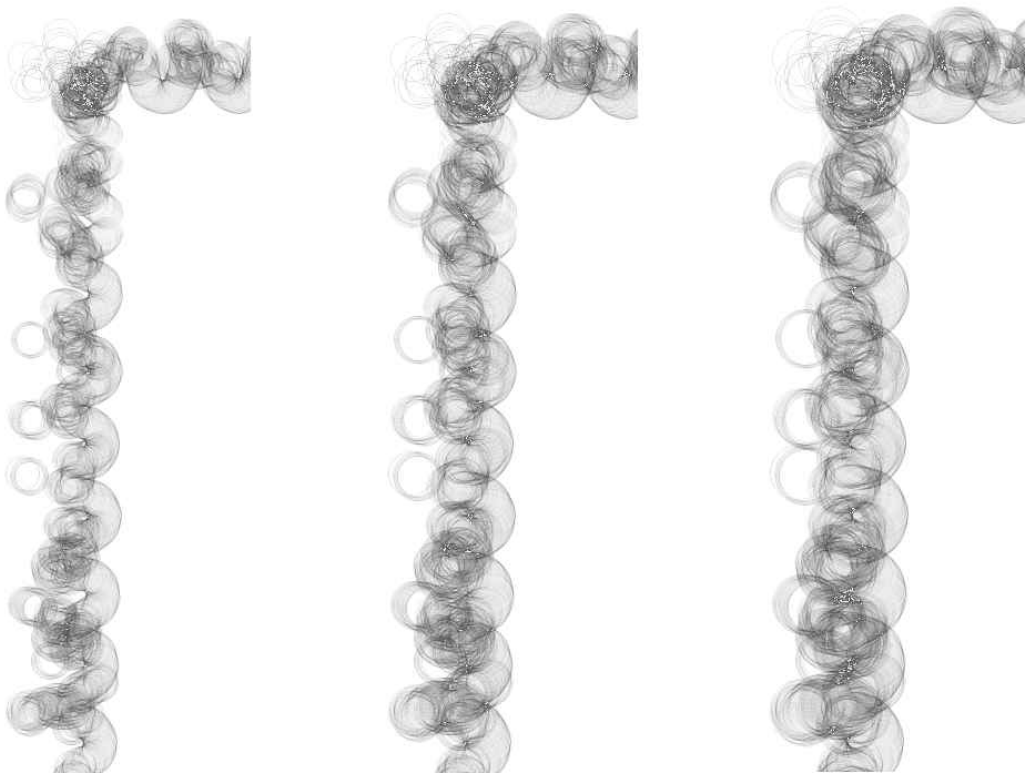
Procházím postupně černobílý obrázek v oblastech definovaných v předchozí části programu - zvlášť pro každou stranu poštovní známky. Pokud narazím na bílý pixel, jeho souřadnice považuji za střed kružnice a pomocí Bresenhamova algoritmu pro kresbu kružnice v Houghově prostoru zvýším intenzitu všech pixelů, které se nacházejí na kružnici definované tímto středem a právě zkoumaným poloměrem o 1. Po průchodu celou zkoumanou oblastí tak dostanu Houghův prostor, ve kterém budou píky (vysoké hodnoty intenzit pixelů) označovat středy perforačních otvorů. V následující části budu jednotlivé středy perforačních otvorů detekovat.

Ukázka Houghova prostoru je na obrázku 5.6

Protože ale Houghova transformace vytvořila kružnice ve velké části zkoumaného prostoru, je nutno nějak rozlišit, kdy se jedná o skutečně detekovaný střed a kdy se jedná jen o šum, který žádnému středu neodpovídá. Toto rozdělení provádím tak, že v Houghově prostoru naleznu pixel s nejvyšší intenzitou a zkoumám pouze pixely, které mají alespoň poloviční intenzitu jako tento pixel.

Procházím postupně Houghův prostor a detekci středů provádím pomocí semínkového vyplňování následujícím postupem

1. Pokud narazím na pixel s intenzitou větší než práh, považuji ho za inicializační semínko a jeho pozici si zapamatuji jako pozici možného středu perforace
2. S takto detekovaným inicializačním semínkem provedu semínkové vyplňování s rozšiřováním do čtyřokolí
3. Pokud narazím na pixel s intenzitou nižší než práh, tento pixel přeskočím
4. Pokud narazím na pixel s vyšší intenzitou, než je aktuálně nejvyšší detekovaná intenzita, tuto novou intenzitu i pozici pixelu si zapamatuji.
5. Pokud narazím na pixel se stejnou intenzitou, jako je aktuálně nejvyšší detekovaná intenzita, upravím pozici nalezeného středu na průměr mezi jeho aktuální pozicí a pozicí právě detekovaného pixelu.



Obrázek 5.6: Ukázka Houghova prostoru pro různé poloměry

6. Po zpracování pixelu snížím jeho intenzitu na 0, aby nedošlo k zacyklování algoritmu.

7. Nově detekovaná semínka ukládám na zásobník. Pokud je tento zásobník prázdný, uložím detekovaný střed do pole středů a přejdu na bod 1.

Toto opakuji, dokud neprojdu celý Houghův prostor.

Protože tento algoritmus generuje i středy perforací, které mohou být velmi blízko u sebe - pravděpodobně se jedná o jediný střed, musíme tyto zdvojené středy nějak ošetřit. Ošetření provádím průchodem detekovanými středy a pokud od sebe dva sousední středy mají vzdálenost menší než je právě zkoumaný poloměr, provedu zprůměrování těchto středů a jeden z těchto středů nahradím tímto novým středem a druhý střed ze seznamu odstraním. Po této operaci zůstane v seznamu každý detekovaný střed pouze jednou.

Výběr nejlepších středů perforací

Nyní jsme zjistili počty středů, které jsou na okrajích poštovní známky pro jednotlivé poloměry středů. V tomto kroku musíme určit, který z těchto poloměrů je nejlepší a bude tedy použit dále v programu.

Výřez obrazu známky s detekovanými středy je na obrázku 6.4.

5.8 Určení zoubkování

Poslední část programu zahrnuje samotné zjištění zoubkování poštovní známky.

Nalezení skutečného počtu středů perforací

Nejprve ale musím ověřit, zda jsem opravdu nalezl všechny středy perforací. To provádím následujícím postupem:

1. Ze seznamu středů detekovaných v předchozí části algoritmu určím vzdálenosti mezi jednotlivými středy.
2. Tyto vzdálenosti uspořádám do histogramu, z něhož zjistím nejčastější vzdálenosti mezi jednotlivými středy.
3. Zjistím vzdálenost mezi nejvzdálenějšími detekovanými středy.
4. Pomocí těchto dvou vzdáleností přepočítám počet nalezených středů.

Tento postup aplikuji, protože zoubkování poštovních známek může být na některých místech poškozeno, což by mohlo vést ke špatnému určení zoubkování poštovní známky.

Tento postup eliminuje případné poškození zoubkování a zjistí přesný počet perforačních otvorů.

Určení rozlišení obrázku

Aby bylo možno určit zoubkování poštovní známky, je dále potřeba určit rozlišení obrázku, na kterém je poštovní známka zobrazena. Vzhledem k tomu, že knihovna OpenCV neobsahuje ve struktuře načítaných obrázků jejich rozlišení, byl jsem nucen v této části využít některého jiného způsobu určování rozlišení obrázku.

Určování rozlišení obrazů jsem prováděl druhým způsobem popsaným v návrhu. Tedy pomocí poloměrů detekovaných středů perforací

Pro každou stranu poštovní známky jsem pomocí tabulky uvedené v části návrhu detektoru určil rozlišení obrazu.

Určení zoubkování poštovní známky

Nyní je již možno přikročit k určování samotného zoubkování poštovní známky.

Určování zoubkování provádím pro každou stranu poštovní známky zvlášť po následujících krocích

1. S využitím rozlišení obrázku vypočítám počet pixelů, které odpovídají vzdálenosti 2 cm.
2. Zjistím největší vzdálenost středů detekovaných na jednotlivých stranách poštovní známky.
3. Počet detekovaných středů perforací přepočítám na počet středů na vzdálenost 2 cm.

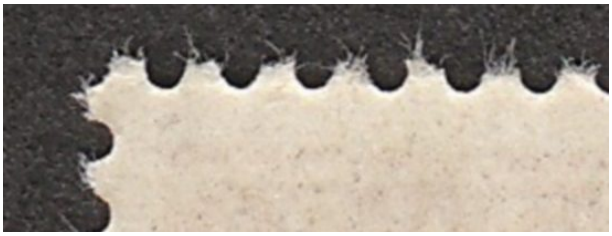
Vypočítané hodnoty odpovídají velikosti zoubkování poštovní známky na vstupním obrázku.

Tyto hodnoty vypíši na standardní výstup.

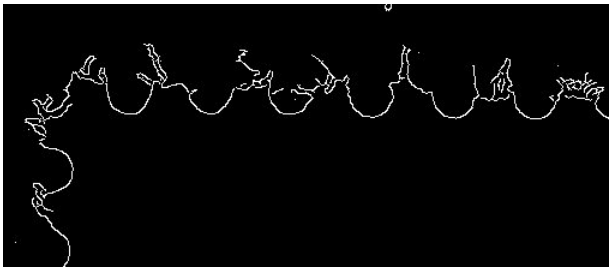
Kapitola 6

Výsledky

Program je schopen detekovat rozměr zoubkování, bohužel úspěšnost není prozatím příliš velká. Výrazný vliv na výsledek má kvalita vstupního obrazu. Příklad okraje známky, s kterým si program neporadí je na obrázku 6.1 . Zpracovaný obraz je na obrázku 6.2. Vlákna papíru,která se vyskytují skoro u všech nepoužitých známek, neumím z obrazu odstranit a výrazně ovlivňují hledání středů perforačních otvorů. V případě jako na obrázku 6.2 dokonce algoritmu vůbec není schopen středy najít.



Obrázek 6.1: Výřez obrazu známky s vlákny

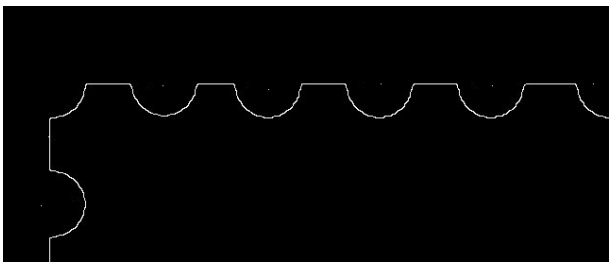


Obrázek 6.2: Výřez obrazu známky s vlákny po zpracování

U čistě perforovaných známek (obr. 6.3) bez otřepů je program schopnen nalézt středy perforačních otvorů (obr. 6.4) a následně vypočítat rozměr zoubkování.

Program také vykazuje horší výsledky pro obrázky s nižším rozlišením. Teprve pro 600 dpi určuje rozměr zoubkování relativně správně. Protože většina obrázků známek na nejrůznějších aukcích na internetu je v nižším rozlišení, program nejde použít pro kontrolu zoubkování známek v těchto aukcích.

Obrázek 6.3: Výřez obrazu čistě perforované známky



Obrázek 6.4: Výřez obrazu čistě perforované známky po zpracování

Program má také problém, pokud je na obrázku výrazná přední strana, popřípadě ještě razítko, také pozadí se strukturou může vést k nesprávným výsledkům. Nejlépe je, pokud je skenována zadní strana známky a okolí je tvořeno tmavým pozadím bez odlesků a struktury.

Na CD jsou přiloženy některé obrázky známek, které byly použity k testování.

Kapitola 7

Závěr

Úkolem práce bylo navrhnout program, který by byl schopen z obrázku poštovní známky určit zoubkování. Vzhledem k použití knihovny OpenCV není velikost vstupního obrázku nijak omezena. Pro dosažení obstojných výsledků zjišťování zoubkování poštovních známek je vhodné použití obrázků, na nichž je známka co nejméně pootočena. Pokud je známka na obrázku pootočena o větší úhel, algoritmy použité v programu nemusejí vykazovat stejné výsledky, jaké bychom mohli získat při měření zoubkování poštovních známek pomocí zoubkoměru. Také v případě, že je zoubkování známky nějak výrazněji poškozeno, není programem zaručeno, že zoubkování zjištěné pomocí programu bude odpovídat skutečnosti.

Program umožňuje dávkové zpracování obrázků, vstupním parametrem je název souboru s obrázkem známky, případně rozlišení, výstupem vypočítaný rozměr zoubkování. Nepodařilo se mně implementovat funkci, která by z rozmístění středů perforačních otvorů byla schopna rozeznat, o jaký typ zoubkování jde, jestli řádkové, hřebenové nebo rámcové zoubkování. O tuto funkcionalitu by bylo možno případně program ještě rozšířit.

Seznam použitých zkratek a symbolů

BMP – Microsoft Windows Bitmap
DPI – dots per inch
Hz – hřebenové zoubkování
JPG – Joint Photographic experts Group
RANSAC – Random SAmple Consensus
Rz – rámcové zoubkování
Řz – řádkové zoubkování

Seznam obrázků