

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

LOKALIZACE ČÁROVÉHO KÓDU V OBRAZE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

PAVEL ŠIMURDA

BRNO 2011



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

LOKALIZACE ČÁROVÉHO KÓDU V OBRAZE

BARCODE LOCALIZATION IN IMAGE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

PAVEL ŠIMURDA

VEDOUCÍ PRÁCE

SUPERVISOR

Doc. Ing. ADAM HEROUT, Ph.D.

BRNO 2011

Abstrakt

Lokalizace čárového kódu je důležitým předpokladem pro jeho čtení. Tato práce se zabývá lokalizací čárových kódů v obraze. Stručně popisuje význam čárových kódů, principy a algoritmy jejich lokalizace. Hlavní část je věnována analýze, návrhu a implementaci nové lokalizační metody. Navržený přístup je postaven na hledání změn v profilu jasových intenzit, díky tomu může metoda pracovat rychle a efektivně. Za účelem testování byla pořízena sada pěti set snímků, podle kterých byla shrnuta úspěšnost algoritmu. Výsledkem je implementace vhodná pro použití v programu určeného ke čtení čárových kódů pracující v reálném čase.

Abstract

The barcode localization is an important prerequisite for its reading. This work deals with the barcode localization in an image. This thesis briefly describes the meaning of barcodes, principles and algorithms of its localization. The main part of this work is concentrated on analysis, concept and implementation of the new localization method. The proposed approach is based on the observation of changes in the intensity profile. Due to this method is fast and accurate. In order to test the rate of success of the algorithm, set of five hundred pictures has been taken. The result is the implementation suitable for real-time usage in the barcode reader application.

Klíčová slova

čárový kód, lokalizace, detekce , EAN

Keywords

barcode, localization, detection, EAN

Citace

Pavel Šimurda: Lokalizace čárového kódu v obraze, bakalářská práce, Brno, FIT VUT v Brně, 2011

Lokalizace čárového kódu v obraze

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Doc. Ing. Adama Herouta Ph.D.

.....
Pavel Šimurda
16. května 2011

Poděkování

Rád bych poděkoval vedoucímu mé práce panu Doc. Ing. Adamu Heroutovi Ph.D. za cenné rady a připomínky nejen odborného rázu, ale také za ochotu, trpělivost a čas, který mi věnoval při konzultacích. Dále bych rád poděkoval Ing. Josefu Mlíchovi za náměty technického charakteru.

© Pavel Šimurda, 2011.

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

Obsah

1	Úvod	2
2	Čárové kódy	3
2.1	Využití	3
2.2	Dělení	4
2.3	EAN	4
2.4	Snímání čárových kódů	6
2.5	Digitální snímání	7
3	Lokalizace a detekce čárových kódů	8
3.1	Existující aplikace pro zpracování čárového kódu	9
3.2	Existující přístupy detekce a lokalizace	9
4	Návrh a analýza přístupů	12
4.1	Metoda rovnoběžných hran	12
4.2	Řádkové čtení	13
4.3	Blokové skenování	16
5	Implementace	19
5.1	OpenCV a jazyk C++	19
5.2	Návrh programu	20
5.3	Anotace	20
5.4	Třída Scanner	20
5.5	zMask	21
5.6	Třída Inter	22
5.7	Třída MyLocator	23
5.8	Reader	24
6	Testování	25
6.1	Test správné detekce	25
6.2	Vyhodnocení lokalizační metody	26
6.3	Nedostatky metody	28
6.4	Srovnání s existující aplikací	30
7	Závěr	32
A	Obsah CD	35
B	Dosažené výsledky	36

Kapitola 1

Úvod

Čárové kódy jsou velmi rozšířeným nosičem informace. Používají se v mnohých odvětvích a slouží především k označení a následné identifikaci zboží. Staly se populárními díky své univerzálnosti. Výhodou čárových kódů je jejich spolehlivost, jednoduchost čtení a označení. Práce s čárovými kódy je užitečná hlavně v oblasti obchodu a logistiky, kde je urychlen a zpřesněn proces rozpoznávání zboží. Kvůli narůstající potřebě zakódovat větší a větší množství informací se čárové kódy neustále vyvíjí. Existuje velké množství typů kódů, kterých je dnes standardizováno okolo dvou set. S novými technologiemi přicházejí i nové typy kódů, které jsou schopny nést daleko větší množství informace.

Tato práce se zabývá především lokalizací čárových kódů v obraze. Popisuje některé existující metody lokalizace a jejím cílem je vytvořit novou metodu. Lokalizace je klíčovým předpokladem k přečtení čárového kódu. Z celého procesu zpracování čárového kódu se jedná o nejsložitější problém. Úkolem je vyhledat v obraze určitý vzorek, což může být velmi složité, protože v reálně pořízeném snímku může být mnoho rušivých jevů. Pro spolehlivou a přesnou lokalizaci je třeba zvolit vhodný kompromis mezi rychlostí a přesností nalezení výsledků. Navržený lokalizační algoritmus pracuje s profily jasových intenzit a provádí vyhodnocení čtvercových bloků, složených ze čtecích řádků intenzit. Díky tomu nepracuje s nadměrným množstvím dat a vyhodnocení probíhá v přijatelné rychlosti.

Spolehlivá a rychlá lokalizace je vhodná především pro použití v oblasti automatické identifikace. Zde je žádoucí, aby se všemi kroky omezil vznik chyb způsobených lidským faktorem. Lokalizování čárového kódu však stále provádí nejčastěji člověk. Cílem této práce je navrhnout vhodný algoritmus lokalizace, který by byl použitelný pro zpracování v reálném čase. Sloužit může jako knihovna pro lokalizaci nebo součást čtečky čárových kódů.

Kapitola 2 pojednává o významu, využití a historii čárového kódu. Následující kapitola 3 je zaměřena na lokalizaci čárových kódů. Popisují se zde různé existující metody lokalizace a principy aplikací, které čtení čárových kódů zajišťují. V kapitole 4 je popsán návrh nových lokalizačních a detekčních metod. Je zde věnována pozornost především jejich analýze a vlastnostem. Kapitola 5 se zabývá implementací programu, který pracuje se snímky obsahující čárové kódy. Hlavní těžiště této kapitoly je zaměřeno na část zabývající se vlastní lokalizací. Kapitola 6 je zaměřena na testování algoritmu, jsou zde popsány dosažené výsledky, odhalené chyby a nedostatky. Poslední kapitola 7 shrnuje celou práci, popisuje její přínos a možná rozšíření.

Kapitola 2

Čárové kódy

Čárový kód je jedním z prostředků automatické identifikace objektů reálného světa. Nejčastěji se používá pro označení zboží nebo jiných produktů. Jedná se o zakódování alfanumerické informace do strojově čitelné, vizuální podoby. Existuje přes dvě stě typů čárových kódů, většina druhů jsou specifické pro svou oblast použití. Klasický lineární čárový kód, kterým se tato práce zabývá, je tvořen rovnoběžným uspořádáním čar a mezer určitých šířek, ty pak vyjadřují zakódovanou informaci.

Použití čárového kódu jako prostředku k identifikaci zboží je velmi praktické. Své využití má v oblasti obchodu, průmyslu nebo logistiky. Může být vytištěn a umístěn na libovolný objekt a z něj také snímán. Je vhodný pro strojové zpracování, tím pádem přináší zrychlení procesu sběru a zadávání dat. Stává se proto praktickým a také levným nástrojem automatické identifikace.

V této kapitole bude čtenář seznámen s nejběžnějšími typy čárového kódu a s jeho charakteristickými vlastnostmi. Přehledově se dozví o jeho historii a uplatnění. Nejvíce pozornosti bude věnováno kódu typu *EAN-13*, který je celosvětově nejrozšířenějším. Především je používán v našich zeměpisných šířkách, a proto je pro nás užitečná jeho lokalizace i dekodování.

2.1 Využití

Jak už bylo zmíněno, čárové kódy slouží hlavně k zakódování informace do určité vizuální podoby a jejímu následnému čtení. Principu, kdy je zboží strojově označeno nebo přečteno a informace z něj je zpracována počítačem se říká automatická identifikace. V padesátých letech minulého století byl tento přístup vyvíjen ve Spojených státech amerických převážně proto, aby se zamezilo vzniku chyb ve sběru dat způsobených lidským faktorem [19]. Inspirací při vytváření čárového kódu byla Morseova abeceda, pro zjednodušení čtení byly v počátcích kódy tisknuty kruhově.

Uplatnění čárových kódů je např. v přepravě zboží, kde jsou všechny zásilky označeny a pomocí této informace je mimo jiné možno sledovat, kde se zásilka nachází a kdo jí přebral. Dalším odvětvím, kde se čárové kódy používají, jsou sklady. Informace o uskladněném a přijatém nebo vydaném zboží jsou za pomoci takového označení lépe kontrolovány. Člověk se s čárovým kódem nejčastěji setkává v obchodních domech, kde jsou pomocí něj označovány veškeré produkty. To zvyšuje rychlost při odbavení nákupu. Kódy typu ISBN a ISSN slouží k označování knih, což je pouze jiná modifikace čárového kódu typu *EAN*. Využití čárového kódu je široké. Je možno jej nasadit prakticky ve všech odvětvích činnosti

člověka. Za zmínku stojí také jejich aplikace v administrativě, bezhotovostním platebním styku, v monitorování výroby, dopravě a medicíně aj.

2.2 Dělení

Čárové kódy lze rozdělit do mnoha kategorií. Často se dělí podle použití na dva základní typy. Kódy používané v průmyslu, to jsou například *Code 39*, *Codebar*, *Code2/5 atd.* a kódy používané v obchodu např. *UPC*, *EAN-8*, *EAN-13 apod.* Další dělení je podle použité symboliky, ta popisuje jakým způsobem jsou kódovány jednotlivé znaky, ale také počáteční a koncové značky kódu nebo zprávy. Je to vlastně specifikace a popis toho, jak je konkrétní typ čárového kódu zakódován, nejčastěji do posloupnosti čar a mezer.

- **Lineární čárový kód** je první a nejznámější, zakódovává informace do jednorozměrného sledu čar a mezer. Dělí se na mnoho dalších typů a je ze všech nejpoužívanější.
- **Kruhový kód** je dalším typem zobrazení jednorozměrného kódu. V tomto případě je použita různá šířka nebo délka oblouku soustředných kruhů. Hlavní výhodou tohoto typu je možnost čtení z jakéhokoliv úhlu, nicméně kruhová symbolika zabírá více místa než ostatní.
- **2D čárový kód** se dělí na další skupiny. Důležité je zmínit, že zde patří v dnešní době velmi rozšířený QR kód. Hlavní výhodou těchto kódů je možnost zakódování velkého množství informací na malém prostoru. Na druhou stranu je tento typ symboliky složitý na lokalizaci a detekci.

Vlastnosti symbolik jako výška, počet šířek, obousměrnost, detekce chyb a typ jsou podstatnými pro návrh snímače čárových kódů. Důležitým parametrem symboliky je znaková sada, ta říká jakou množinu znaků je možno zakódovat. Některé symboliky jsou pouze numerické, ale existují také alfanumerické.



Obrázek 2.1: Ukázka různých druhů čárových kódů.

2.3 EAN

U nás se jedná o nejčastěji používaný typ čárového kódu. Známý je především díky svému použití v obchodní sféře. Původní označení *EAN* (European Article Number), bylo kvůli

celosvětovému rozšíření přejmenováno na *IAN* (International Article Number). Nicméně zkratka byla z důvodu přehlednosti ponechána. Z názvu plyne, že se jedná o symboliku kódující pouze numerické znaky. Používán je především koncovými prodejci zboží k označení a čtení informací z jejich produktů. Jsou různé verze standardu *EAN*. Používané jsou verze 8 a 13. Tato čísla označují kolik číslic je zakódováno, jedná se proto o kódy s numericky pevnou délkou. Kód je také normalizován podle [7]¹. *EAN* kód je složen z několika částí:

- **Kód země**, podle kterého lze identifikovat stát, kde je registrován výrobce produktu. Číslo přiděluje zemím sdružení GS1 [12].
- **Kód výrobce** jsou další čtyři číslice (v závislosti na kódu země) označují výrobce. Tato čísla v naší zemi přiděluje GS1 Czech Republic.
- **Kód výrobku** obsazuje pět číslic a je také obsažen v seznamu spravovaným GS1.
- **Kontrolní číslo** je poslední číslicí kódu jeho výpočet je popsán níže.

Zmíněné informace sdružuje organizace GS1, která zaštiťuje rozvoj a tvorbu standardů, především na poli zpracování informací. V České republice existuje sdružení GS1 Czech Republic.

EAN kódy jsou tvořeny okrajovými *start* a *stop* značkami, dělicím znakem, který se nachází uprostřed tisknuté oblasti, a poté kódovacími znaky pro numerické hodnoty (0 až 9). Také je zde definována klidová zóna, která vymezuje šířku okolního bílého pozadí proto, aby byl kód snadněji nalezen snímacím zařízením. Základním parametrem čárového kódu je modulová šířka, tzv. modul, což je nejmenší šířka mezery i čáry. Je to vlastně bit čárového kódu, který odpovídá logické 0 nebo 1. Každá hodnota je zakódována sekvencemi dvou čar a dvou mezer, to odpovídá sedmi modulům. Pro kódování numerických znaků jsou definovány celkem tři kódovací sady. Jedna pro lichou paritu a dvě pro sudou paritu. U kódu *EAN-13* se vždy používá kódová sada C, sudé parity pro šest datových znaků z pravé poloviny kódu a kódová sada A a B pro znaky z levé poloviny. Volba kódové sady pro levou polovinu je dána první číslicí v kódu. Názorně je vše zobrazeno na obrázku 2.2. *EAN-13* je schopen detekovat chybu v kódu podle kontrolního součtu, jehož hodnota je zároveň poslední číslicí.

ČÍSLICE	ZNAKOVÁ SADA		
	A	B	C
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
OKRAJOVÉ ZNAKY		DĚLÍCÍ ZNAK	

Obrázek 2.2: Znakové sady kódu EAN. Převzato z [2].

¹Norma již není v platnosti.

Výpočet kontrolní číslice je roven součtu číslic na lichých pozicích sečtenou s součtem číslic na sudých pozicích vynásobeným třemi. Celý součet je zaokrouhlen na desítku nahoru. Od hodnoty tohoto zaokrouhleného součtu je odečtena jeho nezaokrouhlená hodnota. Výsledné číslo je hodnota kontrolního součtu.



Obrázek 2.3: Okrajové a dělicí znaky kódu EAN.

2.4 Snímání čárových kódů

Celý proces snímání čárového kódu lze rozdělit na několik základních úkonů. Nejprve je důležité nalezení čárového kódu, s kterým je spojena detekce a lokalizace. Druhá část souvisí s elektronickým zpracováním signálu (čárového kódu) a jeho logické vyhodnocení. Poslední částí je pak dekódér, který slouží k převodu vstupní informace. V tomto případě signálu čárového kódu na výstupní informaci, což je nejčastěji číslo nebo text.

V dnešní době je prakticky nepoužívanější ruční snímání čárového kódu. Existuje nepřehledné množství snímačů, nejčastější jsou laserové, ale používají se také CCD kamery. K připojení s počítačem vyžívají nejčastěji USB nebo RS-232 konektivitu, proto je možno jejich výstup, čili posloupnost přečtených znaků, zpracovávat jako emulaci vstupu z klávesnice. Jsou však i takové čtečky, které v sobě mají integrován celý systém a dekódované znaky jsou reprezentovány na displeji nebo ukládány do vnitřní paměti. Z praktického hlediska jsou také používány bezdrátové snímače. Nevýhodou takových zařízení je jejich cena, která narůstá kvůli miniaturizaci použitých součástek. Poslední součástí snímače je dekódér. Může být externí nebo interní, používá se k přečtení znaků kódu a výstupu informací. Řídí se různými algoritmy podle typu zpracovaného kódu.



Obrázek 2.4: Ruční laserový scanner. Převzato z [3].

Detailnímu popisu čtecího zařízení se věnuje [2]. Zařízení pracují s elektromagnetickým vlněním určité vlnové délky. To je odraženo od čárového kódu a podle této intenzity je signál

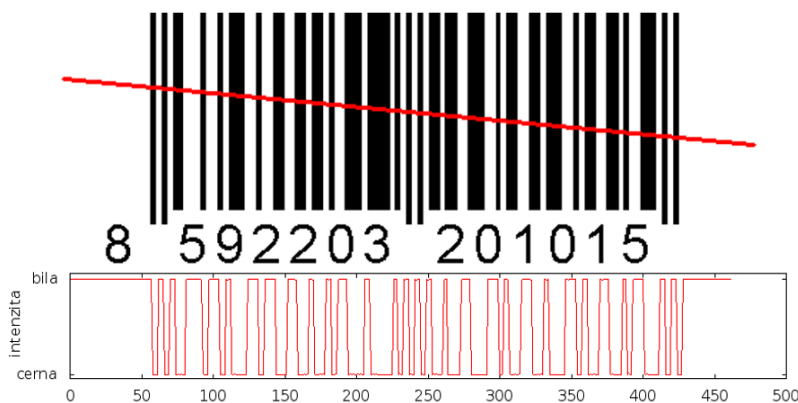
pomocí polovodičových součástek přeměněn na různé napěťové úrovně. Zařízení obsahuje často zvukové nebo světelné signalizační zařízení pro zpětnou vazbu.

Také použití těchto snímačů vyžaduje lidskou manipulaci k detekování a lokalizaci čárového kódu. Namíření čtecího paprsku na kód je nejproblematičtější krokem celého zpracování. Může zde selhat lidský faktor. Rychlost snímání a odezvy snímače bývá závislá na kvalitě čárového kódu a čtecího zařízení.

- **Způsoby čtení:** Čtecí pero, lineární CCD ruční scanner.
- **Možnost výstupu:** Klávesnicový, RS232.

2.5 Digitální snímání

Digitální snímače čárového kódu pracují na velmi podobném principu. Využívají však jiné technologie snímání obrazu. Nejčastěji plošné CCD čipy v klasických digitálních fotoaparátech či kamerách. Tato zařízení pořizují digitální obraz a z něj je poté třeba informaci z čárového kódu nalézt a přečíst. Protože je snímána celá scéna, je zde kladen velký důraz na detekci a lokalizaci čárového kódu. Detekce v tomto případě závisí na kvalitě snímku, ale také přesnosti algoritmu.



Obrázek 2.5: Výstup při snímání lineárního čárového kódu.

Digitální snímání v zásadě využívá přístupy toho klasického manuálního. Digitálním snímáním se rozumí sejmutí obrázku prostřednictvím digitální kamery, fotoaparátu nebo scanneru. A dekódování čárového kódu z pořizovaného snímku. Je jisté, že takový proces se musí skládat z několika částí:

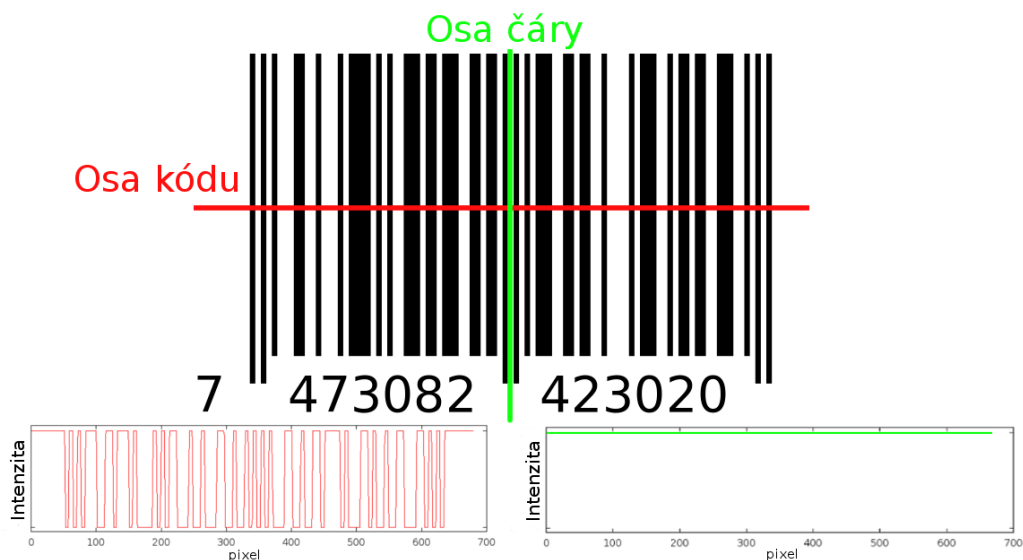
1. Nalezení vzorku čárového kódu v obraze
2. Lokalizace - určení souřadnic oblasti s čárovým kódem
3. Vymezení vhodné části z nalezené oblasti pro dekódování
4. Převedení části nalezené oblasti na signál vhodný pro dekódování
5. Přečtení čárového kódu

Tato práce se zabývá body 1–3 a částečně také bodem 4.

Kapitola 3

Lokalizace a detekce čárových kódů

Při digitálním zpracování čárového kódu je základním předpokladem správného dekódování spolehlivá detekce a lokalizace. Tato práce se zabývá lokalizací 1D neboli lineárních čárových kódů. Tento kód je tvořen posloupností navzájem rovnoběžných čar. Hlavní charakteristika takovéto textury je, že obsahuje velké množství hran v jednom směru, ale žádné ve směru na něm kolmém. Tyto směry odpovídající ose čáry a ose kódu jsou spolu se svými průběhy intenzit zobrazeny na obrázku 3.1. Takový vzorek je možno poměrně dobře lokalizovat. Ve výsledku je po provedení detekce a lokalizace důležité mít informaci o tom, kolik čárových kódů je v obraze obsaženo, kde se nachází a jaké je jejich orientace.



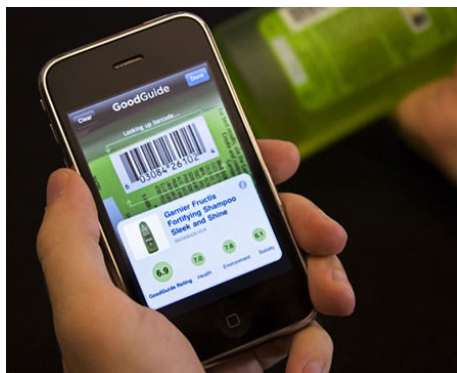
Obrázek 3.1: Charakteristika textury čárového kódu.

Pojmy detekce a lokalizace se do značné míry překrývají, proto přesný význam těchto termínů může být nejednoznačný. V textu této práce bude za detekci považováno zjištění, zda a kde jsou v obraze hledané objekty. Pojem lokalizace bude popisovat proces, který přibližný výsledek detekce zpřesní a poskytne potřebné informace o nalezené oblasti, jakými jsou orientace kódu a souřadnice hraničních bodů. Celý proces vyhledání čárového kódu lze chápat jako detekci spojenou s lokalizací. Je důležité, jak potvrdit nebo vyvrátit výskyt čárových kódů, tak vymezit konkrétní oblast, kde se nacházejí. Detekce a lokalizace je tedy prováděna jako celek.

Čárový kód může být v obraze různě natočen, zastíněn nebo osvětlen. Bývá také často vytištěn na produkty, kde je mnoho rušivých elementů, z lokalizace se proto stává složitý problém. Není možné vytvořit takový algoritmus, který by v přijatelném čase detekoval všechny možné kódy v obraze. Vždy budeme moci takovému algoritmu předat jako vstup tak neobvyklý snímek s kódem, při kterém detekce selže. Pro omezení nepříznivých vlivů některé přístupy provádějí předzpracování obrazu nebo jiné techniky pro zvýšení přesnosti a rychlosti zpracování čárového kódu. Při čtení čárových kódů se s problémy lokalizace a detekce potýkají různé aplikace různými způsoby.

3.1 Existující aplikace pro zpracování čárového kódu

Vývoj aplikací, které zpracovávají čárové kódy zaznamenal úspěch díky rozšíření zařízení snímajících digitální obraz. V dnešní době jsou jednoduše dostupné chytré telefony (*Smart Phones*) a také digitální kamery, které lze snadno připojit k počítači.



Obrázek 3.2: Aplikace Goodguide¹, určena ke čtení čárových kódů (obrázek převzat z[11]).

Většina programů určených ke čtení čárového kódu funguje tak, že je nejprve nutno kód zamířit do připraveného rámečku, ze kterého je následně informace dekodována. Takový přístup se neliší od běžných laserových čteček a celý proces čtení čárového kódu nijak nezjednodušuje. Zaměření čárového kódu je totiž nejproblematictější a nejdéle trvající úkon. Navíc je k jeho výkonu potřeba činnost člověka. Hlavním cílem v této oblasti je proto zautomatizovat lokalizaci čárového kódu tak, aby nebylo nutné mechanicky zaměřovat jeho oblast. Proto existují algoritmy, provádějící lokalizaci čárového kódu v komplexní scéně. Do koncových aplikací nejsou často nasazovány kvůli vysokým výpočetním nárokům a nedostatečné spolehlivosti.

3.2 Existující přístupy detekce a lokalizace

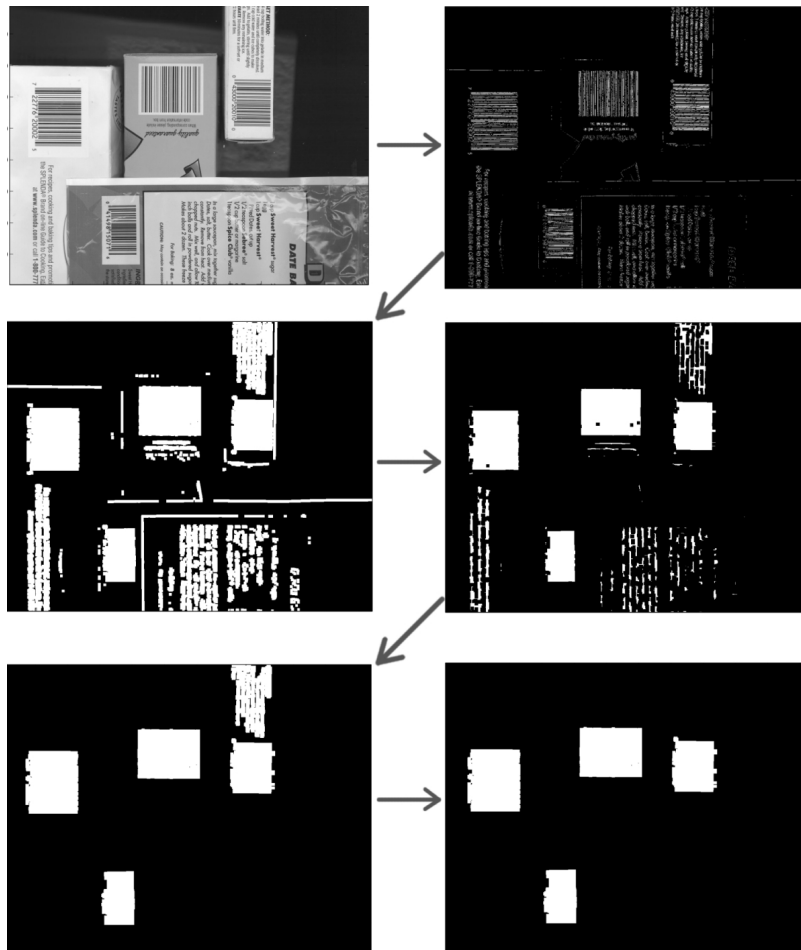
Pro lokalizaci čárových kódů existuje mnoho metod a lze je dělit do několika kategorií. Naprostá většina metod je postavena na principech segmentace obrazu [10], často na základě detekce hran nebo prahování [13]. Tyto metody pracují na takovém principu, že postupně dělí obraz na části splňující daná kritéria a vytvoří tzv. binární obraz. Základní typy těchto metod jsou poměrně robustní a přesné, na druhou stranu vyžadují velký výpočetní výkon

¹Webové stránky aplikace: <http://www.goodguide.com/about/mobile>

a tím pádem nejsou vhodné pro zpracování v reálném čase. Mnoho metod se tento přístup snaží zachovat, ale přidávají různá zjednodušení, která vedou k rychlejšímu vyhodnocení.

Například v [13] jsou k detekci hran použity Gaborovy filtry v různých orientacích, od 0° do 157.5° s krokem 22.5° . Tímto způsobem jsou hledány oblasti s největší lokální frekvencí signálu obrazových dat. Pro klasifikaci různě natočených kódů je zde použita neuronová síť.

Mnohé metody k lokalizaci čárového kódu využívají gradientů [1]. Používají Robinsonův nebo Sobelův operátor s jehož pomocí získají velikost i směr gradientu. Metoda vhodná pro lokalizaci v reálném čase je popsána v [6]. Je to velmi jednoduchý a praktický přístup. Vyhledává v obraze oblasti s vysokou hustotou jednosměrných gradientů. Pro extrakci oblastí s čárovým kódem využívá čtyř binárních obrazů, každý je získán použitím Sobelova operátoru pro jednotlivé směry pod úhlem 0° , 45° , 90° a 135° . Další kroky spočívají v redukci množství zpracovávaných dat, sloučení blízkých oblastí a odstranění těch izolovaných. Zjednodušení této metody je prezentováno v [23]. Používá pouze dvě konvoluční jádra Sobelova operátoru, a to pro horizontální a vertikální směr. Poté je provedeno prahování gradientu, které eliminuje oblasti se zanedbatelnou silou gradientu. V dalších fázích odstraní nespojitě oblasti a provede určité morfologické operace pro zpřesnění nalezených oblastí. Každým krokem se provádějí výpočty nad celým obrazem, to je hlavní nevýhoda těchto přístupů. Celý postup, kterým segmentace obrazu probíhá můžeme vidět na obrázku 3.3.



Obrázek 3.3: Postup při segmentaci obrazu (obrázek je přebrán z [23]).

Přístup popisovaný v [25] pracuje se dvěma obrazy. Počáteční část zpracování je kvůli rychlosti prováděna nad obrázkem v nízkém rozlišení. První krok je odhad hlavní orientace kódu podle směru nalezených hran. Ty jsou získány z gradientů pro čtyři základní směry, stejně jako ve výše zmíněném přístupu. Konečná fáze pak pracuje nad originálním obrazem zpřesňuje natočení čárového kódu a přesně vymezuje jeho oblast.

Pro rychlé zpracování jsou zajímavé metody, které pracují s co nejmenším množstvím obrazových dat [22]. Nejprve se zpracovává obraz zachycený z kamery v malém rozlišení. V něm jsou čteny pixely v řezích odpovídajícím rozkladovým řádkům s orientacemi 0° , 45° , 90° a 135° . Oblast s největší koncentrací na ně kolmých čar je označena jako kandidát čárového kódu. Dalším zpracováním, už ve vyšším rozlišení, je verifikována tato oblast a jsou nalezeny její přesné hranice. Jiná metoda [24] opět čte pouze určité řádky v obraze. Pracuje však s profilem jasové intenzity, ve kterém hledá významné změny a podle nich určí, kde se nachází oblast s čárovým kódem. Pro robustnost provádí předzpracování obrazu, díky tomu je schopna nalézt čárové kódy i v obraze s nižší kvalitou. Také je zde demonstrováno jakým způsobem lze lokalizovat poškozené nebo špatně viditelné kódy, aby je bylo možné dekodovat.

Některé jiné metody používají tzv. superrozlišení [18]. Tato metoda snižuje šum obrázkových dat a přibližuje tak obrázek dokonalému vzoru. Nevýhodou je zdvojnásobení rozlišení obrazu a tím pádem snížení rychlosti prováděné lokalizace.

Další kategorie využívá k lokalizaci morfologické operace v [9], je vypočítána dilatace a eroze. Je zde také použit Cannyho operátor k detekci hran, a protože se jedná o přístup používaný pro lokalizaci ve videu, pracuje se s dvěma snímky. Zajímavé využití Skeletonizace je ukázáno v [5]. Obraz je rozdělen na nepřekrývající se bloky, v nichž je provedena Otsuho metoda prahování. Pro každý blok jsou vytvořeny skelety jednotlivých spojitých částí a je zkoumána jejich rovnoběžnost.

Ostatní principy využívající Houghovy transformace [16] nebo vlnkové transformace [17] jsou velmi zajímavé, ale vzhledem k jejich nízké rychlosti nejsou příliš vhodné pro zpracování v reálném čase.

Kapitola 4

Návrh a analýza přístupů

Ze znalosti existujících přístupů jsem postupně implementoval variace různých metod detekce a lokalizace. Metody by měly být zaměřeny především na rychlost zpracování, aby je bylo možno použít v zařízeních zpracovávajícím sekvenčně v reálném čase a poskytnout tak modul pro systém čtení čárových kódů např. z webové kamery. Při návrhu algoritmů musel být brán ohled na to, že bude pracovat s obrázky, které mohou zobrazovat komplexní scénu. Tím se problém nalezení čárového kódu stává velmi náročným. Je jasné, že ideální algoritmus by měl detekovat a lokalizovat všechny možné čárové kódy. V praxi je však, s ohledem na rychlost zpracování, nemožné takovýto algoritmus navrhnout. Většina metod popisovaných v předchozí kapitole, spoléhá do jisté míry na přibližné vyhodnocení a ta v určitých situacích selhávají. Častým případem selhání bývá především kaz na kódu nebo jeho špatné natočení.

V kapitole budou postupně rozebrány tři implementované algoritmy lokalizace. Nejprve jsou popsány jednodušší metody, které se ukázaly jako nevhodné. Vysvětlené přístupy jsou však vývojovými etapami výsledného algoritmu. Bude rozebráno srovnání metod, a také jejich výhody a nedostatky.

Mnohé principy a vhodné vlastnosti jsou do výsledného algoritmu promítnuty. Poslední přístup založený na blokovém skenování profilů jasových intenzit, se ukázal jako nejvýhodnější. Je proto stěžejním bodem celé kapitoly. Navržený algoritmus si neklade za cíl lokalizovat přesně všechny čárové kódy, ale je určen pro rychlou lokalizaci oblastí, které čárový kód obsahují. Důležité je, aby poskytl výřez obrazu, ze kterého bude čárový kód přečten. Je proto nežádoucí, aby algoritmus nějaký kód vynechal. Falešné lokalizace nejsou až takovým nedostatkem. Takovéto oblasti budou závěrečným zpracováním vyloučeny a nebo je čtečka nebude moci dekodovat. Algoritmy jsou zaměřeny na lokalizaci lineárních čárových kódů hlavně typu *EAN* a *UPC*.

4.1 Metoda rovnoběžných hran

První navrhovaná metoda spočívá na nalezení rovnoběžných hran v obraze. Je velice jednoduchá a nedosahuje příliš přesných výsledků. Nebyla úplně dokončena a nekompletní algoritmus byl poměrně výpočetně náročný. Aby lokalizace probíhala pro všechny orientace čárového kódu, bylo by nutno provést komplexní úpravy. Inspirací pro mě byla práce popsána v [5]. Algoritmus pracoval tak, že nejprve rozdělil obraz na nepřekrývající čtvercové bloky. Pro každý blok bylo provedeno prahování obrazu pomocí Otsuho algoritmu [21]. Ten je výpočetně nenáročný a hledá optimální prah, takže výsledný binarizovaný obraz je

přesnější než při pevně zvoleném prahu. V takto binarizovaném bloku jsou poté pomocí Laplacova operátoru nalezeny hrany a zkontrolována jejich rovnoběžnost. Pokud jsou všechny nalezené hrany rovnoběžné, je blok označen jako část čárového kódu viz. obrázek 4.1. Na obrázku jsou zvýrazněny bloky, odpovídající kódu a vyškrtnut je blok, ve kterém nebyly nalezeny rovnoběžné hrany.

Výhoda této metody spočívá v relativně dobré odolnosti proti zastínění a osvětlení díky dynamickému prahování v blocích. Potýká se s nedostatečnou detekcí oblastí s malým počtem rovnoběžných hran v rámci bloku. Nevýhodou jsou operace pracující s veškerými obrazovými daty, především prahování obrazu a detekce hran. Problematické je také vhodné zvolení velikosti bloku.



Obrázek 4.1: Vyhodnocení rovnoběžných hran.

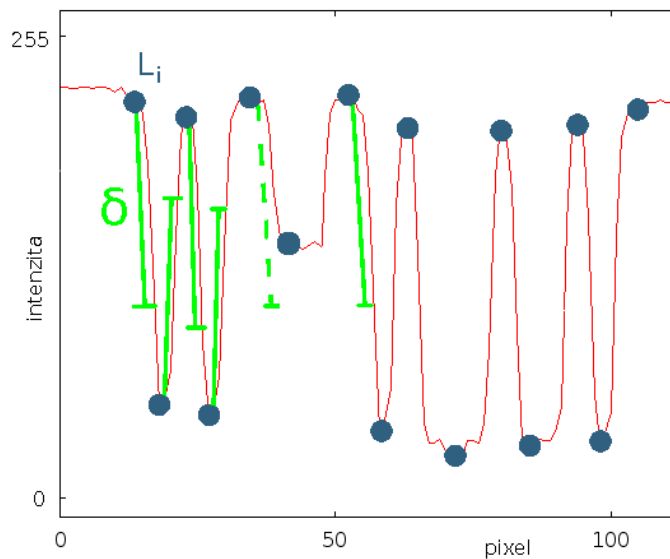
4.2 Řádkové čtení

Pro urychlení procesu lokalizace se ukázalo jako výhodné pracovat s profilem jasových intenzit. Inspirací tohoto přístupu byly práce [15] a [24]. V obraze jsou čteny rozkladové řádky, z nichž se zpracuje informace o intenzitě. Ta je vypočítána podle známého vztahu 4.1, kde I je výsledná hodnota intenzity, vypočítána z hodnot jednotlivých složek barevného modelu R , G a B . Čtené řádky mají vzájemnou orientaci 0° , 45° , 90° a 135° . Experimentálně se ukázalo, že každý takový řádek pokrývá dostatečnou oblast pro zachycení všech natočení čárového kódu.

$$I = 0.229 R + 0.587 G + 0.114 B; \quad (4.1)$$

Úsek odpovídající čárovému kódu je poté vyhodnocen jako oblast řádku s významnou změnou hodnot v profilu jasových intenzit. Významná změna je definována jako rozdíl mezi dvěma po sobě jdoucími lokálními extrémů v signálu intenzity, jehož hodnota je větší než požadovaný prah δ . Ve vztahu 4.2 je L vektor všech lokálních extrémů z čteného řádku. Demonstrativně je vyhodnocení změny zobrazeno na obrázku 4.2. Mimojiné je čárkovane ukázáno, mezi kterými hodnotami intenzit nebyla nalezena změna.

$$|L_i - L_{i+1}| > \delta \quad (4.2)$$



Obrázek 4.2: Vyhodnocení změny v signálu intenzit.

Volba hodnoty prahu určuje, do jaké míry bude záviset přesnost lokalizace na kvalitě, hlavně zaostření obrazu s čárovým kódem. Čím větší bude nastavena hodnota δ , tím větší rozdíl v intenzitě se očekává mezi přechodem černé čáry a bílé mezery v čárovém kódu. Volba této hodnoty není ovšem jednoznačná. Nelze dopředu říci, jak bude čárový kód v obraze nasvícen. V extrémním případě může nastat situace, kdy část kódu bude schována ve stínu a část bude značně osvětlena. Pokud by za prah byla zvolena nízká hodnota, nemusely by být změny v zastíněné části vůbec zaznamenány, protože rozdíl mezi intenzitou zastíněné mezery a čáry kódu by byl nepatrný.¹

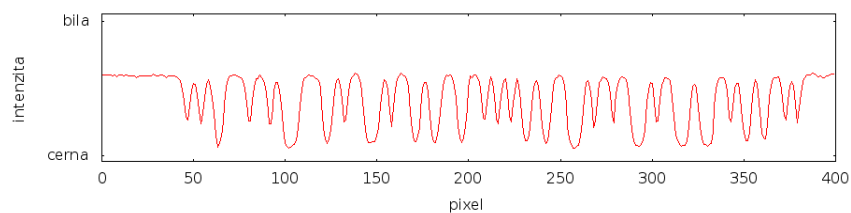
Obrázky pořízené kamerou nebývají ideálně zaostřeny a ani světelné podmínky často nejsou optimální. Na grafech v obrázku 4.3 je vidět srovnání signálů intenzit ideálního obrázku s reálně pořízenou fotografií. Z grafu 4.3(a) můžeme pozorovat, že se profil jasových intenzit k ideálnímu průběhu pouze přibližuje.

Navržený algoritmus čte hodnoty intenzit z obrázku podle určitých řádků. Na obrázku 4.4(a) je mřížkou demonstrováno, které řádky jasových intenzit jsou načítány a vyhodnocovány. Části obsahující odpovídající počet změn jsou na obrázku 4.4(b) vyznačeny červeně. Minimální počet změn je zde stanoven konstantou $T_c = 10$. Pokud v profilu není nalezeno alespoň tolik změn, nelze takovouto posloupnost čar a mezer považovat za čárový kód. Čárový kód typu *EAN-13* obsahuje 60 změn². Protože některé čáry v kódu mohou vlivem špatného zaostření splývat, je třeba počítat i s menším počtem změn. Pokud je takových změn nalezeno méně než T_c je možné, že kód nebude čitelný, a proto je detekce považována za špatnou. Kvůli rozdílnému poměru šířek čar a mezer, který závisí na velikosti kódu a rozlišení snímku, je třeba dynamicky vyhodnocovat přijatelnou vzdálenost mezi jednotlivými změnami. Ta je postupně počítána jako průměrná vzdálenost mezi jednotlivými přijatelnými změnami. Vyhodnocení, zda řádek odpovídá čárovému kódu je postaveno na kombinaci počtu změn a vzdálenostmi mezi nimi.

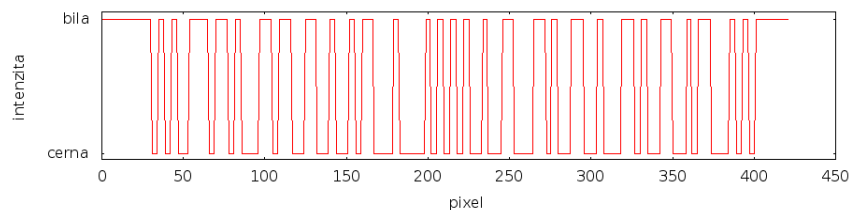
Metoda pracující na bázi řádkového čtení byla také inspirována prací [14], ve které je

¹Experimentálně bylo zjištěno, že je tento prah vhodné nastavit na hodnotu 32.

²V čárovém kódu typu EAN-13 je 30 černých čar, v typu EAN-8 je celkově 22 černých čar a obsahuje tedy 44 změn



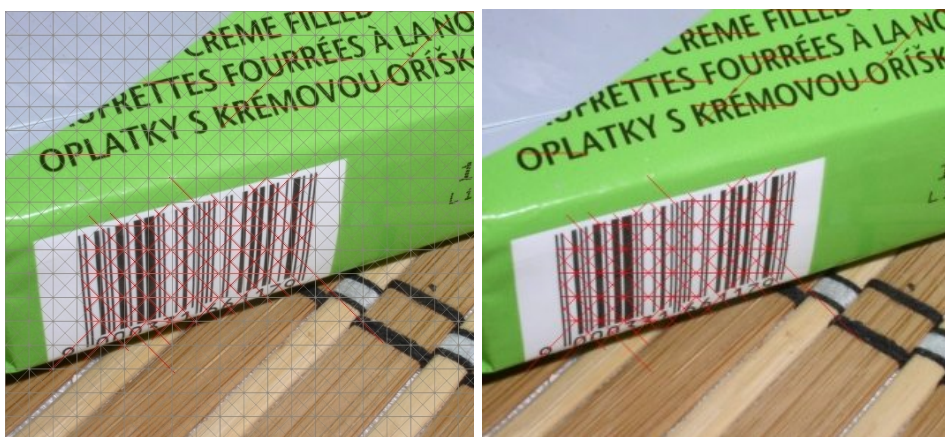
(a) Reálný čárový kód.



(b) Ideální čárový kód.

Obrázek 4.3: Porovnání reálného a ideálního profilu intenzit.

navíc potvrzení oblasti s čárovým kódem prováděno porovnáním sousedních profilů jasových intenzit. Zde metoda funguje tak, že označí začátek a konec místa, kde jsou zaznamenány časté změny intenzity. Výsledkem je pouze množina bodů, které ohraničují oblasti čárového kódu. Informace o směru kódu je dána pouze čtecí řádkou, ve které byly změny nalezeny. Výsledkem tohoto algoritmu jsou zvýrazněné body na čtených řádcích. Z těchto bodů je pomocí algoritmu RANSAC [8] vytyčena souvislá oblast.



(a) Mřížka čtených řádků.

(b) Zvýrazněné řádky detekující čárový kód.

Obrázek 4.4: Princip přístupu řádkového čtení.

Hlavní nevýhoda tohoto přístupu spočívá v náročném vymezení výsledné oblasti a určení rotace kódu. Při nalezení velkého počtu hraničních bodů se vymezení oblastí zpomaluje.



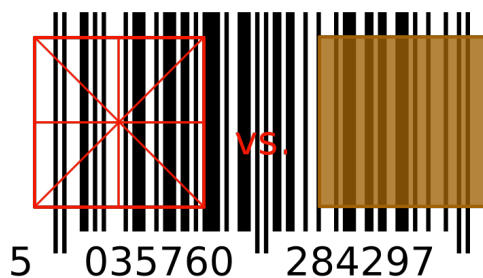
Obrázek 4.5: Výsledek metody řádkového čtení.

Metoda také pracuje s poměrně malým množstvím informace z obrazu. Na druhou stranu je tento algoritmus velmi rychlý a oproti první metodě, která je založená na hledání rovnoběžných hran, prokazuje lepší výsledky.

4.3 Blokové skenování

Pro přesnou lokalizaci čárového kódu je nutné, mimo vymezení jeho oblasti, určit také jeho orientaci. U předchozí metody je vyhodnocení oblasti, ale i orientace poměrně náročné. Pro vylepšení lokalizace a vymezení oblasti kódu je zde popsán algoritmus, který zpracovává obraz po čtvercových blocích. S návrhem tohoto algoritmu jsem se také účastnil soutěže Student EEICT 2011 [20].

Stále se pracuje s profilem jasových intenzit, který je zkoumán pouze v určitých čtených řádcích a nezatěžuje tak procesor zbytečnými operacemi nad celým obrazem. Při takovém zpracování je znatelná úspora načítaných dat. Na obrázku 4.6 je zobrazeno porovnání nad jakým množstvím dat se provádí operace při čtení určitých řádků a při zpracování celého bloku. Červeně jsou vyznačeny pixely, které jsou zpracovávány čtenými řádky. Oproti první navržené metodě, tato neprovádí operace nad všemi pixely v rámci bloku.



Obrázek 4.6: Úspora dat při zpracování řádků jasových intenzit v rámci bloku.

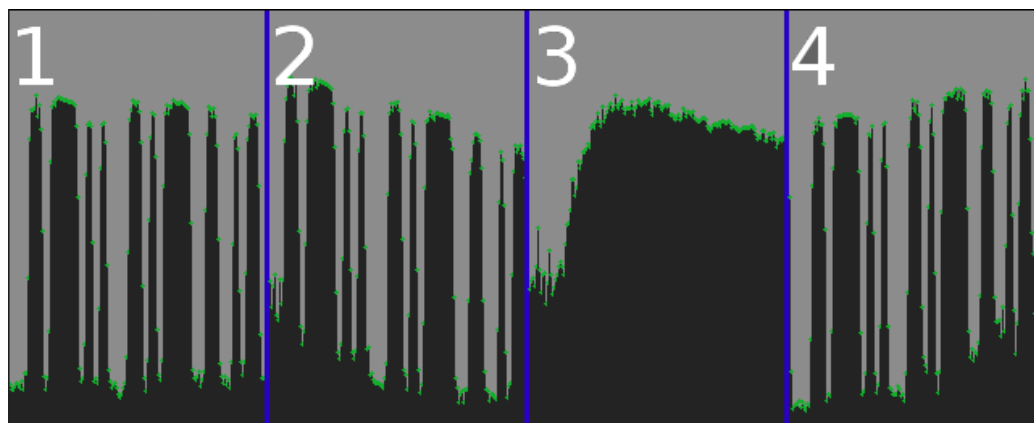
Vyhodnocení je prováděno v rámci čtvercového bloku, nad všemi čtyřmi čtenými řádky. Pozitivně vyhodnocený blok je označen jako kandidátní blok oblasti čárového kódu. Použití tzv. bloku přináší možnost výpočtu dalších kritérií. Při aplikaci metody se však ukázalo, že výpočet vlastností bloku jako např. četnost jednotlivých hodnot intenzit nebo jejich poměr,

průběh metody výrazně zpomaluje. Pro jednoduchost je počítána pouze četnost změn.

V rámci bloku je orientace určena podle řádku s největším počtem změn intenzit. Určení oblasti čárového kódu je provedeno spojením kandidátních bloků detekující jeho stejnou orientaci.



(a) Blok a čtecí řádky.



(b) Odpovídající profily jasových intenzit

Obrázek 4.7: Znázornění blokového skenování.

Vztah pro výpočet počtů změn je určen podle rovnice 4.3. Kde C je počet změn pro určitý směr j . Směry jsou celkem čtyři, horizontální, vertikální a dva diagonální podle orientace čtecích řádků. M_i reprezentuje i -tý lokální extrém ze signálu intenzit přečteného daným řádkem a n je počet těchto extrémů. Rozdíl mezi dvěma po sobě jdoucími extrémy musí být větší než prah T^3 .

$$C_j = \sum_{i=1}^{n-1} \begin{cases} 1 & |M_i - M_{i+1}| > T \\ 0 & jinak \end{cases} \quad (4.3)$$

$$C_1 > K_1 \quad \wedge \quad C_3 \leq K_{min} \quad \wedge \quad (C_2 \approx C_4) > K_2 \quad (4.4)$$

Výsledné počty změn v daném bloku musí splňovat podmínky z 4.4. Tato rovnice platí konkrétně pro kód se svislou orientací (orientace je určena podle směru čar). C jsou počty změn v jednotlivých směrech, pro každý směr je jiný prah. K_1 je prah pro počet změn v kolmém směru (směr osy čar) na orientaci čárového kódu. V tomto směru musí být změn

³Tento práh je podle experimentů nastaven na hodnotu 32

nejvíce. Pro počet nalezených změn ve směru kódu je hodnota prahu minimální K_{min} . Neočekává se zde žádná změna, protože čtecí úsečka kopíruje čáru nebo mezeru tj. místo se stejnou intenzitou. Pro poslední dva diagonální směry je hodnota prahu K_2 stejná, přičemž hodnoty změn musí být přibližně stejné v obou.

Pro zpřesnění výsledku jsou ještě počítány rozdíly dvou po sobě jdoucích hodnot intenzit. Nejsou počítány z lokálních extrémů, ale ze všech hodnot intenzit a říkají kolik tzv. ostrých změn je v daném směru.

Volba prahů je pro spolehlivou lokalizaci nejdůležitějším parametrem. Pokud jsou nastaveny přísně, lokalizační algoritmus mnohé kódy vynechá, naopak při nastavení s velkou tolerancí je lokalizováno mnoho oblastí neodpovídající čárovému kódu. Prahy jsou zvoleny na základě šířky skenovacího bloku. Informace o nastavení hodnot prahů a šířky bloku budou doplněny v následujících kapitolách. Kvůli tomu, aby byly nalezeny čárové kódy všech velikostí, je třeba měnit velikost skenovacího bloku. Jako nejlepší se ukázalo začít s největší šířkou bloku a postupně ji snižovat. Velké kódy budou lokalizovány blokem větší šířky. Při hledání menších kódů s malou šířkou bloku už není nutné vyhodnocovat bloky nad oblastmi s nalezeným kódem.

Hlavní předností je jednoduchost a nízká výpočetní náročnost algoritmu. Oproti předchozí metodě je možné lépe vyhodnotit segmenty obrazu a určit orientaci jím detekovaného kódu. Takto nalezené oblasti lze spojit do příslušných oblastí, ve kterých se nachází čárový kód. Nedostatečnou detekci prokazuje metoda při kódech, které jsou natočeny tak, že přesně neodpovídají žádné ze základních orientací. Metoda také může chybně detekovat části obrazu, ve kterém jsou profily jasových intenzit velmi podobné profilu čárového kódu. To jsou především oblasti s textem.

Kapitola 5

Implementace

V této kapitole jsou nejprve popsány implementační detaily a prostředky, které byly při práci použity. Následuje popis funkčnosti a implementace jednotlivých aplikací, které pro účely testování a implementace navrženého algoritmu vznikly. Výsledná aplikace byla naprogramována s použitím jazyka C++ a knihovny OpenCV, která nabízí vhodné funkce a třídy pro práci s obrázkovými daty. Jelikož je knihovna dostupná pro jazyk C/C++, byla tato kombinace programovacích prostředků užitečná.

5.1 OpenCV a jazyk C++

Pro implementaci byl zvolen programovací jazyk C++, který je nadstavbou nad jazykem C. Původně byl koncipován jako nadstavba jazyka C s třídami (C with classes), třídy jsou rozšířené datové struktury, které mohou obsahovat jak proměnné, tak funkce. Používá prvky nízkoúrovňového jazyka, ale na rozdíl od C poskytuje objektově orientovaný přístup a další vlastnosti jako dědičnost nebo přetěžování operátorů. Umožňuje rychlou a efektivní práci s daty a pamětí. I když se jedná o objektově orientovaný jazyk, podporuje také procedurální a generické programování, až na několik výjimek je zpětně kompatibilní s jazykem C. C++ obsahuje standardní knihovnu šablon (STL Standard Template Library). Obsahuje mnoho užitečných šablon datových struktur, kontejnerů, iterátorů a jiných algoritmů. Pro efektivní práci s intenzitami je použit datový kontejner `vector`.

OpenCV je opensource knihovna zaměřena na počítačové vidění a zpracování obrazu. Je napsána v C/C++ a podporována mnohými platformami, existuje pro nejrozšířenější operační systémy Windows, Linux a Mac OS. Byla navržena pro rychlé a efektivní zpracování, použitelná je zejména pro real-time aplikace. Obsahuje různé optimalizační rutiny pro procesory Intel, tím pádem dosahuje rychlejšího zpracování. Poskytuje programátorovi pohodlné prostředí především pro práci s obrazovými daty. Zapouzdřuje implementaci mnoha známých a používaných algoritmů z oblasti počítačové grafiky, strojového učení a počítačového vidění. Zvolena byla hlavně pro svou robustnost při zpracování vstupních dat, podporuje mnoho formátů, jak obrázků tak videa, dokáže i zpřístupnit vstupní data z webové kamery. Nabízí přístup k obrázkovým datům pro zápis nebo čtení a dokáže je zobrazit i uložit do obrázku. Pro účely aplikace představené v této práci byla knihovna využívána hlavně pro načítání, ukládání a zobrazování obrázků. K tomu slouží moduly CXCORE, CV a HighGui. Další moduly, které knihovna nabízí jsou CVCAM, CVAUX a MLL[4].

5.2 Návrh programu

Celý systém pro lokalizaci čárového kódu by měl fungovat jako konzolová aplikace. Jádro programu demonstrující lokalizaci čárového kódu pracuje tak, že nejprve otevře daný soubor (obrázek), provede detekční a lokalizační algoritmy a zpracuje výsledek do výstupního souboru. Pro dekódování kódu je také důležité vytyčit přímkou, podle které půjde kód přečíst. Příмка by měla protínat celý kód a kopírovat osu kódu. Podle koncepce návrhu čárového kódu je jedno, jakým směrem bude kód přečten. Čtecí linie také nutně nemusí procházet osou kódu. Poměr šířek čar a mezer zůstává vždy stejný, ať je nakloněna jakkoliv.



Obrázek 5.1: Výsledek demonstrační aplikace.

Aplikace je zaměřena na to, aby dokázala najít takovou čtecí linii, kterou je možno čárový kód dekódovat. Výsledkem je kopie vstupního obrázku se zvýrazněnými nalezenými kódy a čtecími liniemi viz. obrázek 5.1. Zvýrazněny jsou také kandidátní bloky, jejich orientace jsou od sebe odlišeny barevně.

5.3 Anotace

Třída `anote` je používána hlavně pro manipulaci s obrázkovými daty. Obsahuje statické metody, které umožní načítání a ukládání obrazu. S její pomocí lze také vykreslovat do obrázku základní tvary a tím pádem vizualizovat výsledky prováděných operací. Pomocí této třídy je vytvořena jednoduchá anotační aplikace, která slouží k anotaci snímků. Takové snímky mohou posloužit k testování.

5.4 Třída Scanner

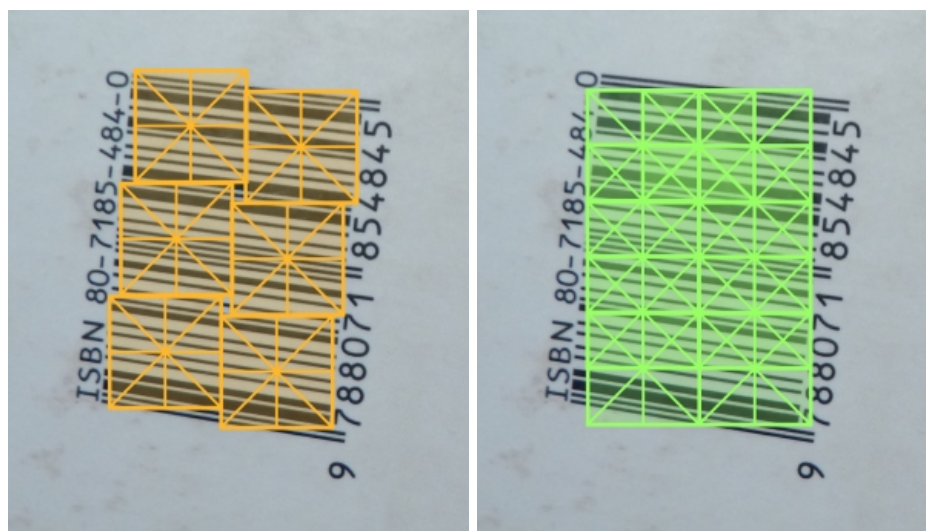
Jádrem aplikace je třída `Scanner`, ta obstarává detekci čárového kódu. Z otevřeného snímku načítá obrázková data podél čtecích řádků. Z nich vypočítá hodnoty jasových intenzit, které ukládá do patřičných vektorů. Metoda `BlockScan()` má za parametr šířku bloku, načítá vektory intenzit pro každý blok a provádí vyhodnocení jednotlivých bloků. Určí výsledný směr bloku a uloží jej do seznamu kandidátních bloků, ten je poskytnut objektem třídy `MyLocator`.

Snímek je zpracováván čtvercovými bloky. Nejprve je blok umístěn v levém horním okraji obrazu, oblast pokrývaná tímto blokem je vyhodnocena a blok je posunut. Kvůli přesnosti lokalizace je blok posouván o polovinu své šířky, tím pádem se bloky překrývají, jak je vidět v obrázku 5.2(b). Může se zdát, že to vede k redundantnímu vyhodnocování a

zbytečnému zatížení. Ve výsledku je ovšem metoda lokalizace o mnoho přesnější. Větším překrytím by bylo dosaženo ještě přesnějšího vyhodnocení, kvůli rychlosti je však použito pouze poloviční překrytí bloků.

Jakmile je celý obraz zpracován blokem původní šířky, je šířka bloku zmenšena (míra zmenšení závisí na rozlišení snímku) a metoda `BlockScan()` provedena znovu se změněnou šířkou. Nyní už se nenačítají vektory intenzit z oblastí, které jsou vyhodnoceny kladně. Informace o vyhodnocených oblastech jsou obsaženy v tzv. masce obrazu, implementovanou třídou `zMask`, které je věnována následující podkapitola. K výraznému zrychlení přispívá vyhodnocení nezajímavých oblastí, s profilem intenzit který neobsahuje četné změny. Tyto oblasti jsou uloženy do masky a díky tomu nejsou podrobeny dalším zbytečným výpočtům.

Důležité je zmínit, že při načítání dat ze snímku je také provedeno rozhodování o počáteční šířce bloku a kroku její změny. Velikosti čárových kódů mohou být různé. Proto je složité zvolit takovou velikost bloku, aby pokryla všechny možné čárové kódy. Další faktor, pro který je volba šířky bloku podstatná, je poměr šířky čar mezer, který závisí na rozlišení snímku a vzdálenosti kódu od objektivu při jeho pořizování. Blok musí v náležitém řádku detekovat velký počet změn, aby byl vyhodnocen kladně. Z tohoto důvodu je volba šířky klíčová pro kvalitu vyhodnocení. Hodnoty počáteční a koncové šířky bloku a také kroku, se kterými je měněna byly zvoleny na základě experimentování s pořízenými obrázky. Rozhodovací kritéria pro vyhodnocení bloku jsou nastavena tak, aby blok o velikosti alespoň poloviny výšky čárového kódu detekoval jeho oblast viz. obrázek 5.2(a).



(a) Bloky o šířce poloviny výšky kódu musí detekovat jeho oblast.

(b) Překrytí bloků.

Obrázek 5.2: Některé implementační vlastnosti algoritmu.

5.5 zMask

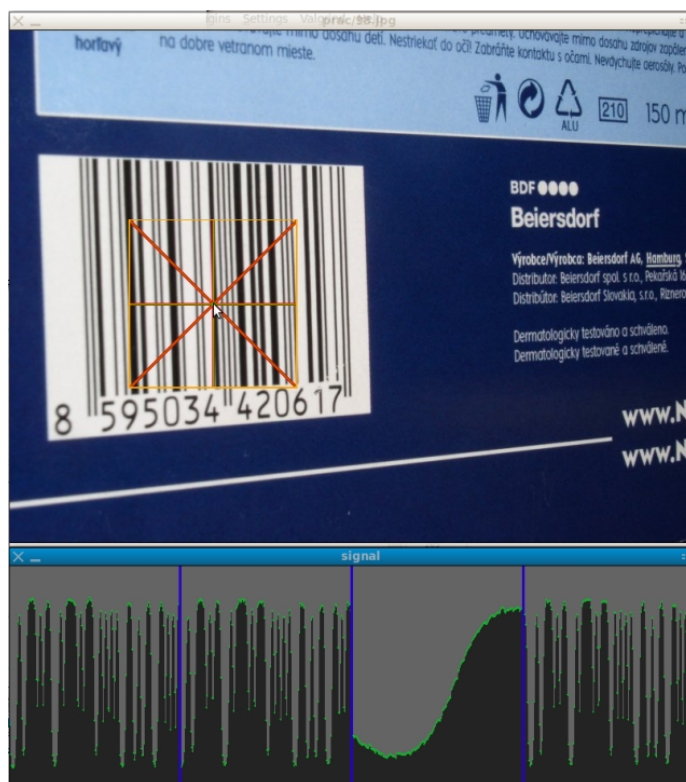
Proces lokalizace pracuje s postupným snižováním šířky skenovacího bloku. Toto nutně vede k vyhodnocování podoblastí, které už jsou označeny jako bloky náležící čárovému kódu. Maska je matice o stejných rozměrech jako vstupní snímek. Do masky je zaznamenáno, v jakých oblastech detektor našel blok odpovídající čárovému kódu. Není zde nutné

provádět nová vyhodnocování s nižším krokem. Také je možno vyhodnotit a označit oblasti, které jistě nebudou obsahovat čárový kód, protože v žádném směru nejsou zaznamenány vysoké změny intenzit. V masce zůstanou nevyplněny pouze oblasti, ve kterých se může potencionálně vyskytovat čárový kód. Použití tohoto principu výrazně zlepšuje rychlost algoritmu.

Tuto třídu lze použít také pro vybrání výsledných oblastí. To znamená, že je zde možnost upravení tohoto algoritmu. Spojení bloků do oblastí je však realizováno objektem třídy MyLocator.

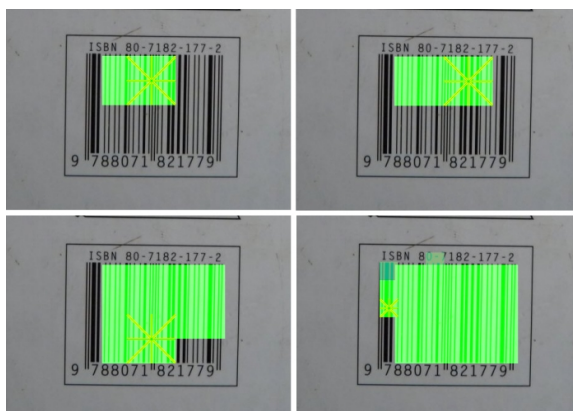
5.6 Třída Inter

S touto třídou je spojena Interaktivní aplikace zobrazena na obrázku 5.3, která je vedlejším produktem této práce. Pro zkoumání obrazových dat bylo třeba vytvořit aplikaci, která dokáže vybrat a zobrazit jasové intenzity z určitého bloku. Aby byla uživatelsky přívětivá, pozice bloku je nastavitelná myší. Je možné také měnit šířku bloku a nechat vypsát na standardní výstup hodnoty různých proměnných související se signálem.



Obrázek 5.3: Ukázka interaktivní aplikace.

Při testování detekční části algoritmu byla sestavena také aplikace, která demonstruje jak po krocích probíhá celý proces detekce. V každém kroku je zobrazen zkoumaný blok a vypsány informace o jeho charakteristice. Na obrázku 5.4 jsou pak barevně znázorněny vyhodnocené oblasti příslušící čárovému kódu.



Obrázek 5.4: Několik kroků detekční metody.

5.7 Třída MyLocator

Posledním a velmi důležitým prvkem celé aplikace je výběr a sjednocení detekovaných bloků do souvislých oblastí, které odpovídají čárovému kódu. Ve třídě `MyLocator` jsou obsaženy metody, zaměřující se na tento problém. Třída `Scanner` naplní seznam všech detekovaných kandidátních bloků čárových kódů. Nyní je třeba určit, jaké bloky odpovídají kterému kódu. Z tohoto seznamu jsou bloky postupně vybírány a ke každému jsou hledány bloky, jejichž hranice s ním mají průsečík. Ty jsou ukládány do seznamů odpovídajících společných oblastí. Každý blok, který už nemá žádné sousední bloky, je potvrzen a pokračuje se v hledání protínajících se bloků jeho souseda. Jakmile jsou v oblasti všechny bloky potvrzeny, tzn. neexistuje další blok se společnou nebo protínající se hranicí, přejde se k vymezení další oblasti.

Pokud jsou nalezeny protínající se bloky s orientacemi, které spolu nemohou být spojeny (např. vodorovná se svislou), není tento blok přidán do společné oblasti a zůstává nevyhodnocen. Naopak různá natočení kódu způsobují, že je žádoucí spojit bloky oblastí příbuzných orientací. Příbuzné orientace jsou vertikální a diagonální nebo horizontální s diagonální. Důležité je pouze, aby nebyly v jedné oblasti spojeny bloky se vzájemně kolmou orientací.

Algoritmus skončí při vyprázdnění seznamu všech kandidátních bloků. Může mít v nejhorším případě kvadratickou časovou složitost, proto může celý proces lokalizace značně zpomalit. Je jasné, že čím více kandidátních bloků bude nalezeno, tím pomaleji bude algoritmus vyhodnocovat spojitě oblasti kódu.

V závěrečném zpracování oblastí jsou podle orientace oblasti předpočítány body, jimiž bude veden řádek, kterým může být kód přečten. Je zde také provedena zpětná kontrola. V řádku, který má přečíst čárový kód je vypočítán počet změn a poměr jejich průměrných vzdáleností v závislosti na délce tohoto řádku. Tímto se částečně eliminují oblasti, které mají podobné charakteristiky jako čárový kód. Zde by ke kontrole měl sloužit modul čtečky čárových kódů. Protože není jasné jaký druh kódu umí čtečka číst, není vhodné provádět takovéto kontroly při procesu lokalizace. Vzhledem k tomu, že implementace čtečky není náplní této práce, je algoritmus zaměřen na co nejpřesnější nalezení kódu typu *EAN-13*.

Podstatné je, že z obrazu lokalizuje pouze taková místa, ve kterých by čárový kód mohl být čitelný. Pokud lokalizační algoritmus selže a poskytne čtečce oblast, která neodpovídá čárovému kódu, čtečka nebude takovéto posloupnosti intenzit rozumět a nedetekuje kód.

Hlavní je, že lokalizátor eliminuje podstatnou část oblastí a nebude zatěžovat čtečku zbytečnou snahou o čtení vzorku, který neodpovídá kódu.

5.8 Reader

Třída `Reader` je nástavbou nad celým programem. Ve skutečnosti neslouží k dekodování oblasti s čárovým kódem, ale pouze k provedení výřezu takové části obrazu, ze které je možno kód přečíst. Je schopna uložit a zobrazit obraz tvořen profilem jasových intenzit a obraz vzniklý jejich binarizací. Ta probíhá jednoduchou metodou Otsuho prahování. Vizualizace metody, prováděnou objektem třídy `Reader`, je patrná na obrázku 5.5. Barevně



Obrázek 5.5: Názorná ukázka modulu Reader.

je označena oblast kódu a orámovaná oblast je čtecí linie kódu. Okno v horní části obrázku demonstruje přečtený řádek a jeho binarizovaný obraz.

Kapitola 6

Testování

Pro testování algoritmů byla pořízena sada 545 fotografií v různých rozlišeních a z několika přístrojů. Pro zaručení objektivitu testů bylo nutno pořídit snímky v různých kvalitách. Je žádoucí, aby sada obsahovala také snímky, kde je čárový kód poškozen, zastíněn, nasvětlen, rozmazán nebo jinak deformován. Všechny snímky byly řádně anotovány, aby bylo možno automaticky vyhodnotit přesnost lokalizační metody. Anotování probíhalo za využití anotační aplikace. Anotovaný textový soubor má následující strukturu:

```
#obraz.jpg # 2
[192;103] [221;329] [312;365] [295;116]
[416;107] [408;311] [726;281] [741;115]
```

První řádek poskytuje informaci o názvu souboru a počtu čárových kódů. Každý další řádek uchovává informaci o čtyřech hraničních bodech čarového kódu.

Testování je nezbytné pro reprezentaci jakýchkoliv výsledků. Z výsledků testů je možno usoudit nakolik je algoritmus spolehlivý. Za úspěch se považuje nalezení všech kódů a správné určení jejich rotace. Při takovémto vyhodnocení lze nalézt linii, podle které bude čárový kód přečten. Algoritmus by se měl chovat tak, že nalezne všechny oblasti, ve kterých odhalil výskyt čarového kódu. Vynechání oblasti s čarovým kódem se považuje za větší neúspěch než detekce falešné oblasti s velmi podobnou charakteristikou. Vyhodnocení kvality algoritmu navrženého v této práci bude rozebráno podle několika kritérií.

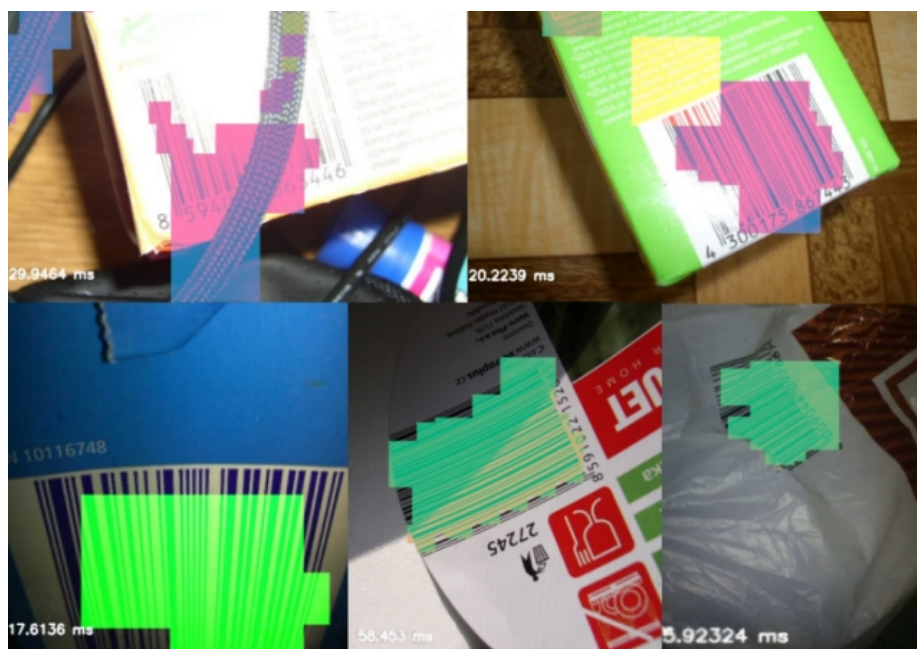
6.1 Test správné detekce

Testování správné detekce bylo prováděno na celé sadě snímků. Vyhodnocení probíhalo vizuálně. Za úspěch bylo považováno správné nalezení oblasti s čarovým kódem a určení jejich orientace. Algoritmus je navržen tak, aby detekoval i oblasti, které neodpovídají kódu, ovšem takovéto oblasti by měly být pozdějším provedením lokalizace a zpětné kontroly vyloučeny. Jedná se tedy o test správného vyhodnocování bloků. Výsledky jsou uvedeny v tabulce 6.1. Detekovaným kódem se rozumí takový kód, nad kterým byla správným vyhodnocením bloku vyznačena oblast příslušné orientace. Pokud byla orientace určena špatně nebo byl kód detekován pouze z velmi malé části či vůbec, je to považováno za neúspěch.

Výsledky detekce lze pozorovat na obrázku 6.1. Je nutno podotknout, že mnohé kódy obsažené v pořízených obrázcích jsou náročné pro detekci. Výsledky jsou i přes přítomnost stínů poměrně uspokojivé. Jsou zde vidět i falešné detekce v části obrazu obsahující text nebo jiné oblasti s charakteristikami podobnými čarovému kódu.

Skupina snímků	Počet snímků (kódů na snímcích)	Detekováno	Úspěšnost [%]
640x480 px	150 (205)	190	92
800x600 px	80 (110)	106	96
1024x768 px	100 (124)	109	87
1280x1024 px	45 (63)	50	79
1600x1200 px	70 (96)	90	93
Ostatní	100 (112)	85	75

Tabulka 6.1: Výsledky algoritmu detekce, čárového kódu.



Obrázek 6.1: Výsledky detekce čárových kódů (barva označuje příslušnou orientaci).

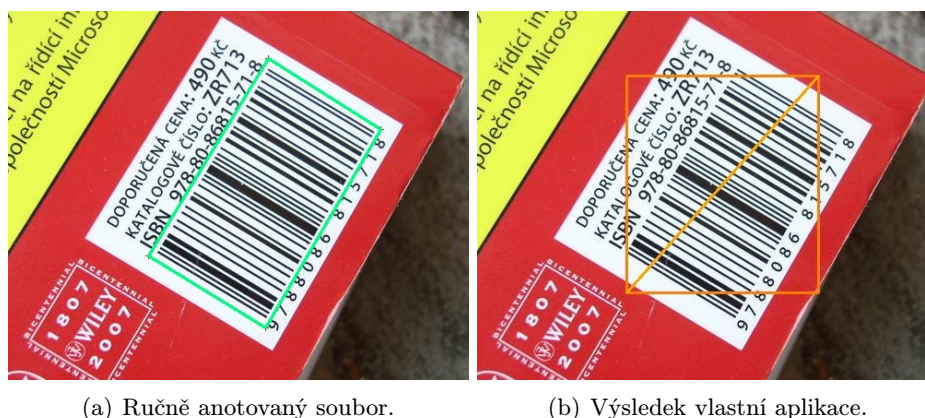
Celková úspěšnost detekce se pohybuje okolo 88%, což se přibližuje úspěšnosti existujících metod. Na druhou stranu zde nejsou zahrnuty dopady falešných detekcí, které se promítnou do chybné lokalizace. Velkým problémem je totiž nastavit algoritmus tak, aby vhodně detekoval všechny kódy a zároveň se v detekci neprojevila chyba. Parametry je nutno přispůsobit co největší množině možností, existuje ale mnoho extrémních případů, které nemůže jednoduchý algoritmus pokrýt. Protože se zaměřuji především na rychlost zpracování, je nutné detekovat převážně ty kódy, které mohou být čitelné. Příliš rozmazané či jinak deformované kódy nemusí být správně detekovány.

6.2 Vyhodnocení lokalizační metody

Při implementaci lokalizační metody byl brán ohled na to, že bude používána pro zpracování snímků v reálném čase. Mohla by tak sloužit pro lokalizaci čárových kódů snímaných přímo webovou kamerou. V současné době jsou pro zaznamenávání obrazu webové kamery velmi rozšířeny. Nejčastěji pořizují obraz v nižším rozlišení, proto byla lokalizační metoda

implementována s tímto ohledem. Můžeme si všimnout, že nejlepší výsledky jak lokalizace tak detekce jsou patrné právě na takovýchto snímcích.

K testování správnosti lokalizační metody bylo použito anotovaných snímků. Později se ukázalo, že vyhodnocení není příliš adekvátní kvůli ne zrovna vhodnému počítání okrajových bodů. V anotačních souborech jsou přesně vyznačeny souřadnice hranic kódu. Při lokalizaci je především nutné správně zobrazit a vypočítat čtecí linii, proto nebyl kladen velký důraz na zaznamenání přesné pozice hranic kódu. Jak vidíme na obrázku 6.2. Zvýraznění lokalizovaného kódu je provedeno osově zarovnaným obdélníkem, který nebývá natočen stejně jako čárový kód. Proto různě natočené kódy nebudou mít hraniční body umístěny přesně podle anotovaného souboru.



(a) Ručně anotovaný soubor.

(b) Výsledek vlastní aplikace.

Obrázek 6.2: Srovnání testovacích obrázků.

Pro spolehlivou lokalizaci není důležité jen vymezení oblasti, ale hlavně určení linie, podél které bude kód přečten. Toto je ve velké míře dosaženo. Nepřesné určení čtecí linie může být způsobeno tím, že řez v obraze byl proveden místem, ve kterém změna intenzit odpovídá čárovému kódu. Za chybu při lokalizaci je považováno přílišné odchýlení od oblasti s čárovým kódem. Špatné vymezení čtecí linie je druhá kategorie chyby, na kterou se testování také zaměřuje. V následující tabulce 6.2 jsou uvedeny výsledky testování aplikace. Je zde sloupec s poměrem správně lokalizované oblasti ku počtu detekovaných kódů, počet chybně nalezených čtecích linií a úspěšnost.

Skupina snímků	Správná lokalizace	Chybná linie	Úspěšnost [%]
640x480 px	146/190	30	76
800x600 px	91/106	22	85
1024x768 px	90/109	20	82
1280x1024 px	45/50	14	90
1600x1200 px	77/90	21	85
Ostatní	70/85	15	82

Tabulka 6.2: Přesnost lokalizace.

Algoritmus je navržen především pro rychlé zpracování a proto není nastaven tak, aby spolehlivě lokalizoval i problematické kódy. Celková úspěšnost lokalizace, když vezmeme v potaz i chybu detekce, se pohybuje okolo 73%. Čtecí linie je s ohledem na rychlost vypočítána pouze podle výsledného směru nalezené oblasti. Ke nutno zdůraznit, že u většiny

existujících metod je testování prováděno na ideálních snímcích, často stejného rozlišení.

Pro korektní vyhodnocení lokalizační části byly brány v potaz pouze snímky se správně provedenou detekcí. Úspěšnost lokalizace je možné zlepšit eliminací detekce oblastí s textem. Kvůli podobnosti textur řádků textu a čárového kódu je to velmi problematické. Zpřísnění parametrů detekce by také mohlo přinést nedostatečnou detekci rozostřených kódů. V testovací sadě jsou obsaženy snímky s komplexní scénou, také kvality některých kódů jsou nedostatečné a pro lokalizaci a detekci kódu složité. Proto je nutno zvážit tento fakt při vyhodnocení úspěšnosti.

Vyhodnocení rychlosti je prováděno nad sadou obrázků o různých rozlišeních¹. Dosažené výsledky lze možno pozorovat v tabulce 6.3, kde je uvedeno, jakou rychlostí je zpracován snímek. Takovéto rychlosti jsou velmi pozoruhodné. Málokteré existující metody se k nim přibližují. Na druhou stranu lokalizace je na úkor rychlosti méně spolehlivější než v existujících aplikacích. V dostupných zdrojích [25, 24] byla nejrychleji prováděna metoda, která lokalizovala kód v obraze o rozlišení 512x512 pixelů za 62ms. Pro ověření rychlosti byly provedeny testy na několika videosekvencích, které jsou obsaženy na příloženém CD. Při tomto testování se ukázalo, že je metoda vhodná pro lokalizaci čárových kódů v reálném čase.

Výsledky jsou zobrazeny v příloze na obrázcích B.1 a B.2. Vybrány jsou i snímky horších kvalit, aby bylo demonstrováno jaké netriviální kódy je algoritmus schopen lokalizovat.

Rozlišení snímku [px]	Rychlost		
	Nejlepší [ms]	Nejhorší [ms]	Průměrná [ms]
640x480	4	13	8
800x600	6	31	11
1024x768	8	36	19
1280x960	16	57	32
1600x1200	30	120	68

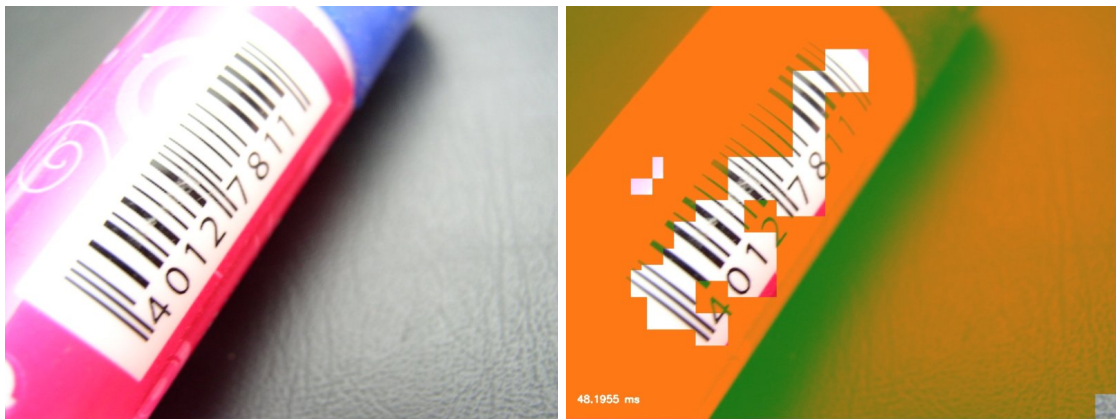
Tabulka 6.3: Rychlost zpracování jednotlivých snímků.

6.3 Nedostatky metody

Při testování se ukázaly nedostatky implementovaného algoritmu. Už dříve byl nastíněn problém s výběrem šířky skenovacího bloku. Hlavní nedostatek navržené metody je v nedostatečné detekci kódů malé výšky, a také kódů neostrých nebo vzdálených od objektivu. Ty ale bývají často nečitelné. Nejlepší výsledky lokalizace byly zaznamenány na snímcích, ve kterých kód pokrývá 15-65 %.

Na obrázku 6.3 můžeme vidět příklad selhání lokalizační metody. Zobrazený čárový kód je poměrně složitý pro lokalizaci. Má malou výšku, je natočen a osvětlen. Z toho vyplývá, že oblast bude detekována až blokem malé velikosti. Tento kód má navíc velké šířky čar a mezer, proto v rámci malého bloku nebude zaznamenáno dostatečné množství změn. Pokud by byla nastavena méně přísná kritéria, metoda by v nepozměněné oblasti zobrazené na obrázku 6.3(b) bloky kladně vyhodnotila. V jiných snímcích by taková změna přísnosti vedla k nadměrným falešným detekcím oblastí s textem. Není proto jednoduché nastavit kritéria pro všemožné rotace a orientace kódů.

¹Testy byly prováděny na notebooku s procesorem Intel Core 2 Duo T6600 2.2GHz



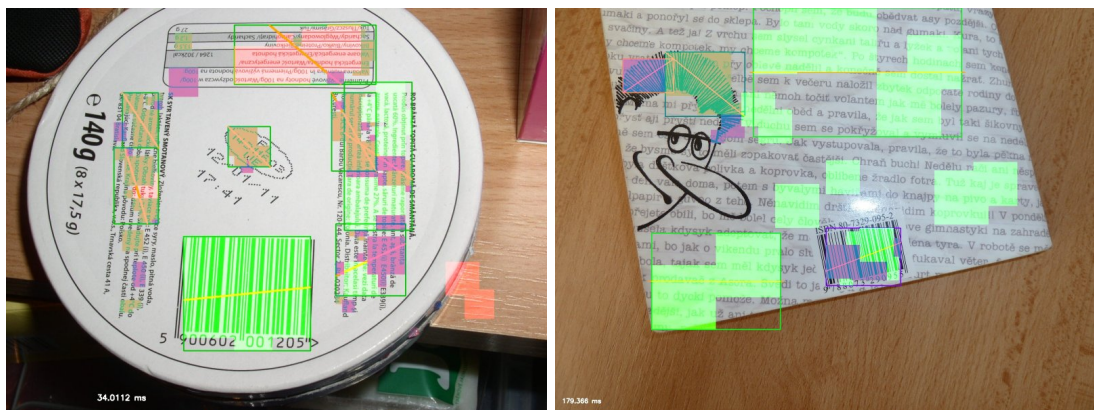
(a) Chyba v detekci kódu.

(b) Eliminace nekandidátních bloků pomocí zMask.

Obrázek 6.3: Selhání lokalizace.

Problémy v detekci jsou způsobeny především nevhodnou volbou šířky bloku a kroku, se kterým se snižuje. Následkem je, že okrajové oblasti čárového kódu nejsou kvůli přítomnosti ochranné zóny, tedy oblasti bez změn v hodnotách intenzit, detekovány. Tato záležitost je patrná pouze pokud je snaha detekovat kód svislé nebo vodorovné orientace. Proto je při vymezení oblasti takto natočených kódů prodlužována čtecí linie tak, aby pokrývala celý kód.

Největším problémem navrhované metody je nedostatečná detekce kódů ve snímcích s vysokým rozlišením. V takových snímcích bývá poměr šířek čar a mezer kódu značně jiný než ve snímcích s nízkým rozlišením. Je to způsobeno především tím, že snímky dosahují větší kvality a kód pokrývá velkou plochu obrazu. V rámci bloku tak není detekováno mnoho změn. Řešení tohoto problému je však možné. Vyžadovalo by to ale podrobnější analýzu problematických snímků a změnu parametrů na základě rozlišení a kvality snímku. Ideální by bylo provést vhodné předzpracování obrazu, které by zpřesnilo informaci o velikostech očekávaných kódů ve zkoumaném obraze.



(a) Chybná detekce textu.

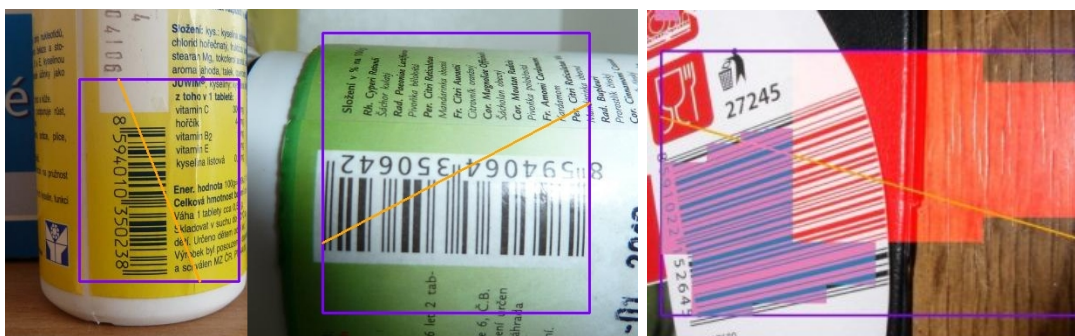
(b) Nalezení chybných oblastí.

Obrázek 6.4: Chyby v lokalizaci.

Další potíže nastaly při lokalizaci kódů, které byly zastíněny, zpřehýbány či deformovány nebo nedodržovaly parametry, jakými jsou velikost klidové zóny a barva čar. Lokalizace také

selhala na špatně zaostřených nebo přexponovaných snímcích. Takové nedostatky se však daly očekávat. Často také dochází k falešným detekcím, zejména bývá takto detekován text. Ten má velice podobné profily jasových intenzit jako čárový kód.

Pro eliminaci detekce a lokalizace oblastí s textem je možné kontrolovat změny v řádcích rovnoběžných s nalezenou čtecí linií. Většina takových kontrol způsobuje výrazný pokles rychlosti zpracování. Navíc tento princip není vždy přesný a v praxi se zcela neosvědčil. Proto byl zvolen takový kompromis, aby lokalizace probíhala rychle s tím, že může čteče poskytnout malé množství chybných výsledků.



(a) Chybná čtecí linie.

(b) Spojení nesouvisejících oblastí.

Obrázek 6.5: Chyby při výpočtu čtecí linie.

Posledním problémem, který se projevuje je špatný výpočet koncových bodů čtecí řádky. Zobrazeno na obrázku 6.5. Pro spolehlivější výpočet koncových bodů je možné zkusit čtecí linii na základě orientace oblasti natočit. Za lepší výsledek je možno považovat takovou linii, která pokrývá větší množství čar v kódu. Tyto dodatečné výpočty by však metodu příliš zatěžovaly.

6.4 Srovnání s existující aplikací

Výsledná práce byla porovnána s existující aplikací pro čtení čárových kódů ZBar². Tato aplikace funguje jako komplexní čtečka čárových kódů v obraze nebo videu. Celý systém je schopen pracovat v reálném čase. Autoři uvádějí, že pracuje na podobném principu jako 1D lineární čtečka čárových kódů. Nezpracovává obraz různými filtry, ale snaží se vyhodnocovat jasové intenzity, podobně jako mnou navržený přístup. Výsledky jsou porovnány v tabulce 6.4. V závorce je uvedeno u kolika kódů se podařilo najít správnou čtecí linii.



Obrázek 6.6: Ukázka programu ZBar pro čtení čárových kódů.

²Dostupné z <http://zbar.sourceforge.net>

Aplikace	Sada snímků (počet kódů)	Nalezené kódy (čtecí linie)	Průměrná rychlost [ms]
vlastní ZBar	Ostatní (112)	70 (55) 114	120 600
vlastní ZBar	640x480 (205)	146 (116) 162	8 50
vlastní ZBar	800x600 (110)	91 (69) 94	11 80
vlastní ZBar	1024x768 (124)	90 (20) 110	19 150
vlastní ZBar	1280x960 (63)	45 (31) 46	32 230
vlastní ZBar	1600x1200 (96)	77 (56) 79	68 360

Tabulka 6.4: Srovnání s jinou aplikací.

Rychlost vyhodnocení programu ZBar v sobě zahrnuje i dobu čtení čárového kódu, logicky proto bude pomalejší. Přechtení čárového kódu nebývá náročné. Pokud je navíc k dispozici vhodný výřez, ze kterého bude dekódování probíhat, je možno číst kódy řádkově v desítkách milisekund [25]. Aplikace ZBar je velmi robustním a dlouho vyvíjeným nástrojem. Jelikož v sobě zahrnuje i dekódování, nelze spolehlivě vyhodnotit, zda je chyba způsobena nedostatečnou lokalizací nebo nevhodným kódem pro čtení. Z testů je patrné, že mnou navržená metoda podává velmi podobné výsledky lokalizace a v rychlosti vyhodnocení je na tom ještě lépe.

Kapitola 7

Závěr

Práce se zaměřuje na návrh a implementaci vhodné metody lokalizace čárového kódu v obraze. Obecně popisuje využití a typy čárových kódů. Seznamuje také s existujícími přístupy lokalizace čárových kódů. Po nastudování a zhodnocení různých přístupů lokalizace byly navrženy tři lokalizační metody, z nichž jedna byla rozvíjena do konečné podoby. Jako užitečné se ukázalo lokalizovat na základě změn v profilu jasových intenzit. Tento přístup nepracuje s velkým objemem dat ze zkoumaného obrazu, a proto je jeho vyhodnocení rychlé a použitelné pro zpracování snímků v reálném čase.

Výsledná aplikace umožňuje lokalizovat lineární čárové kódy různých typů, přičemž důraz je kladen na čárové kódy typu *EAN*. K testování byla použita sada 545 snímků, která k zaručení objektivnosti testování poskytla vstupní data různých kvalit. Z výsledků testování algoritmu je zřejmé, že lokalizace pokryje naprostou většinu čitelných čárových kódů, což je pro zpracování čárového kódu klíčové. Chyby v lokalizaci byly způsobeny různými faktory, které byly diskutovány.

Pro dosažení větší přesnosti lokalizace by bylo nutné zvolit jiné parametry algoritmu, což by vedlo ke zpomalení vyhodnocení. Vzhledem k tomu, že je metoda navržena především pro rychlé zpracování, je důležitá její nízká náročnost. Tím pádem je doba zpracování snímku minimální. Ta byla testována jak nad testovací sadou snímků, tak nad videosekvencemi pořízenými webovou kamerou. Aplikace byla také srovnána s existující aplikací určenou k lokalizaci a čtení čárových kódů. Výsledky lokalizace byly srovnatelné a rychlost mého algoritmu dokonce lepší.

Možné rozšíření této práce by bylo v přidání modulu, který by v reálném čase dokázal dekodovat informaci z lokalizovaného čárového kódu. Výsledkem by potom byl systém pro čtení a lokalizaci čárového kódu, který by mohl fungovat v reálném čase. Své uplatnění by určitě našel ve formě automatického odbavení v obchodních domech. Dále by šlo zpřesnit výsledky lokalizace například přísnějšími kontrolami a rozšířit lokalizační metodu o podporu dalších druhů kódů.

Literatura

- [1] Aas, K.; Eikvil, L.: Decoding bar codes from human-readable characters. In *Pattern Recognition Letters 18*, Turkey, 1997, s. 1519–1527.
- [2] Adriana, B.; Štefan, M.; Stanislav, W.: *Čárové kódy - automatická identifikace*. Praha: Grada, 1994, ISBN 80-85623-66-8, 252 s.
- [3] Barcode Products: Barcode Products Ltd.
http://www.barcodeproducts.co.nz/IM_Custom/ContentStore/Assets/5/45/2e3b79f4ece0e79238537c4d6120e94a.jpg, 2011, [Citováno 10.5.2011].
- [4] Bradski, G.; Kaehler, A.: *Learning OpenCV*. O'Reilly Media, první vydání, 2008, ISBN 978-0-596-51613-0, 576 s.
- [5] Chai, D.; Hock, F.: Locating and Decoding EAN-13 Barcodes from Images Captured by Digital Cameras. In *Information, Communications and Signal Processing, 2005 Fifth International Conference*, Bangkok, 2005, ISBN 0-7803-9283-3, s. 1595–1599.
- [6] Christian, V.-G.; Nicolas, N.; Dominique, B.: A Bar Code Location Algorithm Using a Two-Dimensional Approach: s. 45–48.
- [7] ČSN 97 7101: Systém EAN. Kód UCC/EAN 128. Aplikační identifikátory. 1993.
- [8] Fischler, M. A.; Bolles, R. C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, ročník 24, 1981: s. 381–395, ISSN 0001-0782.
- [9] Gamage, R. J. E.; Kularatne, L.; Pannipitiya, K.; aj.: A low cost optical barcode reader using a webcam. In *Engineering Research Unit (ERU) Symposium*, Sri Lanka, 2003.
- [10] Gonzalez, R. C.; Woods, R. E.: *Digital Image Processing*. New Jersey: Prentice-Hall, Inc., druhé vydání, 2001, ISBN 0-201-18075-8, 793 s.
- [11] Greenlaunches: GoodGuide Launches iPhone App That Scans Green Products.
<http://www.greenlaunches.com/GoodGuide-iPhone-green-products1-thumb-450x360.jpg>, 2011, [Citováno 10.5.2011].
- [12] GS1: <http://www.gs1.org/>, [Citováno 10.5.2011].
- [13] Jain, A. K.; Chen, Y.: Bar Code Localization Using Texture Analysis. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, Japan, 1993, ISBN 0-8186-4960-7, s. 41–44.

- [14] Kostiha, M.: *Dekódování čárového kódu v obraze v reálném čase*. Diplomová práce, FIT VUT v Brně, 2010.
- [15] Kulyukin, V.; Kutiyawala, A.: *Eyes-Free Barcode Localization and Decoding for Visually Impaired Mobile Phone Users*. San Antonio, Texas, 2010.
- [16] Muniz, R.; Junco, L.; Otero, A.: A Robust Software Barcode Reader Using the Hough Transform. *Information, Intelligence, and Systems, International Conference on*, ročník 0, 1999: str. 313.
- [17] Oktem, R.: Bar code localization in wavelet domain by using binary morphology. In *Signal Processing and Communications Applications Conference, 2004. Proceedings of the IEEE 12th*, april 2004, s. 499–501.
- [18] Oktem, R.; Oktem, L.: A Superresolution approach for Bar code Reading. In *13th European Signal Processing Conference (EUSIPCO-2005, Turkey, 2005*.
- [19] Palmer, R. C.: *The Bar Code Book*. Helmers Pub, 2001, ISBN 978-0911261134, 395 s.
- [20] Šimurda, P.: Barcode localization in image. In *Proceedings of the 17th Conference STUDENT EEICT 2011 Volume 1*, Brno: FIT VUT, 2011, ISBN 978-80-214-4271-9, s. 169–171.
- [21] Som, H. M.; Zain, J. M.; Ghazali, A. J.: Application of Threshold Techniques for Readability Improvement of Jawi Historical Manuscript Images. *Advanced Computing: An International Journal*, ročník 2, 2011.
- [22] Tekin, E.; Coughlan, J. M.: A Mobile Phone Application Enabling Visually Impaired Users to Find and Read Product Barcodes. In *ICCHP'10 Proceedings of the 12th international conference on Computers helping people with special needs*, Berlin, 2010, ISBN 3-642-14099-8, s. 290–295.
- [23] Tunistra, T. R.: *Reading Barcodes from Digital Imagery*: str. 18.
- [24] Xiangju, L.; Guoliang, F.; Yunkuan, W.: A Robust Barcode Reading Method Based on Image Analysis of a Hierarchical Feature Classification. In *IEEE/RSJ International Conference on Intelligent Robots and Systems 2006*, Beijing, 2006, ISBN 3-642-14099-8, s. 3358–3362.
- [25] Zhang, Z.; Wang, J.; Han, S.; aj.: Automatic real-time barcode localization in complex scenes. In *Image Processing, 2006 IEEE International Conference*, Atlanta, 2006, ISBN 1-4244-0480-0, s. 497–500.

Příloha A

Obsah CD

- Zdrojové kódy aplikací
- Programová dokumentace (`doxygen`)
- Manuál k aplikacím
- Testovací sada anotovaných snímků
- Výsledky aplikace
- Video demonstrace
- Technická zpráva
- Zdrojové texty technické zprávy v $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$
- Plakát pro prezentaci práce
- Soutěžní článek EEICT [20]

Příloha B

Dosažené výsledky



Obrázek B.1: Výsledky lokalizační aplikace.



Obrázek B.2: Výsledky lokalizační aplikace.