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

DEKÓDOVANIE ČIAROVÉHO KÓDU V OBRAZE V REÁLNOM ČASE

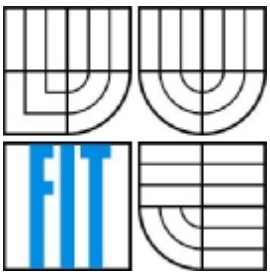
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN KRUPA

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

DEKÓDOVÁNÍ ČÁROVÉHO KÓDU V OBRAZE V REÁLNÉM ČASE

DECODING BARCODE IN IMAGE IN REAL TIME

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN KRUPA

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. Ing. ADAM HEROUT, Ph.D.

BRNO 2011

Abstrakt

Cílem práce je popsat čárové kódy používané běžně v praxi a vytvořit aplikaci, která bude za pomocí počítačového vidění zvolené kódy detekovat a rozpoznávat v reálnem čase. Nejdříve popisuje klasické, lineární čárové kódy, pak následuje popis dvourozměrných maticových kódů, jejich vznik, použití a struktura. Detailněji je rozebrán princip lokalizace, kódování a dekódování QR Kódu, s kterým se dále v práci pracuje. Taktéž je uveden stručný popis počítačového vidění a knihovny OpenCV. Následuje popis návrhu a implemenace aplikace na lokalizaci a rozpoznávání QR Kódů. Nakonec je popsané testování aplikace a jsou zhrnuty výsledky a možnosti pokračování práce.

Abstract

The goal of this thesis is to describe barcodes used in real world applications and to create application for real time detecting and decoding of chosen barcodes using computer vision. Normal linear barcode is first introduced, and then two dimensional matrix barcode is described with focus on its origin, usage and structure. Especially the process of localization, encoding and decoding of QR Code, which is chosen for future use in work, is described mainly. Also briefly description of computer vision and OpenCV library is available. Next the concept and implementation of application for localization and recognition of QR Codes is described. Finally description of testing, evaluation of thesis and ways of improvements are discussed.

Klíčová slova

čiarový kód, dvojrozmerný čiarový kód, QR Kód, počítačové videnie, OpenCV, spracovanie obrazu

Keywords

barcode, two dimensional barcode, QR Code, computer vision, OpenCV, image processing

Citace

Krupa Martin: Dekódovanie čiarového kódu v obraze v reálnom čase, diplomová práce, Brno, FIT VUT v Brně, 2011

Dekódovanie čiarového kódu v obraze v reálnom čase

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Doc. Ing. Adama Herouta, Ph.D.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Martin Krupa
24.5.2011

Poděkování

Děkuji vedoucímu práce Doc. Ing. Adamovi Heroutovi, Ph. D., za cenné rady, informace a podporu při vypracovaní této diplomové práce.

Obsah

1 Úvod

Klasický čiarový EAN kód je jednorozmerný kód, kde je informácia kódovaná pomocou hrubých a tenkých čiar oddelených medzerami. Jednorozmerný, alebo tiež lineárny sa nazýva preto, že kód je uložený a taktiež aj čítaný iba v jednom smere. Takéto čiarové kódy slúžia väčšinou na strojom jednoducho čitateľné a rozpoznateľné označovanie tovarov pomocou identifikačných čísel.

S čiarovými kódmi prichádzame denne do styku, či už vo forme označenia tovaru v obchodoch, alebo značenie a sledovanie zásielok na poštách pomocou čiarových kódov. Možno si to väčšina ľudí ani neuvedomuje, ale tieto čiarové kódy sú veľmi užitočné, pretože nám uľahčujú niektoré každodenné úkony, ako napríklad spomínané nakupovanie, kde podstatne prispievajú k skráteniu čakacej doby pri pokladniach. Keby totiž neboli výrobky označované číslami kódovanými prostredníctvom čiarových kódov, museli by byť všetky tieto čísla výrobkov zadávané ručne. Hlavným nedostatkom týchto jedno rozmerných čiarových kódov je schopnosť ukladať pomerne málo informácií pri zachovaní malých rozmerov.

Klasické čiarové kódy už na mnohé účely nepostačujú. Je to predovšetkým z dôvodu obmedzeného množstva informácie, ktorú sú schopné uchovávať. Väčšina týchto čiarových kódov totiž dokáže kódovať iba číslice a aj to v dosť obmedzenom počte. Preto vznikli čiarové kódy, ktoré neukladajú kódovanú informáciu iba v jednom smere, ale kód ukladajú do dvoch smerov. Takéto čiarové kódy sa z tohto dôvodu nazývajú dvojrozmerné čiarové kódy, alebo niekedy sa používa tiež označenie maticové čiarové kódy. Rozdiel medzi maticovými kódmi a lineárnymi je vidieť na obrázku 1.1. Tieto kódy umožňujú ukladať nielen číslice, ale aj text a špeciálne znaky. Dvojrozmerné kódy umožňujú kódovať dokonca aj japonské a iné ázijské znakové písma, preto sú v Japonsku veľmi rozšírené. Tieto čiarové kódy sa rýchlo rozvíjajú a existujú dokonca už aj dvojrozmerné farebné čiarové kódy, ktoré dokážu na relatívne malej ploche zakódovať obrázky, hudbu alebo dokonca krátke video.

Pôvodne boli dvojrozmerné kódy vyvinuté pre priemyselné aplikácie, kde bolo potrebné uložiť čo najviac informácii na malej ploche. Dnes predstavujú tieto kódy štandard v označovaní priemyselných výrobkov. Neskôr sa tieto kódy dostali aj na bežné výrobky, ako lieky, alebo elektronika. Okrem údaja základného identifikátora (čo je základným poslaním jednorozmerných kódov) obsahujú v sebe tieto kódy aj ďalšie informácie. Dvojrozmerný kód môže obsahovať napríklad inštrukcie pre zaobchádzanie s výrobkom, údaje o výrobkoch v balení, údaje povinne archivované (napr. identifikátory výrobného postupu), diagnózu pacienta, údaje na preukazoch (vrátane fotografie) a podobne. Dvojrozmerné čiarové kódy predstavujú aj zvýšenie bezpečnosti, pretože tým, že obsahujú viac informácií, umožňujú sledovanie jednotlivých výrobkov, alebo odhaľovanie nepravých výrobkov.

Veľkou výhodou maticových kódov je schopnosť chybovej korekcie. Niektoré tieto kódy totiž v sebe obsahujú zakódovaný samo opravný reťazec. Pomocou tohto reťazca je možné zakódovanú informáciu dekódovať aj z poškodeného alebo nekompletného kódu.

V poslednej dobe došlo k veľkému rozšíreniu dvojrozmerných čiarových kódov prostredníctvom mobilných telefónov. Tie totiž umožňujú zaznamenať čiarový kód pomocou kamery a ten následne veľmi rýchlo rozpoznať pomocou programov priamo na mobilom telefóne. Z mobilného telefónu sa tak stáva prenosná čítačka čiarových kódov. Tento systém je používaný prevažne na kódovanie internetových stránok, posielanie emailov, kontaktov, zdieľanie informácií o zariadeniach. Čiarový kód obsahujúci adresu internetovej stránky sa môže vyskytovať napríklad v novinách, časopisoch, plagátoch alebo hocikde inde a užívateľ disponujúci telefónom s kamerou

a príslušnou aplikáciou tak môže odfotením a rozpoznaním tohto kódu rýchlo otvoriť stránku priamo v prehliadači mobilného telefónu.

Obrázok 1.1: Jednorozmerný EAN čiarový kód a dvojrozmerný QR kód

Cieľom tejto diplomovej práce je vytvoriť aplikáciu, ktorá bude v reálnom čase rozpoznávať dvojrozmerné čiarové kódy. Detekcia a rozpoznanie bude prebiehať prostredníctvom počítačového videnia, ktoré sa zaoberá spracovávaním obrazu, a extrahovaním informácií z obrazu. Konkrétne použijeme knižnicu OpenCV, ktorá bola vytvorená práve pre riešenie problémov spojených s počítačovým videním. Zameriame sa najprv na stručný popis klasických a taktiež dvojrozmerných čiarových kódov. Potom bude nasledovať spôsob detekcie a rozpoznania niektorých kódov a taktiež nástrojov vhodných na strojové spracovanie dvojrozmerných kódov. Venovať sa budeme predovšetkým najpoužívanejšiemu z nich, ktorým je QR kód. Potom nasleduje popis návrhu a implementácie vytvoreného programu na rozpoznávanie čiarových kódov. Na koniec je popísané testovanie aplikácie, výsledky a možnosti pokračovania v práci.

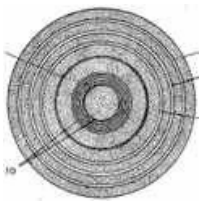
2 Čiarové kódy

Čiarový kód patrí k najrozšírenejšia metódam automatickej identifikácie. Existuje viacero typov čiarových kódov. Rozdeľujú sa do dvoch hlavných kategórií a to jednorozmerné a dvojrozmerné čiarové kódy [7]. Najprv boli vyvinuté jednorozmerné čiarové kódy pre identifikáciu výrobkov a tovaru. Neskôr ale, hlavne z dôvodu obmedzenia týchto kódov z hľadiska veľkosti zakódovanej informácie, vznikli dvojrozmerné čiarové kódy. Táto kapitola obsahuje stručný popis klasických, jedno rozmerných čiarových kódov, ako aj dvojrozmerných kódov. Je zameraná hlavne na najrozšírenejšie a najpoužívanejšie kódy, na ich vznik a použitie a predovšetkým na popis ich štruktúry a formátu.

2.1 Jednorozmerný čiarový kód

Jednorozmerný čiarový kód je štandardný kód používaný na označovanie väčšiny produktov. Ide o sériu rovnobežných čiar s rôznou hrúbkou, navzájom oddelených medzerami. Názov jednorozmerný vznikol preto, že pri tomto type kódu je informácia kódovaná iba v jednom smere, u väčšiny jednorozmerných kódov zľava doprava kolmo na čiary kódu. Existuje veľké množstvo jednorozmerných čiarových kódov, preto sa zameriam iba na tie najpoužívanejšie, viac v [7].

Prvá zmienka o čiarových kódoch je z roku 1948, kedy Bernard Silver, študent Drexelovho Inštitútu technológie vo Filadelfii a Norman Joseph Woodland, začali vytvárať prvý systém čiarových kódov. Podnet na vytvorenie takéhoto systému prišiel od prezidenta lokálnej sieti obchodov s jedlom Food Fair, ktorý potreboval vyvinúť systém pre automatizované čítanie informácií o produkte na pokladni. Prvý systém, ktorý vytvorili bol založený na vzore z atramentu, ktorý žiaril pod ultrafialovým svetlom. Vytvorili systém, ktorý fungoval, ale mal problémy s nestabilitou atramentu a taktiež tlač takýmto atramentom bola príliš drahá pre nasadenie v praxi. Neskôr, v roku 1949, sa im podarilo vytvoriť čiarový kód bull's eye, na obrázku 2.1, založený na kruhovej štruktúre. Bola to séria sústredených kruhov rôznej šírky, ktoré sa čítali zo stredu kruhu. Tento princíp striedania čiar s rôznou hrúbkou zostal v podstate zachovaný, až do dnes, jediným rozdielom je, že čiary nie sú uložené do kruhu, ale v jednej rade.



Obrázok 2.1: Historicky prvý čiarový kód bull's eye z roku 1949

Ďalším, kto výrazne prispel k rozvoju čiarových kódov bol David Collins, ktorý ihneď po skončení štúdia na MIT začal s vývojom systému pre automatickú identifikáciu vlakových vozňov. V roku 1959 vynašiel systém, ktorého základom boli modré a žlté reflexné pruhy na bokoch vozňov, v ktorých bolo zakódované šesťmiestne identifikačné číslo prepravnej spoločnosti a štvormiestne číslo identifikujúce vozeň. Tento systém však nebol veľmi úspešný a nebol nasadený v praxi príliš dlho.

Skupina veľkých obchodných potravinárskych spoločností definovala číselný formát kódu UPC (Universal Product Code). Viacero firiem, medzi nimi aj IBM, sa podieľalo na vzniku a vývoji

tohto kódu. Kód vychádzal z pôvodného bull's eye kódu od Silvera a Woodlanda. Prvý krát bol skener čiarového kódu UPC použitý v roku 1974 v supermarkete v Ohio, USA. Nasnímanie výrobku týmto skenerom sa považuje za prvé použitie klasických čiarových kódov na výrobkoch, ktoré sa používa odvtedy až do dnes. Identifikovanie výrobkov čiarovými kódmi sa po tomto úspechu začalo veľmi rýchlo rozširovať, a čoskoro bolo rozšírené do celého sveta.

Čiarové kódy čoskoro prenikli aj do priemyselných oblastí. V roku 1981 Americké Ministerstvo Obrany začalo používať čiarový kód Code 39 na označovanie a identifikáciu všetkých produktov predaných armáde. Tento systém je stále používaný Americkým Ministerstvom Obrany, a jeho zavedenie podnietilo prudké rozšírenie používania čiarových kódov vo všetkých priemyselných odvetviach.

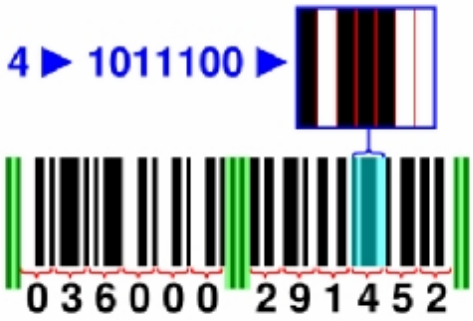
Medzi dnes najpoužívanejšie jednorozmerné čiarové kódy patria UPC (Universal Product Code), ktorý má viac verzií ale najviac používanou je UPC-A a druhým najpoužívanejším je EAN (European Article Number). Existuje však veľké množstvo ďalších typov čiarových kódov, viac v [7]. Jednotlivé typy sa od seba líšia množstvom kódovaných informácií, veľkosťou a spôsobom kódovania. K ďalším jednorozmerným čiarovým kódom patria napríklad Kód ITF, ktorý umožňuje vysokú hustotu zápisu, až 8 znakov na 1cm dĺžky kódu a je využívaný predovšetkým v priemysle. Ďalej je to Kód 39, ktorý je štandardom v automobilovom priemysle, zdravotníctve a v obrane. Okrem číslic dokáže zakódovať aj písmená a má veľmi veľkú odolnosť voči chybám. CODABAR je kód, ktorý je medzinárodne využívaný na označovanie krvných bánk v transfúznych staniciach. Okrem číslic môže kódovať šesť špeciálnych znakov. Prehľad týchto kódov je znázornený na obrázku 2.2, kde je aj vidieť rozdiely medzi týmito kódmi pri zakódovaní rovnakého čísla.

Obrázok 2.2: Prehľad najpoužívanejších jednorozmerných kódov

UPC kód dokáže kódovať 12 číslic na ktorých reprezentáciu potrebuje celkovo 95 bitov, kde každé číslo je kódované siedmimi bitmi a ostatné bity sú vyhradené pre špeciále značky určené formátom UPC kódu. Špeciálne značky označujúce začiatok a koniec kódu, a taktiež značky označujúce stred kódu, sú dlhšie ako ostatné, pre ich jednoduchšiu identifikáciu. Princíp kódovania UPC kódu je znázornený na obrázku 2.3. UPC kód je navrhnutý tak aby zaberal čo najmenej miesta a aby nemal viac ako štyri sekvencie bielych a čiernych bitov nasledujúcich po sebe. UPC kód umožňuje kódovať iba číslice. Obsahuje kontrolné bity, ktoré odhalia poškodený, alebo zle načítaný kód. Každé číslo je kódované postupnosťou dvoch čiernych a dvoch bielych čiar. Celkový počet čiar UPC kódu je presne 30. Pre každú číslicu definuje UPC dve špeciálne sekvencie bitov, podľa toho či sa číslice nachádzajú na pravej alebo ľavej strane kódu. Samotná kódovaná informácia je zložená z identifikačného čísla a z prefixu, ktorý určuje druh výrobku a kontrolného súčtu. Variáciou UPC

kódu je UPC-E kód, ktorý kóduje iba 6 číslic a používa sa tam kde nie je potrebných všetkých 12 čísiel. Bol vytvorený pre použitie tam kde z dôvodu veľkosti nie je možné umiestniť klasický UPC kód.

Digit	L Pattern	R Pattern
0	0001101	1110010
1	0011001	1100110
2	0010011	1101100
3	0111101	1000010
4	0100011	1011100
5	0110001	1001110
6	0101111	1010000
7	0111011	1000100
8	0110111	1001000
9	0001011	1110100



Obrázok 2.3: Princíp kódovania UPC a EAN kódu

K najpoužívanejším čiarovým kódom patrí aj kód EAN. Jedná sa o najznámejší kód pre tovar predávaný v obchodnej sieti. Tento kód môže užívať každý štát zapojený do medzinárodného združenia IANA EAN. Čiarový kód EAN dokáže kódovať číslice 0 až 9 pričom každá číslica je kódovaná 2 čiarami a 2 medzerami. Môže obsahovať buď 8 číslic (EAN-8) alebo trinásť číslic (EAN-13). EAN-13 kód je lineárny čiarový kód a predstavuje štandard v označovaní výrobkov čiarovými kódmi. Ide vlastne iba o rozšírenie UPC kódu, kedy je do neho pridaná trinásta číslica. Zmenený bol z dôvodu, že UPC kód nebol schopný dostatočne pokryť medzinárodné označovanie produktov. Princíp kódovania je rovnaký ako pri UPC kóde a je zobrazený na obrázku 2.3. EAN kód, ktorý začína nulou je totožný s odpovedajúcim UPC kódom. Preto identifikátory krajiny pôvodu začínajúce nulou sú priradené USA, aby bola zaistená kompatibilita s UPC kódmi. Kód je zložený z identifikačného čísla výrobku a identifikátora pôvodu výrobku. Prvé dve alebo 3 číslice vždy určujú štát pôvodu, ďalšie číslice, väčšinou 4 až 6, určujú výrobcu a nasledujúce číslice okrem poslednej určujú konkrétny tovar. Posledná číslica je kontrolná, tá overuje správnosť dekódovania.

Na skenovanie čiarových kódov sa používajú špeciálne optické skenery, ktoré sú schopné kód prečítať a rozpoznať v reálnom čase. Fungujú na princípe osvetlenia kódu svetelným lúčom a následne zachytia odraz svetla, alebo zachytávania svetla prirodzene vyžarovaného čiarovým kódom. Takto dokážu zachytiť digitálny obrázok čiarového kódu a previesť ho do elektronickej podoby. Existujú tri typy zariadení na snímanie čiarových kódov [7]. Prvým typom je dotykové pero, kde proces skenovania prebieha pretiahnutím pera cez celú oblasť čiarového kódu kolmo na jednotlivé čiary kódu. Nevýhodou tohto spôsobu je že ide o dotykovú technológiu, čo obmedzuje jej dosah a taktiež nie je vhodná pre použitie na nerovných povrchoch. Druhým typom senzoru je aktívny, bezkontaktný snímač. Na nasnímanie kódu používa usmernený lúč svetla, laser. Väčšinou ide o laser, ktorý je odrazený systémom zrkadiel, alebo pohyblivý lúč pre zachytenie celej oblasti čiarového kódu. Tento typ skenerov je bežne používaný v obchodoch, či už ako stacionárne skenery zabudované v pokladniach, alebo ručne ovládané skenery. Ich hlavnou výhodou je, že umožňujú prečítať čiarový kód aj na väčšiu vzdialenosť. Posledným typom sú pasívne bezkontaktné senzory. Tieto senzory používajú kameru na zachytenie obrázka čiarového kódu. Niekedy sú nazývane aj CCD skenery, pretože používajú CCD jednotku na zachytenie obrazu. Rozdiel oproti predchádzajúcim je, že tieto skenery nemajú žiadny zdroj svetla, a teda nemerajú svetlo odrazené od čiarového kódu. Sú to síce bezkontaktné snímače, ale ich dosah je pomerne malý v porovnaní s laserovými snímačmi.

Dôvodov pre používanie čiarových kódov je mnoho. Čiarový kód má veľa výhod a predností. Jednou z nich je napríklad presnosť. Používanie čiarových kódov je jedna z najpresnejších a

najrýchlejších metód k registrácii väčšieho množstva dát. Pri ručnom zadávaní dát dochádza k chybe v pomerne často, pri použití čiarového kódu sa počet chýb znižuje minimum, pričom väčšina chýb môže byť eliminovaná, ak je do kódu zakomponovaný kontrolný znak, ktorý overuje správnosť čítania všetkých ostatných znakov. Dôležitá je taktiež rýchlosť zadávania, ktorá je v porovnaní s manuálnym zadávaním čiarového kódu oveľa pomalší ako akýkoľvek snímač. Technológia čiarových kódov je mnohoúčelová, spoľahlivá a má jednoduché používanie. Čiarové kódy sa môžu používať v najrôznejších a extrémnych prostrediach a terénoch. Je možné ich tlačiť na materiály odolné vysokým teplotám alebo naopak extrémnym mrazom, na materiály odolné kyselinám, organickým rozpúšťadlám, oderu, nadmernej vlhkosti a pod. Ich rozmery môžu byť dokonca prispôsobené tak, aby sa mohli používať i na miniatúrne elektronické súčiastky.

Čiarové kódy sa stali všadeprítomným prvokom nášho sveta. Stretávame sa s nimi každodenne. Čiarové kódy majú najrozmanitejšie možnosti využitia. Skoro každý výrobok zakúpený v obchode, má na sebe vytlačený EAN čiarový kód. Taktiež umožňujú sledovanie tovaru od výrobcu, cez distribútora, až po koncového spotrebiteľa. Ďalšou aplikáciou čiarových kódov je použitie v doručovacích službách, kde sú používané na identifikáciu a sledovanie zásielok. Zásielka je pri prevzatí kuriérom označená etiketou vytlačenou na prenosnej tlačiarni, ktorá je pripojená k ručnému terminálu a súčasne zapísaná do evidencie prijatých zásielok. Odpadá tak zložitá manipulácia s papiermi. Používané sú aj na evidenciu majetku. V tomto prípade je každý objekt hmotného majetku označený etiketou s čiarovým kódom a načítaný do systému. Zdravotníctvo taktiež prináša mnoho možností na využitie čiarových kódov. Napríklad náramky pre pacientov, označovanie liekov, označovanie laboratórnych vzoriek a krvných preparátov. Farmaceutické spoločnosti môžu lepšie kontrolovať a lokalizovať každú dávku lieku, ktorá bola podaná a dokáže sa zamedziť aj používaniu falošných liečiv. Bezpečnosť pacientov môže byť značne zvýšená využitím čiarových kódov a to tým spôsobom, že lekári identifikujú pacienta pred podaním lieku, odobratím krvi, pri operačnom zákroku alebo pri prevoze pacienta. Čiarové kódy môžu byť použité aj na netradičné účely. Zaujímavé je napríklad použitie čiarových kódov na sledovanie včiel. Na každú včelu bol pripevnený čiarový kód veľmi malých rozmerov. Takto označené včely boli identifikované pri opustení, alebo pri návrate do úľa optickým snímačom s veľmi vysokým rozlíšením, pretože čiarové kódy boli veľmi malé.

2.2 Dvojrozmerný čiarový kód

Dvojrozmerné, alebo tiež maticové čiarové kódy majú zakódovanú informáciu uložené v dvoch smeroch, teda horizontálnom aj vertikálnom. Takéto usporiadanie umožňuje uložiť niekoľkonásobne väčšie množstvo informácie, ako klasické, lineárne čiarové kódy, ale zároveň je tento kód náročnejší na spracovanie. Pôvodne boli dvojrozmerné kódy vyvinuté pre priemyselné aplikácie, kde bolo potrebné uložiť čo najviac informácii na malej ploche. Dnes predstavujú tieto kódy štandard v označovaní priemyselných výrobkov. Neskôr sa tieto kódy dostali aj na bežné výrobky, ako lieky, alebo elektronika. V súčasnosti existuje veľké množstvo dvojrozmerných čiarových kódov s rozličnou štruktúrou a spôsobom kódovania.

Jednorozmerné čiarové, kódy síce predstavovali účinný spôsob identifikácie produktov, stále viac ich užívateľov, však vyžadovalo kódovanie väčšieho množstva informácií. Chceli predovšetkým, aby čiarový kód neposkytoval len nejakú identifikáciu do databáze ďalších informácií o produkte, ale aby sám čiarový kód obsahoval všetky tieto informácie o produkte. Bolo potrebné, aby čiarové kódy umožňovali kódovať viac znakov, teda číslice všetky znaky abecedy a aj špeciálne znaky. Prvé pokusy o zvýšenie informácie ktorú nesie kód boli v zväčšení kódovej oblasti alebo skladaniu

viacerých čiarových kódov. To však neviedlo k úspešnému riešeniu, pretože hlavný problém bol v zväčšovaní plochy čiarového kódu a taktiež sa objavili problémy s čitateľnosťou.

Prvé náznaky dvojrozmerného čiarového kódu boli zaznamenané v roku 1984. Nešlo úplne o dvojrozmerný kód, ako ho poznáme dnes, ale o naskladanie viacerých lineárnych kódov na seba. Spoločnosť Automotive Industry Action Group publikovala normu pre prepravné a identifikačé označovanie súčiastok. To pozostávalo s štyroch na seba naskladaných lineárnych kódov typu Code 39. Tento prvý dvojrozmerný čiarový kód obsahoval číslo súčiastky, ich množstvo, dodávateľa a sériové číslo.

Skutočný dvojrozmerný čiarový kód sa však objavil v roku 1988. Spoločnosťou Intermec Corporation bol vtedy predstavený nový spôsob označovania a identifikácie produktov, Kód 49. Ide o viacriadkový čiarový kód, ktorý mohol zakódovať všetky znaky ASCII tabuľky a mohol kódovať 49 alfanumerických znakov, alebo 81 číslic. Krátko po vzniku tohto kódu bolo vynájdených šesť ďalších dvojrozmerných kódov a začali sa rýchlo rozširovať, hlavne pre ich schopnosť uložiť malú prenosnú databázu na veľmi malom priestore.

Spočiatku boli dvojrozmerné čiarové kódy určené k použitiu na miestach, kde bolo dostupné iba veľmi málo miesta. Tieto kódy používané predovšetkým v zdravotníckom a elektrotechnickom priemysle. Tu sa totiž často používali malé produkty, ktoré mali príliš málo miesta na umiestnenie klasického lineárneho kódu. Postupne sa dvojrozmerné čiarové kódy rozšírili aj do oblastí, kde veľkosť kódu nebola až tak obmedzujúcim faktorom. Záujem bol predovšetkým o schopnosť uchovávať čo najviac informácií. Účinne boli takéto kódy nasadené napríklad na poštových zásielkach, kde obsahovali okrem identifikačného čísla aj celú adresu a meno. To umožňovalo v kombinácií s rýchlym prečítaním týchto informácií zrýchlenie identifikácie a času spracovania poštových zásielok. Praktické využitie týchto dvojrozmerných kódov je aj v oblastiach, kde nie je možné zabezpečiť on-line spojenie. Informácie v kóde sú totiž nezávislé od fungovania hlavného informačného systému. Dvojrozmerný kód môže teda obsahovať napríklad inštrukcie pre zaobchádzanie s objektom, údaje o výrobkoch, identifikátory výrobného postupu, diagnózu pacienta, údaje na preukazoch, vrátane fotografie a podobne. V poslednom čase sa rozmohlo používanie čiarových kódov na komerčné účely. Je to hlavne z dôvodu možnosti dekódovania týchto kódov na mobilných telefónoch. V časopisoch, novinách, na plagátoch alebo iných reklamných plochách tak môže byť čiarový kód obsahujúci internetovú adresu, kontaktné informácie, alebo akékoľvek ďalšie údaje, ktoré je možné okamžite využiť. Takto sa dá čiarový kód použiť napríklad v múzeách, kde je možné s ich pomocou rýchlo získať popisy jednotlivých exponátov, alebo vo forme vizitky, z ktorej je po odfotení možné získať kontaktné informácie a previesť ich rovno do adresára mobilného telefónu. Tieto kódy je možné použiť skoro všade tam, kde je potrebné zdeliť nejakú informáciu. Existujú už dokonca aj kódy schopné zakódovať multimediálny obsah, ako napríklad obrázok, hudbu alebo krátke video.

Obrovskou výhodou maticových kódov je aj ich schopnosť chybovej korekcie, čiže možnosťou správneho dekódovania aj v prípade čiastočne poškodeného, alebo nekompletného kódu. Do zakódovaných údajov je vložený kód na opravu chyby, pomocou, ktorého je možné dekódovať aj nekompletné dáta. Táto schopnosť opravy chyby je odlišná medzi jednotlivými typmi čiarových kódov a závisí od použitého spôsobu kódovania. Aj keď táto schopnosť obnovy čiarových kódov je pôvodné určená k prirodzenému poškodeniu, veľa používateľov ju používa na označovanie čiarových kódov vlastným logom. Časť čiarového kódu môže byť totiž vynechaná a namiesto nej umiestnené logo firmy, bez vplyvu na dekódovanie pôvodnej informácie.

Princíp skenovania dvojrozmerných čiarových kódov je rovnaký ako v prípade jednorozmerných. Z dôvodu ich zložitejšej štruktúry sú najviac rozšírené optické senzory využívajúce CCD technológiu. K obrovskému rozšíreniu prispelo umožnenie snímania dvojrozmerných kódov pomocou mobilných telefónov. Takto sa môže každý mobilný telefón vybavený fotoaparátom

a príslušným softvérom zmeniť na prenosnú čítačku kódov. Existujú voľne dostupné programy na mobilné telefóny, ktoré dokážu rozpoznávať maticové kódy rôznych formátov.

Dvojrozmerných čiarových kódov existuje veľké množstvo, viac v [7]. Tieto kódy sa navzájom líšia hlavne svojou štruktúrou, spôsobom kódovania a schopnosťou korekcie chyby. Nasleduje popis niektorých najpoužívanejších z nich. Podrobnejšie sa zameriam predovšetkým na QR kód, s ktorým budem ďalej pracovať.

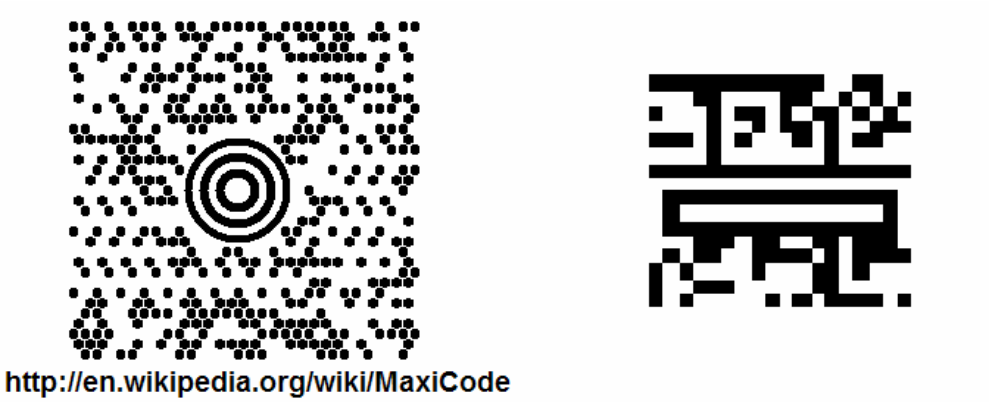
Aztec kód bol vytvorený s dôrazom na čo najjednoduchšiu tlač a jednoduché dekódovanie. Má štvorcový tvar s štvorcovými symbolmi a zameriavacím symbolom v strede kódu. Jeho rozmery sa pohybujú od 15x15 štvorcov do 151x151 štvorcov. Najmenší kód dokáže zakódovať 13 číslic, alebo 12 znakov abecedy, kým najväčší 3832 číslic, 3067 abecedných znakov, alebo 1914 bytov dát. Hodnoty 0 až 127 sú interpretované ako ASCII znaky a hodnoty 128 až 255 sú interpretované ako znaky abecedy v kódovaní ISO 8859-1. Okolo kódu nie je potrebné žiadne prázdne miesto. Poskytuje voliteľnú hranicu opravy kódu, od 5% do 95% celkovej plochy kódu, pričom odporúčaná hodnota je 23%. Jednou jeho variantou je Small Aztec Code, ktorý je zmenšenou verziou Aztec kódu a kóduje informácie maximálnej dĺžky 95 znakov, alebo 512 bitov. Miesto šetrí aj odstránením jedného štvorca z vyhľadávacieho vzoru. Aztec kód sa používa väčšinou v doprave napríklad na označovanie cestovných lístkov zakúpených on-line a vytlačených zákazníkmi, alebo na identifikáciu elektronických cestovných lístkov v leteckej doprave. Príklad Aztec kódu na obrázku 2.4

Obrázok 2.4: Príklad Aztec kódu a mini Aztec kódu

Ďalším kódom, rozšíreným hlavne v doprave, pri identifikácií osôb a na poštách, je kód PDF 417. Ide o kód s veľmi vysokou informačnou hustotou a taktiež dobrou schopnosťou detekcie a opravy chýb pri poškodenom kóde. Každé kódové slovo pozostáva z štyroch čiar a štyroch medzier so šírkou minimálne jedného a maximálne šiestich bodov. Celkovo je bodov v slove vždy presne 17. PDF 417 dokáže okrem bežného textu aj obrázky, alebo špeciálne programovacie inštrukcie. Maximálny limit kódovaných dát je 1,1 kB. To umožňuje zakódovať celú databázu údajov, čím sa kód stáva nezávislý na systéme. Používaný je pri identifikačných kartách, vodičských preukazoch, alebo na zakódovanie diagnózy pacientov. Princíp použitého kódovanie umožňuje bezchybne dekódovať kód, ktorý je až z 50% poškodený. Tento kód má aj svoju obmedzenú verziu, ktorou je Micro PDF 417. Ten je schopný kódovať 250 alfanumerických znakov, 366 číslic, alebo dáta do veľkosti 150 bytov. Príklad kódu PDF 417 na obrázku 2.5.

Obrázok 2.5: Príklad PDF417 kódu

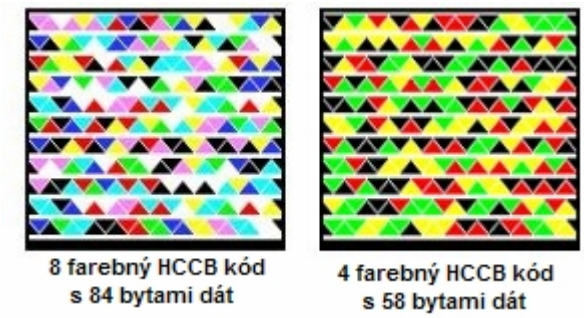
MaxiCode, znázornený na obrázku 2.6, je kódovací systém vytvorený logistickou firmou UPS na označovanie prepravovaných zásielok. Kódové pole je rozdelené do šesťuholníkovej mriežky s maximálnou rozlohou jeden štvorcový palec (6,45 štvorcových centimetrov). V strede je vyhľadávací vzor vo forme sústredených kružníc. Použitie tejto štruktúry namiesto klasickej štvorcovej mriežky zlepšuje hustotu kódu a umožňuje zakódovať o 15% viac dát. Schopnosť obnovy chyby dosahuje 24%. Dokáže zakódovať približne 100 ASCII znakov, a jeho štruktúra umožňuje spájanie viacerých kódov do jedného. Z dôvodu použitia neštvorcovej mriežky je tento kód ťažšie rozpoznateľný, a taktiež musí byť vytlačený na tlačiarňach s vyšším rozlíšením.



Obrázok 2.6: Príklad MaxiCode a Code 1 kódu

Code 1, na obrázku 2.6, používa vyhľadávací vzor horizontálnych a vertikálnych stĺpcov krížiacich sa v strede kódu. Symbol zvláda kódovať ASCII symboly alebo binárne dáta. Existuje 8 veľkostí v rozsahu od Code 1A, ktorý kóduje 13 alfanumerických znakov, alebo 22 číslic, až Code 1H, ktorý zaznamená 2218 alfanumerických znakov alebo 3550 číslic. Samotný kód môže mať viac tvarov. Code 1 sa v súčasnosti používa v lekárstve na označovanie liečiv, alebo napríklad v triedení druhotných surovín pre označovanie kontajnerov.

High Capacity Color Barcode (HCCB) je príklad farebného čiarového kódu, ktorý vytvorila spoločnosť Microsoft. Namiesto štvorcových bodov, ktoré používa väčšina maticových čiarových kódov, HCCB používa farebné trojuholníky usporiadané v riadkoch. Normálne sú trojuholníky čierne, alebo biele, ale hustota dát môže byť zvýšená použitím štyroch, prípadne šiestich farieb. Týmto spôsobom kódovania je možné dosiahnuť pri použití najvyššieho rozlíšenia kódu a ôsmich farieb hustotu až 2000 bytov binárnych dát, alebo 3500 abecedných znakov na štvorcový palec (6,45 štvorcových centimetrov). Pretože dokáže kódovať veľké množstvo dát, môže byť použitý pri elektronickom šifrovaní a overovaní pravosti. Do kódu môže byť zakódovaný digitálny podpis, alebo kľúč a ten následne použitý na overenie pravosti. Príklad farebného HCCB kódu je na obrázku 2.7.



Obrázok 2.7: Dve farebné verzie kódu High Capacity Color Barcode

Kód Datamatrix, na obrázku 2.8, patrí tiež k maticovým kódom, ktoré sa snažia o uloženie čo najväčšieho množstva informácií na veľmi malom priestore. Patrí k jedným z najpoužívanejších

dvojrozmerných kódov. Umožňuje uložiť 2335 alfanumerických znakov. Informácie sú kódované absolútnou pozíciou prvkov, nie relatívnou, preto nie je taký citlivý na tlačové chyby ako ostatné čiarové kódy a umožňuje prečítať kód aj keď je časť kódu odtrhnutá. Má štvorcový, prípadne obdĺžnikový tvar a každý symbol Datamatrix má dva susedné okraje vytlačené ako plné čiary a ostatné dva okraje ako séria rovnako sa opakujúcich štvorcových bodov. Tieto vzory sa používajú ako referenčné pri určení polohy kódu a na určenie hustoty. Datamatrix kód je najčastejšie používaný na označovaní malých súčiastok v priemysle, ako sú napríklad integrované obvody. Stále častejšie sa sním ale môžeme stretnúť aj na bežných výrobkoch v obchodoch. Taktiež je používaný skoro vo všetkých spomínaných oblastiach využiteľnosti maticových čiarových kódov.

Obrázok 2.8: Príklad Datamatrix kódu

3 QR Kód

Quick Response (QR) je dvojrozmerný čiarový kód ktorý vyvinula japonská spoločnosť Denso-Wave v roku 1994. V roku 2000 bol schválený ako medzinárodný štandard. Skratka „QR“ pochádza z anglického označenia Quick Response (Rýchla reakcia), keďže kód je navrhnutý s ohľadom na možnosť rýchleho dekódovania. Pôvodne slúžil na identifikáciu súčiastok pri výrobe automobilov. Dnes je jeho použitie oveľa rozšírenejšie. Slúži predovšetkým na identifikáciu a sledovanie produktov. Veľkou oblasťou jeho použitia sú aplikácie zamerané na mobilné telefóny. Na použitie QR Kódu sa nevzťahuje žiada licencia. Je definovaný a publikovaný spoločnosťou DENSO-WAWE, ktorá vlastní patentové práva na tento kód. QR Kód je však otvorený štandard, pretože spoločnosť DENSO-WAVE zverejnila špecifikáciu a neuplatňuje patentovú ochranu. Príklad QR Kódu je na obrázku 3.1.

Obrázok 3.1: Príklad QR Kódu

Veľkosť kódu	21 x 21 – 177 x 177 bodov (zväčšenie o 4 body na každú verziu)	
Typ a maximálne množstvo kódovaných informácií	Číslice	Max. 7089 znakov
	Alfanumerické znaky	Max. 4296 znakov
	Binárne dáta (8 bitov)	Max. 2953 bytov
	Kanji	Max. 1817 znakov
Obnova dát (koľko kódových slov môže byť obnovených)	Level L	Približne 7% kódových slov
	Level M	Približne 15% kódových slov
	Level Q	Približne 25% kódových slov
	Level H	Približne 30% kódových slov

Tabuľka 3.1: Špecifikácia QR Kódu

Medzi základné vlastnosti QR Kódu patrí jeho vysoká kapacita kódovaných dát. Dokáže uchovávať niekoľkonásobne viac dát ako lineárne čiarové kódy. Je prispôsobený na kódovanie všetkých typov dát, numerické, alfanumerické, symboly Japonskej abecedy (Kanji symboly), a binárne dáta. Tieto znaky je možné ľubovoľne kombinovať. V jednom symbole môže byť zakódovaných až 7089 znakov viď tabuľka 3.1. QR Kód je spomedzi všetkých dvojrozmerných čiarových kódov najvhodnejší pre Japonské znaky, pretože ich kóduje veľmi efektívne do 13 bitov, čo umožňuje kódovať o viac ako 20% viac znakov ako ostatné kódy. Vďaka tejto vlastnosti sú QR kódy veľmi rozšírene najmä v ázijských krajinách používajúcich znakové písmo.

QR kód má dobrú schopnosť obnovovať chýbajúce, alebo poškodené dáta. Dáta môžu byť dekódované aj v prípade čiastočne poškodeného alebo inak nečitateľného kódu. Na výber sú 4 rôzne úrovne chybovej korekcie, z ktorých má užívateľ na výber. S každou ďalšou úrovňou sa zvyšuje schopnosť opravy chyby, ale taktiež sa zväčšuje celková veľkosť QR Kódu. Zvolená úroveň závisí od viacerých faktorov, ako napríklad pravdepodobnosť poškodenia kódu, alebo nároky na jeho veľkosť. Zvyčajne to býva úroveň M. Prehľad jednotlivých úrovní je v tabuľke 3.1. Na opravovanie chýb sa používa metóda Reed-Solomon, viac v [1]. Miera obnovy dát závisí od množstva dát. Napríklad pre 100 kódových slov, z ktorých chceme 50 obnoviť, je potrebných 100 Reed-Solomon kódových slov, pretože táto metóda potrebuje vždy dvojnásobok slov, ako je schopná obnoviť.

Verzia čiarového kódu určuje predovšetkým jeho veľkosť. Dostupné sú od Verzie 1 do Verzie 40. Každá má odlišnú konfiguráciu modulov alebo počet modulov. Modul je označenie pre biele a čierne body QR Kódu. Najmenšia verzia 1 má rozmery 21 x 21 modulov a najväčšia má rozmery 177 x 177 modulov. Každá ďalšia verzia má o 4 moduly viac. Maximálna kapacita jednotlivých verzií je odvodená od verzie kódu, typu a množstva dát a úrovne opravného kódu.

QR Kód má štvorcový tvar. Celá jeho oblasť je rozdelená do pravidelnej mriežky. Obsahuje okrem samotnej kódovej oblasti aj časti, ktoré sú definované formátom QR Kódu a sú nemenné. Ide o informácie o formáte a verzii kódu, ďalej sú to pozičné a zarovnávacie vzory. Okolo kódu musí vyť voľné miesto o šírke minimálne štyri body. Samotná kódovaná oblasť pozostáva zo striedajúcich sa bielych a čiernych bodov, viď obrázok 3.3.

3.1 Modely QR kódu

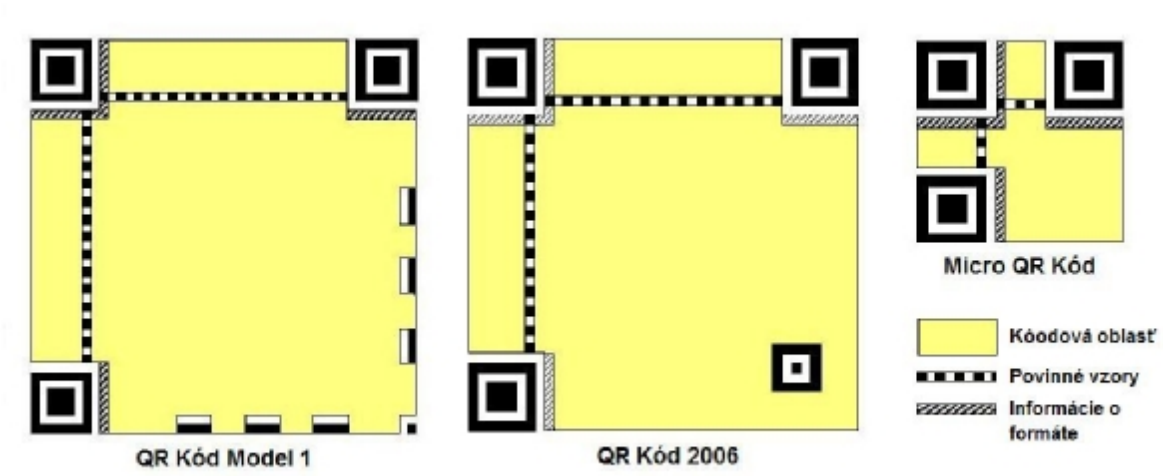
Spoločnosť DENSO-WAWE definovala spolu štyri modely QR kódu [9]. Sú to QR Kóde Model 1, Model 2, QR Kód 2006 a Micro QR Kód. V súčasnosti sa výhradne používajú iba modely QR Kód 2006 a Micro QR Kód. Rozdiely medzi jednotlivými modelmi QR Kódu sú znázornené na obrázku 3.2.

Prvý, pôvodne vytvorený model bol QR Kód Model 1, ktorý je špecifikovaný medzinárodným štandardom ISO/IEC 18004:2000. Tento kód bol neskôr vylepšený o niektoré špecifické funkcia a to hlavne o pridanie zarovnávacieho vzoru pre jednoduchšiu a presnejšiu lokalizáciu kódu v obraze. Kód bol označený ako QR Kód Model 2 a je popísaný štandardom SO/IEC 18004:2000.

Najnovšou verziou QR Kódu je QR Kód 2006, bol revidovaný v roku 2006. Ide o kód, ktorý je veľmi podobný ako QR Kód model 2, iba má pridané niektoré špecifické rysy, ako je podpora inverzného označovanie kódu, to znamená biele znaky na čiernom pozadí, zrkadlovo otočeného kódu a podpora viacerých znakových sád. Tento model QR kódu popisuje medzinárodný štandard ISO/IEC 18004:2006 [1].

Posledným modelom QR kódu je Micro QR Kód. Ide o redukovaný QR Kód 2006 kde je obmedzený počet vyhľadávacích vzorov z troch na jeden a taktiež počet kódovaných znakov je zmenšený. Jeho hlavnou výhodou sú veľmi malé rozmery a rýchle rozpoznanie. Vlastnosti a formát Micro QR Kódu sú popísané rovnakým štandardom ako v prípade QR Kódu 2006.

QR Kódy Modelu 2 a Kód 2006 sú navzájom kompatibilné, čo znamená, že je možný prevod medzi týmito kódmi a čítacie zariadenia pre QR Kódy 2006 sú schopné prečítať aj QR Kódy Modelu 2. Kódy vytvorené podľa normy pre QR Kód Model 1 nemusia byť kompatibilné s kódmi Modelu 2, ani kódmi Modelu 2006. Kompatibilita Micro QR Kódu s ostatnými modelmi nie je definovaná štandardom [1], pretože tieto kódy sú väčšinou používané samostatne v aplikáciách, kde nie je možné použiť QR Kód 2006, predovšetkým z dôvodu obmedzenej veľkosti. Micro QR Kódy teda nemusia byť kompatibilné s kódmi Modelu 1, Modelu 2 ani Modelu 2006.

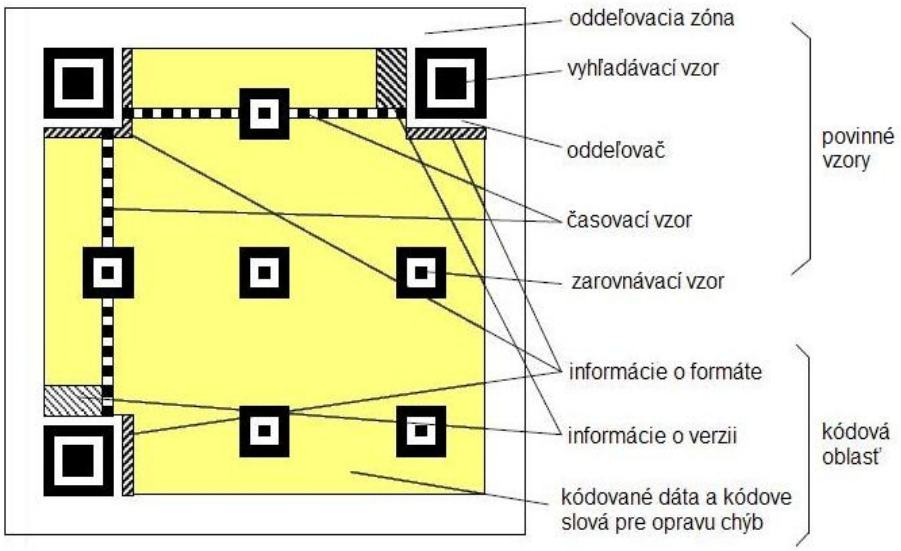


Obrázok 3.2: Rozdiely medzi modelmi QR Kódu

Kód v tomto dokumente označovaný ako QR Kód zodpovedá QR Kódu 2006. Ak sa jedná o iný model je to vždy uvedené.

3.2 Štruktúra QR Kódu

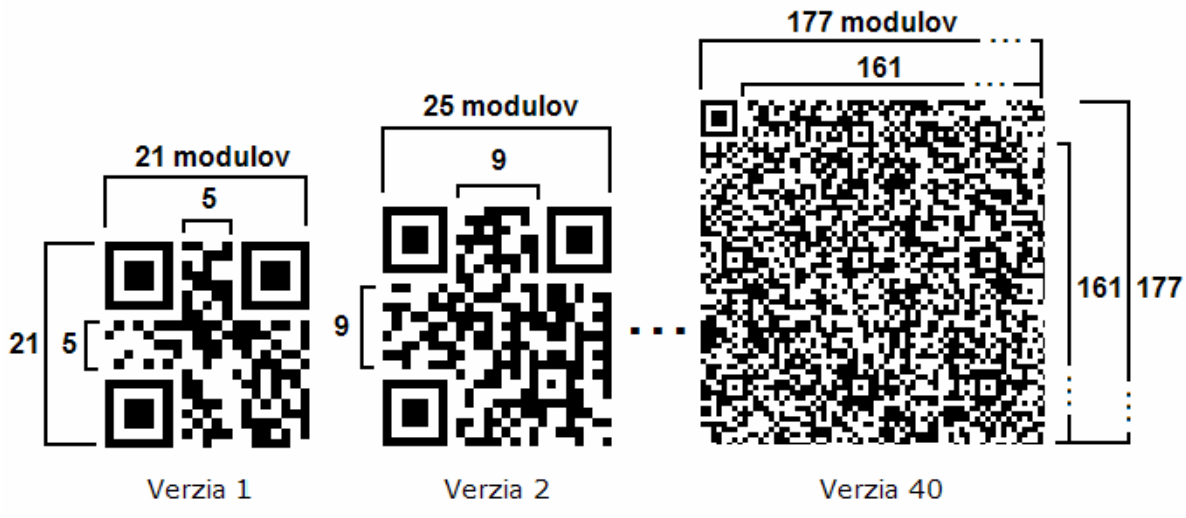
QR Kód má štvorcový tvar. Celá jeho oblasť je rozdelená do pravidelnej mriežky. Tá obsahuje okrem samotnej kódovej oblasti aj časti, ktoré sú definované formátom QR Kódu a sú nemenné. Ide o informácie o formáte a verzii kódu, ďalej sú to pozičné a zarovnávacie vzory, konkrétne vyhľadávací vzor, zarovnávací vzor, časovací vzor a oddeľovače. Okolo kódu musí byť oddeľovacia zóna, čo je voľné miesto o šírke minimálne štyri moduly z každej strany kódu. Kompletná štruktúra QR Kódu je znázornená na obrázku 3.3.



Obrázok 3.3: Štruktúra QR Kódu

3.2.1 Verzie a veľkosti QR Kódu

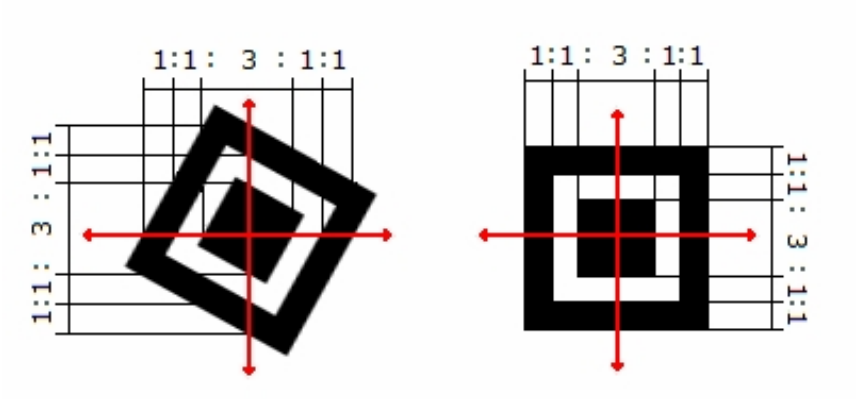
Je dostupných 40 verzií QR Kódov od Verzie 1 do Verzie 40. Verzia čiarového kódu určuje predovšetkým jeho veľkosť. Každá má odlišnú konfiguráciu modulov a čím je verzia vyššia, tým obsahuje viac modulov. Modul je označenie pre jednotlivé biele a čierne body QR Kódu. Najmenšia verzia, Verzia 1 má rozmery 21 x 21 modulov, Verzia 2 má rozmery 25 x 25 modulov, a tak ďalej, pričom každá ďalšia verzia obsahuje o štyri moduly na stranu viac, takže najväčšia Verzia 40 obsahuje 177 x 177 modulov. Maximálna kapacita jednotlivých verzií je odvodená od verzie kódu, typu a množstva dát a úrovne opravného kódu. Postupnosť zväčšovania jednotlivých verzií od Verzie 1 až po verziu 40 je znázornená na obrázku 3.4.



Obrázok 3.4: Verzie a veľkosť QR Kódu

3.2.2 Vyhľadávacie vzory

QR Kód obsahuje celkovo tri vyhľadávacie vzory umiestnené v ľavom hornom, pravom hornom a ľavom dolnom rohu kódu. Sú štvorcového tvaru a sú to tri sústredné štvorce preložené cez seba. Najväčší z nich má rozmery 7 x 7 modulov a je čiernej farby. V ňom je umiestnený menší, o rozmeroch 5 x 5 modulov bielej farby, v ktorom je ďalej štvorec o rozmeroch 3 x 3 modulov čiernej farby. To zaručuje, že pri prechode týmto vyhľadávacím vzorom akýmkoľvek smerom je vždy zachovaný rovnaký pomer čiernych a bielych bodov, viď obrázok 3.5. Tento pomer je teda pri vyhľadávacom vzore 1:1:3:1:1. Keďže sú čitateľné z každej strany a v kódovej oblasti by sa nemal vyskytovať podobný vzor, slúžia na detekciu QR Kódu v obraze. Taktiež ich detekcia umožňuje určiť orientáciu QR Kódu, uhol natočenia a prípadné zrkadlové obrátenie.



Obrázok 3.5: Invariantnosť pomeru bodov voči rotácií vzoru

3.2.3 Zarovnávacie vzory

Zarovnávacie vzory sú podobného formátu ako vyhľadávacie vzory. Sú štvorcového tvaru a sú to dva sústredné štvorce preložené cez seba. Väčší z nich má rozmery 5 x 5 modulov a je čiernej farby. V ňom je umiestnený menší, o rozmeroch 3 x 3 modulov bielej farby, ktorý má v strede jeden modul čiernej farby. Slúžia na zistenie prípadného skosenia čiarového kódu, jednoduchšiu detekciu čiarového kódu v obraze a správnu detekciu jednotlivých bitov kódu pri väčších kódoch. Taktiež sa používa na perspektívnu transformáciu obrazu pri zle zosnímanom obrázku. QR Kód Verzie 1 neobsahuje zarovnávací vzor, vyššie verzie obsahujú vždy aspoň jeden vyhľadávací vzor a ich počet závisí od verzie kódu. Pri prechode cez vyhľadávací vzor je tiež zachovaný rovnaký pomer čiernych a bielych bodov v každom smere, a tento pomer je 1:1:1:1:1.

3.2.4 Časovací reťazec a oddeľovacia zóna

Oddeľovacia zóna obklopuje QR Kód z každej strany a jej šírka je minimálne 4 moduly bielej farby. Jej účelom je zvýrazniť QR Kód v obrázku a tým uľahčuje vyhľadávania kódu v obraze. Podobnú úlohu majú oddeľovače, ktoré majú šírku jeden modul bielej farby a oddeľujú vyhľadávacie vzory a kódovú oblasť.

Vnútorné rohy vyhľadávacích vzorov sú spojené postupnosťou, pravidelne sa striedajúcich čiernych a bielych bodov o veľkosti jeden modul, nazývanou časovací vzor. Ten slúži predovšetkým na určenie veľkosti a verzie QR Kódu, a ďalej na spresnenie určenia pozície jednotlivých bitov kódu v skosenom, otočenom, alebo inak zdeformovanom obraze. V kóde sú dva časovacie reťazce jeden na šiestom riadku kódu, postupujúci rovnobežne s vrchnou hranou kódu a spájajúci dva horné vyhľadávacie reťazce. Druhý sa nachádza v šiestom stĺpci kódu, postupuje rovnobežne s ľavou hranou kódu a spája ľavý horný a pravý dolný vyhľadávací vzor.

3.2.5 Dátová oblasť

Poslednou časťou štruktúry QR Kódu je dátová oblasť. Je to oblasť kódu, ktorá zahrňuje cely kód okrem vyššie spomínaných oblastí. Obsahuje samotné kódované dáta, kódové slová na opravu chýb a informácie o verzii a formáte kódu. Informácie o verzii a formáte kódu, ich štruktúra a pozícia sú detailne popísané v kapitole 9 o dekódovaní QR Kódu. Kapacita dátovej oblasti sa líši v závislosti na verzii kódu. Podrobný prehľad dátovej kapacity jednotlivých verzií QR Kódov ako aj rozloženie modulov v tejto oblasti je možné nájsť v [1].

4 Princíp kódovania QR Kódu

K tomu aby sme boli schopní dekódovať dáta zakódované v QR Kóde je potrebné najprv pochopiť spôsob, akým sú tieto dáta v kóde zakódované. Preto v tejto kapitole nasleduje popis hlavných krokov vedúcich k úspešnému zakódovaniu dát do QR Kódu. Medzi tieto hlavné kroky kódovania dát do QR Kódu patrí: (prevzaté z [1])

1. Analýza vstupných dát
2. Zakódovanie vstupných dát do kódových slov
3. Vytvorenie kódových slov pre opravu chýb
4. Štruktúrovane dát do konečnej postupnosti
5. Uloženie dát do štruktúry kódu
6. Maskovanie dát
7. Pridanie informácie o formáte a verzii do štruktúry kódu

4.1 Analýza vstupných dát

Prvým krokom zakódovania dát do QR Kódu je analýza a vstupných dát. Dáta sa analyzujú z dôvodu výberu najvhodnejšieho formátu reprezentácie dát a ich prevodu do binárnej podoby. Nesprávne zvolený formát dát môže mať za následok zbytočné plytvanie miesta v QR Kóde. Snaha kódovania je previesť vstupné dáta do čo najkratšej bitovej postupnosti, a tú následne vložiť do štruktúru kódu. QR Kód poskytuje štyri formáty prevodu dát do bitovej postupnosti. Sú to nasledujúce:

- Numerický, predovšetkým pre ukladanie číslic. V tomto móde sú vždy 3 vstupné číslice kódované do postupnosti 10 bitov.
- Alfanumerický, pre kódovanie znakov abecedy. Vždy dva znaky sú kódované do postupnosti 11 bitov.
- 8-bitový dáta formát, pre reprezentáciu dát uložených do postupností bytov, každý o dĺžke 8 bitov.
- Posledným je Kanji mód pre kódovanie znakového písma. V tomto móde je každý znak zakódovaný do postupnosti 13 bitov.

Jednotlivé módy je možné medzi sebou kombinovať, pre dosiahnutie čo najefektívnejšieho kódovania s výsledkom čo najmenšej bitovej postupnosti pre vstupné dáta.

V tomto kroku sa podľa vstupných dát taktiež zvolí úroveň detekcia a opravy chýb. Tá je väčšinou daná užívateľom. Rovnako užívateľ zvolí aj požadovanú veľkosť a verziu kódu, ak nie je definovaná, je použitá najmenšia verzia QR Kódu, ktorá je schopná dáta zakódovať. Referencie k výberu úrovne opravy chýb a k určeniu verzie kódu dostupné v [1].

4.2 Zakódovanie vstupných dát

V tomto kroku ide o zakódovanie vstupných dát do kódových slov, kde každé kódové slovo sa skladá z postupnosti 8 bitov. Najprv je vytvorená tzv. hlavička, ktorá obsahuje označenie formátu dát a indikátor počtu zakódovaných znakov daným formátom. Bitová dĺžka označujúca počet znakov taktiež závisí od zvoleného formátu. Označenie formátu dát a dĺžka indikátoru počtu znakov sú nasledovné:

- Numerický formát, kód: 0001, indikátor počtu znakov: 10 bitov.
- Alfanumerický formát, kód: 0010, indikátor počtu znakov: 9 bitov.
- 8-bitový dáta formát, kód: 0100, indikátor počtu znakov: 8 bitov.
- Kanji formát, kód: 1000, indikátor počtu znakov: 8 bitov.

Potom nasleduje dátová postupnosť zakódovaných vstupných dát. Samotné kódovanie vstupných dát do binárnej postupnosti prebieha v prípadne numerických dát tak, že dáta sú rozdelená po 3 znakoch (čísliciach). Každá takáto skupina š číslic je zakódovaná do 10 bitov dlhej binárnej reprezentácie. Špeciálny prípad je, ak je počet oddelených dát na konci postupnosti menší ako 3. V tomto prípade ak máme iba jednu číslicu, tá je zakódovaný do 4-bitovej binárnej reprezentácie. Ak zostali iba 2 číslice, tie sú zakódované do 7-bitov dlhej postupnosti.

V alfanumerickom móde je postupnosť znakov rozdelená po dvoch znakoch a každý znak je prevedený na príslušnú hodnotu podľa tabuľky 4.1. Hodnota prvého znaku vo dvojici je vynásobená číslom 45 a hodnota druhého znaku je pripočítaná k tejto hodnote. Výsledok, vždy pre dvojicu čísel, je následne zakódovaný do 11-bitovej binárnej reprezentácie. Ak je na konci postupnosti znakov iba jeden znak, je v tabuľke vyhľadaná jeho hodnota, a tá je prevedená do 6-bitovej postupnosti.

hodnota	znak	hodnota	znak	hodnota	znak	hodnota	znak	hodnota	znak	hodnota	znak	hodnota	znak
0	0	7	7	14	E	21	L	28	S	35	Z	42	.
1	1	8	8	15	F	22	M	29	T	36	SP	43	/
2	2	9	9	16	G	23	N	30	U	37	$	44	:
3	3	10	A	17	H	24	O	31	V	38	%		
4	4	11	B	18	I	25	P	32	W	39	*		
5	5	12	C	19	J	26	Q	33	X	40	+		
6	6	13	D	20	K	27	R	34	Y	41	-		

Tabuľka 4.1: Kódovacia/Dekódovacia tabuľka pre alfanumerické znaky

Pri 8-bitovom dátovom móde každá 8-bitová postupnosť priamo reprezentuje binárnu hodnotu vstupných dát podľa ASCII tabuľky pre ISO/IEC 8859-1 znakovú sadu.

V Kanji móde, kde sa kódujú znaky znakovej abecedy, je každý znak prevedený na 13-bitovú binárnu reprezentáciu. Viac informácií o kódovaní Kanji znakov dostupných v [1].

Koniec kódovanej postupnosti je označený 4-bitovou postupnosťou znakov 0. Tento kód je pridaný na koniec, za všetky zakódované dáta. Nepridáva sa v prípade, že zakódovaná bitová postupnosť kompletne zaplní celú kódovú oblasť QR Kódu, alebo ak zbytková kapacita kódu je menšia ako 4 bity.

Posledným krokom je zakódovanie vytvorenej postupnosti bitov do kódových slov. Každé kódové slovo má dĺžku 8 bitov. Ak po zakódovaní ostane na konci menej ako 8 bitov je kódové slovo vytvorené doplnením 0 za posledný bit, až do veľkosti 8 bitov. Ak je počet takto vytvorených kódových slov menší ako je kapacita QR Kódu zvolenej verzie, je kapacita vyplnená náhodným pridávaním kódových slov 11101100 alebo 00010001, až do zaplnenia celej dátovej oblasti QR Kódu.

Kapacita zvolenej verzie QR Kódu pre zvolenú úroveň opravy chýb je určená podľa tabulky 4.2. Je uvedená iba relevantná časť tabuľky, pre verzie kódov s ktorými pracujeme. Kompletná tabuľka je k dispozícii v [1] .

Verzia QRKódu	Úroveň opravy chýb	Počet kódových slov	Počet bitov	Kapacita/Dátový formát		
				Numerický	Alfa-numerický	8-bitivy byte mód
1	L	19	152	41	25	17
	M	16	128	34	20	14
	Q	13	104	27	16	11
	H	9	72	17	10	7
2	L	34	272	77	47	32
	M	28	224	63	38	26
	Q	22	176	48	29	20
	H	16	128	34	20	14
3	L	55	440	127	77	53
	M	44	352	101	61	42
	Q	34	272	77	47	32
	H	26	208	58	35	24
4	L	80	640	187	114	78
	M	64	512	149	90	62
	Q	48	384	111	67	46
	H	36	288	82	50	34
5	L	108	255	255	154	106
	M	86	202	202	122	84
	Q	62	144	144	87	60
	H	46	106	106	64	44
6	L	136	1088	322	195	134
	M	108	864	255	154	106
	Q	76	608	178	108	74
	H	60	480	139	84	58

Tabuľka 4.2: Počty symbolov a dátová kapacita pre vybrané verzie QR Kódov

4.3 Vytvorenie kódových slov pre opravu chýb

QR Kód používa na detekciu a opravovanie chybných údajov v zakódovanej informácii opravný kód Reed-Solomon. Z dát získaných predchádzajúcimi krokmi sú vypočítané kódové slová na opravu chýb, ktoré sú následne pridané za tieto dáta a vložené spolu s nimi do štruktúry QR Kódu. To umožňuje dekódovať aj poškodené, alebo zle prečítané informácie, bez straty dát. K dispozícii sú 4 voliteľné úrovne opravy chýb. Každá z nich sa líši počtom potrebných kódových slov na úspešné obnovenie informácií a počtom poškodených dát, ktoré je schopná obnoviť z týchto kódových slov. Schopnosť obnovy sa pohybuje od 7% kódových slov pri najnižšej úrovni, až po 30% kódových slov pri najvyššej úrovni Reed-Solomonovho opravného kódu. Týmto spôsobom je možné obnoviť

chýbajúce bity, keby nebolo možné na niektorých pozíciách prečítať informácie. Taktiež dokáže opraviť chyby vzniknuté chybným prečítaním jednotlivých bitov v kóde.

Počet chýb ktoré je možné touto metódou opraviť je počítaný podľa vzťahu

$$e + 2 * t \leq d - p,$$

kde *e* je počet chýbajúcich bitov, *t* je počet chybných bitov, *d* je počet kódových slov na opravu chýb *p* je počet chybne dekódovaných kódových slov.

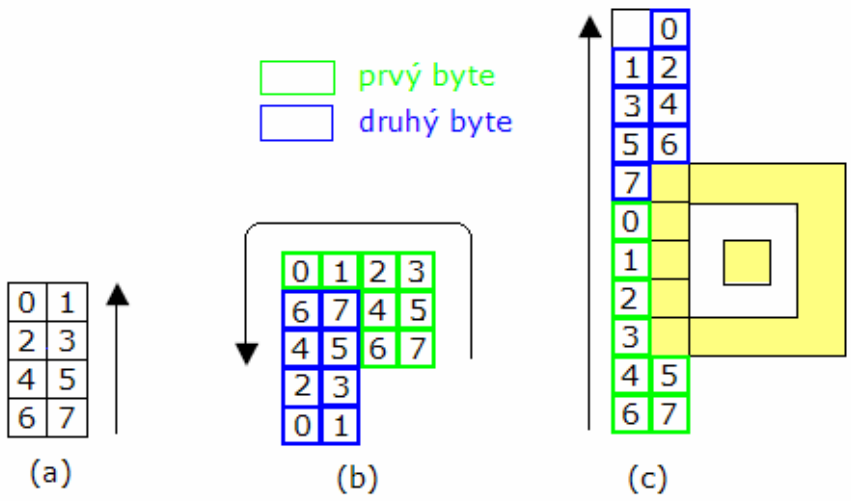
Podľa verzie a počtu kódových slov je niekedy vhodné rozdeliť zakódovaný vstup a príslušné kódové slová na opravu chýb do viacerých blokov, v ktorých prebieha oprava chýb samostatne, nezávisle na ostatných blokoch. Detailnejší postup procesu tvorby kódových slov je možné nájsť v uvedenej literatúre [1].

4.4 Štruktúrovane dát do konečnej postupnosti

V tejto časti kódovacieho procesu ide o rozdelenie zakódovaných dát, a k nim prislúchajúcim kódovým slovám na opravu chýb, do jednotlivých blokov. Počet takýchto blokov je pre každú verziu QR Kódu presne určený [1] . Dáta a príslušné slová opravy chýb sú navzájom prekladané podľa príslušnosti k jednotlivým blokom, pričom najprv sú zoradené všetky kódované dáta a až za nimi nasledujú dáta na opravu chýb. Tiež je kontrolované zaplnenie celej dátovej oblasti QR Kódu. V prípade že nie je zaplnená celá, je voľne miesto doplnené nulami.

4.5 Uloženie dát do štruktúry kódu

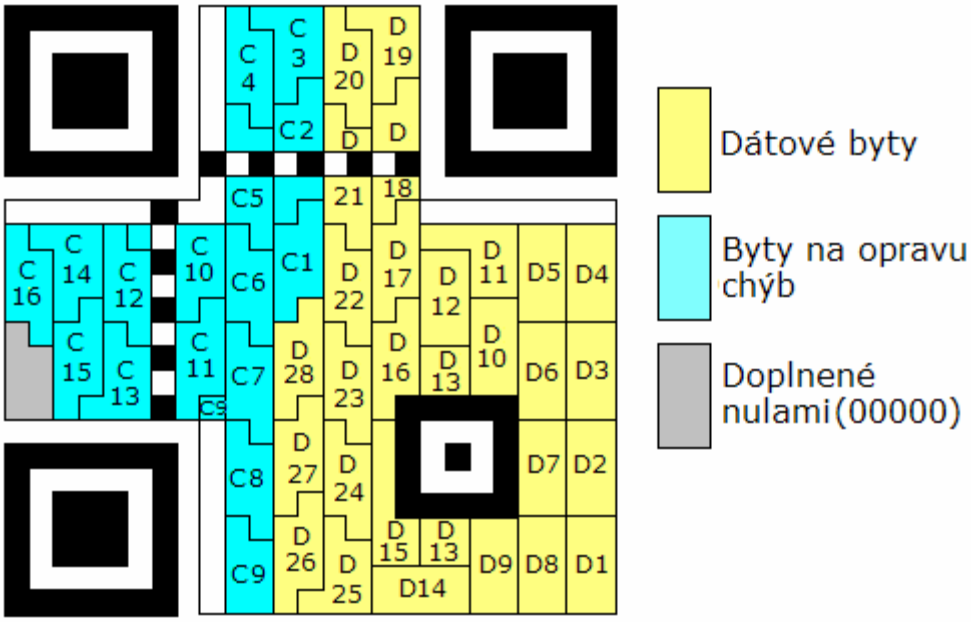
Ak máme výslednú bitovú postupnosť kódových slov a slov na opravu chýb je ich potrebné správne uložiť do štruktúry QR Kódu.



Obrázok 4.1: Ukladanie jednotlivých bitov a zmena smeru pri dosiahnutí miesta, na ktoré nie je možné bit vložiť

Dáta sa ukladajú do stĺpcov o šírke dva moduly. Počiatočný bod je v pravom dolnom rohu kódu. Postupuje sa po tomto stĺpci najprv smerom nahor, ako náhle sa dosiahne vrchný koniec kódu

prechádzaný stĺpec sa posunie o dva moduly doľava a zmení sa smer prechádzania smerom dole, obrázok 4.1b. Takto sa pokračuje až sa dosiahne ľavý koniec kódu. Pri postupnom prechádzaní každého stĺpca sa ukladajú jednotlivé bity kódového slova, vždy najprv na pravý modul príslušného stĺpca, ďalší bit na ľavý modul a nasleduje posun hore (prípadne dole), a zase sa začína pravým modulom a pokračuje ľavým, až do vyplnenia celej kódovej oblasti QR Kódu, obrázok 4.1a. Ak sa narazí na hranicu kódu, alebo na nemenné časti QR Kódu, tak sa tieto moduly ignorujú a pokračuje sa ďalej preskočením týchto modulov. Ak sa teda nachádzame v pravom module stĺpca a ľavý modul je voľný, tak vložíme bit na pravý a prejdeme na ľavý. Ak je ale v ľavý obsadený, vložíme bit na pravý a pokračujeme ďalej v danom smere. Ak sú obsadene obidva moduly, aj pravý aj ľavý, obidva sa preskočia a pokračuje sa v danom smere až na najbližší voľný bit, alebo pri dosiahnutí hranice kódu sa presunieme na ďalší stĺpec a pokračujeme rovnakým spôsobom opačným smerom, obrázok 4.1c. Stĺpec ktorý neobsahuje žiadny voľný bit, je v každom kóde práve jeden, a celý sa preskočí. Dáta sa ukladajú spôsobom, že sa postupuje od najvýznamnejšieho bitu kódového slova, ktorý je uložený na prvé voľné miesto. Na obrázkoch 4.1 a 4.2 je znázornený princíp ukladania jednotlivých bitov do modulov QR Kódu a konečné rozloženie kódových slov v kóde.



Obrázok 4.2: Rozloženie kódových a opravných slov v QR Kóde

4.6 Maskovanie dát

Aby bolo čítanie QR Kódu jednoduchšie a spoľahlivejšie, je potrebné aby boli biele a čierne moduly v kóde rovnomerne rozmiestnené. Je nežiaduce, aby boli tvorené veľké zhluky modulov rovnakej farby. Taktiež sa v dátovej oblasti nesmú nachádzať vzory podobné vyhľadávaciemu, alebo zarovnávaciemu vzoru. Preto sú bity uložené v kóde ďalej maskované. Maskovanie sa aplikuje iba na dátovú oblasť, na povinné a nemenne časti nie. Existuje 8 rôznych maskovacích vzorov, ktoré sú generované rovnicami z tabuľky 8.1 dosadením súradnice riadku a stĺpca počítaného modulu.

Konkrétny maskovací vzor sa vyberá podľa stanovených pravidiel popísaných v [1]. Detailnejší popis maskovania dátovej oblasti kódu je uvedený v kapitole 9, ktorá sa venuje dekódovaniu QR Kódu.

4.7 Pridanie informácie o formáte a verzii do štruktúry kódu

Posledným krokom pri vytváraní QR Kódu je vloženie informácie o formáte a verzii kódu do jeho štruktúry. Miesta pre uloženie týchto informácii sú presne stanovené, vid. obrázok 8.5.

Úroveň obnovy chýb	Binárny kód
L	01
M	00
Q	11
H	10

Tabuľka 4.3: Označenie chybovej korekcie pre formát QR Kódu

Informácia o formáte obsahuje označenie maskovacieho reťazca a úroveň chybovej korekcie v 15 bitovom reťazci. Tieto dve informácie sú spolu zakódované do 5 bitov, kde prvé dva bity označujú úroveň chybovej korekcie (tabuľka 4.3) a tri ďalšie maskovací vzor (tabuľka 8.1). Za nimi nasleduje 10 bitový kód pre opravu chýb, kódovaný podľa Bose-Chaudhuri-Hocquenghem (15, 5) algoritmu, viac o tomto algoritme a spôsobe kódovania v [1]. Týchto 15 bitov je tiež maskovaných a to exkluzívnym logickým súčtom s reťazcom 101010000010010. Presné umiestnenie informácie o formáte a jej vytvorenie je popísané v kapitole 9.

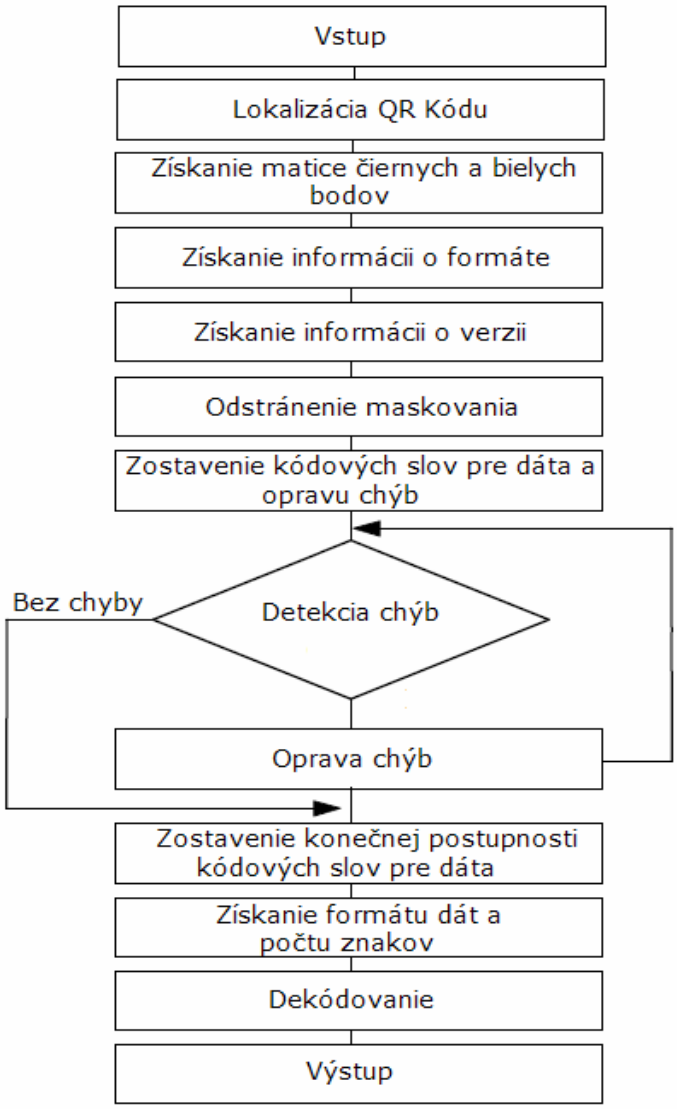
Informácia o verzii je prítomná až od siedmej verzie QR Kódu a obsahuje číslo verzie, od 7 do 40. Pozostáva z 18 bitov, kde je 6 dátových bitov a 12 bitov na opravu chýb, kódovanými algoritmom Bose-Chaudhuri-Hocquenghem (18, 6). Podobne ako informácia o formáte je uložená v kóde 2 krát, z dôvodu presnejšieho dekódovania. Informácia o verzii nie je maskovaná.

5 Dekódovanie QR Kódu

Proces dekódovania čiarového kódu je vlastne obrátený proces jeho zakódovania, kedy sa postupne postupuje od posledného kroku, až k prvému kroku kódovacieho procesu. Tým sa z matice bitov, získaných zo štruktúry QR Kódu postupne extrahujú všetky potrebné informácie a postupnosť získaných bitov sa dekóduje až na pôvodné vstupné dáta, čo je vlastne samotným výsledkom dekódovacieho procesu.

5.1 Hlavné kroky dekódovacieho procesu

Nasleduje popis hlavných krokov dekódovacieho procesu. Väčšina krokov je aplikovaná pri dekódovaní akéhokoľvek QR Kódu, avšak niektoré kroky sú aplikované iba pri dekódovaní určitých verzií QR Kódu, alebo rôznych modelov QR Kódu. Postup je zameraný hlavne na QR Kódy modelu 2006, viac o dekódovaní ostatných modelov [1].



Obrázok 5.1: Postup dekódovania QR Kódu

1. Získanie matice jednotlivých bitov QR Kódu. To zahrňuje lokalizáciu QR Kódu v obraze, následne rozpoznanie bielych a čiernych bitov a prevod do príslušnej kódovej matice znakov 0 pre biele moduly kódu, a 1 pre čierne moduly kódu.
2. Druhým krokom je získanie informácií o formáte. Tá sa získa z vytvorenej kódovej matice a aplikuje sa na ňu algoritmus na opravu chýb. Tým sa získa informácia o úrovni opravy chýb pre zakódované dáta a maskovací vzor, ktorý bol použitý na maskovania dátovej oblasti QR Kódu pri procese zakódovania.
3. Získanie informácii o verzii QR Kódu. Tento krok je aplikovaný iba pre QR Kódy Verzie 7 a vyššej.
4. Odstránenie maskovania dátovej oblasti kódu, aplikáciou maskovacieho vzoru získaného v kroku 2. Tým sa získajú pôvodné bity a je možné zostaviť jednotlivé kódové slová.
5. Prečítanie konečnej postupnosti bitov a vytvorenie príslušných kódových slov pre kódované dáta a taktiež kódové slová pre opravu chýb.
6. Kontrola chýb zo získaných kódových slov na úrovni získanej v kroku 2. Ak sú detegované chyby nasleduje ich oprava, až kým nie je konečná postupnosť bez chyby.
7. Získanie označenia formátu uloženia dát a počtu zakódovaných znakov. Rozdelenie získanej bitovej postupnosti na jednotlivé kódové slová, na základe získaného formátu dát.
8. Dekódovanie takto získanej bitovej postupnosti na jednotlivé znaky.

Na obrázku 5.1 je znázornená schéma dekódovacieho procesu. Detailnejší popis jednotlivých krokov dekódovania je dostupný v kapitole 9 venovanej detailnejšiemu popisu dekódovacieho procesu.

6 Počítačové videnie

Počítačové videnie je technologická disciplína, ktorá sa snaží počítačovými prostriedkami napodobniť ľudské videnie. Ide o súbor technologických postupov zaoberajúcich sa zachytávaním obrazu, jeho spracovaním , reprezentácie a získavaním informácií zo získaného obrazu. Predmetom spracovanie je obrazová informácia o reálnom svete. Počítačové videnie rieši úlohu vytvorenia popisu fyzických objektov v obraze. Moderné počítačové videnie dokáže pracovať s 2D aj 3D vstupnými dátami.

K typickým a najbežnejším úlohám počítačového videnia patrí rozpoznávanie. Častým problémom je práve určenie, či dané dáta obsahujú špecifický objekt alebo majú určitú vlastnosť. Aj keď je táto úloha človekom jednoducho riešiteľná, stále nie je algoritmicky spoľahlivo vyriešená pre zovšeobecnený prípad, teda ľubovoľný objekt v ľubovoľnom prostredí. Medzi základné problémy patrí rozpoznávanie, identifikácia a detekcia objektov. Pri rozpoznávaní objektov ide o vyhľadávanie jedného, prípadne viacerých vopred známych a špecifikovaných objektov v obraze. Identifikácia je rozpoznanie konkrétneho objektu, napríklad pri identifikácii osôb alebo vozidiel. Detekcia je vyhľadávanie skupín objektov, ktoré spĺňajú dopredu stanovené podmienky, alebo majú hľadané vlastnosti. Počítačové videnie je tiež často používané pri analýze pohybu. Ide o odhad smeru pohybu objektov na základe videa, prípadne určene smeru a pohybu pohľadu. Taktiež sem patrí sledovanie skupiny záujmových bodov, či už osôb, alebo vozidiel, v sekvencii obrázkov. K typickým úlohám pre počítačové videnie tiež patrí rekonštrukcia trojrozmernej scény z viacerých obrázkov, alebo obnovovanie poškodených obrázkov.

Počítačové videnie má uplatnenie v mnohých oblastiach. Je to napríklad ovládanie procesov v autonómnych vozidlách alebo priemyslových robotoch, sledovanie a detekcia rôznych javov alebo objektov. V poslednej dobe sa veľmi často používa aj na interakciu počítača s človekom. Jedno z najčastejších použití počítačového videnia je zapracovávanie obrazu v medicíne. Ide o získavanie informácií z obrazových dát za účelom stanovenia diagnózy pacienta. Druhou veľkou oblasťou použitia je priemysel, kde je často označované ako strojové videnie, kde sú informácie z kamier získavané za účelom podpory výrobného procesu. Ide napríklad o kontrolu kvality výrobku, alebo určovanie orientácie a polohy objektov. Jednou z najväčších oblastí využitia počítačového videnia je armádne použitie. Ide hlavne o detekciu nepriateľských objektov a navádzanie rakiet, alebo bezpilotných vozidiel a lietadiel. S počítačovým videním úzko súvisia aj ďalšie oblasti. Je to hlavne spracovávanie a analýza obrazu, pretože na získavanie informácie z obrazu je ho často potrebné vhodne upraviť. S počítačovým videním úzko súvisí aj spracovávanie obrazu, pretože obrazové dáta bývajú často vo forme signálov. Ďalšou oblasťou je aj umelá inteligencia, ktorej časť rozpoznávanie vzorov je veľmi dôležitá pre počítačové videnie.

Počítačové videnie všeobecne pozostáva z dvoch hlavných krokov. Prvým je získanie a načítanie obrazu jeho predspracovanie. Druhým krokom je spracovanie a analýza obrazu. Postup spracovania obrazu v počítačovom videní sa dá rozdeliť do piatich základných krokov.

1. Snímanie, digitalizácia a uloženie obrazu v počítači. V tomto kroku dochádza k digitalizovaniu vstupného obrazového signálu. Vstupnou informáciou môže byť napríklad informácia o jase z kamery alebo skeneru, intenzita žiarenia, tepelné žiarenie, prípadne iné fyzikálne veličiny, ako hĺbka, alebo odrazovosť zvukových alebo elektromagnetických vĺn.
2. Predspracovanie obrazu. Cieľom predspracovanie je hlavne potlačiť šum a chyby vzniknuté pri digitalizácií obrazu. Niekedy môže byť cieľom predspracovania aj zvýraznenie určitých rysov (hrany, záujmové body) obrazu podstatných pre ďalšie spracovanie.

3. Segmentácia obrazu. Je komplikovaná časť počítačového videnia, pretože ide o rozdelenie obrazu podľa výskytu hľadaných objektov. Ide v podstate o nájdenie tých častí objektov, ktoré sú významné z hľadiska ďalšieho spracovania.
4. Popis nájdených objektov. Objekty je možné popísať buď kvantitatívne pomocou množiny číselných charakteristík, alebo kvalitatívne pomocou vzťahov medzi objektmi. Spôsob popisu objektov závisí na ďalšom použití tohto popisu.
5. Porozumenie obsahu. V obecnom prípade predstavuje porozumenie interpretácie obrazových dát. Je založené na znalostiach, cieľoch a využití spätných väzieb medzi rôznymi úrovňami spracovania. Ide o napodobnenie procesu rozhodovania u človeka na základe informácie získanej z obrazu.

6.1 OpenCV

Open Source Computer Vision (OpenCV) je voľne dostupná knižnica na riešenie problémov počítačového videnia. Knižnica je distribuovaná pod licenciou BSD. Je zameraná predovšetkým na počítačové videnie a spracovávanie obrazu v reálnom čase. Táto knižnica ponúka cez 500 optimalizovaných algoritmov súvisiacich s počítačovým videním. Knižnica je pôvodne vyvinutá firmou Intel. Počiatky vzniku siahajú do roku 1999, ale prvá verzia OpenCV 1.0 bola vydaná v roku 2006. Aktuálna verzia knižnice je OpenCV 2.2 Je multiplatformová, teda použiteľná pod rôznymi operačnými systémami, ako Windows, Linux aj Mac OS. Knižnica je vytvorená pre programovací jazyk C/C++, ale je možné ju používať aj v ostatných ako C#, Java Python, alebo Perl. Bola vytvorená hlavne za účelom spraviť počítačové videnie univerzálne dostupné.

Knižnica je použiteľná pre takmer všetky oblasti počítačového videnia. Používa sa napríklad na detekciu osôb a objektov, rozpoznanie tváre, gest, na sledovanie objektov vo videosekvenciách, segmentáciu obrazu, alebo aj v robotike. Knižnica taktiež podporuje strojové učenie, čo ešte viac rozširuje jej univerzálnosť.

Štruktúra OpenCV pozostáva z nasledujúcich modulov:

CXCORE obsahuje definície dátových štruktúr, správu pamäti, obsluhu chýb, kreslenie, text a základnú matematiku.

CV (Computer Vision) slúži na spracovávanie obrazu, rozpoznávanie vzorov, detekciu pohyb, sledovanie objektov kalibráciu kamery.

ML (Machine Learning) obsahuje algoritmy na zhlukovanie, klasifikáciu a analýzu.

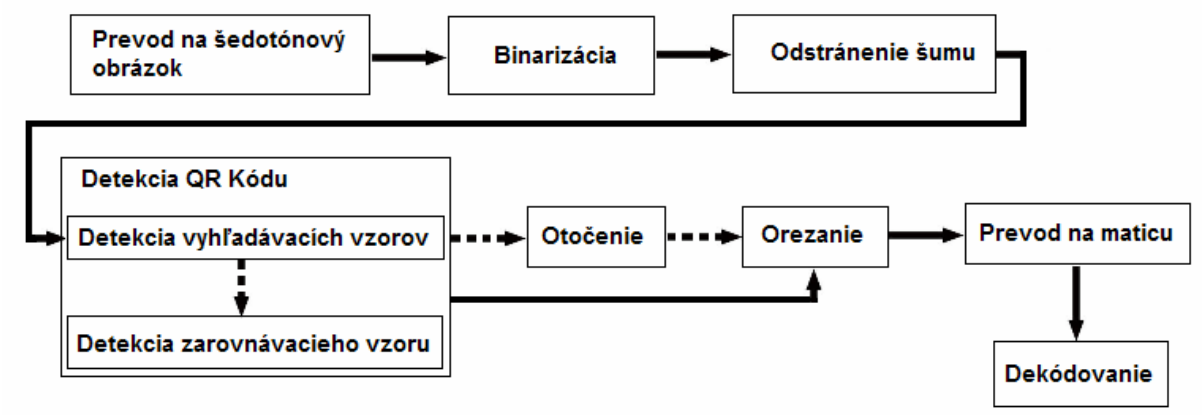
HighGUI obsahuje grafické užívateľské rozhranie prácu so vstupmi a výstupmi, ako aj prácu s obrázkami a videom

CVCAM je modul na spracovávanie a ovládanie kamery

7 Návrh riešenia

Cieľom je vytvoriť aplikáciu, ktorá bude schopná v reálnom čase detegovať a rozpoznať čiarový kód v obraze. Obrázok bude načítaný zo súboru z pevného disku. Zvolený typ čiarového kódu, ktorý bude aplikácia schopná rozpoznať je QR Kód 2006. Verzie, teda veľkosti s ktorými bude aplikácia pracovať sú od QR Kódu Verzie 1 po QR Kód Verzie 6. Obmedzenie je hlavne z dôvodu spracovania QR Kódu v reálnom čase.

Pri detekcii je dôležitá presnosť rozpoznania, ako aj rýchlosť. Aplikácia by mala zvládať rozpoznať aj čiarové kódy natočené, s nehomogénnym osvetlením, skosené, alebo obrázky nekvalite nasnímané, obsahujúce šum. Vyhľadávanie ale musí prebiehať v reálnom čase, preto je potrebné nájsť vhodný kompromis medzi rýchlosťou rozpoznávania a schopnosťou rozpoznávať znehodnotené obrázky kódov. Medzi hlavné kroky patrí predspracovanie obrazu, lokalizácia QR Kódu, prevod na binárnu maticu a samotné dekódovanie informácie zakódovanej v QR Kóde. Schéma návrhu je znázornená na obrázku 7.1.



Obrázok 7.1: Blokové schéma návrhu

7.1 Predspracovanie obrazu

Pred samotnou detekciou čiarového kódu v obrázku, je potrebné, kvôli presnejšej lokalizácii kódu, obrázok najprv upraviť. Najdôležitejšími krokmi je odstránenie šumu, vplyvov nerovnomerného osvetlenia a iné nežiaduce deformácie obrazu, ktoré by mohli znemožňovať detekciu a rozpoznane čiarového kódu. Keďže QR Kód patrí k čiernobielym čiarovým kódom, nie je potrebné aby bol zdrojový obrázok farebný. Obrázok sa preto najprv prevedie z farebného modelu na šedotónový obrázok obsahujúci iba jeden farebný kanál. Následne je obrázok potrebné binarizovať. Voľba správnej metódy binarizácie je veľmi dôležitá, pretože nevhodná metóda by mohla mať negatívny vplyv na detekciu a rozpoznanie kódu. Pre tieto účely je najvhodnejšia metóda prahovania [2]. Osvetlenie pri snímaní obrázku však nemusí byť rovnomerné, preto sa v obrázku môžu objaviť miesta s vyšším, respektíve nižším jasom. To by však pri metóde globálneho prahovania spôsobovalo problémy, pretože by mohlo dôjsť k znehodnoteniu častí čiarového kódu. Preto je obrázok prevedený do binárnej reprezentácie metódou lokálneho, adaptívneho prahovania. Táto metóda používa na výpočet hodnoty pre prahovanie okolie daného bodu a nie celý obrázok ako globálne prahovanie. Preto sa týmto spôsobom odstránia vplyvy nehomogénneho osvetlenia, ktoré vznikli pri snímaní QR Kódu.

Ďalej je z obrázku odstránený šum. To je realizované použitím vhodného filtra. Pre tento prípad vyhovuje napríklad Gaussov filter [2]. Vhodnou metódou pre odstránenie šumu môže byť aj podvzorkovanie obrázku a následne jeho prevzorkovanie. Ide v podstate o zmenšenie obrázku a potom jeho zväčšenie na pôvodný rozmer. To má za následok odstránenie šumu, ale samotné body obrazu reprezentujúce QR Kód v obraze zostanú neporušené.

7.2 Lokalizácia QR Kódu

Aby bolo možné QR kód rozpoznať je potrebné ho najprv v obraze lokalizovať. Práve táto časť úzko súvisí s počítačovým videním, ktorého rôzne funkcie implementuje knižnica OpenCV. Preto na lokalizáciu týchto znakov použijeme knižnicu. OpenCV knižnica taktiež obsahuje rôzne algoritmy a nástroje na transformáciu a úpravu obrazu, preto môže byť použitá aj vo fáze predspracovania obrazu.

Na detekciu vyhľadávacích prípadne aj zarovnávacích vzorov v obraze môžu byť použité dve rôzne techniky. Prvá je založená na vyhľadávaní týchto vzorov ako štvorcov v obraze, a následne kontrolovať, či ich tvar zodpovedá príslušnému vzoru, definovanému formátom QR Kódu. Táto metóda je vhodná pretože v kódovej oblasti QR Kódu sa iné vzory s rovnakou štruktúrou nevyskytujú vďaka maskovaniu aplikovanému na kódovú oblasť v procese kódovania [1]. Druhým možným prístupom pri vyhľadávaní kódu v obraze môže byť prechádzanie celého obrázku po jednotlivých riadkoch, prípadne stĺpcoch a vyhľadávať miesta ktoré by zodpovedali štruktúre vyhľadávacích reťazcov. Detegované musia byť všetky tri vyhľadávacie vzory, inak je vyhľadávanie neúspešné. Z lokalizácie všetkých troch vyhľadávacích vzorov je možné určiť približnú lokalitu vyhľadávacieho vzoru a ten potom presne lokalizovať prechádzaním obrázku rôznymi smermi z tejto približnej polohy vzoru. Následne je možné z pozície vyhľadávacích vzorov zistiť prípadne natočenie kódu a kód otočiť do pôvodnej pozície. Obrázok môže byť následne orezaný tak, aby v ňom zostal iba samotný QR Kód.

7.3 Prevod na maticu čísel

Upravený obrázok, ktorý už obsahuje iba samotný QR Kód je potrebné previesť do štvorcovej binárnej matice , ktorá bude reprezentovať jednotlivé moduly kódu. Vhodná reprezentácia je 0 pre biele moduly kódu a číslo 1 pre tmavé moduly kódu. Veľkosť matice sa rovná počtu modulov kódu a určí sa z verzie QR Kódu. Samotný prevod je možné realizovať preložením obrázku pravidelnou mriežkou, podľa veľkosti jedného modulu v QR Kóde, a prevedením do matice podľa hodnoty jednotlivých pixelov v mriežke. Najrýchlejším spôsobom prevodu je skontrolovať farbu iba jedného bodu, konkrétne v strede daného štvorca v obrázku. Takéto riešenie ale môže viesť k chybnému priradeniu hodnoty danému modulu z dôvodu šumu v obraze. Vhodnejšie je kontrolovať väčšiu oblasť každého štvorca mriežky. Kontrola celej mriežky by poskytovala najlepšie výsledky, ale keďže sa v tomto prípade kontroluje celá oblasť kódu, nie je táto metóda vhodná pre spracovanie v reálnom čase. Najlepším spôsobom je kontrolovať určitú oblasť v okolí stredu daného štvorca, kedy sa hodnoty jednotlivých bodov akumulujú a podľa výslednej hodnoty sa určí, či je modul biely, alebo svetlý. Takto sa prejde celá oblasť QR Kódu , až sú všetky moduly kódu prevedené na príslušné body v matici. Z maticou sa ďalej pracuje podľa schémy na obrázku 5.1 kde sa z matice extrahujú informácie o verzii a formáte kódu a jednotlivé bity sú prevedené na kódové slová a prebehne samotné dekódovanie.

8 Implementácia

Pred samotnou implementáciou je najskôr potrebné vybrať vhodné vývojové prostredie pre vytvorenie aplikácie. Dôraz pri výbere bol kladený najmä na podporu rôznych knižníc, predovšetkým knižnice pre počítačové videnie OpenCV, podporu tvorby aplikácií pracujúcich v reálnom čase, a objektovo orientovaného programovania.

Ďalej je potrebné naštudovať techniky spojené s počítačovým videním, aby pomocou neho bolo možné QR Kód v obraze správne lokalizovať. Taktiež je potrebné zoznámiť sa s programovaním aplikácií počítačového videnia pomocou nástrojov knižnice OpenCV. Zistiť princíp, na akom nástroje tejto knižnice pracujú, preskúmať rôzne dostupné algoritmy a zoznámiť sa s dátovými štruktúrami používanými touto knižnicou. Tiež je pre programovanie dekodéru QR Kódov potrebné detailne preštudovať a pochopiť princíp kódovania dát do QR Kódu a tiež spôsob akým sa tieto dáta z kódu dekódujú. Problémom je ale slabá dostupnosť dokumentácie a podporných materiálov popisujúcich princíp kódovania a dekódovania QR Kódov.

Postup práce pri vytváraní aplikácie pozostáva z viacerých samostatných krokov, ktoré sú načítanie obrázku s QR Kódom, predspracovanie obrazu, lokalizácia QR Kódu, prevod na binárnu maticu a dekódovanie. Jednotlivé kroky zodpovedajú schéme z obrázku 5.1. Postup pri implementovaní samotného procesu dekódovania QR Kódu sa riadil podľa krokov popísaných v kapitole 5.

Pri vytváraní aplikácie pre dekódovanie QR Kódov je taktiež potrebné vytvoriť databázu testovacích snímok rôznych QR Kódov. Tá musí obsahovať kódy rôznych veľkostí a verzií. Musia tu byť obrázky nielen dobre čitateľné bez ďalšieho upravovania, ale aj snímky ktoré sú rôzne poškodené, napríklad šumom, vplyvmi zlého osvetlenia, alebo zlého nasnímania. Tiež musí obsahovať snímky neplatných kódov, teda kódov ktoré nie sú vytvorené v súlade s pravidlami QR Kódov, alebo sú zámerne poškodené, aby nebolo možné ich dekódovať.

8.1 Vývojové prostredie

Pre vývoj každej aplikácie je nevyhnutné použitie vhodného vývojového prostredia. V tomto prípade je aplikácia na dekódovanie QR Kódov implementovaná v prostredí Microsoft Windows za použitia Visual Studia 2008. Zvolený implementačný jazyk je C++, a to hlavne z dôvodu jeho objektovo orientovaného prístupu a širokej podpore rôznych knižníc. Hlavnou podmienkou pri výbere jazyka bolo možnosť použiť knižnicu pre programovanie aplikácií počítačového videnia OpenCV. Jazyk C++ je na použitie knižnice OpenCV najvhodnejší [4]. Jazyk C++ umožňuje používať všetky nástroje tejto knižnice. Taktiež je vhodný pre programovanie aplikácií prebiehajúcich v reálnom čase.

8.2 Predspracovanie obrazu

Predspracovanie obrazu zahrňuje operácie vykonané s vstupným obrazom ešte pred jeho samotným spracovávaním. Z dôvodu schopnosti aplikácie čo najlepšie lokalizovať QR Kód vo vstupnom obraze, je obraz ešte pred samotnou detekciou upravovaný. Väčšina úprav súvisí s prevodom do odtieňov šedej farby a binarizáciou obrazu, pretože QR Kód je čierno-biely a nepotrebuje farebnú reprezentáciu. Ďalším stupňov úprav patriacich do predspracovania je odstránenie šumu odrazu a odstránenie efektov nehomogénneho osvetlenia pri snímaní obrázku. Väčšina úprav je realizovaná

prostredníctvom funkcií knižnice OpenCV. Príklad výsledkov predspracovania je znázornený na obrázku 8.1.

Prvou úpravou je prevod farebného vstupného obrázku do šedo-tónového obrázku. Ide o prevod z farebného RGB modelu na jednokanálový 8 bitový obrázok, obsahujúci 256 odtieňov šedej farby. Obraz je prevedený na jednokanálový najmä z dôvodu nasledujúceho prevodu na binárny obraz. Binarizácia obrazu prebieha metódou prahovania. Z dôvodu lepších vlastností, najmä pri obraze s nehomogénnou intenzitou jasu v celom obrázku je použitá metóda adaptívneho prahovania. Tá dosahuje pri binarizácii lepšie výsledky, ako klasická metóda globálneho prahovania [3]. Pri obyčajnom prahovaní môže totiž dochádzať k určitej strate informácii v oblastiach s väčším, prípadne menším osvetlením. Táto metóda má používa prah globálny, rovnaký pre celý obrázok a prechádza postupne celý obraz a porovnáva hodnoty obrazu. Ak je hodnota väčšia ako prah, je nahradená maximálnou hodnotou, naopak, ak je hodnota menšia je nahradená nulou. Výsledkom je potom binárny obraz. Lokálne adaptívne prahovanie funguje na princípe použitia rôznych hodnôt prahu pre rôzne časti obrazu. Do úvahy sa berie stanovené okolie bodu, z ktorého sa vypočíta prah a podľa neho je daný bod nahradený maximálnou hodnotou, alebo nulou podobne ako pri globálnom prahovaní. To odstraňuje rozdiely v nehomogénnom osvetlení obrazu a výsledkom je binarizovaný pôvodný obrázok vhodný na ďalšie spracovanie. Prevod obrázku na čierno-biely, ako aj binarizácia sú realizované prostredníctvom funkcií knižnice OpenCV. Pri adaptívnom prahovaní je okolie bodu, ktoré sa berie v úvahu pri počítaní prahu 43 bodov obrazu na výšku aj šírku.

Vstupný obraz môže obsahovať šum, čo je nežiaduce pre ďalšie spracovanie. Šum je odstránený veľmi jednoduchou a rýchlou metódou a to podvzorkovanie obrazu, teda jeho zmenšenie, na polovičnú veľkosť, a následne jeho zväčšenie na pôvodný rozmer. Opäť sú použité funkcie knižnice OpenCV, ktoré používajú na aproximáciu pri zmenšovaní, respektíve zväčšovaní Gaussov filter a konvolučnú maticu veľkosti 5 x 5 bodov.

Obrázok 8.1: Pôvodný a binarizovaný obrázok

8.3 Lokalizácia čiarového kódu

Ďalším krokom je lokalizácia čiarového kódu v obraze. Tá pozostáva z vyhľadania troch vyhľadávacích vzorov, prípadne aj zarovnávacieho vzoru, v obraze. Tým sú lokalizované okraje QR Kódu a môžeme presne lokalizovať kódovú oblasť kódu.

8.3.1 Vyhľadávacie vzory

Najskôr sú detegované vyhľadávacie vzory. Ich počet musí byť presne tri. Ak je ich nájdených menej ako tri vyhľadávanie končí neúspešne. Väčšinou sa pre vyhľadávanie QR Kódu používa metóda riadkového prechádzania celého obrazu [1, 2]. Táto metóda ale musí po jednom bode prehľadávať celý obraz a kontrolovať jeho okolie, čo je časovo náročné. Rýchlejšou metódou, viď. výsledky testov v kapitole 10, je metóda založená na vyhľadávaní štvorcov v obraze a ich následná kontrola. Princíp vyhľadávania vyhľadávacích vzorov je založený na upravenom algoritme pre vyhľadávanie štvorcov v obraze z [4].

Táto metóda vyhľadávania štvorcov funguje na nasledujúcom princípe. Najprv sú v obraze zvýraznené hrany pomocou Cannyho hranového detektoru implementovaného v knižnici OpenCV. Následne, pomocou získaných hrán, sú v obraze vyhľadané obrysy štvorcových oblastí, ktoré musia byť obklopené vždy minimálne štyrmi hranami, ktoré zvierajú uhol približne 90°. Tieto oblasti sú následne filtrované pre odstránenie všetkých oblastí okrem vyhľadávacích vzorov QR kódu. Odstránené sú oblasti ktorých veľkosť je príliš malá alebo naopak moc veľká aby mohli byť vyhľadávacími reťazcami a takisto oblasti, ktorých štruktúra nezodpovedá vyhľadávacím reťazcom QR Kódu, obrázok 3.5. Všetky oblasti ktoré vyhovujú sú uložené a sú ďalej filtrované podľa pozície v obraze. Je vypočítaný stred každej oblasti. Štvorce s podobnými súradnicami stredu sa ukladajú do jedného poľa a následne sú vymazané štvorce, ktoré sú v obraze iba raz, prípadne majú menšie zastúpenie. Taktiež sa štvorce porovnávajú navzájom podľa veľkosti, pretože všetky tri vyhľadávacie vzory musia mať približne rovnakú veľkosť. Oblasti v ktorých je zhoda najvyššia, teda s najväčším počtom štvorcov predstavujú vyhľadávacie reťazce QR Kódu. To, že každý vyhľadávací vzor je zastúpený viacerými štvorcami je zabezpečené tým, že sa tento postup opakuje viac krát. Pri každej aplikácii tohto algoritmu je na obrázok aplikované prahovanie vždy s inou hodnotou prahu. Viacnásobným vyhľadávaním štvorcom pri rôznom prahu sa zvyšuje šanca na správnu lokalizáciu vyhľadávacieho vzoru v obraze.

Posledným krokom je určenie presnej polohy stredu vyhľadávacieho vzoru spriemerovaním stredov všetkých štvorcov, ktoré mu prislúchajú. Rovnakým spôsobom sú určené aj pozície rohov pre každý vyhľadávací vzor. Príklad úspešnej detekcie vyhľadávacích vzorov je znázornený na obrázku 2,5. Vyhľadávacie vzory v obrázku sú v dokumente ďalej označované nasledovne: ľavý horný je označený ako prvý, pravý horný ako druhý a ľavý dolný ako tretí.

Obrázok 8.2: Správne lokalizované vzory v obrázku

Následne sú všetky stredy štvorcov okolo každého vyhľadávacieho vzoru spriemerované, čím je nájdený presný stred vyhľadávacieho vzoru. Potom je zistená pozícia týchto vzorov navzájom. Z pozícií stredov horných dvoch štvorcov je zistené prípadne otočenie obrázku vzhľadom k horizontálnej osi a vypočítaný uhol otočenia. Ak je obrázok natočený o viac ako 2 stupne tak dôjde k jeho zarovnaniu vzhľadom v horizontálnej osi. V opačnom prípade je obrázok orezaný podľa pozícií vyhľadávacích reťazcov. Príklady úspešnej detekcie sú na obrázku 8.2.

8.3.2 Zarovnávacie vzory

Pre správne dekódovanie QR Kódu je nevyhnutné lokalizovať pozíciu zarovnávacieho vzoru. Ten sa nachádza v QR Kódoch od Verzie 2 a je umiestnený v pravom dolnom rohu kódu. Stred zarovnávacieho vzoru sa nachádza na rovnakom riadku ako najvyšší modul tretieho vyhľadávacieho vzoru a na stĺpci ako modul najviac vľavo v druhom vyhľadávacom vzore.

Najprv je určený približný stred vyhľadávacieho vzoru ako priesečník dvoch priamok. Prvá začína v ľavom hornom bode druhého vyhľadávacieho vzoru a prechádza ľavým dolným bodom toho istého vzoru. Druhá priamka má počiatok v ľavom hornom rohu tretieho vyhľadávacieho vzoru a prechádza cez pravý horný roh tohto vyhľadávacieho vzoru. Priesečník dvoch priamok, ktoré sú určené dvomi bodmi sa vypočíta ako

$$x = \frac{(x_1 * y_2 - y_1 * x_2) * (x_3 - x_4) - (x_1 - x_2) * (x_3 * y_4 - y_3 * x_4)}{(x_1 - x_2) * (y_3 - y_4) - (y_1 - y_2) * (x_3 - x_4)} \text{ a}$$

$$y = \frac{(x_1 * y_2 - y_1 * x_2) * (y_3 - y_4) - (y_1 - y_2) * (x_3 * y_4 - y_3 * x_4)}{(x_1 - x_2) * (y_3 - y_4) - (y_1 - y_2) * (x_3 - x_4)} ,$$

Kde *x, y* sú súradnice výsledného priesečníka, *x1, y1* a *x2, y2* sú súradnice dvoch bodov prvej priamky, *x3, y3 a x4, y4* sú súradnice dvoch bodov druhej priamky.

Ďalej sa vymedzí štvorcová oblasť, v ktorej sa bude zarovnávací vzor vyhľadávať. Veľkosť tejto oblasti je rovná šírke 10 modulov QR Kódu, čo postačuje aj pri nepresnom určení stredu

zarovnávacieho vzoru. Oblasť pre vyhľadávanie je zvýraznená v obrázku 8.2 modrou farbou. Táto oblasť je postupne celá prehľadávaná po jednotlivých riadkoch. Hľadajú sa prechody medzi bielymi a tmavými modulmi kódu. Ak je zaznamenaná zmena, je skontrolované, či nejde iba o chybu spôsobenú šumom alebo rozostrením obrazu. Najprv sa uloží hodnota pôvodnej oblasti (hodnota 0, alebo 255). Pri detekcii zmeny sa akumulujú hodnoty bodov v okolí bodu v ktorom nastala zmena, a to vždy 5 bodov pre každý skúmaný bod. Následne je táto hodnota vydelená počtom bodov, teda 5, a ak je rozdiel výslednej hodnoty a pôvodnej hodnoty farby bodu väčší ako 128, znamená to že došlo v danom bode k zmene farby. Výsledkom je postupnosť čísiel, ktoré predstavujú dĺžky oblastí rovnakej farby na jednom riadku. Takisto je pre následnú kontrolu vzoru ukladaná aj farba jednotlivých oblastí. V tejto postupnosti sa potom hľadajú také oblasti, ktorých pomer bodov a farba vyhovujú štruktúre zarovnávaciemu reťazcu QR Kódu, popísaného v kapitole 3. Tolerancia pri porovnávaní je $\frac{w}{4}$, kde *w* je veľkosť modulu, aby nedochádzalo k chybám pri menej kvalitných obrázkoch.

Ak je detekcia úspešná, daný bod sa skontroluje aj vo vertikálnom smere podobným spôsobom. Ak splňuje podmienku aj vo vertikálnom smere, znamená to že je to zarovnávací reťazec, pretože v kóde by sa nemala nachádzať iná oblasť s podobnou štruktúrou [1]. Pri úspešnej lokalizácií je ešte vypočítaná presná poloha pravého dolného konca zarovnávacieho vzoru, znázornené na obrázku 8.2 zelenou farbou, a vyhľadávanie končí.

8.3.3 Rotácia

Zo získaných pozícií vyhľadávacích vzorov je určené prípadne natočenie, alebo prevrátenie QR Kódu. Uhol sa počíta medzi dvomi vrchnými vyhľadávacími vzormi, konkrétne medzi ich stredmi a je počítaný podľa vzťahu:

$$angle = \arctan(y2 - y1, x2 - x1) * 180 / \pi ,$$

kde *x1*, *y2* sú súradnice stredu ľavého vyhľadávacieho vzoru a *x2*, *y2* súradnice pravého. Výsledný uhol *angle* je v stupňoch.

Ak je výsledný uhol väčší ako 2 stupne, je QR Kód otočený o príslušný uhol. Ak je uhol menší ako 2 stupne rotáciu môžeme zanedbať, pretože je príliš malá a nebude mať vplyv na ďalšie spracovanie obrazu. Pre rotáciu obrázku je potrebné vypočítať transformačnú maticu, ktorá má tvar:

$$\begin{bmatrix} \cos\theta & \sin\theta & t_x \\ -\sin\theta & \cos\theta & t_y \end{bmatrix}$$

kde θ je získaný uhol rotácie v radiánoch a t_x ,t_y sú súradnice stredu rotácie, v tomto prípade je to stred obrázka s QR Kódom. Na samotnú rotáciu obrázku je použitá funkcia z knižnice OpenCV, ktorá podľa zadanej rotačnej matice transformuje celý obrázok.

8.3.4 Verzia kódu a veľkosť modulu

Ak je známa presná pozícia vyhľadávacích vzorov v obrázku, je možné pomocou nich určiť šírku celej oblasti kódu a taktiež veľkosť jedného modulu. Z týchto dvoch informácií sa následne získa počet modulov prislúchajúcich na šírku, resp. výšku kódu. Toto rozlíšenie kódu v moduloch predstavuje vlastne jeho verziu, kedy Verzia 1 má rozlíšenie 21 modulov, a každá ďalšia verzia má o č moduly vyššie rozlíšenie. Výška aj šírka kódu v moduloch musí byť totožná, inak ide o neplatný

QR Kód. Takéto určenie verzie kódu nemusí byť vždy presné, pretože šírka modulu sa určuje ba na základe šírky vyhľadávacieho reťazca, ktorý ma šírku 7 modulov. Na presnejšie určenie je možné použiť hodnotu priemeru širok všetkých strán vyhľadávacích modulov, ale najpresnejšou metódou je určenie pomocou lokalizácie časovacieho reťazca. Ten obsahuje pravidelne sa strádajúce tmavé a biele moduly a je napojený na vyhľadávacie vzory takže spolu s nimi pokrýva celú šírku, resp. výšku QR Kódu. Preto umožňuje presné určenie rozlíšenia kódu v moduloch a určenie správnej verzie kódu.

Šírka kódu je vypočítaná ako vzdialenosť medzi ľavým horným rohom prvého vyhľadávacieho vzoru a pravým horným rohom druhého vyhľadávacieho vzoru, označovanie vzorov viď. podkapitola 8.3.1. Rovnakým spôsobom je vypočítaná veľkosť vyhľadávacích vzorov, ako vzdialenosť ich bodov, vždy na jednej strane. Vzdialenosť dvoch bodov v obraze je počítaná podľa vzťahu

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \ ,$$

kde *x1*, *y2* sú súradnice stredu prvého bodu, *x2*, *y2* súradnice druhého bodu a *d* je vzdialenosť medzi nimi v bodoch obrazu.

Šírka jedného modulu v bodoch obrazu je určená ako podiel $w_M = w_{FP} / n$, kde w_{FP} je šírka vyhľadávacieho vzoru a n je počet modulov na vyhľadávací vzor, v tomto prípade je to 7 modulov, podľa štruktúru QR Kódu. Následne je vypočítaná približná hodnota dimenzie kódu, teda počtu modulov na šírku, resp. výšku kódu. Tá je určená ako podiel šírky kódu a šírky jedného modulu.

Pre presnejšie určenie dimenzie QR Kódu je použitý časovací reťazec. Tie sa v kóde nachádzajú dva, pre čo najpresnejšie určenie dimenzie. Jeden vertikálne medzi prvým a tretím vyhľadávacím vzorom a jeden horizontálne medzi prvým a tretím vyhľadávacím vzorom. Dimenzia je vypočítaná tak že sa zoberie počiatočný a koncový bod, v tomto prípade rohy vyhľadávacích vzorov, a uložia sa všetky hodnoty intenzity bodov na spojnici medzi týmito bodmi. Z tej je potom určený počet prechodov medzi bielymi a tmavými bodmi, čím je určený počet modulov časovacieho vzoru. K nemu je ešte potrebné pripočítať počty modulov vo vyhľadávacích vzoroch na každej strane časovacieho vzoru, konkrétne 2 * 7 a výsledná hodnota predstavuje dimenziu QR Kódu.

8.3.5 Perspektívna korekcia

Perspektívna korekcia slúži na odstránenie deformácií obrazu, ako skosenie alebo natočenie. Hlavným cieľom je odstránenie chýb snímacieho zariadenia, alebo nesprávnej polohy kódu pri snímaní. Ide v podstate o transformáciu QR Kódu, ktorý je natočený alebo skosený do štvorcovej zarovnanej oblasti a to pomocou lokalizovaných bodov.

Pre perspektívnu korekciu je najskôr potrebné získať transformačnú maticu, podľa ktorej bude aplikovaná transformácia pre celý obraz. Pre jej výpočet je potrebné poznať pozície aspoň štyroch bodov v pôvodnom obraze a ich príslušné pozície v novom obraze, teda súradnice po transformácií. K tomu použijeme krajné body QR Kódu, teda rohové body troch vyhľadávacích reťazcov. Problém nastáva pri určení polohy štvrtého, pravého dolného rohu QR Kódu však nepoznáme. Nájsť polohu tohto bodu v QR Kóde je problém , viď [1], pretože v tomto bode nie je presne stanovená štruktúra modulov, ale je tu už dátová oblasť, ktorá môže mať ľubovoľné rozmiestnenie modulov. Najbližší bod ktorého poloha je známa je pravý dolný roh zarovnávacieho reťazca, preto sa použije ako štvrtý referenčný bod práve ten. Ak sa dekóduje QR Kód Verzie 1, ktorý neobsahuje zarovnávací vzor, tak sa predpokladá, že je QR Kód tvaru štvorca a štvrtý bod je približne dopočítaný z pozície troch vyhľadávacích vzorov.

Pred samotnou transformáciou je potrebné vytvoriť cieľový obrázok, ktorého veľkosť sa rovná dimenzii, teda počtu modulov, vynásobená zaokrúhlenou veľkosťou jedného modulu. Veľkosť je

podobná ako v prípade pôvodného obrázku, aby nedošlo k strate informácie pri zmenšení obrazu. Obrázok má štvorcový tvar pretože QR Kód má rovnaký počet modulov na šírku, aj na výšku.

Z pozície bodov v referenčnom a novom obrázku sa pomocou vstavanej funkcie získa transformačná matica o rozmeroch 3 x 3. Transformácia je aplikovaná na každý bod pôvodného obrázku. Na to je opäť v knižnici OpenCV pripravená funkcia. Jej vstupmi sú referenčný obrázok, nový obrázok a transformačný matica. Výsledkom je pravidelný, štvorcový obrázok v ktorom sa už nachádza iba QR Kód. Princíp a výsledky transformácie je znázornený na obrázku 8.3.

Obrázok 8.3: Perspektívna korekcia

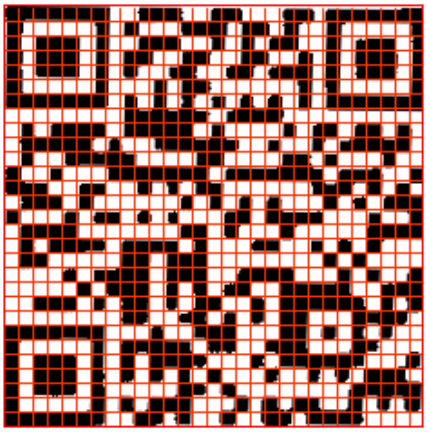
8.4 Prevod QR Kódu na maticu

Hodnoty jednotlivých modulov z obrázka QR Kódu je potrebné previesť do binárnej matice, ktorá bude obsahovať hodnotu 0 pre biely modul a hodnotu 1 pre tmavý modul. Matica má štvorcové rozmery podľa dimenzie QR Kódu. Obrázok je preložený mriežkou s rozmermi odpovedajúcimi vytváranej matici, viď obrázok 8.4. Hodnotu bitu v jednotlivých bodoch obrazu nie je vhodné určiť iba z hodnoty v jednom bode. Mohlo by dôjsť k chybám spôsobeným šumom v obraze alebo nepresným zvolením bodu, ktorý môže byť namiesto v strede modulu, na jeho okraji, čo môže skresľovať výsledky. Preto je použité väčšie okolie bodu, konkrétne 3 x 3 body. Na získanie hodnoty daného bodu sa použije aritmetický priemer hodnôt z okolia bodu. Na výpočet týchto bodov je použitá maska

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Najväčšiu váhu pri počítaní hodnoty majú body bližšie k stredu, pretože body, ktoré sú ďalej môžu byť skreslené šumom, prípadne zlým určením polohy stredu. Ak je takto získaná hodnota väčšia ako 128, je výsledný bit v matici reprezentujúci jeden modul kódu nastavený na 1, inak 0.

Výsledkom je binárna matica reprezentujúca QR Kód ktorá je ďalej použitá pri procese dekódovania informácie uloženej v kóde.



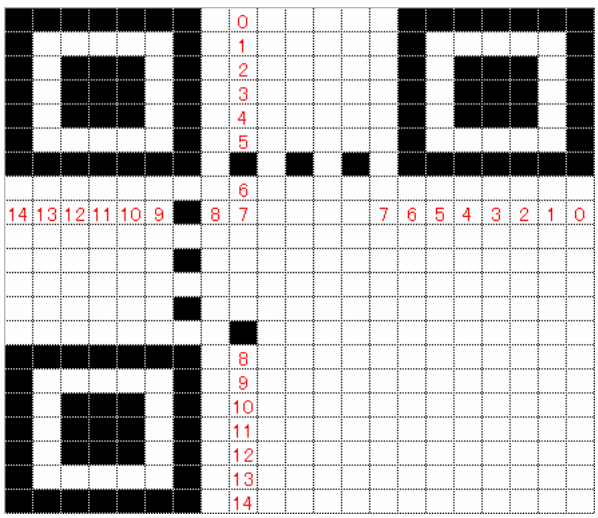
Obrázok 8.4: Rozdelenie na moduly pri prevode na maticu

8.5 Dekódovanie

Na reprezentáciu jednotlivých bitových postupností pri procese dekódovania QR Kódu je použitá knižnica bitset. Tá umožňuje veľmi efektívne ukladať bitové dáta, pretože jeden bit je v skutočnosti naozaj reprezentovaný iba jedným bitom v pamäti počítača. Taktiež obsahuje rôzne operácie nad bitovými postupnosťami, vrátane bitových operácií.

8.5.1 Získanie formátu

Prvým krokom dekódovania informácie je získanie formátu QR Kódu. Formát predstavuje úroveň chybovej korekcie a maskovací vzor použitý na maskovanie dát v procese kódovania. Kompletná informácia o formáte je zložená z 15 bitov, kde prvých 5 bitov je samotný formát a 10 bitov je kód pre prípadnú opravu chýb. Týchto 5 bitov je zakódovaných algoritmom Bose-Chaudhuri-Hocquenghem (15,5). Keďže formát pozostáva iba z 5 bitov, nie je potrebné pri dekódovaní počítať opravný kód, ale príslušné informácie vyhľadať v tabuľke, ktorá obsahuje všetky kombinácie a je dostupná v [1].



Obrázok 8.5: Rozmiestnenie modulov obsahujúcich informáciu o formáte

Jednotlivé bity informácie o formáte sú rozmiestené v štruktúre kódu podľa obrázku 8.5. Kód obsahuje hneď dve rovnaké takéto postupnosti, pretože určenie chybovej korekcie a maskovacieho vzoru je pre ďalšie dekódovania kľúčovým krokom. Informácia je prečítaná spôsobom, že najvýznamnejší bit je na čísle 14. Obidve takto získané postupnosti sú porovnané s referenčnou tabuľkou, kde obidve musia zodpovedať rovnakému riadku a môžu sa líšiť maximálne v troch bitoch. Výslednou postupnosťou reprezentujúcou informáciu o formáte je teda informácia z tabuľky najbližšia obidvom získaným postupnostiam.

Takto získanú postupnosť je potrebné najprv odmaskovať exkluzívnym logickým súčtom s bitovou postupnosťou 101010000010010. Teraz už je možné extrahovať samotné informácie o úrovni opravy chýb a maskovacom vzore. Prvé dva bity predstavujú informáciu o úrovni chybovej korekcie, zakódovanej podľa tabuľky 4.3. Nasledujúce tri bity obsahujú podmienku pre maskovací vzor, zakódovanú podľa tabuľky 8.1.

Zakódovaná hodnota	Podmienka pre maskovanie
000	$(i + j) \bmod 2 = 0$
001	$i \bmod 2 = 0$
010	$j \bmod 3 = 0$
011	$(i + j) \bmod 3 = 0$
100	$((i \text{ div } 2) + (j \text{ div } 3)) \bmod 2 = 0$
101	$(i\, j) \bmod 2 + (i\, j) \bmod 3 = 0$
110	$((i\, j) \bmod 2 + (i\, j) \bmod 3) \bmod 2 = 0$
111	$((i+j) \bmod 2 + (i\, j) \bmod 3) \bmod 2 = 0$

Tabuľka 8.1: Podmienky pre generovanie maskovacieho vzoru

8.5.2 Odstránenie maskovania

Podľa podmienky pre maskovanie je vytvorená maskovacia matica rovnakej veľkosti, ako má matica získaná z QR Kódu. Prechádza sa po jednotlivých bitoch a do maskovacej podmienky sa za premenné *i* z *j* dosádza aktuálne číslo riadku a stĺpca daného bitu v matici. Ak je podmienka splnená, bit je nastavený na hodnotu 1, inak je ponechaná hodnota 0.

Samotné odstránenie maskovania z matice získanej z QR Kódu prebieha tak, že sa prechádza postupne celá matica a pre príslušný bit sa vyhľadá bit s rovnakou pozíciou v maskovacej matici. Ak má tento bit v maskovacej hodnotu 1, je bit v pôvodnej matici invertovaný, teda hodnota 0 sa zmení na 1 a naopak, hodnota 1 na 0. Maskovanie je aplikované iba na dátovú oblasť QR Kódu.

Posledným krokom je zamaskovanie všetkých bitov, ktoré nebudeme ďalej potrebovať a ich nastavením na hodnotu 2. Ide o funkčné vzory QR Kódu, ako vyhľadávacie vzory, zarovnávacie vzory, časovacie vzory a oddeľovače. Informácia o formáte je tiež maskovaná. Takéto maskovanie je aplikované z dôvodu jednoduchšej tvorby jednotlivých kódových slov v ďalšom procese dekódovania.

8.5.3 Získanie bitovej postupnosti z QR Kódu

Ďalším krokom procesu dekódovania je získanie bitovej postupnosti z dátovej oblasti QR Kódu. Celá oblasť kódu sa postupne prechádza, od pravého dolného rohu, až po ľavý koniec kódu. Prechádza sa vždy dvojica stĺpcov QR Kódu naraz, pričom najprv sa kontroluje pravý bit a potom ľavý. Ak je na kontrolovanom mieste hodnota 0, alebo 1, priradí sa táto hodnota konkrétnemu bitu. Ak obsahuje hodnotu 2, toto miesto sa ignoruje a pokračuje sa ďalej. Pri narazení na hranicu kódu sa obracia smer

prechádzania a posúva sa o dva stĺpce vľavo. Takto sa pokračuje, až kým sa nedosiahne ľavá hranica kódu. Princíp prechádzania oblasti kódu a zostavovania bitovej postupnosti je popísaný v kapitole 4.

8.5.4 Určenie dátového módu a počtu kódovaných znakov

Z dátovej postupnosti je najprv potrebné určiť dátový mód, ktorým je informácia v kóde zakódovaná. Tá sa nachádza v prvých 4 bitoch postupnosti a je zakódovaná podľa tabuľky 8.2. Za označením dátového módu nasleduje informácia o počte zakódovaných znakoch. Dĺžka tejto informácie sa líši v závislosti na zvolenom dátovom móde a je znázornená v tabuľke 8.2. Takto získaná postupnosť je prevedená z binárnej reprezentácie do decimálnej reprezentácie, a toto číslo predstavuje počet znakov zakódovaných do dátovej oblasti QR Kódu.

Dátový mód	Kód módu	Dĺžka označenia počtu znakov
Numerický	0001	10 bitov
Alfanumerický	0010	9 bitov
8-bitový dátový mód	0100	8 bitov

Tabuľka 8.2: Označenie dátového módu a dĺžka označenia počtu znakov

8.5.5 Zostavenie kódových slov

Z bitovej postupnosti je potrebné získať jednotlivé kódové slová. Dĺžka kódových slov a počet zakódovaných znakov v jednom kódovom slove je pre každý dátový mód odlišná.

Pre numerický je to 10 bitov na kódové slovo a v každom slove sú zakódované 3 znaky. Ak ide o posledné znaky a ich počet je menší ako 3 je použitých 7 bitov pre zakódovanie dvoch znakov, prípadne 4 bity pre jeden znak. Pre alfanumerický mód je dĺžka jedného kódového slova 11 bitov. V každom kódovom slove sú zakódované dva znaky. Ak ide kódovanie posledného znaku, vtedy je zakódovaný do kódového slova samostatne a jeho dĺžka je 6 bitov. V prípade 8-bitového módu majú všetky kódové slová dĺžku 8 bitov, a v každom je zakódovaný jeden znak.

Takto sú zostavené všetky kódové slová podľa indikátoru počtu znakov. Následne sú jednotlivé kódové slová prevedené z binárnej reprezentácie do dekadickej, pretože tá je vhodnejšia pre ďalšie spracovanie.

8.5.6 Dekódovanie informácie

Z predchádzajúceho postupu sme získali postupnosť čísel, kde každé číslo reprezentuje určitý počet znakov, podľa dátového módu. Posledným krokom dekódovacieho procesu je prevod týchto čísel na jednotlivé znaky. Samotný prevod sa líši pre každý dátový mód.

V prípade numerického módu každá číslica čísla zo získanej postupnosti predstavuje jeden znak zakódovaného vstupu. Každé číslo teda reprezentuje 3 číslice, okrem posledných čísel, ktoré môžu reprezentoval dve prípadne jednu číslicu. Špeciálny prípad nastáva, ak je prvou číslicou 0. Tá je totiž pri prevode z binárnej reprezentácie stratená a musí byť v procese dekódovania doplnená.

V alfanumerickom móde predstavuje každé číslo dva znaky. Prvý znak sa získa celočíselným delením číslom 45. Výsledok delenia je prevedený na znak podľa tabuľky 4.1. Druhý znak predstavuje zvyšok po delení číslom 45, ktorý je rovnakým spôsobom prevedený na odpovedajúci znak. Ak ide o posledný znak, teda číslo predstavuje iba jeden znak, je ten priamo podľa tabuľky 4.1 prevedený na odpovedajúci znak.

V 8-bitovom dátovom móde predstavuje každé číslo jeden znak, ktorý je určený podľa ASCII tabuľky pre ISO/IEC 8859-1 znakovú sadu.

9 Testovanie

Na testovanie vytvorenej aplikácie bola vytvorená testovacia množina obrázkov. Tá je zameraná na čo najlepšie otestovať schopnosti implementovaných algoritmov lokalizovať a dekódovať QR Kódy v obraze. Preto táto množina obsahuje okrem obrázkov, ktoré sú dobre čitateľné, bez šumu, aj obrázky zámerne zdeformované alebo poškodené. Celkový počet testovacích obrázkov je 100 a na tvorbu väčšiny z nich boli použité generátory QR Kódov voľne dostupné na internete. Tie sú vhodné najmä z toho dôvodu, že umožňujú zvoliť formát kódovaných dát, úroveň chybovej korekcie a verziu vytváraného QR Kódu. Testovacia sada obrázkov obsahuje obrázky rôznych verzií od QR Kódu Verzie 1, až po QR kódy Verzie 6, na rozpoznávanie, ktorých je aplikácia zameraná. Taktiež obsahuje rôzne úrovne opravy chýb, napriek tomu že chybová korekcia nie je v aplikácií implementovaná. Je to z toho dôvodu aby bola otestovaná schopnosť aplikácie prečítať celú kódovú oblasť QR Kódu.

Väčšina QR Kódov je ďalej upravovaná. Sú použité rôzne transformácie, ako otočenie, skosenie alebo rozostrenie obrazu. Taktiež je do niektorých obrázkov umelo pridaný šum, aby otestovali schopnosť aplikácie prečítať QR Kód aj napriek nekvalitnému obrázku. Niektoré obrázky boli ďalej vytlačené, v rôznej kvalite a nefotené. Na snímanie obrázkov boli použité dve snímacie zariadenia s rôznou kvalitou snímania. Prvým bol fotoaparát v mobilnom telefóne s rozlíšením 3,2 Megapixlu, bez zostrovania takže snímky ním fotené neboli úplne ostré a obsahovali pomerne veľa šumu. Druhým snímacím zariadením bol digitálny fotoaparát s rozlíšením 10 Megapixlov. Obrázky boli fotené za rôznych svetelných podmienok a z rôznych uhlov. Niektoré boli pri znímaní zámerne zatienené aby otestovali schopnosť spracovať obraz s nehomogénnym osvetlením. Taktiež boli nasnímane QR Kódy nekompletné, kedy bola časť kódu zakrytá. Veľkosť obrázkov sa pohybuje od 115 x 115 bodov pre najmenší obrázok a najväčšie obrázky dosahujú veľkosť približne 500 x 500 bodov. Väčšie obrázky nie je možné použiť z dôvodu spracovávania obrázkov v reálnom čase. Na reprezentáciu QR Kódu však nie je väčšie rozlíšenie potrebné, pretože QR Kód Verzie 6 má hustotu 41 modulov, čo predstavuje pri rozlíšení 500 bodov 12 bodov na modul čo je pre prečítanie kódu dostatočné. Testovacia sada obsahuje aj niekoľko zámerne zlých QR Kódov, ktoré nie je možné dekódovať.

Pri vyhodnocovaní testovaných obrázkov sa brali do úvahy hlavne tri faktory. Sú to schopnosť aplikácie vyhľadať v obraze vyhľadávacie vzory a tým lokalizovať celú oblasť QR Kódu, schopnosť QR Kód prečítať a dekódovať a posledným faktorom bola rýchlosť celého procesu snímania. Z týchto bola najdôležitejšia schopnosť lokalizácie kódu v obraze a čas vyhodnotenia, pretože rozpoznanie QR Kódu musí prebehnúť v reálnom čase. Pri spracovávaní podobných aplikácií je ťažko určiť časovú hranicu, ktorá je ešte považovaná za spracovanie v reálnom čase. Pri testovaní bola zvolená hranica maximálne 0,1 sekundy pre celkové spracovanie obrázku, od jeho načítania, až po zobrazenie výsledku.

Tiež bola testovaná rýchlosť dvoch prístupov pri lokalizácií vyhľadávacích reťazcov. Boli implementované a testované obidva prístupy, teda riadkové skenovanie a lokalizácia pomocou štvorcov, popísaná v kapitole 8. Testovanie bolo zamerané iba na rýchlosť, koľko trvalo kým boli všetky vyhľadávacie reťazce nájdené. Po načítaní obrázka bol spustený časovač a po lokalizácií bol odpočítaný približný čas detekcie.

10 Výsledky

Najprv sú v tabuľke 10.1 uvedené výsledky testovania dvoch algoritmov pre určovanie polohy vyhľadávacích vzorov v obraze. Algoritmy boli testované na rýchlosť spracovania jednotlivých obrázkov z testovacej sady. Rýchlosť ale nezávisí iba od rozlíšenia obrazu, ale aj od rozmiestnenia modulov v QR Kóde. Preto tento test iba približne určuje rýchlosť jednotlivých algoritmov slúžil na výber použitej metódy. Keďže metóda vyhľadávania štvorcov mala lepší priemerný čas pri väčších obrázkoch, je pre lokalizáciu vyhľadávacích vzorov v aplikáciu zvolená práve táto metóda. Algoritmus riadkového skenovania je zasa použitý pri lokalizácií zarovnávacieho vzoru.

Rozlíšenie obrázku v bodoch	Priemerná rýchlosť spracovania	
	Riadkové skenovanie	Metóda štvorcov
100 – 200	8 ms	10 ms
200 – 300	15 ms	16 ms
300 – 400	35 ms	28 ms
400 – 500	70 ms	37 ms

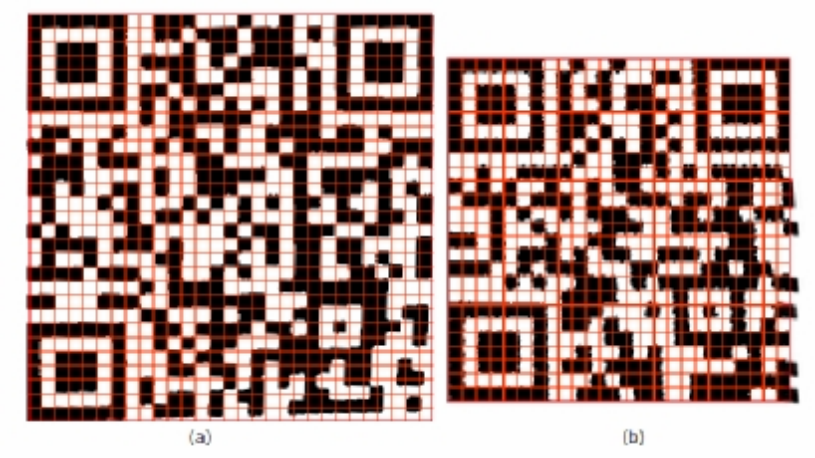
Tabuľka 10.1: Porovnanie algoritmov vyhľadávania

Obrázok 10.1: Príklady menej kvalitných obrázkov, ktoré aplikácia dokáže dekódovať

Vytvorená aplikácia bola testovaná na vytvorenej sade dát. Úspešnosť vyhľadávania vyhľadávacích vzorov dosahuje vysokú úroveň. Zo 100 testovaných obrázkov nie je aplikácia schopná detegovať vyhľadávacie vzory v 5 prípadoch, čo predstavuje 95% úspešnosť. Ďalej nasledoval test vyhľadania zarovnávacieho reťazca, kde sa úspešnosť vyhľadávania pohybuje na testovacej sade obrázkov okolo 92%. Obrázky na ktorých zlyháva vyhľadávanie zarovnávacích vzorov väčšinou obsahujú príliš veľa šumu, alebo sú zle zaostrené. Z toho dôvodu je pri prahovaní stredný modul zarovnávacieho vzoru zmenšený, čo má za následok zlyhanie pri jeho detekcii, pretože jeho veľkosť je menšia ako zvolená tolerancia, príklad na obrázku 10.2b. V prípade 10.2a vyhľadávanie zlyháva je to z dôvodu zmenšenia kódu vo vertikálnom smere, kedy sa pri riadkovom prehľadávaní nájde zarovnávací vzor, ale pri vertikálnej kontrole nevyhovuje zadanej tolerancii.

Obrázok 10.2: Príklady obrázkov na ktorých zlyhala detekcia zarovnávacieho vzoru

Ďalším častým dôvodom neúspešného dekódovania je perspektívna korekcia. Dekódovanie zlyháva práve v chybnom prevedení tohto kroku v 13% prípadoch. Problém je hlavne v chybnom určení bodov pre perspektívnu transformáciu, čo má za následok, že niektoré body sa zobrazia mimo cieľovej plochy a sú nesprávne prevedené do výslednej matice. Na obrázku 10.3a je chybné posunutie modulov v pravom dolnom rohu, za zarovnávacím reťazcom dovnútra kódu, na obrázku 10.3b chybné posunutie mimo oblasť z ktorej je následne počítaná matica.



Obrázok 2,5: Chyba pri perspektívnej korekcii

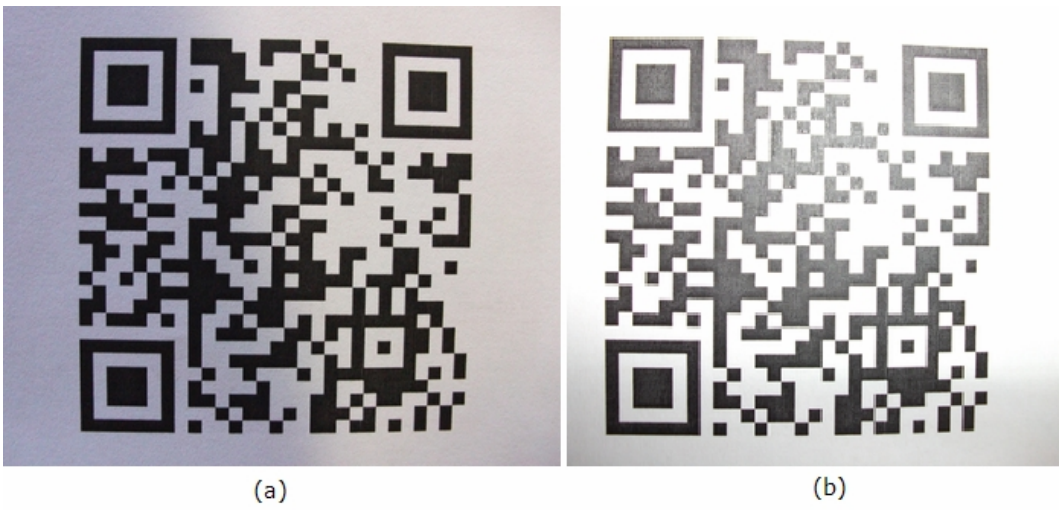
Ďalej nasleduje popis testovania dekódovania QR Kódu. Dekódovanie bolo testované na rovnakých obrázkoch ako vyhľadávanie. Tu bola úspešnosť nižšia ako pri vyhľadávaní a to z dvoch hlavných dôvodov. Prvým bolo, že v aplikácií nie sú implementované algoritmy na opravovanie chýb. To spôsobovalo, že už pri zlom určení hodnoty jedného modulu boli výsledky dekódovania chybné. Táto chyba čiastočne súvisí s chybným prevodom na maticu z dôvodu zlého určenia bodov pri perspektívnej korekcii. To bolo príčinou, že asi v 20% obrázkoch, bola síce informácia dekódovaná, ale obsahovala chyby.

Ďalším problémom pri dekódovaní je, že od QR Kódu Verzie 6 a úrovne opravy chyb vyššej ako M dochádza k rozdeleniu kódových slov a slov na opravu chýb do viacerých blokov, ktoré sú navzájom prekladané vo výslednej postupnosti kódových slov, čo nie je v aplikácií implementované. Táto chyba je príčinou chybného dekódovania v 10% prípadoch, kedy je výstupná informácia úplne odlišná od pôvodne zakódovanej informácie.

V 2% testovaných QR Kódov došlo k zlému určeniu formátu QR Kódu, čo malo za následok zlyhanie dekódovania. Išlo ale o prípady kedy boli moduly obsahujúce informácie o formáte nečitateľné, alebo zakryté cudzím predmetom, preto je toto zlyhanie správne.

Obrázok 10.4: Príklady úspešnej detekcie v obraze so šumom, rozostrenom a skosenom obraze

Spolu teda lokalizácia a dekódovanie zlyháva v pomerne vysokom počte prípadov, blížiacemu sa až 50% prípadov. V ostatných prípadoch je lokalizácia a detekcia QR Kódu v obraze úspešná. Vysoká miera neúspechu je čiastočne spôsobená tým, že obrázky v testovacej sade sú väčšinou horšie čitateľné. Aplikácia napriek tomu dokáže dekódovať aj obrázky obsahujúce šum, obrázok 10.4a, zle zaostrené obrázky, obrázok 10.4b, natočené obrázky a skosené obrázky, obrázok 10.4c. Niektoré ďalšie príklady menej kvalitných obrázkov, ktoré dokáže aplikácia úspešne dekódovať sú na obrázku 10.1. Dobre dokáže taktiež korigovať dôsledky nehomogénneho osvetlenia, obrázok 10.5a a odlesky spôsobené pri snímaní čiarového kódu, obrázok 10.5b.



Obrázok 10.5: Obrázky so zlým osvetlením

Čas spracovania od načítania obrázku až po dekódovania a vypísanie výsledku je závislý hlavne na veľkosti vstupného obrázku. Nezanedbateľná ale je z pohľadu aj štruktúra kódu a rozmiestnenie jednotlivých modulov. Čím viac miest má totiž podobný tvar ako vyhľadávací vzor, tým dlhšie trvá proces lokalizácie. Každé takéto miesto sa totiž musí kontrolovať, či nie je hľadaným vzorom. Priemerný Čas spracovania sa pohybuje od 20 milisekúnd pre menšie obrázky s rozlíšením okolo 200 bodov, až do 70 milisekúnd pre väčšie obrázky s rozlíšením do 500 bodov. Podmienka spracovania v reálnom čase je teda splnená.

10.1 Možnosti pokračovania

Možnosti pokračovania v práci na dekodéri QR Kódov sú široké. Jednou z vecí ako podstatne vylepšiť skóre dekódovania je napríklad implementovať Reed – Solomonove algoritmy na detekciu a korekciu chýb. To by odstránilo chyby často sa vyskytujúce pri dekódovaní. Ďalšou možnosťou je rozšírenie schopnosti dekódovať kódy vyšších verzií, kde je potrebné lokalizovať väčší počet zarovnávacích reťazcov a takisto schopnosť spracovávať dáta uložené vo viacerých blokoch. Zlepšiť by sa dala aj perspektívna korekcia, kde je vhodné pridať viacero referenčných bodov, aby zarovnanie kódu bolo čo najpresnejšie. Vhodným rozšírením je aj podpora pre dekódovanie Micro QR Kódov.

11 Záver

V úvode práce sme sa zoznámili s problematikou čiarových kódov a s ich každodenným využitím v najrôznejších odvetviach ľudskej činnosti. Taktiež boli popísané rôzne metódy na lokalizáciu a dekódovanie rozličných typov kódov. V práci sú popísané jednotlivé spôsoby kódovania pomocou čiarových kódov. Zameraná je hlavne na dvojrozmerné čiarové kódy, ale poskytuje aj prehľad klasických čiarových kódov, pretože spolu úzko súvisia.

Taktiež popisuje princíp počítačového videnia a programovaciu knižnicu pre počítačové videnie OpenCV, ktoré poskytujú vhodné prostriedky na dekódovanie čiarového kódu v obraze. Detailnejšie je popísaná štruktúra QR Kódov, a taktiež princíp kódovania dát do QR Kódov, pretože ďalej pracujeme práve s týmito kódmi.

Podarilo sa implementovať aplikáciu na rozpoznávanie QR Kódov. Postup jej vytvárania je popísaný od návrhu až po samotnú implementáciu. Vytvorená aplikácia dokáže lokalizovať QR Kód v obraze pomocou počítačového videnia a takto nájdený kód dekódovať.

Aplikácia je otestovaná na vzorovej sade QR Kódov, ktorá je vytvorená za účelom čo najlepšie otestovať schopnosti vytvoreného dekodéru. Pri kvalitných snímkach je dekódovanie QR Kódu bezproblémové a netrvá viac ako 100 milisekúnd. Vyhľadávanie a zároveň dekódovanie kódov prebieha teda v reálnom čase. To je tiež jedným z dôvodov pomerne vysokej chybovosti pri dekódovaní informácie. Schopnosť lokalizovať kód v obraze je ale na vysokej úrovni a dosahuje 93%, ale pri dekódovaní informácie sa často vyskytujú chyby, hlavne z dôvodu zlého určenia hodnoty niektorých modulov. Podarilo sa vytvoriť aplikáciu, ktorá je odolná proti rôznym transformáciám obrazu. Dokáže dekódovať obrázky obsahujúce šum, natočené, skosené a inak zdeformované. Taktiež si dokáže poradiť s meniacou sa intenzitou osvetlenia v rôznych častiach kódu alebo s odleskami vzniknutými pri snímaní čiarového kódu.

Schopnosť rozpoznávať čiarové kódy môže byť ďalej rozširovaná, a to hlavne implementáciou postupov na detekciu a korekciu chýb.

Literatúra

[1] ISO/IEC 18004:2005, Information technology — Automatic identification and data capture techniques — QR Code 2005 bar code symbology specification.

[2] Liu, Y., Liu, M.: Automatic Recognition Algorithm of Quick Response Code Based on Embedded System. ISDA '06 Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications - Volume 02, China, 2006.

[3] Liu, Y., Yang, J., Liu, M.: Recognition of QR Code with mobile phones. Control and Decision Conference, Yantai, Shandong, 2008.
Ohbuchi, E., Hanaizumi, H., Hock, H. A., : Barcode Readers using the Camera Device in Mobile Phones. International Conference on Cyberworlds, 2004.

[4] Bradski, G., Kaehler, A. : Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly Media, Inc., Sebastopol, CA, 2008.

[5] Hlaváč, V., Šonka, M. : Počítačové vidění. Garda a.s., Praha 1992.
Karpus, A.: An OpenCV implementation of a QR Barcode reader for the Iphone and Windows Mobile platforms, Honours Project, Carleton University, 2009.

[6] Swetake, Y.: How to create QRcode [online]. Aktualizované: 4.2.2007,
Dostupné na URL: <http://www.swetake.com/qr/qr1_en.html>.

[7] Adams, R.: BarCode 1, A Web Of Information About Bar Code [online]. Aktualizované 9.3.2009, Dostupné na URL: <http://www.adams1.com/history.html>

[8] Microsoft Corporation: High Capacity Color Barcode Technology [online]. 2011, Dostupné na URL: <http://research.microsoft.com/en-us/projects/hccb/about.aspx>

[9] DENSO WAVE INCORPORATED: About 2D Code [online].
Dostupné na URL: < http://www.denso-wave.com/qrcode/aboutqr-e.html>

[10] QR Code [online]. Naposledy modifikované 11.1.2011,
Dostupné na URL: <http://en.wikipedia.org/wiki/QR_Code>

[11] Brian, S.: 2D Barcodes [online]. 2009,
Dostupné na URL: <http://optional.is/required/2009/06/02/what-are-2d-barcodes>

Zoznam príloh

Príloha 1. CD so zdrojovými súbormi, aplikáciou, testovacími snímkami a plagátom prezentujúcim prácu.