UNIVERZITA PARDUBICE

Fakulta elektrotechniky a informatiky

# Kodér / dekodér QR kódů

Michal Horáček

Diplomová práce

2016

**Univerzita Pardubice**
**Fakulta elektrotechniky a informatiky**
Akademický rok: **2014/2015**

# ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

| | |
|---|---|
| Jméno a příjmení: | **Bc. Michal Horáček** |
| Osobní číslo: | **I13374** |
| Studijní program: | **N2612 Elektrotechnika a informatika** |
| Studijní obor: | **Komunikační a řídicí technologie** |
| Název tématu: | **Kodér/dekodér QR kódů** |
| Zadávající katedra: | **Katedra elektrotechniky** |

Zásady pro vypracování:

V teoretické části proveďte analýzu kodéru QR kódů, a to z různých možností zakódování požadované informace (např. zakódování různých vstupních dat, velikosti výsledného QR kódu, možností poškození QR kódu). Další část bude věnována dekódování ideálního QR kódu (neporušeného), a také poškozených QR kódů.
V praktické části vytvořte SW modul s GUI umožňující pro zadanou či importovanou zprávu kódování/ dekódování této zprávy se zobrazením jednotlivých kroků kódování/dekódování. Součástí SW bude i možnost analýzy kvality daného QR kódu.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:


**1. Adámek, J.: Kódování, SNTL, PRaha, 1989**
**2. Costello, L.: Error Control Coding, 2ed., Pearson, 2004**
**3. QR Code Tutorial [online]. 2014 [cit. 2014-01-07]. Dostupné z: http://www.thonky.com/qr-code-tutorial/**
**4. Dobeš, J. Žalud, V.: Moderní radiotechnika, BEN, Praha, 2006**


Vedoucí diplomové práce: **Ing. Jan Pidanič, Ph.D.**
Katedra elektrotechniky

Datum zadání diplomové práce: **31. října 2014**
Termín odevzdání diplomové práce: **15. května 2015**

Univerzita Pardubice
Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Zdeněk Němec, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2014

**Prohlášení autora**

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 20. 5. 2016 Michal Horáček

**Poděkování**

Chci poděkovat všem, kteří mě podporovali v psaní této diplomové práce, především panu Ing. Janu Pidaničovi za odborné rady a trpělivost při jejím zpracování. Také chci poděkovat své rodině a přátelům za jejich podporu a pomoc během mého studia na vysoké škole.

**Anotace**

Diplomová práce se zabývá kódováním a dekódováním QR kódů, které slouží k uchování informace v obrazové matici. Práce je rozdělena na část teoretickou, kde je popsán proces kódování a dekódování QR kódů, a část praktickou, kde jsou pomocí aplikace vytvořené v programu MATLAB uvedeny příklady demonstrující výše uvedené procesy.

**Klíčová slova**

QR kód, kodér, dekodér, Reed-Solomonovy kódy, Zpracování obrazu

**Title**

QR coder and decoder

**Annotation**

The diploma thesis describes techniques for encoding and decoding QR codes that offer storing a message in a visual scene. This work is divided into a theoretical part, describing a process of coding and decoding the QR code, and a practical part containing examples of processes mentioned above using the application created in MATLAB.

**Keywords**

QR code, coder, decoder, Reed-Solomon codes, Image processing

# Obsah

## Seznam zkratek

QR Quick Response

ISO International Organization for Standardization

RS Reed Solomon

# Seznam obrázků

## Seznam tabulek

# Úvod

Cílem této diplomové práce je podrobné popsání a vysvětlení principu fungování QR kódů, jejich vytvoření a možnosti použití v praxi. Dalším cílem je vytvoření sofistikovaného softwaru, který dokáže jak vytvořit QR kód podle zadaných parametrů, tak jej detekovat v obrazové scéně a vyčíst z něj původní informaci.

První kapitola stručně pojednává o historickém vývoji čárových kódů a jejich použití v praxi. Dále jsou zmíněny některé druhy dvourozměrných kódů včetně QR kódů.

Ve druhé kapitole jsou popsány jednotlivé kroky vytváření kódu. Čtenář se zde dozví, jakým způsobem jsou data zakódována a jak jsou zabezpečena proti chybám vzniklým při snímání QR kódu. Popsána je samostatně matice QR kódu a části, ze kterých se skládá.

Ve třetí kapitole je popsán princip snímání QR kódu a následné dekódování. Na začátku procesu je vstupní obrazová scéna, na kterou jsou aplikovány metody zpracování obrazové informace, díky kterým lze nalézt polohu QR kódu v této scéně. Dekódování probíhá podobně jako kódování, navíc jsou v datech nalezeny a opraveny případné chyby.

Poslední čtvrtá kapitola je věnována praktické části práce. Hlavní náplní je popis aplikace umožňující kódování a dekódování QR kódů podle postupu popsaného v teoretické části. Součástí jsou i vzorové příklady, na kterých je celá problematika snáze pochopitelná.

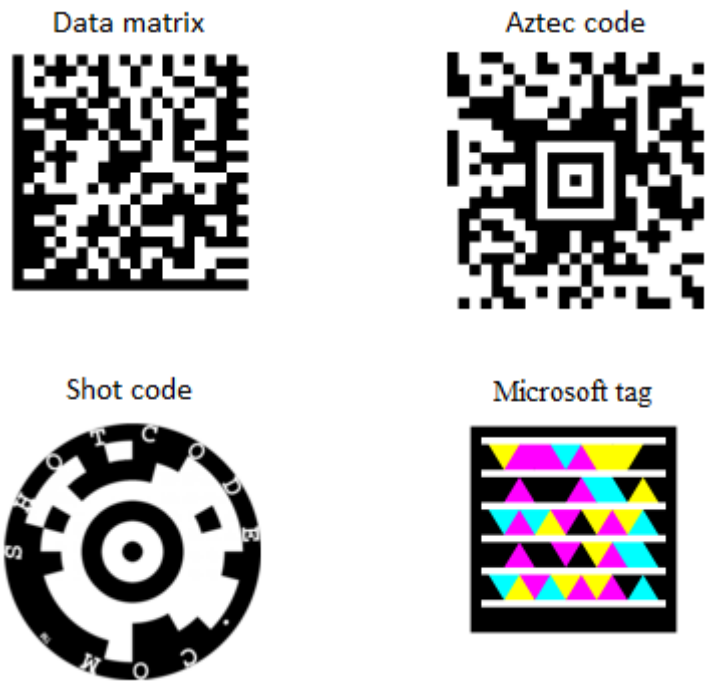
# 1 Historie snímacích kódů

QR kód patří mezi nejpoužívanější snímací kódy. Byl vytvořen japonskou firmou Denso Wave [4] v roce 1994 a je dnes popsán standardem ISO 18004 [5]. Samotný název vychází z anglického „Quick Response“, v překladu rychlá odezva. Je primárně určen k rychlému poskytnutí velkého množství dat prostřednictvím čtecího zařízení. Ve světě se stal velice populárním především mezi širokou veřejností, která získává informace z QR kódů pomocí fotoaparátu mobilního telefonu. Jednou z hlavních výhod je také vysoká schopnost opravy chyb při dekódování. Kód využívá tzv. Reed Solomonových samoopravných kódů [1], díky kterým je schopen opravit až 30% poškozených dat.

Historie snímacích kódů spadá až do poloviny 20. století, kdy Američan Joseph Woodland vynalezl čárový kód. Původním záměrem bylo urychlit zpracování zboží u pokladen ve velkých obchodech. S vynalezením laserového čtecího zařízení se začaly čárové kódy používat právě na obalech zboží. Nejstarším je kód typu 2/5, který zakóduje pouze numerické znaky. Později byl vyvinut novější typ známý pod zkratkou EAN, z anglického „European article number“. Ten dokáže pojmout více dat. Obsahuje například informaci o zemi původu výrobce, dokáže zakódovat také celý ISBN kód používaný u knih.


Obrázek 1 - Ukázka čárových kódů [6]

Později se začaly objevovat dvourozměrné (2D) kódy. Příčinou vzniku byla nutnost dalšího navýšení kapacity dat obsažených v kódu. 2D kódy se staly velmi populárními a začaly postupně nahrazovat klasické čárové kódy. Většina kódů má čtvercový vzhled i tvar datových modulů. Mimo QR kód sem patří například Data Matrix [7] nebo Aztec Code [8]. Vyskytují se ale i kódy poněkud exotičtější. Zajímavý je například Shot Code [9], jehož data jsou zapsána do souvislých kružnic (tento formát již není podporován). Mezi nejnovější patří barevný High Capacity Color Barcode (HCCB) od firmy Microsoft, též zvaný Microsoft tag [10]. Základní modul pro interpretaci dat tvoří trojúhelník, který může nabývat čtyř nebo osmi barevných možností. Tímto způsobem lze do kódu uložit mnohem více dat než do klasického černobílého kódu. S rostoucím počtem barev ovšem stoupají požadavky na kvalitu skenovacího zařízení.



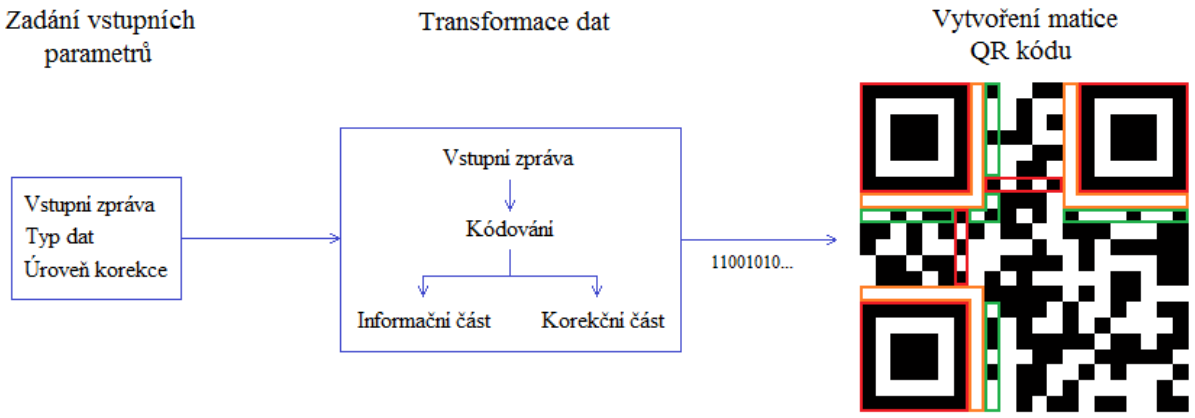

Obrázek 2 - Ukázka 2D kódů [7, 8, 9, 10]

QR kód by původně vyvinut pro automobilový průmysl, dnes je jeho použití značně rozšířené. Slouží například ke komerčním účelům - zadavatel do QR kódu zakóduje webový odkaz na stránky se svým produktem či službou, uživatel jej dekóduje pomocí aplikace v telefonu a ta jej na tyto stránky přímo odkáže. Prospěšného využití se našlo také v lékařství, jedná se o francouzský projekt MyQRV [11] – QR kód obsahuje osobní informace důležité pro ošetřujícího zdravotníka, například krevní skupina, alergie apod. Ten je poté vyražen do kovového štítku, který lze upevnit provázkem na ruku jako elegantní přívěšek.

Obrázek 3 - praktické využití QR kódu [11]

# 2 Vytvoření QR kódu

Proces vytvoření QR kódu lze rozdělit do tří základních fází – zadání vstupních parametrů, transformace dat a vytvoření matice QR kódu, jak je znázorněno na schématu níže:




Obrázek 4 – Schéma vytvoření QR kódu

Vstupními parametry jsou vstupní zpráva, typ dat a úroveň zabezpečení dat proti poškození. Tyto parametry mají vliv především na rozměry výsledné matice. Nejlepší vlastnosti QR kódu z hlediska získání původní informace dostaneme tak, že velikost dat bude co nejmenší a úroveň zabezpečení nejvyšší. Snímání QR kódu skenovacím zařízením je přesnější, pokud jsou jednotlivé „čtverečky“ v obrazové scéně dostatečně velké. Pro QR kódy s vysokým objemem dat je potřeba, aby skenovací zařízení mělo velkou rozlišovací schopnost, jinak dojde k chybnému vyhodnocení datové matice. Úroveň zabezpečení zajistí, že pokud bude například při skenování část QR kódu překrytá nějakým předmětem či bude jinak znemožněno čtení všech bodů matice, budeme stále schopní získat původní informaci.

Po nastavení vstupních parametrů je vstupní zpráva transformována do binárního kódu. Ten tvoří dvě části – informační a korekční. Informační část obsahuje hlavičku s informacemi o vstupních parametrech a počtu znaků zprávy. Za hlavičkou je připojena samotná zpráva, která je upravena do vhodné formy. Korekční část dat má tu vlastnost, že společně s informační částí tvoří kód schopný nalézt a opravit chyby, které se v datech mohou objevit. Konkrétně pro QR kód se používají Reed Solomonovy kódy [1].

Poslední fází je vytvoření matice QR kódu. Ta je tvořena s několika částí, z nichž některá obsahují klíčová data a některá slouží k ohraničení dat za účelem snadnější detekce skenovacím zařízením a tvoří jakousi kostru (ve schématu barevně ohraničená místa). Transformovaná data jsou poskládána do zbylého prostoru.

Při celém postupu je samozřejmě nutné dodržovat standard definující přesný formát [4].

## 2.1 Analýza dat

Prvním krokem je výběr typu dat, se kterým budeme chtít pracovat. Celkový obsah dat, který může QR kód nést, závisí na počtu symbolů zdrojové abecedy, které chceme použít pro zakódování. QR kód má definované různé druhy módů z hlediska obsahu znaků ve zdrojové abecedě.

Módy QR kódu:

**Numerický mód**
Tento mód používá pouze číselné znaky 0 – 9.

**Alfanumerický mód**
Na rozdíl od numerického módu navíc používá velké znaky abecedy A – Z a také znaky $, %, *, +, -, ., / a mezeru. Dohromady je to 45 znaků.

**Byte mód**
Jak už název napovídá, tento typ dat je složen z bytových, tedy osmi-bitových znaků podle normy ISO-8859-1. Využitelných je tedy 256 znaků.

**Kanji mód**
Kanji je pojmenování pro sadu symbolů, které se používají například v čínštině či japonštině. Tento mód nebude dále rozebrán.

## 2.2 Kódování dat

Zde jsou vstupní data zakódována na základě vybraného módu. Před vlastním zakódováním je nutné zvolit základní parametry kódování. Postup zpracování je následující:

1. Volba úrovně ochrany proti poškození
2. Výpočet verze QR kódu
3. Vytvoření hlavičky
4. Transformace vstupních dat

**Volba úrovně ochrany proti poškození**
Ochranou proti poškození se myslí schopnost detekovat celý obsah zprávy i přesto, že je část dat poškozen či zcela chybí. Rozlišujeme 4 úrovně ochrany podle procentuální schopnosti korekce poškozených dat, viz Tabulka 1:

Tabulka 1 - Úrovně ochrany

| Úroveň ochrany | Schopnost korekce |
|---|---|
| L – Low | Opraví až 7% dat |
| M - Medium | Opraví až 15% dat |
| Q - Quality | Opraví až 25% dat |
| H - high | Opraví až 30% dat |

Volba úrovně závisí na požadavku ochrany dat. Vyšší ochrana znamená nižší množství obsažené informace. Pro korekci dat se používají tzv. Reed – Solomonovy kódy, které jsou součástí skupiny BCH kódů. Tyto kódy se později připojí k originálním datům. Princip bude vysvětlen v další kapitole.

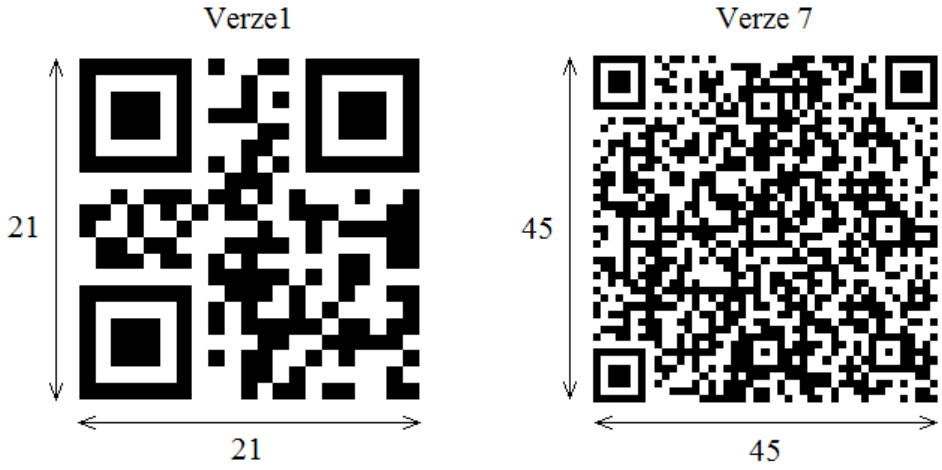
**Výpočet verze QR kódu**

Definovány jsou verze 1 – 40. Čím více dat má být v QR kódu uloženo, tím vyšší verze bude použita a tím větší budou rozměry výsledné matice. Rozměr matice je udáván v počtu tzv. modulů na obou hranách matice. Vypočítá se vztahem

$$N = 17 + v \cdot 4 \ ,$$

kde $v$ je číslo verze.

Příklad pro QR kód verze 1 a 7:




Obrázek 5 - Verze QR kódu

Standard uvádí, kolik znaků se může maximálně vyskytovat ve vstupní zprávě, v závislosti na všech výše uvedených parametrech – typ dat, úroveň ochrany a verze QR kódu. V následující tabulce jsou vypsány kapacity pro verzi 1 a 2:

Tabulka 2 - Kapacita QR kódu v počtu znaků

| Verze QR kódu | Úroveň ochrany | Numerický mód | Alfanumerický mód | Byte mód | Kanji mód |
|---|---|---|---|---|---|
| 1 | L | 41 | 25 | 17 | 10 |
| | M | 34 | 20 | 14 | 8 |
| | Q | 27 | 16 | 11 | 7 |
| | H | 17 | 10 | 7 | 4 |
| 2 | L | 77 | 47 | 32 | 20 |
| | M | 63 | 38 | 26 | 16 |
| | Q | 48 | 29 | 20 | 12 |
| | H | 34 | 20 | 14 | 8 |

Verze QR kódu se automaticky přiděluje podle datového objemu zprávy, nicméně bychom mohli data vložit i do verze vyšší.

**Vytvoření hlavičky**
Hlavička udává informace o vstupním proudu dat. Obsahuje index zvoleného módu a počet znaků vstupní zprávy. Index módu je vždy tvořen 4 bity, počet bitů vyhrazených pro počet znaků závisí na verzi a módu a pohybuje se mezi 8 a 16 bity.

**Transformace vstupních dat**
Při zadávání vstupní zprávy jsou symboly obvykle tvořeny 8 bity, jako je tomu v ASCII tabulce. Naší snahou je data převést do formy zabírající co nejméně bitů.

Pokud kódujeme zprávu numerického módu, vezmeme postupně skupinu 3 čísel a získané trojmístné číslo převedeme do binárního kódu o 10 bitech (standard ošetřuje výjimku pro zbytková čísla na konci řady). Kdybychom trojčíslí nechali v původní formě, zabraly by 24 bitů. Pokud jsme na začátku zvolili Byte mód, data se neupravují a jsou rovnou předány k dalšímu zpracování.

Hlavička a transformovaná data jsou poté spojena do bitového řetězce podle schématu:

| Index módu | Počet znaků | Transformovaná data | Doplňkový kód |
|---|---|---|---|
| 0100 | 00010011 | 00100101 11011001 01011100 … | 000 |

Podle počtu bitů hlavičky a maximálního počtu znaků zprávy vypočítáme celkovou kapacitu vstupního řetězce. Ten se proto doplní doplňkovým kódem, abychom celý datový prostor vyplnili. Tento celek nazýváme „Informační částí QR kódu“. V terminologii se pro každou skupinu osmi bitů (Byte) používá pojem „Data codeword“, který bude dále v textu použit také.

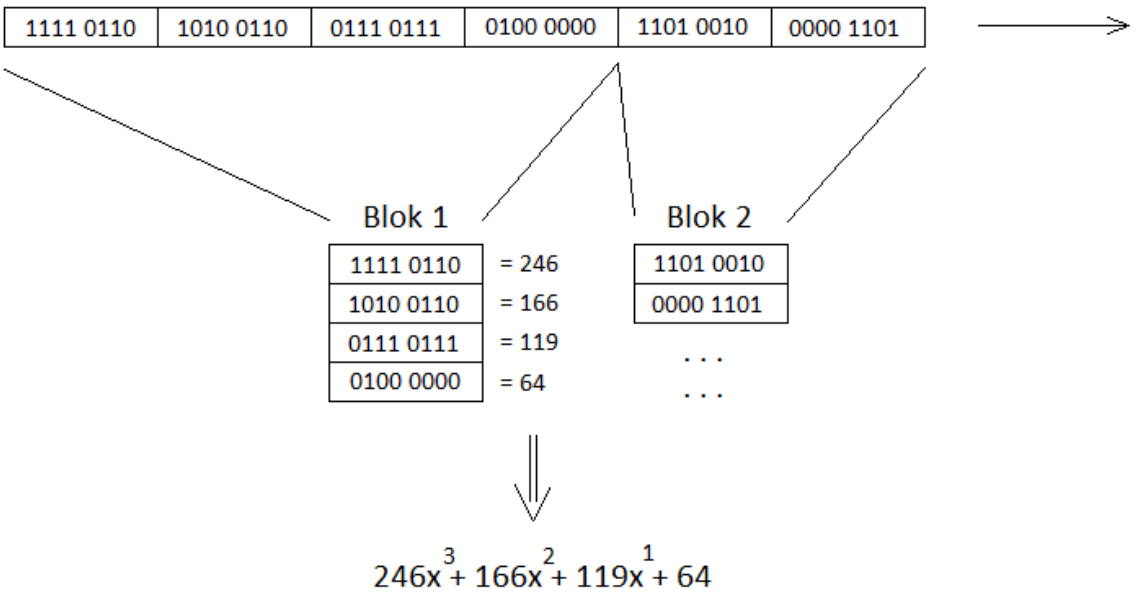
## 2.3 Výpočet samoopravných kódů

Jak již bylo zmíněno výše, samoopravné kódy slouží k získání původních dat i přes zanesení chyb do přijatého kódu. QR kód používá Reed Solomonovy kódy, které patří do skupiny tzv. blokových cyklických kódů. Blokové kódy se používají tak, že se připojí za původní data, nejsou tedy do nich přímo zakódována. Principiální popis RS kódů je uveden níže. Detailní popis RS kódů lze najít v literatuře [1, 5, 12].

Princip detekce a korekce chyby spočívá v tom, že se z určité části dat vytvoří polynom, na který je následně aplikováno dělení tzv. generujícím polynomem. Tím je získán zbytek po dělení a ten se poté připojí k vybrané části dat. Pokud bychom tento celek opět vydělili generujícím polynomem, logicky bychom získali nulový zbytek. Pokud ale některé bity pozměníme, nulu nezískáme, což nám signalizuje přítomnost chyb.

Před samotným výpočtem samoopravných kódů se datová část rozdělí na osmibitové části, ty jsou převedeny na čísla v decimálním tvaru. Vzniklá sada je následně rozdělena do tzv. bloků.

Každý z bloků obsahuje určitý počet čísel, ze kterých je následně vytvořen polynom. Na následujícím obrázku je tento proces zřetelnější (uvedený rozsah dat je pouze ilustrující):




Obrázek 6 - Rozdělení datové části do bloků

Ke každému polynomu, který vytvoříme z datové části, následně vypočítáme polynom reprezentující korekční kód. Koeficienty polynomu jsou dále zpětně převedeny do bitové formy. Na konci získáme tolik polynomů, kolik máme bloků. Všechna získaná data jsou sloučena jednoho celku. Nazvěme jej „Korekční částí QR kódu“. Podobně jako v Informační části, i zde má každá skupina 8 bitů svůj název, a to „Error codeword“.

## 2.4 Uspořádání dat

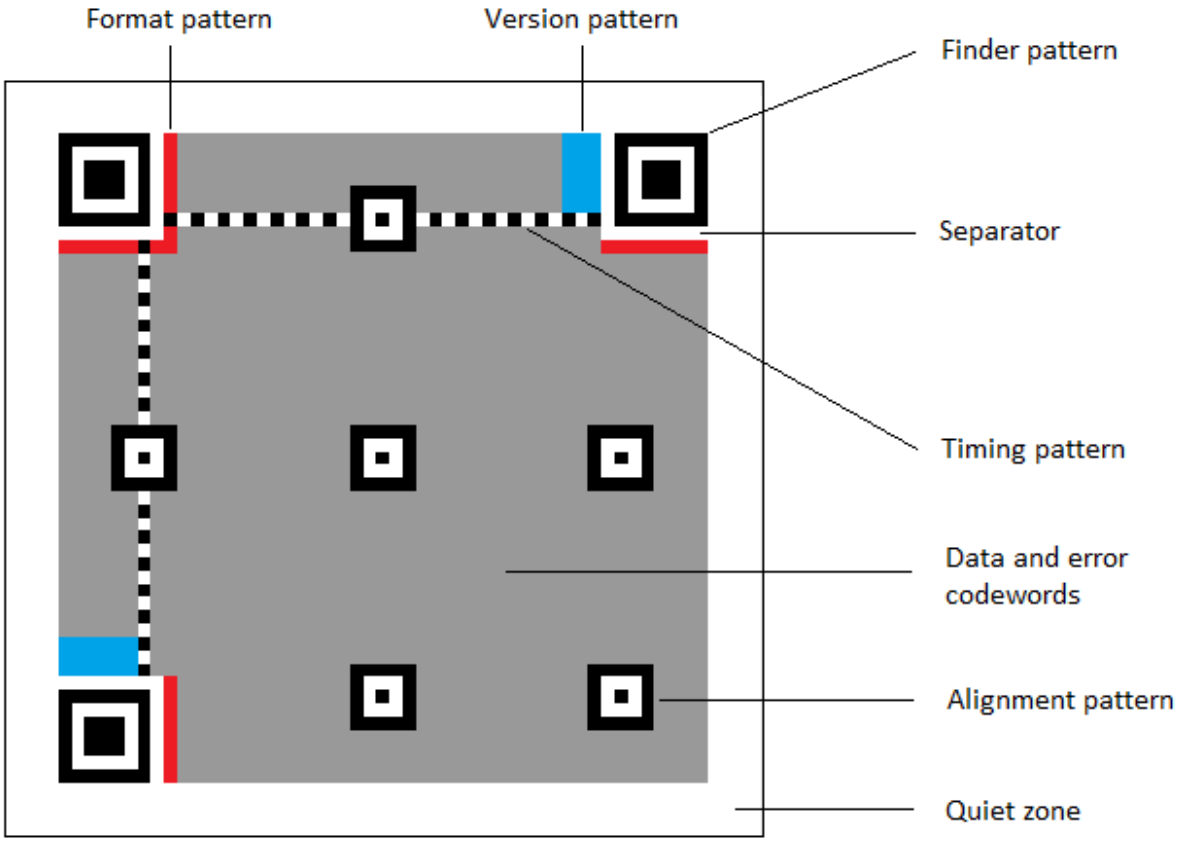
Nyní máme k dispozici informační část a korekční část dat. V tomto kroku tyto části spojíme v jednu, a to podle pravidel, která nám dává standard. Obě části se rozdělí na bloky obdobně, jak tomu bylo při tvorbě korekčních kódů. Postupně jsou odebírána první hodnota z prvního bloku, poté první hodnota z druhého bloku až do bloku posledního. Poté jsou odebírány druhé hodnoty ze všech bloků a tak dále. Tím jsou data mezi sebou promíchána. Pokud je nějaká oblast v QR kódu poškozena (shlukové chyby), jsou chyby rozloženy do různých částí dat (náhodné chyby) a mohou být snáze opraveny. Stejným způsobem jsou „proházena“ data v korekční části.

Obě části jsou poté spojeny k sobě a zakončeny doplňkovým kódem obdobně jako na konci informační části kódu. Takto shromážděná data budou následně umístěna do matice QR kódu.

| Informační část | Korekční část | Doplňkový kód |
|---|---|---|
| 01000001 00110010 0101 … | 11011001 01011100 1000 … | 000 |

## 2.5 Umístění dat do výsledné matice

Matice QR kódu je zobrazena na obrázku 7. Je tvořena osmi prvky, z nichž některé mají povahu funkční a některá povahu informační. Následující schéma vyznačuje umístění těchto prvků:




Obrázek 7 – Části matice QR kódu

Základním prvkem obrazové matice je čtvercový modul reprezentující datový bit. Všechny grafické prvky mají velikost v násobcích velikosti modulu.

**Quiet zone**
V překladu tichá zóna, pomáhá k oddělení matice od okolí a tím i k lepšímu nalezení QR kódu v obrazové scéně. Doporučuje se šířka okraje alespoň 4 body.

**Finder patterns**
Tyto čtvercové vzory patří k nejdůležitějším částem matice. Při hledání QR kódu v obrazové scéně jsou přednostně vyhledávány. Při tvorbě matice se klade důraz na to, aby se podobný tvar neobjevil v datové části a nedošlo tak k chybné detekci pozice. Vždy jsou umístěny ve třech rozích kromě pravého dolního a mají stálou velikost – vnější rozměr je 7x7 bodů, vnitřní čtverec má rozměr 3x3 body. Podle vzájemné pozice vzorů lze obraz QR kódu natočit pod správným úhlem.

**Separators**
Slouží k vnitřnímu oddělení Finder patterns od ostatních prvků, důvodem je opět usnadnění detekce. Mají vždy bílou barvu.

**Timing patterns**
Timing patterns se nachází mezi rohovými vzory, jsou tvořeny pravidelně se střídajícími se bílými a černými moduly. Pomáhají lépe určit rozestupy jednotlivých modulů.

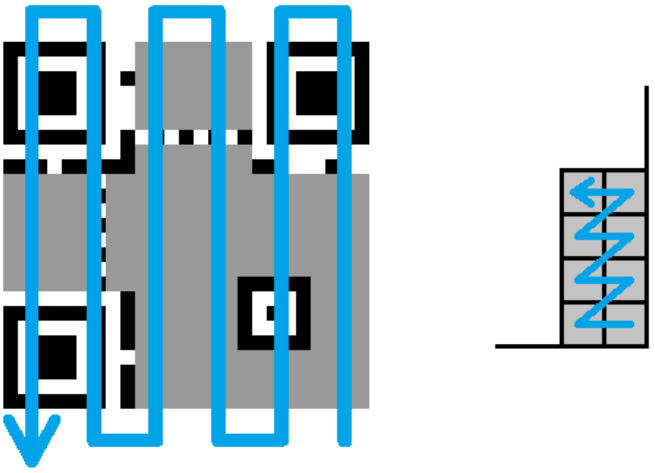
**Alignment patterns**
Jsou to malé čtvercové vzory podobné rohovým, jejich vnější rozměr je 5x5 bodů. Jsou umístěny do prostoru vyhrazeného pro data. Jejich rozmístění a počet závisí na verzi QR kódu. Standard se o jejich funkci nezmiňuje, nicméně bychom pomocí nich mohli lépe detekovat různé deformace obrazu.

**Format information**
Jedná se o vyhrazený prostor v matici, do kterého se umístí informace o úrovni ochrany proti poškození a o tzv. masce – o té bude pojednáno později.

**Version information**
Definuje prostor, ve kterém je uložena informace o verzi QR kódu. Vkládá se do matice u verze 7 a vyšší, kde se hůře rozeznává skutečný počet modulů podél hran.

**Data / Error codewords**

Po umístění všech vzorů do matice definované velikosti zbývá prostor pro umístění datové části, čili jednotlivých bitů. Ty jsou do matice umístěny podle vyhrazené cesty připomínající pohyb hada. Toto rozložení bitů má své opodstatnění z hlediska poškození dat – Informační část dat se vloží na pravou stranu matice, korekční část na levou.

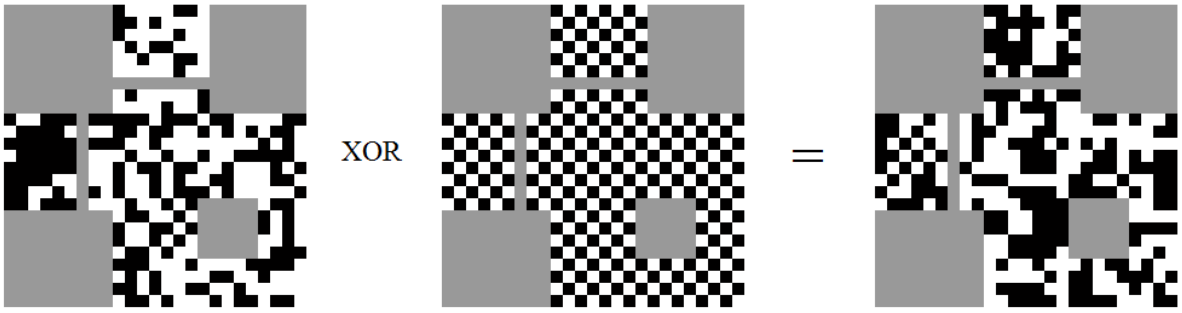


Obrázek 8 - Postup při umisťování bitů

## 2.6 Maskování dat

Maskování je proces, při kterém se datová matice upraví tak, aby měl výsledný QR kód lepší vlastnosti z hlediska dekódování. Zde mohou být problémem například větší shluky modulů stejné barvy, což znesnadňuje přesnou pozici každého z nich. Dalším problémem může být podobnost úseku v datové části matice s některým z Finder patterns a tím i nesprávné vyhodnocení pozice hledaného QR kódu ve scéně. Tyto problémy řeší právě maskování dat.

Maska je bitová matice, která se pomocí operátoru XOR vynásobí s datovou částí matice QR kódu, a vznikne tak transformovaná datová část. Princip je zobrazen na schématu:




Obrázek 9 - Maskování dat

Celkem je definováno 8 druhů masky. Postupně jsou porovnávány výsledné matice z hlediska kvality rozložení modulů, na konci vybereme masku s nejkvalitnějším výstupem a tu použijeme pro maskování.

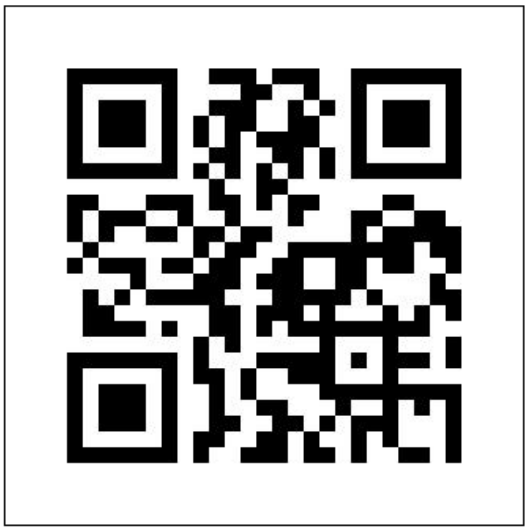
Kvalita datové části matice se posuzuje podle 4 kritérií, podle kterých jsou vypočítány tzv. trestné body a jsou započteny do celkového součtu. Čím méně trestných bodů maska získá, tím vhodnější je její použití. Kritéria jsou následující:

1. Souvislá řada nebo sloupec modulů stejné barvy - za každou řadu 5 a více stejných modulů jsou přičítány trestné body.
2. Čtvercový vzor tvořený moduly stejné barvy – platí především pro čtverce velikosti 2x2. Tímto se snažíme odstranit podobnosti se vzory Finder a Alignment patterns.
3. Řada nebo sloupec tvořený definovanou posloupností – vyhledává jistou podobnost s Finder patterns, respektive střídání modulů v tomto vzoru. Jelikož je tento jev nepříznivý, je také penalizován.
4. Poměr bílých a černých modulů – pro dekódování matice QR kódu je nevhodnější stejný počet modulů obou druhů. Čím větší odchylka vznikne, tím více je připočteno trestných bodů.

## 2.7 Kompletace

Po výběru vhodné masky, je daná maska aplikována na datovou část. Dále je třeba dopočítat Format information – vzor obsahující informace o úrovni ochrany a o použité masce. Tím získáme všechny potřebné části matice, které spojíme do jednoho celku.

V posledním kroku vytvoříme z bitové matice její grafickou podobu. Zpravidla se pro znázornění bitů užívají bílé a černé čtverce, není to však podmínkou. Lze vybrat i různé barvy, avšak důležité je, aby nebyly barevně či jasově příliš podobné.



Obrázek 10 - Výsledný QR kód

# 3 Dekódování QR kódu

Pro vyhledávání QR kódu v obrazové scéně lze použít mnoho metod zpracování obrazové informace [2, 3]. Každá z metod se zaměřuje na některý z charakteristických rysů QR kódu. Jednou z metod je například vyhledávání vzorů Finder patterns, které vymezují pozici a natočení QR kódu. K tomu se používá tzv. diskrétní vlnková transformace, která dokáže tyto vzory v obraze detekovat. Výhodou je možnost vyhledat více QR kódů v obraze, nevýhodou je výpočetní náročnost.

Metoda použitá v této práci je zaměřena na vyhledávání určité mřížky, která je základem grafické podoby matice QR kódu. Mřížka je tvořena řadou vertikálních a horizontálních přímek, jejichž počet a pozice přímo vymezují hledaný kód. Tyto přímky můžeme detekovat pomocí tzv. hranové detekce. Tato metoda je oproti předchozí jednodušší na výpočet, je ovšem zaměřena na detekci jednoho konkrétního QR kódu v obraze.

## 3.1 Vymezení požadavků pro detekci

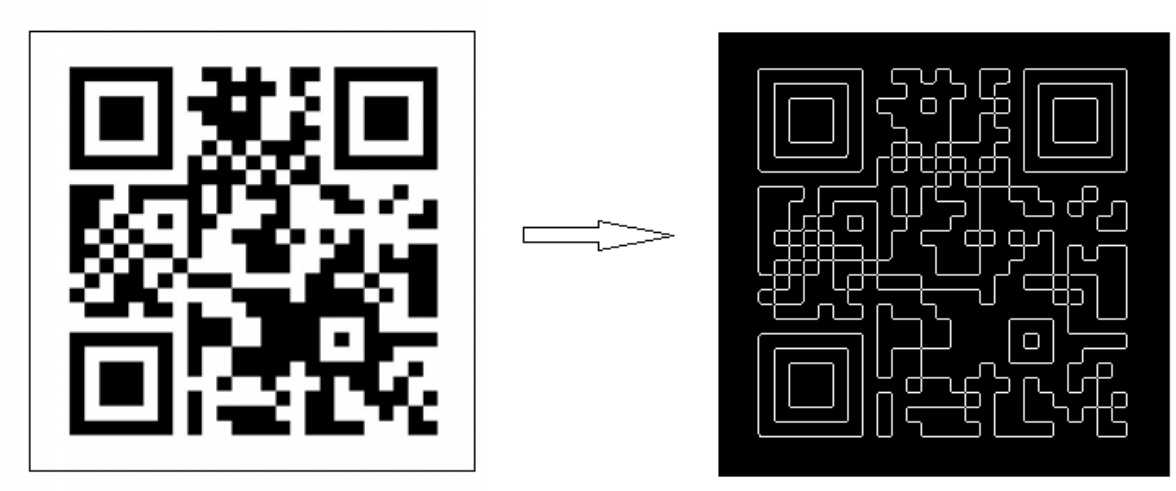
Jak už bylo zmíněno výše, prvním z požadavků je lokalizace a dekódování jednoho konkrétního QR kódu. Nejčastěji se ve zpracovávaném obraze bude vyskytovat pouze jeden a bude tedy dominantním prvkem. Algoritmus tedy nebude navržen pro nalezení více QR kódů najednou. Druhým požadavkem je, abychom kód vyhledali nezávisle na poloze v obraze a natočení okolo své osy, tedy aby nebylo nutné mít kód striktně uprostřed scény s nulovým úhlem natočení. Algoritmus nebude přizpůsoben natočení prostorovému. Třetí požadavek se týká estetického vzhledu, jednotlivé moduly nemusí být čtvercového tvaru, ani mít černobílý odstín, aby byly korektně rozlišeny. Posledním cílem je nízká výpočetní náročnost, která je dána zvolenou metodou a výše zmíněnými požadavky na detekční algoritmus.

## 3.2 Předzpracování obrazu

Před vlastní analýzou obrazové scény je potřeba upravit některé parametry toho obrazu. Jedním z nich je rozměr v pixelech – většina fotoaparátů v dnešní době poskytuje vysoké rozlišení, které může být pro další zpracování nadbytečné. Pro úspěšnou detekci je postačující rozlišení hrubší, navíc se sníží datový objem a klesne výpočetní náročnost. Z tohoto důvodu je tedy vstupní obraz zmenšen a takto je ponechán v průběhu celého algoritmu. Další úpravou je přeměna barev v obraze na odstíny šedi, což je vyžadováno při následné hranové detekci.

## 3.3 Vyhledání QR kódu ve scéně

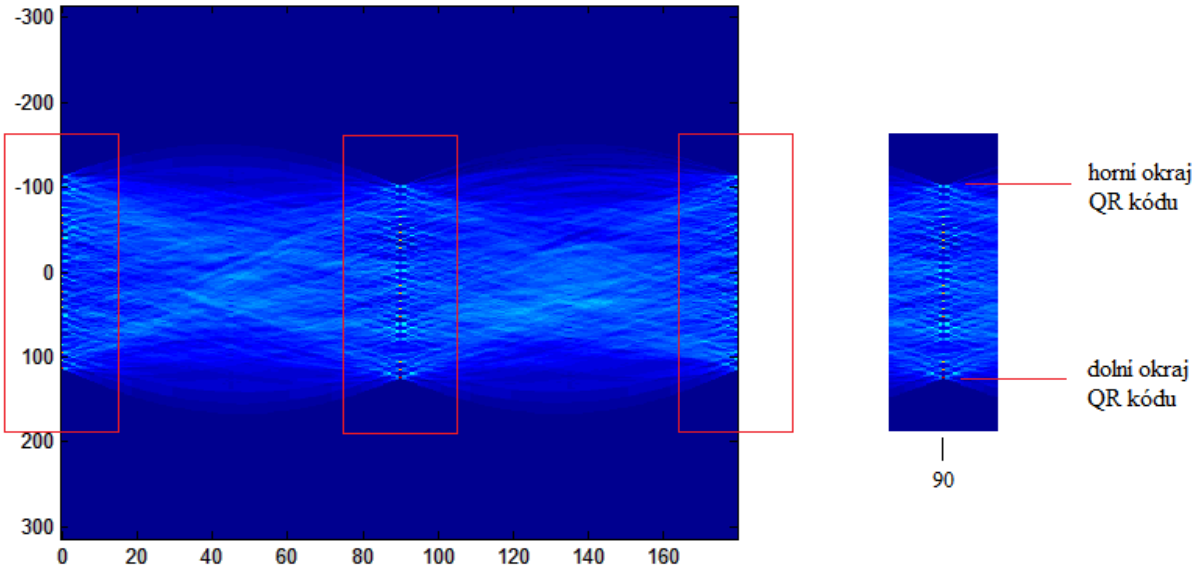
Prvním krokem při vyhledání QR kódu je detekce hran v obraze. Hrana se vyznačuje ostrým přechodem jasové složky v určité oblasti. Použitím vhodného algoritmu získáme nový obraz, který obsahuje pouze tyto hrany:



Obrázek 11 - Detekce hran

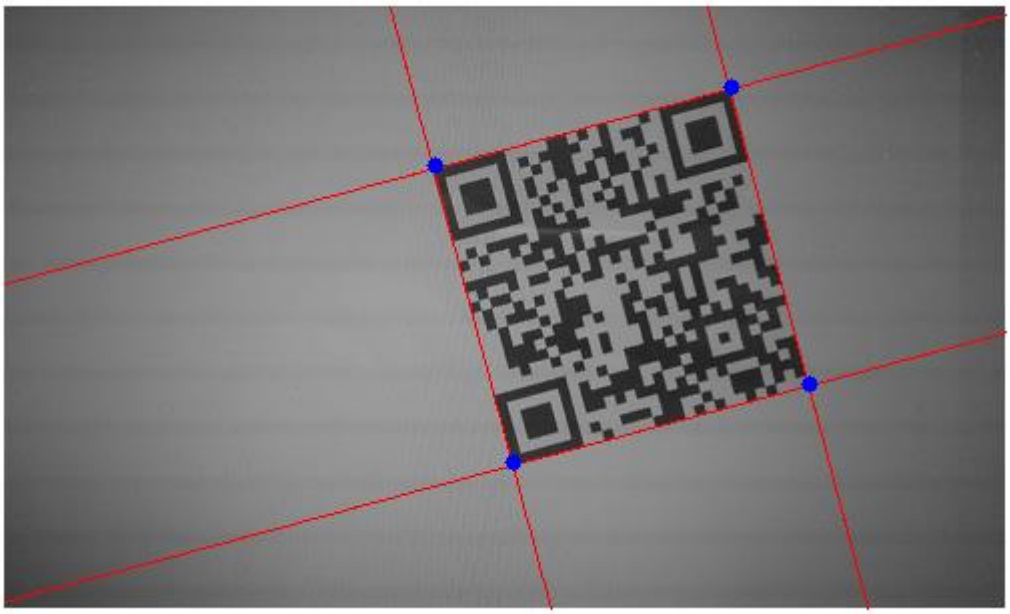
Nově získaný obraz využijeme k nalezení charakteristické mřížky, do které jsou zasazeny moduly. Ta je tvořena horizontálními a vertikálními přímkami. Pro vyhledání těchto přímek použijeme další techniku nazvanou Radonova transformace [2].

Princip Radonovy transformace je založen na postupném prokládání přímky ve všech možných polohách a natočení v obraze hranové detekce. Pokud je v místě proložení přímky mnoho „bílých“ pixelů, tím jasněji bude indikována přítomnost přímky v tomto místě. Výsledkem je tzv. Radonův prostor:




Obrázek 12 - Radonův prostor

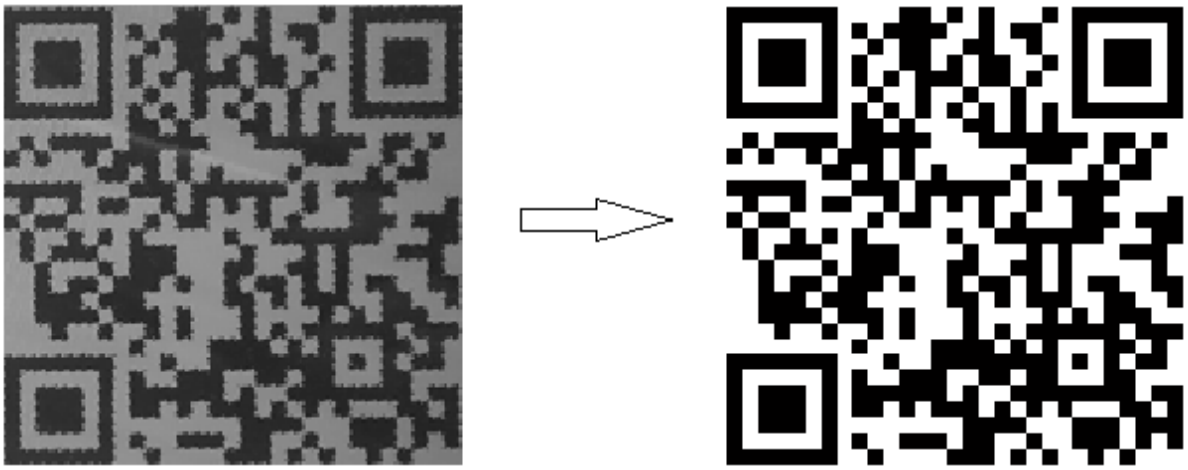
Na horizontální ose je údaj o natočení přímky okolo středu od 0 do 180 stupňů, na vertikální ose je poloha středu přímky od centrálního bodu v obraze. Každý bod v Radonově prostoru indikuje „výraznost“ přímky v obraze hranové detekce. Na pravé straně obrázku 10 je zobrazen výřez Radonova prostoru v místě 90 stupňů. Jsou zde patrné vysoké hodnoty, které jsou od sebe vzdáleny v pravidelných intervalech. Tato vzdálenost je právě odsazení přímek tvořících hledanou mřížku QR kódu. Ve výsledku tedy můžeme dopočítat přesnou polohu hledaného QR kódu v původním obraze a také jeho rozměr v počtu modulů podél hran.



Obrázek 13 - Lokalizace QR kódu v obraze

## 3.4 Rozpoznání modulů

Po nalezení QR kódu v obraze získáme výřez, se kterým budeme dále pracovat. Tento výřez bývá zašuměn, a je potřeba korektně vyhodnotit barvu jednotlivých modulů. Zde využijeme informaci počtu modulů podél hrany, ze které můžeme zjistit rozměr modulu v pixelech. Postupně poté procházíme všechny úseky obrazu a pomocí součtu hodnot pixelů v místě modulu usoudíme, který z modulů se zde nachází. Pro zjištění, která hodnota součtu náleží modulu světlejšímu či modulu tmavšímu, zjistíme stanovením průměrného odstínu, kterým je charakterizován výřez s QR kódem. Hodnota nad průměrem bude indikovat bílý modul, hodnota pod průměrem naopak černý modul. Můžeme tedy moduly převést na matici bitových hodnot.



Obrázek 14 - Rozpoznání modulů prahováním

V této fázi již máme k dispozici bitovou reprezentaci hledaného kódu.

V případě, že je QR kód nesprávně natočen, je potřeba toto natočení opravit. Tento problém lze vyřešit tak, že postupně porovnáme jednotlivé rohy s referenčním Finder pattern vzorem a budeme sledovat shodu. Roh, ve kterém jsme nalezli shodu nejmenší, umístíme do pravého dolního rohu natočením bitové matice.

## 3.5 Dekódování matice QR kódu

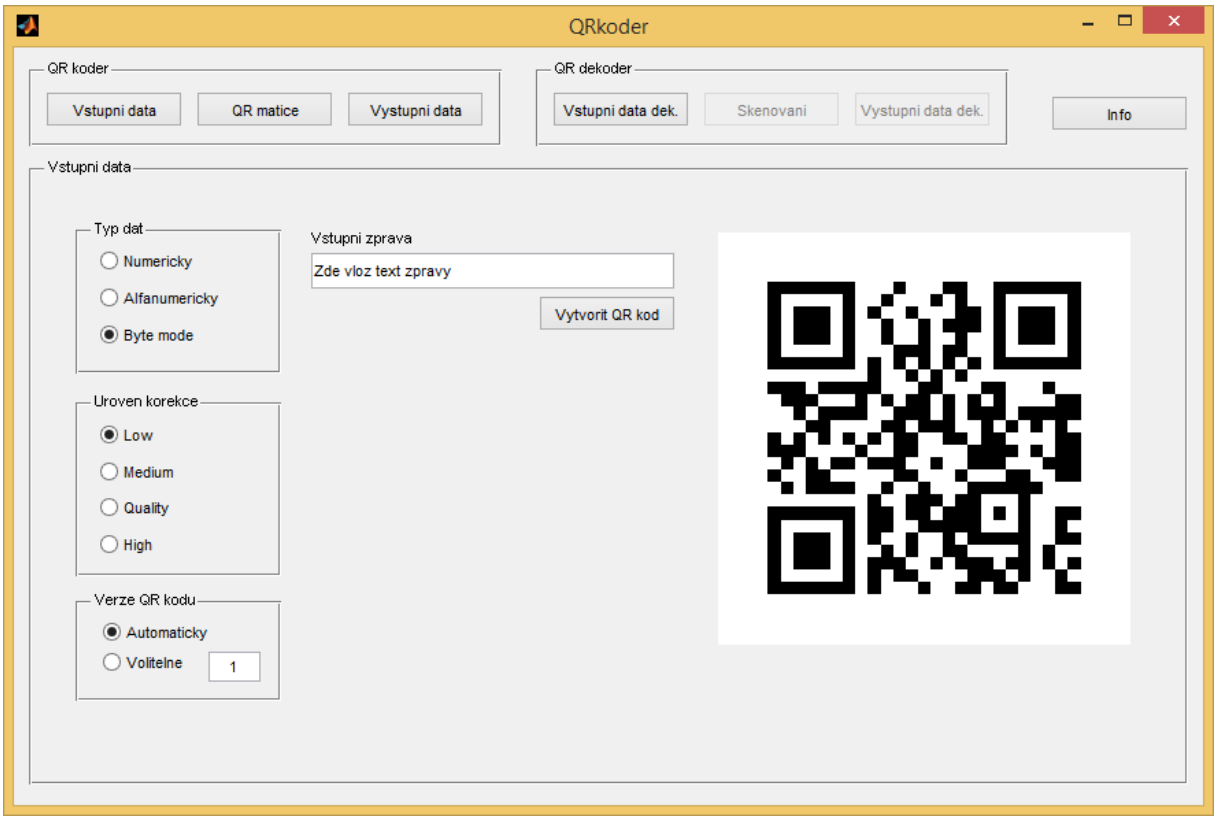
Tento proces je v principu inverzní k procesu vytváření kódu popsaného ve druhé kapitole. Pokud máme k dispozici datovou matici QR kódu, získáme postupně všechny funkční vzory a poté i samotná data. Ze vzoru Version pattern odečteme údaj o verzi QR kódu, který zkontrolujeme s rozměry matice, zda velikostně odpovídají. Dále ze vzoru Format pattern získáme informaci o úrovni korekce chyb a o indexu masky, která byla aplikována na datovou část matice. Tuto masku poté opět aplikujeme na tato data, abychom dostali jejich původní formu.

Data rozdělíme na Informační část a Korekční část. Pomocí algoritmu pro dekódování Reed Solomonových kódů detekujeme a případně opravíme chyby vzniklé v dekódovací části při snímání QR kódu ve scéně. Může se stát, že chyb bude více, než je možné opravit, v tom případě se chyby projeví i na původní zprávě, kterou QR kód obsahuje. V praktické části je uveden názorný příklad se zavedením chyb do matice QR kódu.

Informační část na začátku nese hlavičku, ve které jsou informace o typu dat a o počtu znaků ve zprávě. Pomocí nich dokážeme informační část opět převést do původní podoby.

# 4 Praktická část

Na základě teoretických poznatků o vytvoření a následném dekódování QR kódů byla v programu MATLAB sestavena aplikace s názvem „QR Kodér“. Tato aplikace názorně ukazuje oba tyto procesy v jednotlivých krocích a usnadňuje tak pochopení celé problematiky. Uživatelské prostředí je rozděleno do šesti sekcí, z nichž první tři jsou zaměřeny na kódování a poslední tři na dekódování, viz následující obrázek:




Obrázek 15 – Aplikace: Vstupní data

## 4.1 Popis jednotlivých sekcí

### Vstupní data

V horní části okna jsou tlačítka pro přepínání mezi jednotlivými sekcemi. Sekce z Obrázku 13, která se objeví při spuštění aplikace, se nazývá „Vstupní data“ a obsahuje komponenty pro nastavení vstupních parametrů procesu kódování.

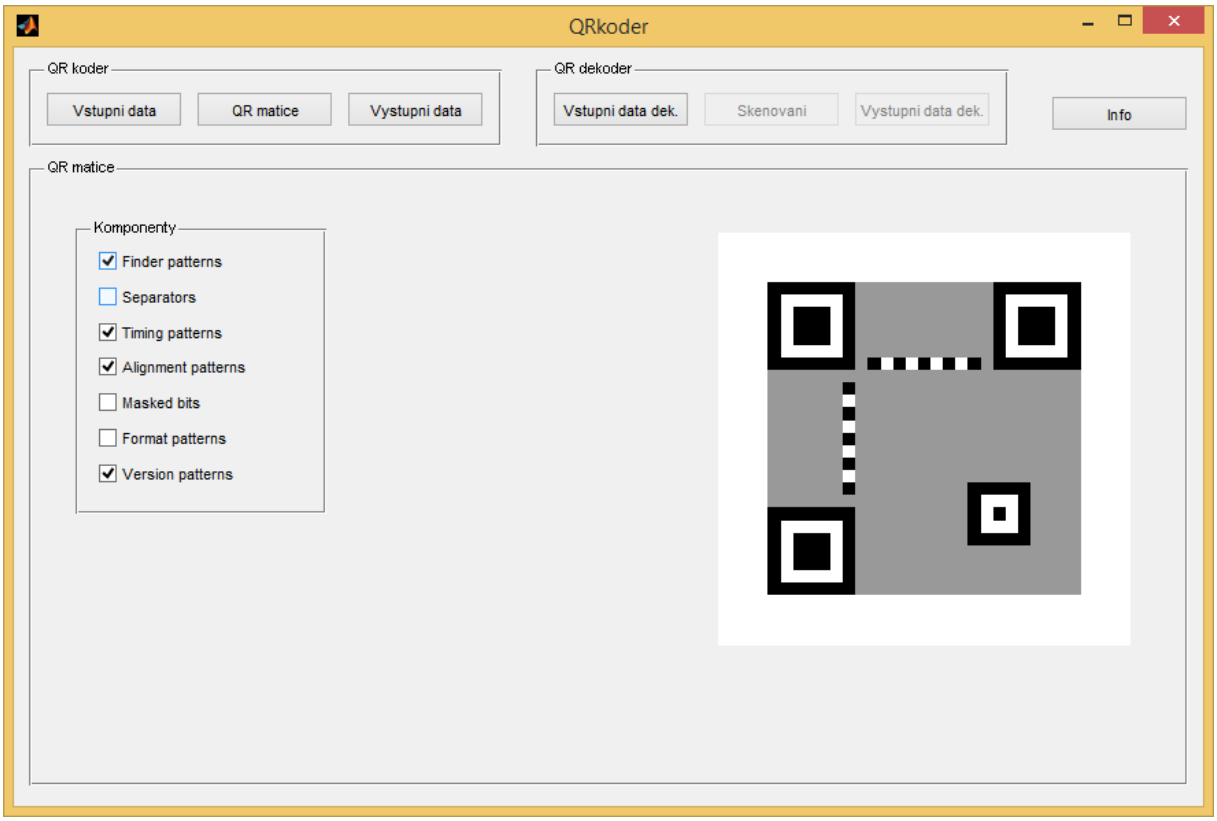
Při vytváření vlastního QR kódu postupujeme takto:

Do pole s názvem „Vstupní zpráva“ zadáme text, který chceme zakódovat. Podle typu znaků použitých ve zprávě vybereme na levé straně okna typ dat. Při najetí myši na jednotlivé volby se zobrazí nápověda se znaky, které může text obsahovat. Poté vybereme úroveň korekce dat – viz Tabulka 1. Pokud chceme vytvořit QR kód o vyšší kapacitě než je potřeba, můžeme zvolit v rámečku „Verze QR kódu“ volbu „Volitelné“ a nastavit vlastní hodnotu. Po nastavení všech parametrů stiskneme tlačítko „Vytvořit QR kód“.

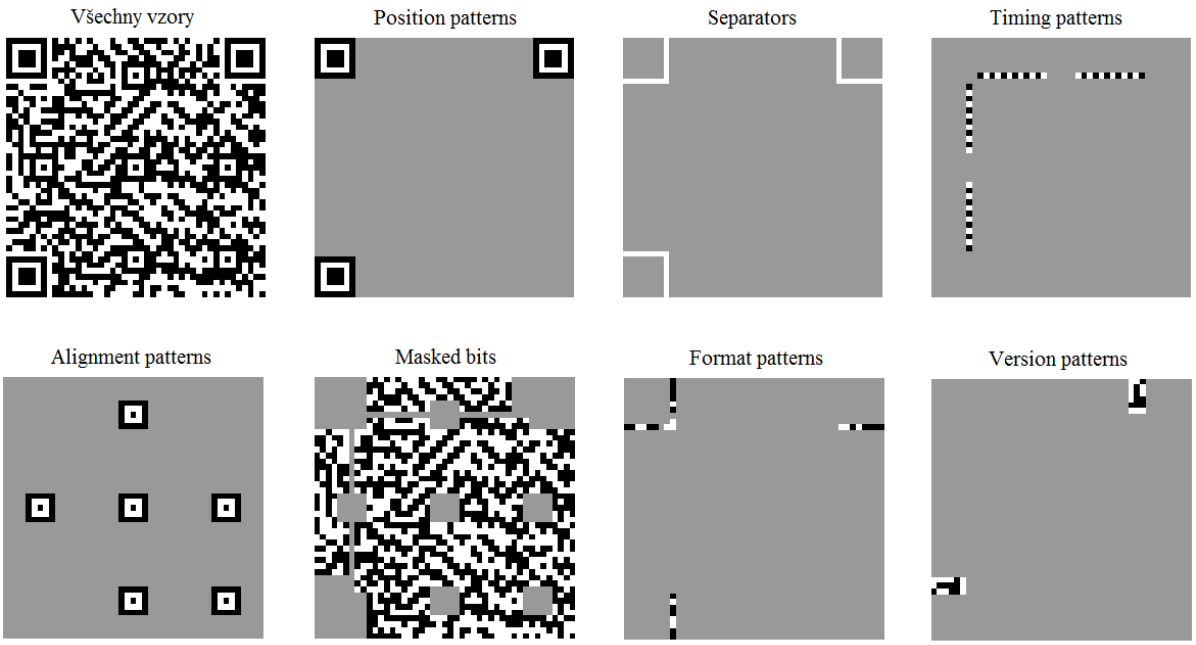
Na pravé straně se zobrazí výsledný QR kód. Také se zpřístupní sekce „QR matice“ a „Výstupní data“.

### QR matice

Tato sekce obsahuje velice názornou pomůcku pro grafické zobrazení jednotlivých částí matice QR kódu popsaných na Obrázku 5. V levé části okna jsou zaškrtávací pole pro výběr částí, které chceme vykreslit do grafu na pravé straně okna. Takto vypadá ukázka výběru:



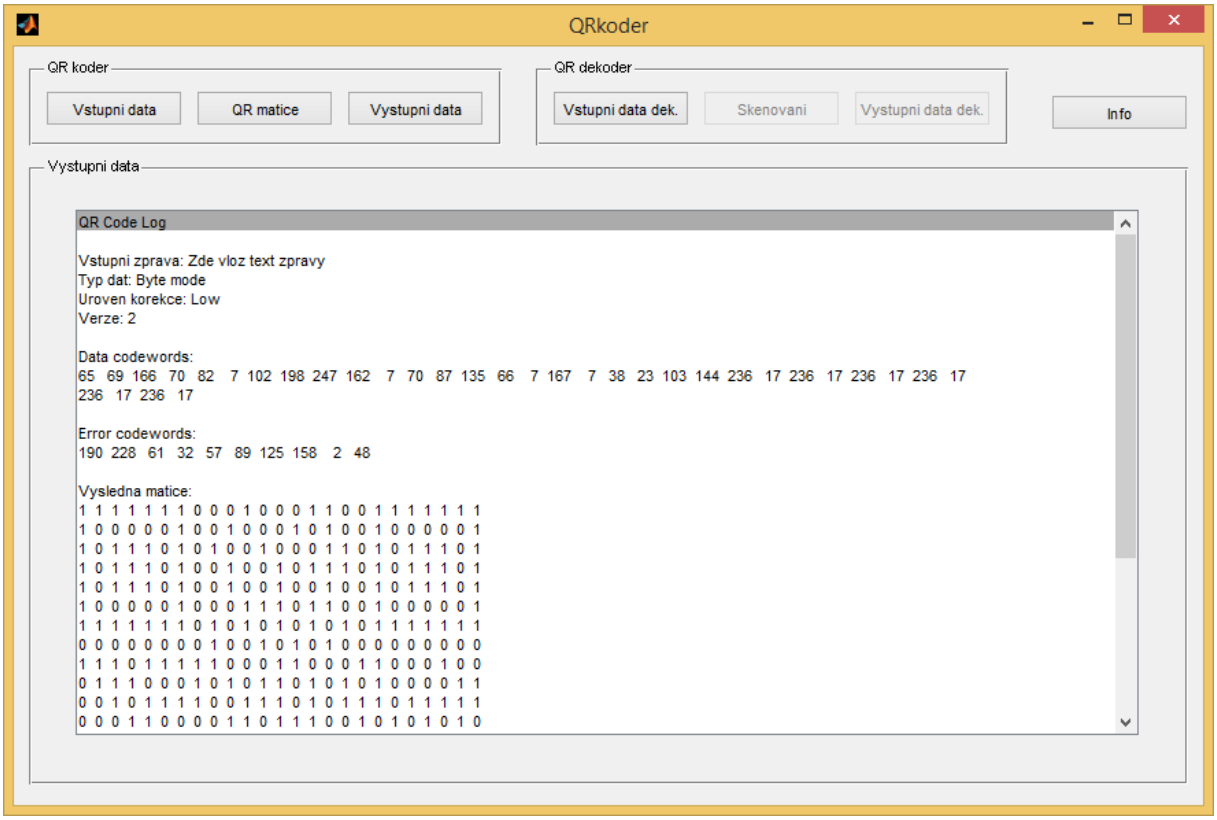

Obrázek 16 – Aplikace: QR matice




Obrázek 17 - Ukázka dekompozice matice

### Výstupní data

Zde jsou do textového pole vypsány informace o vstupních parametrech, dále zakódovaná data v podobě Data codewords a Error codewords. Zde můžeme sledovat, jak se mění poměr počtu Data a Error codewords v závislosti na volbě úrovně korekce. Nakonec je vypsána i samotná matice QR kódu v binární podobě.




Obrázek 18 – Aplikace: Výstupní data

### Vstupní data dekodéru

Tato sekce slouží k načtení obrazové scény, ve které bude QR kód vyhledáván. Podporovány jsou všechny základní formáty jako JPEG, PNG či BMP. Nejprve vybereme příslušný obrázek pomocí tlačítka „Načíst obrázek“, v pravé části okna se zobrazí náhled. Poté spustíme proces detekce a dekódování pomocí tlačítka „Dekódovat“. Pod tímto tlačítkem je umístěn text, který informuje o průběhu zpracování. V případě, že se nepodaří nalézt QR kód v obraze, vypíše se zpráva „Chyba při zpracování obrazu“. Pokud nastane chyba při dekódování dat z binární matice získané po detekci, zobrazí se zpráva „Chyba při dekódování dat“.


Obrázek 19 – Aplikace: Vstupní data dekodéru

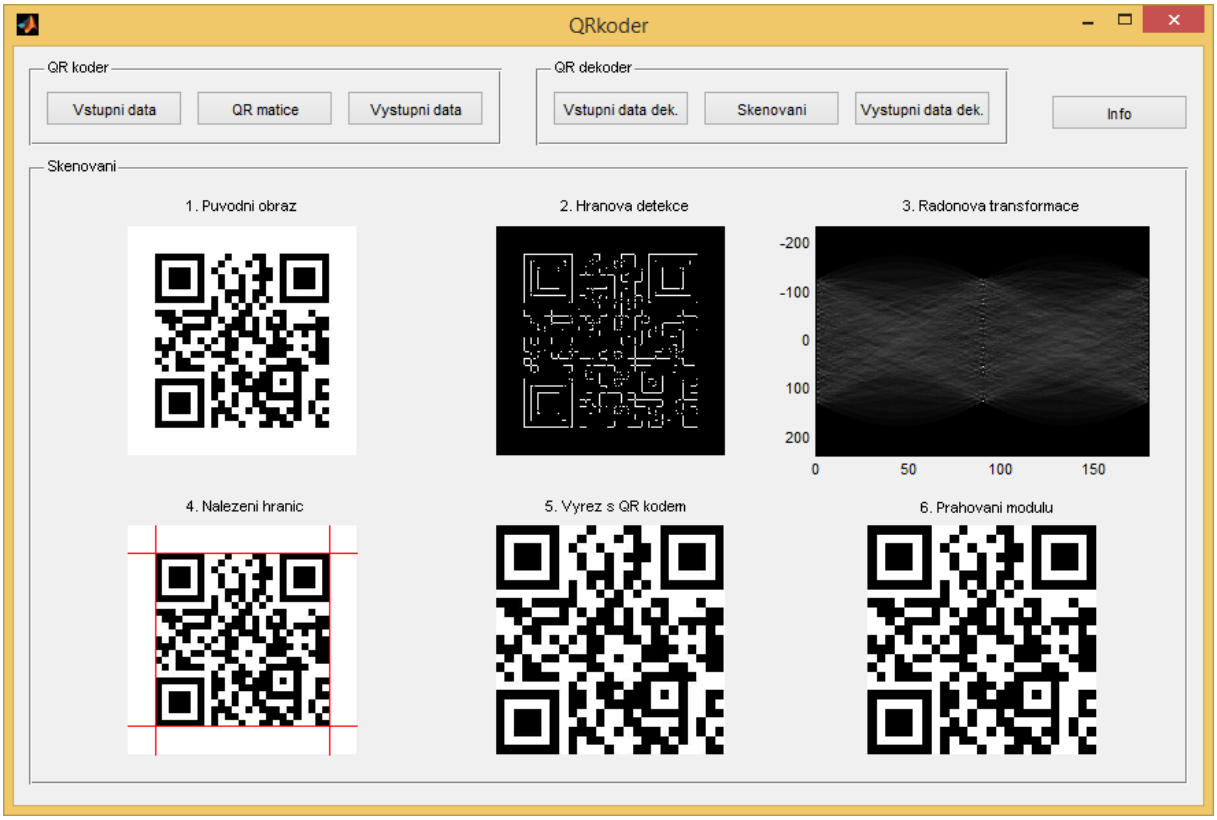
Pokud nenastal při zpracování obrazu a dat žádný problém, vypíše se zpráva „Dokončeno“. Následně se zpřístupní poslední dvě sekce „Skenování“ a „Výstupní data dekodéru“, kde jsou zaznamenány výsledky celého procesu.

**Skenování**

Sekce Skenování zaznamenává průběh detekování QR kódu v obraze. Obsahuje 6 grafů chronologicky očíslovaných podle postupu detekce QR kódu v obraze, který je popsán ve druhé kapitole. Těmito grafy jsou:

1. Původní obraz – zmenšená verze uživatelem zvoleného vstupního obrázku,
2. Hranová detekce – obraz po transformaci hranovým detektorem,
3. Radonova transformace – zobrazuje Radonův prostor získaný z předchozího procesu,
4. Nalezení hranic – vyznačuje prostor, ve kterém algoritmus předpokládá QR kód,
5. Výřez s QR kódem – výřez s QR kódem z původního obrazu,
6. Prahování modulů – rozlišení modulů aplikací metody prahování na předchozí obraz.

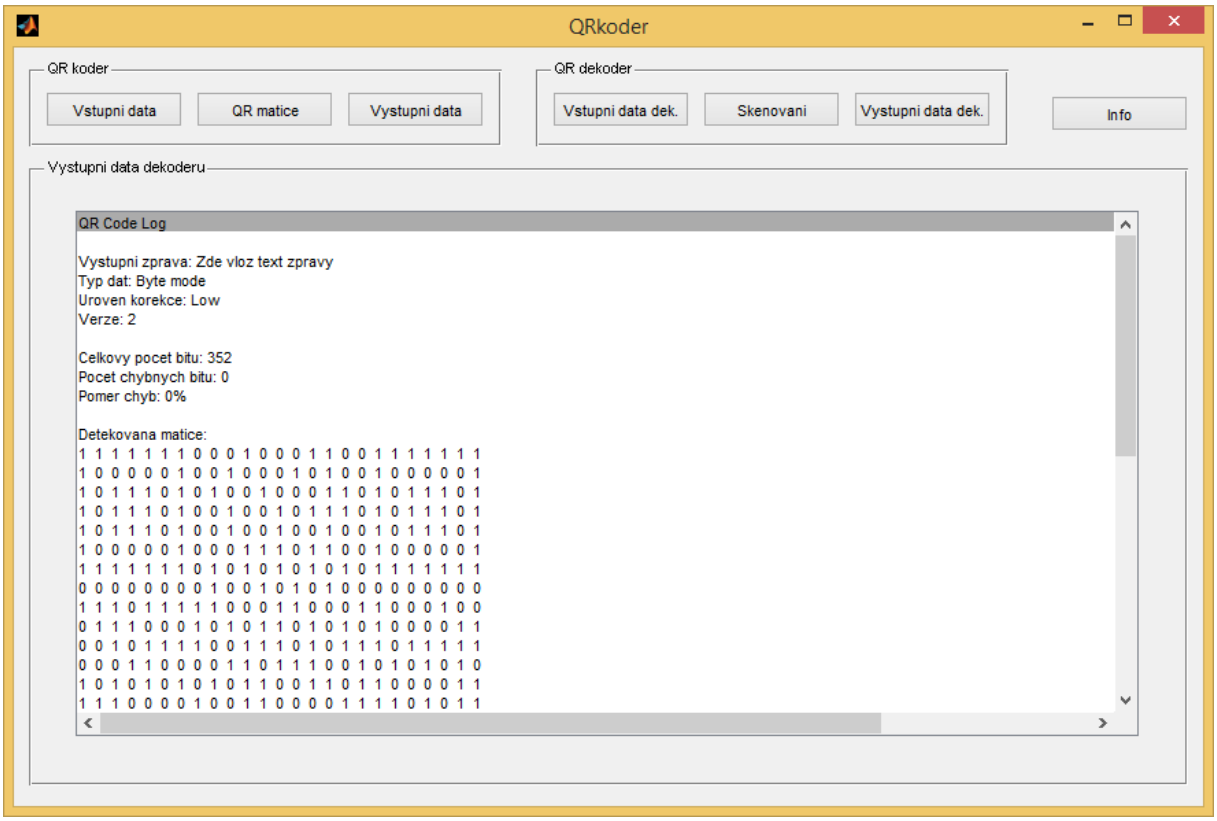
Na dalším obrázku jsou znázorněna ukázka zpracování obrázku obsahujícího QR kód:




Obrázek 20 – Aplikace: Skenování

### Výstupní data dekodéru

Podobně jako v sekci Výstupní data týkající se nově vytvořeného QR kódu, i zde je umístěno textové pole. V něm jsou vypsány výsledky z fáze dekódování binární matice, která navazuje na fázi detekce (skenování).




Obrázek 21 – Aplikace: Výstupní data dekodéru

Obsahem výstupních dat je především původně zakódovaná zpráva, dále typ dat, úroveň korekce a použitá verze. Dále jsou zde statistické údaje o poškozených bitech, respektive těch, které byly ve fázi detekce špatně vyhodnoceny. Vypsán je celkový počet bitů tvořících data, počet bitů poškozených a poměr mezi nimi udávající procentuální poškození QR kódu. Informace o poškození dat jsou získány algoritmem pro opravu Reed Solomonových kódů. Následuje výpis binární matice QR kódu, data rozdělená na Data codewords a Error codewords. Na konci textu je uveden zápis o korekci RS kódů pro každý blok dat.

## 4.2 Import a Export dat

Data pro import a export jsou umístěna v kořenovém adresáři ve složkách „Input“ a „Output“. Testovací obrázky pro import do dekodéru se nachází ve složce „Input“. Export dat aplikace provádí automaticky. Výpis kodéru ze sekce „Výstupní data“ je ve složce „Output“ uložen v textovém souboru pod názvem „Zápis o kódování“. Výpis dekodéru je ve stejné složce uložen pod názvem „Zápis o dekódování“. Zde je také uložen samotný QR kód vytvořený v aplikaci a je uložen pod názvem „QR code“ a lze jej přímo analyzovat v dekodéru pro kontrolu bezchybnosti kódování.

# 5 Praktické ukázky

## 5.1 Příklady kódování

V této části textu budou porovnány vlastnosti QR kódu v závislosti na vstupních parametrech.

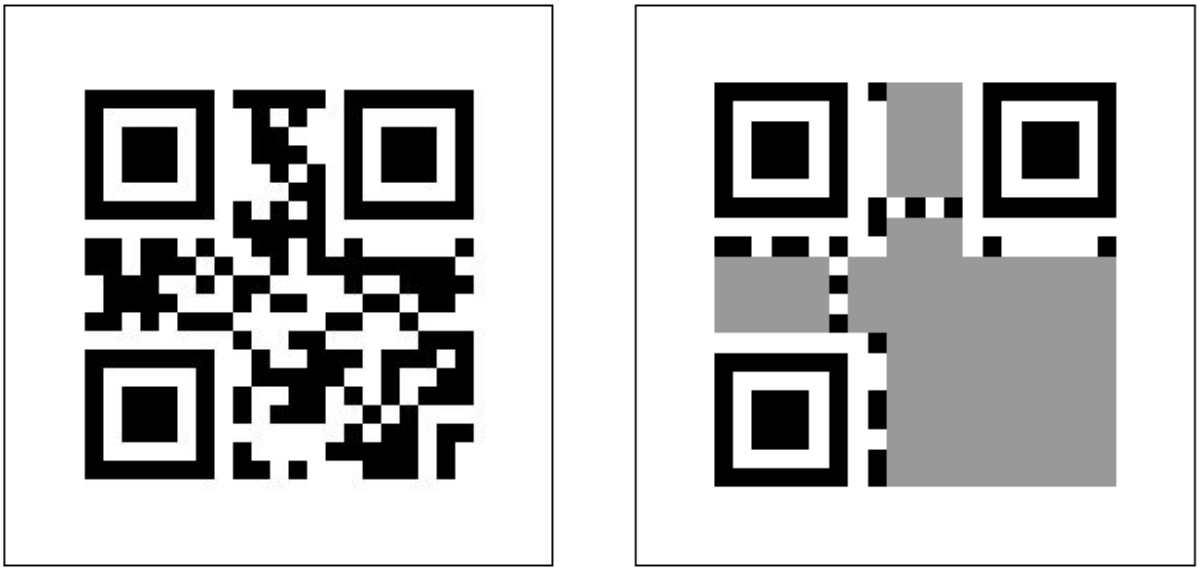
**QR kód verze 1**

Prvním příkladem je vytvoření QR kódu verze 1, tedy nejmenší velikosti. Jeho kapacita vystačí ve většině případů. Vstupní parametry jsou zaznamenány v tabulce:

Tabulka 3 – QR kód verze 1: Vstupní parametry

| | |
|---|---|
| Vstupní zpráva | „ROCK N ROLL“ |
| Typ dat | Alfanumerický |
| Úroveň korekce | Low |
| Verze | Automaticky |

Jelikož je zpráva tvořena pouze velkými písmenky latinské abecedy, můžeme zvolit alfanumerický typ dat. Mohli bychom zvolit také Byte mode, zbytečně bychom ale snížili datovou kapacitu QR kódu. Vytvořením QR kódu získáme výstupní grafy:



Obrázek 22 – QR kód Verze 1: Výstupní grafy

Na Obrázku 22 vidíme vlevo QR graf se zakódovanou vstupní zprávou. Napravo je graf ze sekce „QR matice“, kde jsme nechali odstranit datovou část matice nazvanou „Masked bits“ (data v bitové podobě, na které již byla aplikována maska).

V sekci „Výstupní data“ jsou do textového pole vypsány tyto informace:

```
QR Code Log

Vstupní zpráva: ROCK N ROLL
Typ dat: Alfanumerický
Úroveň korekce: Low
Verze: 1

Data codewords:
32   92  215   70   25  175   55  196  213   64  236   17  236   17  236   17
236   17  236

Error codewords:
3   37   50  163  169  115  188

Výsledná matice:
1  1  1  1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  1  1  1
1  0  0  0  0  0  1  0  0  1  0  1  0  0  1  0  0  0  0  0  1
1  0  1  1  1  0  1  0  0  1  1  0  0  0  1  0  1  1  1  0  1
1  0  1  1  1  0  1  0  0  1  1  1  0  0  1  0  1  1  1  0  1
1  0  1  1  1  0  1  0  0  0  1  0  1  0  1  0  1  1  1  0  1
1  0  0  0  0  0  1  0  0  0  0  1  1  0  1  0  0  0  0  0  1
1  1  1  1  1  1  1  0  1  0  1  0  1  0  1  1  1  1  1  1  1
0  0  0  0  0  0  0  0  1  1  1  1  1  0  0  0  0  0  0  0  0
1  1  0  1  1  0  1  0  0  1  1  0  1  0  1  0  0  0  0  0  1
1  1  0  1  1  1  0  1  0  0  1  0  1  1  1  1  1  1  1  1  0
0  1  1  1  0  0  1  0  1  1  0  0  0  0  1  0  0  1  1  1  1
0  1  1  1  1  0  0  0  1  0  1  1  0  0  0  1  1  0  1  1  0
1  1  0  1  0  1  1  1  0  0  0  0  0  1  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  1  0  0  1  1  0  0  0  0  0  1  1  1
1  1  1  1  1  1  1  0  0  1  0  0  1  1  1  0  1  1  1  0  1
1  0  0  0  0  0  1  0  0  1  1  1  1  1  0  0  1  0  0  1  1
1  0  1  1  1  0  1  0  1  0  0  1  1  0  1  0  1  0  1  1  1
1  0  1  1  1  0  1  0  1  0  1  1  1  0  0  1  0  1  0  0  0
1  0  1  1  1  0  1  0  0  0  0  0  0  0  1  0  1  1  0  1  1
1  0  0  0  0  0  1  0  1  0  0  0  0  1  1  1  1  1  0  1  0
1  1  1  1  1  1  1  0  1  1  0  1  0  0  0  1  1  1  0  1  0
```

Zde si můžeme všimnout především datové části v podobě Data a Error codewords. Jelikož jsme zvolili nejnižší úroveň korekce, je počet opravných slov oproti datovým méně jak poloviční.

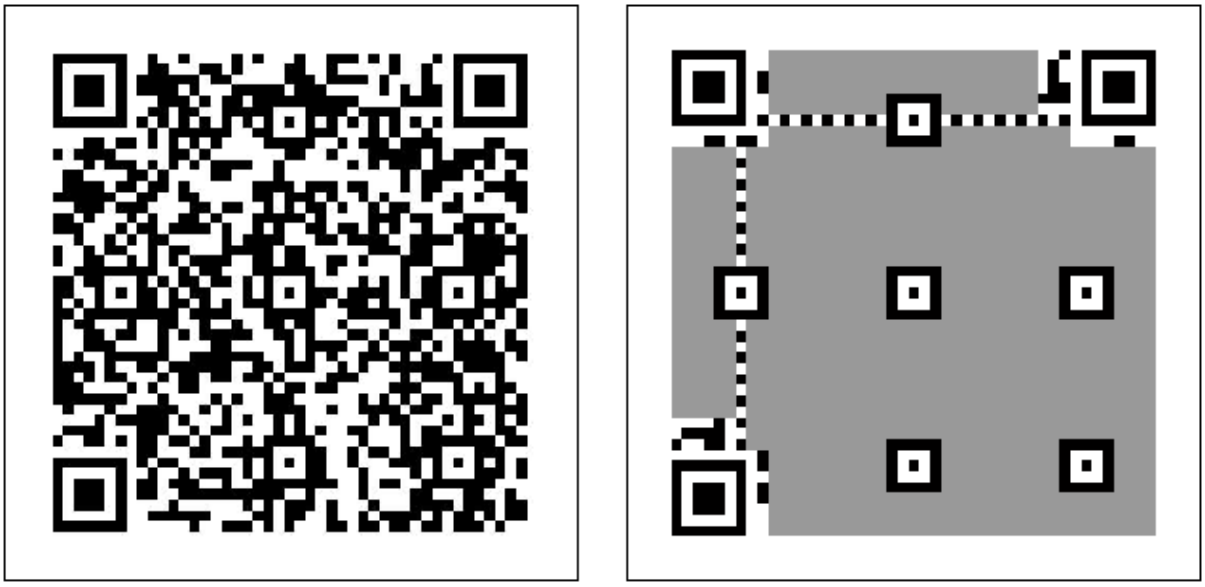
**QR kód verze 7**

Druhým příkladem je vytvoření QR kódu vyšší verze a úrovně zabezpečení proti chybám, jeho popis bude stručnější než v předchozím příkladu:

Tabulka 4 – QR kód verze 7: Vstupní parametry

| Vstupní zpráva | „Jakoby dlouhý text…“ |
|---|---|
| Typ dat | Byte mode |
| Úroveň korekce | High |
| Verze | Volitelné - 7 |

Na výstupu dostaneme tyto grafy:



Obrázek 23 – QR kód verze 7: Výstupní grafy

Při porovnání snímku 23 s předchozím si v prvé řadě všimneme zvětšených rozměrů QR kódu. Dále můžeme na pravém grafu vidět, že zde přibyly „Alignment patterns“ a „Version patterns“, které se vyskytují právě u vyšších verzí.

Nyní se podíváme do textového výstupu kodéru:

```
QR Code Log

Vstupní zpráva: Jakoby dlouhý text...
Typ dat: Byte mode
Úroveň korekce: High
Verze: 7

Data codewords:
65   84  166   22  182  246   39  146    6   70  198  247   86  143  210    7
70   87  135   66  226  226  224  236   17  236   17  236   17  236 17  236
17  236   17  236   17  236   17  236   17  236   17  236   17  236   17  236
17  236   17  236   17  236   17  236   17  236   17  236 17  236   17  236
17  236
```

```
Error codewords:
139  216   85  232  255  120   16  231  161  183  173  237  117  114  211
64  231  100   32  102  152  115  211   66  125  233   11  230  237    5  121
209  123   12   61  203  122   40   12    6  106  206  164   56  204  199
50  253  155   20  214  174  181  108  202  247  125  123  113  148  156  128
230   95  101   64  206  180  148  222   33   90   75  145  232  181  216
130  162  108  134   18   65   26  127  217  122  247   42   99   55   22   31
207   79   19  231    1   85  185  236   30   88  130  233   55  232    4
18  239   94   12  145    2   62  110  123  211   76   97  194  142   83  241
153  138  101  201  116    3
```

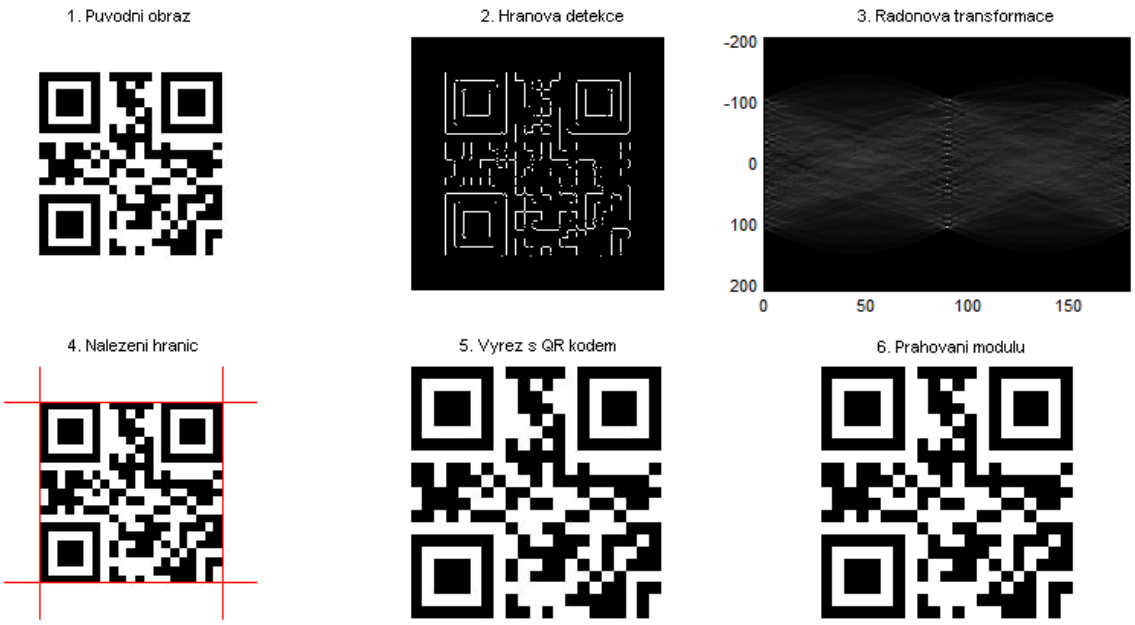
V tomto příkladu je názorně vidět vliv změny úrovně korekce na poměr datových a korekčních slov – korekčních slov je zhruba dvakrát více.

## 5.2 Příklady dekódování

V této části budou uvedeny příklady, které odhalí kvality a nedostatky algoritmu pro obrazovou detekci s ohledem na vymezené požadavky (3.1). Dále bude testována opravitelnost dat v závislosti na úrovni jejich poškození.

### Základní obrazová scéna

V následujícím příkladu vyzkoušíme detekci QR kódu na jednoduchém obrázku, vybereme proto obrázek QR kódu verze 1. Matice v obraze je uprostřed a má nulový úhel natočení, scéna neobsahuje žádné náhodné jevy. Vstupní obraz necháme skenovat:




Obrázek 24 – Základní obrazová scéna: Skenování

Po zhlédnutí výstupních grafů je zřejmé, že byl QR kód detekován bez sebemenších problémů. Zápis o dekódování vypadá takto:

```
QR Code Log

Výstupní zprava: ROCK N ROLL
Typ dat: Alfanumerický
Úroveň korekce: Low
Verze: 1

Celkový počet bitů: 208
Počet chybných bitů: 0
Poměr chyb: 0%

Detekovaná matice:
1  1  1  1  1  1  1  0  1  1  1  1  1  0  1  1  1  1  1  1  1
1  0  0  0  0  0  1  0  0  1  0  1  0  0  1  0  0  0  0  0  1
1  0  1  1  1  0  1  0  0  1  1  0  0  0  1  0  1  1  1  0  1
1  0  1  1  1  0  1  0  0  1  1  1  0  0  1  0  1  1  1  0  1
1  0  1  1  1  0  1  0  0  0  1  0  1  0  1  0  1  1  1  0  1
1  0  0  0  0  0  1  0  0  0  0  1  1  0  1  0  0  0  0  0  1
1  1  1  1  1  1  1  0  1  0  1  0  1  0  1  1  1  1  1  1  1
0  0  0  0  0  0  0  0  1  1  1  1  1  0  0  0  0  0  0  0  0
1  1  0  1  1  0  1  0  0  1  1  0  1  0  1  0  0  0  0  0  1
1  1  0  1  1  1  0  1  0  0  1  0  1  1  1  1  1  1  1  1  0
0  1  1  1  0  0  1  0  1  1  0  0  0  0  1  0  0  1  1  1  1
0  1  1  1  1  0  0  0  1  0  1  1  0  0  0  1  1  0  1  1  0
1  1  0  1  0  1  1  1  0  0  0  0  0  1  1  0  0  1  0  0  0
0  0  0  0  0  0  0  0  1  0  0  1  1  0  0  0  0  0  1  1  1
1  1  1  1  1  1  1  0  0  1  0  0  1  1  1  0  1  1  1  0  1
1  0  0  0  0  0  1  0  0  1  1  1  1  1  0  0  1  0  0  1  1
1  0  1  1  1  0  1  0  1  0  0  1  1  0  1  0  1  0  1  1  1
1  0  1  1  1  0  1  0  1  0  1  1  1  0  0  1  0  1  0  0  0
1  0  1  1  1  0  1  0  0  0  0  0  0  0  0  1  0  1  1  0  1  1
1  0  0  0  0  0  1  0  1  0  0  0  0  1  1  1  1  1  0  1  0
1  1  1  1  1  1  1  0  1  1  0  1  0  0  0  1  1  1  0  1  0

Data codewords:
32   92  215   70   25  175   55  196  213   64  236   17  236   17  236   17
236   17  236

Error codewords:
3   37   50  163  169  115  188

Zápis o opravě chyb:

Vstupní polynom
32   92  215   70   25  175   55  196  213   64  236   17  236   17  236   17
236   17  236    3   37   50  163  169  115  188
Syndromy
0  0  0  0
Polynom neobsahuje chybu.
```

Zápis je tvořen dvěma oddíly – informace o dekódovaných datech a zápis o opravě chyb. Data, která jsme po dekódování získali, se podle očekávání shodují s námi zadanými parametry v tabulce 4. Celkový počet bitů je 208, chybných bitů je 0%.

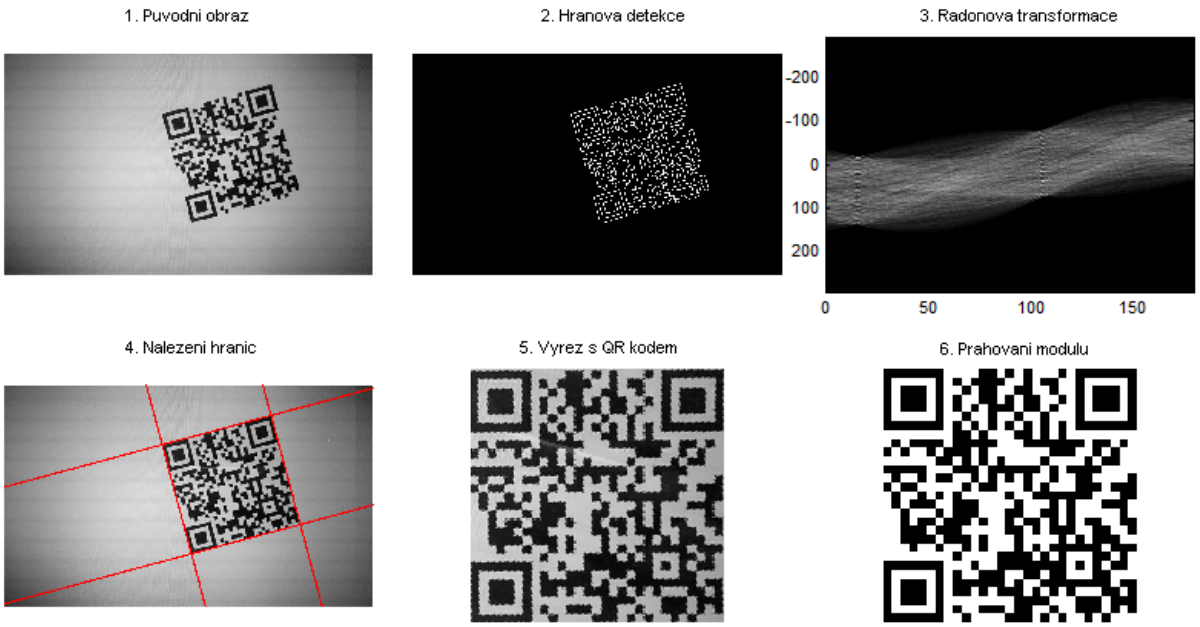
Zápis o opravě chyb podává informace o tom, jak algoritmus postupuje při hledání a opravě RS kódů. Postupně bere jeden blok Data codewords, jeden blok Error Codewords (bloky zde nejsou graficky odděleny) a spojí v jeden polynom, který tvoří vstup do dekodéru RS kódů. V případě, že polynom neobsahuje chybu, je vypsán text „Polynom neobsahuje chybu“. Pokud obsahuje opravitelný počet chyb, dekodér vypíše výstupní polynom, který je těchto chyb zbaven. Je-li polynom příliš poškozen, vypíše se text „Data jsou příliš poškozena“ a zpráva obsažená v datech bude chybně interpretována.

**Reálná obrazová scéna**

Jako vstup použijeme snímek z digitálního fotoaparátu, QR kód byl vyfocen na obrazovce monitoru. Záměrně byl také posunutý mimo střed obrazu a mírně natočený.

Obrázek 25 – Reálná obrazová scéna

Na rozdíl od základní scény, zde se náhodně mění úroveň jasu, kvůli šumu jsou přítomny vyšší frekvence. Nicméně QR kód je stále dobře čitelný. Hrany v obraze jsou stále dostatečně výrazné, hlavní roli při dekódování bude mít fáze prahování modulů:




Obrázek 26 – Reálná obrazová scéna - Skenování

Čtení QR kódu proběhlo bez problému, moduly byly správně vyhodnoceny. Algoritmus pro detekci je odolný proti posunu a natočení matice v obraze.

**Prostorové natočení**

Prostorové natočení matice v obraze vykazuje nové vlastnosti především v Radonově obraze. Přímky získané z hranové detekce nejsou natočeny pod stejným úhlem, viz obrázek 27:


Obrázek 27 – Prostorové natočení QR kódu

Algoritmus není pro prostorové natočení přizpůsobený, lze jej ale upravit i pro tento případ naprogramováním složitějšího vyhledávání dominantních bodů v Radonově prostoru.

**Zavedení chyb do matice QR kódu**

RS kódy jsou primárně zaměřeny na opravu shlukových chyb, tj. poškozených bitů blízko u sebe. Naopak při rozprostřených chybách vykazuje silné nedostatky. Nejprve vyzkoušíme první případ. Vytvoříme QR kód s úrovní korekce „Quality“ a pozměníme hodnotu 9 bitů. Na obrázku níže jsou vyznačeny červenou barvou:

Obrázek 28 – Zavedení shlukových chyb

Textový výstup programu:

```
Celkový počet bitů: 208
Počet chybných bitů: 9
Poměr chyb: 4.3%

Data codewords:
32   92  215   70   25  175   55  196  213   64  236   17  236
Error codewords:
116  150  178  187  164  158  167  158   53   85  182   81   67
Vstupní polynom
32   92  215   70   25  175   55  196  213   64  236   17  236  116  147  242
187  164  157   87  158   53   85  182   81   67
Syndromy
182  104  245  174   78   67   39   45  187  238  114   66
Lokátor chyb
1   92  147   82   78    0    0
Pozice chyb
6   7  10  11
Hodnoty chyb
240    3   64    5
Výstupní polynom
32   92  215   70   25  175   55  196  213   64  236   17  236  116  150  178
187  164  158  167  158   53   85  182   81   67
```

Proces opravy RS kódů je rozdělen do několika fází (princip a matematický zápis je podrobně popsán ve standardu). Počet chybných bitů je 9, což odpovídá počtu chyb zavedených do matice. Protože jsou ale data rozdělena po 8 bitech do bytů, počet chybných slov (codewords) v rámci RS kódů je pouze 4!

Podle teorie je počet opravitelných slov maximálně polovina počtu opravných slov v polynomu. Zde je počet opravných slov 13, poškozeno může být až 6 slov z celkových 26, neboli 48 bitů. V tom případě by byl poměr chyb 23%, což odpovídá úrovni korekce „Quality“ podle tabulky 1. V praxi by se takový počet chybných bitů rozprostřel do více než 6 slov a chyby by nešly odstranit.

Můžeme se o tom přesvědčit v druhé ukázce, kdy rozprostřeme stejný počet chyb do větší plochy:

Obrázek 29 – Zavedení rozprostřených chyb

Textový výstup programu:

```
Celkový počet bitů: 208
Počet chybných bitů: neznámý
Poměr chyb: neznámý
Data codewords:
32   92  215   70   25  175   55  196  212   68  204   17  236
Error codewords:
116  151  178  187  164  150  167  190   53   81   54   81   67
Vstupní polynom
32   92  215   70   25  175   55  196  212   68  204   17  236  116  151  178
187  164  150  167  190   53   81   54   81   67
Syndromy
136  243   61  121   94  172   55   52  112   93  134   62
Lokátor chyb
1  181  239  209  238  184  129
Pozice chyb
Data jsou příliš poškozena.
```

Jelikož bylo chybami zasaženo více než 6 slov, nelze data opravit.

# Závěr

Cílem diplomové práce bylo vytvořit softwarovou aplikaci umožňující kódování a dekódování QR kódů. Aplikace je určena pro výuku, důraz byl proto kladen na názorné vysvětlení jednotlivých kroků při práci s QR kódy a jejich vlastností.

V teoretické části byl popsán proces vytváření QR kódů od vstupních parametrů přes zakódování dat až po sestavení binární matice reprezentující výsledný QR kód. Vyzdviženy byly základní principy kódování vstupní zprávy, naopak byly vynechány podrobné informace a tabulky potřebné k sestavení celého algoritmu a případě potřeby jsou dohledatelné v literatuře.

V další části textu byl vysvětlen jeden ze způsobů vyhledávání QR kódu v obrazové scéně pomocí metod pro zpracování obrazové informace. Použitá metoda se v praxi dosud nevyskytla, před napsáním teorie. Její použití však bylo řádně testováno a vede ke správné detekci QR kódu. Dále byl objasněn postup při dekódování binární matice získané detekcí. Prakticky jde o proces opačný k procesu kódování. Stručný popis dekódování Reed Solomonových kódů a vliv zavedení chyb do datové části byl přesunut do praktické části, jelikož je vysvětlení na konkrétním příkladu mnohem více pochopitelné.

V rámci praktické části byl vytvořena aplikace v programu MATLAB. Je rozdělena na dvě základní části – kodér a dekodér. Aplikace umožňuje vytvořit QR kód podle zadaných parametrů, dále zobrazí výpis o procesu kódování, kde jsou souhrnně vypsány zadané parametry, dále data rozdělená na informační a korekční část a také binární matice vytvořeného QR kódu. Program byl navíc doplněn o graf zobrazující jednotlivé vzory použité v této matici – tzv. „patterns“. Lze pomocí tohoto grafu například zobrazit QR kód bez samotných dat či čtvercových vzorů v rozích. Následuje dekodér, který chronologicky zpracovává QR kód nejprve v obrazové scéně, jeho detekci je výřez s QR kódem převeden do binární matice a následně dekódován. Celý proces detekce je v aplikaci graficky znázorněn v šesti krocích. Před samotným dekódováním zprávy ukryté v datech jsou případě poškození dat při detekci tyto data opravena pomocí korekčních kódů. Algoritmus byl vytvořen ručně bez funkcí nabízených knihovnou. Díky tomu jsou k nahlédnutí jednotlivé kroky nalezení chyb v datech a jejich opravy. Poté jsou data zpětně převedena na původní zprávu. Podobně jako u kodéru, i zde je zobrazen výpis obsahující parametry QR kódu, detekovanou binární matici, data rozdělená na informační a korekční část a na konci také celý proces opravy dat.

Na příkladech byl především testováno vyhledávání QR kódu v obraze, jelikož je tato fáze nejnáročnější. Aplikace si poradila s velkým množstvím obrazových scén, jak ideálních (bez rušivých prvků) a reálných a plně postačuje pro výukové účely. Dále byly testovány zabezpečovací vlastnosti ochrany dat QR kódů. Podle výsledků závisí opravitelnost dat na prostorovém rozložení chyb v matici QR kódu. Údaje uvedené v tabulce 1 jsou jen výjimečně dosažitelné.

Všechna výstupní data jsou exportována do složky „Output“ v kořenovém adresáři programu.

## Použitá literatura

[1] ADÁMEK, Jiří: *Kódování*. Státní nakladatelství technické literatury (SNTL), Praha 1989.

[2] KLÍMA, Miloš. *Zpracování obrazové informace*. Vyd. 1. V Praze: České vysoké učení technické, 1996, iii, 177 s. ISBN 80-01-01436-3.

[3] DOBROVOLNÝ, Martin: *Přednášky z předmětu Zpracování obrazu*, Univerzita Pardubice.

[4] Oficiální stránky QR kódu [online] [cit. 9. 5. 2016] Dostupné na WWW: <http://www.qrcode.com/en/>.

[5] Standard ISO 18004:2015 [online] [cit. 9. 5. 2016] Dostupné na WWW: <https://www.iso.org/obp/ui/#iso:std:62021:en>.

[6] Čárové kódy [online] [cit. 9. 5. 2016] Dostupné na WWW: <https://cs.wikipedia.org/wiki/%C4%8C%C3%A1rov%C3%BD_k%C3%B3d >.

[7] Data matrix [online] [cit. 9. 5. 2016] Dostupné na WWW: <http://en.wikipedia.org/wiki/Data_Matrix>.

[8] Aztec code [online] [cit. 9. 5. 2016] Dostupné na WWW: <http://cs.wikipedia.org/wiki/Azt%C3%A9ck%C3%BD_k%C3%B3d>.

[9] Shot code [online] [cit. 9. 5. 2016] Dostupné na WWW:< http://en.wikipedia.org/wiki/ShotCode>.

[10] Microsoft Tag [online] [cit. 9. 5. 2016] Dostupné na WWW: <https://en.wikipedia.org/wiki/High_Capacity_Color_Barcode >.

[11] MyQRV [online] [cit. 9. 5. 2016] Dostupné na WWW: < https://www.myqrv.com/ >.

[12] QR Code Tutorial [online] [cit. 9. 5. 2016] Dostupné na WWW: <http://www.thonky.com/qr-code-tutorial/>.

## Přílohy

Aplikace včetně zdrojových kódů a text Diplomové práce ve formě PDF jsou k dispozici na přiloženém CD.