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

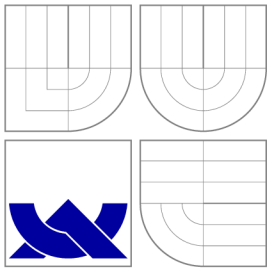
POKROČILÉ METODY DITHERINGU BAREVNÝCH OBRAZŮ, REDUKCE BAREVNÉHO PROSTORU NA N-PRVKOVOU PALETU

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

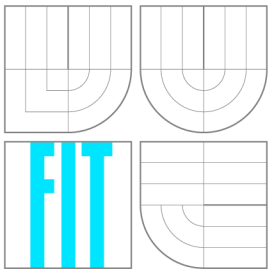
AUTOR PRÁCE
AUTHOR

Bc. PETR ŠTĚPÁN

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

POKROČILÉ METODY DITHERINGU BAREVNÝCH OBRAZŮ, REDUKCE BAREVNÉHO PROSTORU NA N-PRVKOVOU PALETU

COLOR IMAGES ADVANCED DITHERING METHODS, REDUCTION OF COLOR SPACE ON N-DIMENSIONAL PALETTE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. PETR ŠTĚPÁN

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. Ing. PŘEMYSL KRŠEK, Ph.D.

BRNO 2011

Abstrakt

Tato práce se zabývá sumarizací a vzájemným porovnáním pokročilých metod ditheringu obrazu používaných při redukci barevného prostoru na n-prvkovou paletu. Cílem je vytvořit seznam všech základních metod včetně jejich vylepšení a modifikací se zaměřením na kvalitu výstupního obrazu a jeho maximální vizuální věrnost s obrazem původním. Další částí této práce je analýza jednotlivých metod za pomoci vlastní porovnávací metodiky a návrh vlastního algoritmu ditheringu kombinujícího jejich výhody.

Abstract

This thesis describes summarizing and comparsion of advanced dithering techniques used to reduce the image color space on n-dimensional palette. The goal is to create a set of basic methods including theirs enhancements and modifications focusing on the output image quality and maximum visual fidelity to the original image. Another part of this thesis is to analyze this methods using own comparsing method and design new dithering algorithm combining their advantages.

Klíčová slova

Dithering, barva, barevná paleta, redukce barevného prostoru, obrázek.

Keywords

Dithering, color, color palette, reduction of color space, image.

Citace

Petr Štěpán: Pokročilé metody ditheringu barevných obrazů, redukce barevného prostoru na n-prvkovou paletu, diplomová práce, Brno, FIT VUT v Brně, 2011

Pokročilé metody ditheringu barevných obrazů, redukce barevného prostoru na n-prvkovou paletu

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana doc. Ing. Přemysla Krška, Ph.D.

.....
Petr Štěpán
18. května 2011

Poděkování

Chtěl bych tímto poděkovat vedoucímu své diplomové práce, panu doc. Ing. Přemyslu Krškovi, Ph.D. za odborné vedení a cenné rady při tvorbě této práce.

Obsah

Kapitola 1

Úvod

Cílem této práce je sumarizace existujících metod ditheringu obrazu a jejich vzájemné porovnání. Pro toto porovnávání je také nezbytné navrhnout určitou metodiku, pomocí které se bude vyhodnocovat jejich kvalita. V neposlední řadě chci na základě vlastností těchto metod navrhnout vlastní vylepšení jejich funkčnosti.

V kapitole Teoretický rozbor jsou nejprve vysvětleny základní pojmy jako dithering a histogram. Následuje důkladný popis pokročilých metod pro redukci barevného prostoru za pomocí ditheringu z teoretického hlediska, a to včetně existujících modifikací těchto algoritmů. Mezi popsané metody patří prahování, náhodný rozptyl, maticový rozptyl a distribuce chyby. Poslední sekce je věnována aplikaci ditheringu při redukci barevného prostoru u vícekanálových obrazů.

Kapitola Návrh a implementace obsahuje návrh nového algoritmu ditheringu, porovnávací metodiky a aplikace, která bude tuto metodiku implementovat.

V části Výsledky budu prezentovat dosažené výsledky.

V závěru poté shrnu dosažené poznatky a nastíním další možné oblasti výzkumu tohoto tématu.

Kapitola 2

Teoretický rozbor

V této části práce se zaměřím zejména na teoretickou část problematiky, po nastínění základních pojmů popíšu všechny základní používané metody ditheringu, včetně příkladů použití. Posléze uvedu některá specifika týkající se ditheringu v souvislosti s barevnými paletami.

2.1 Digitální obrazy

V této práce se věnuje problematice redukce barevného prostoru digitálního obrazu s využitím metod ditheringu ke zlepšení kvality výsledného obrazu. Tato kapitola má za cíl objasnit základní pojmy týkající se této oblasti.

Digitální obraz je číselná reprezentace dvojrozměrného obrazu za pomoci rastrové mřížky hodnot, tzv. pixelů (bodů). Každý z těchto bodů reprezentuje hodnotu světelné intenzity na odpovídajícím místě původního obrazu. Tímto způsobem jsme schopni vyjádřit diskrétní matici světelných intenzit v jednotlivých bodech obrazu, která se dá snadno uložit a dále zpracovávat.

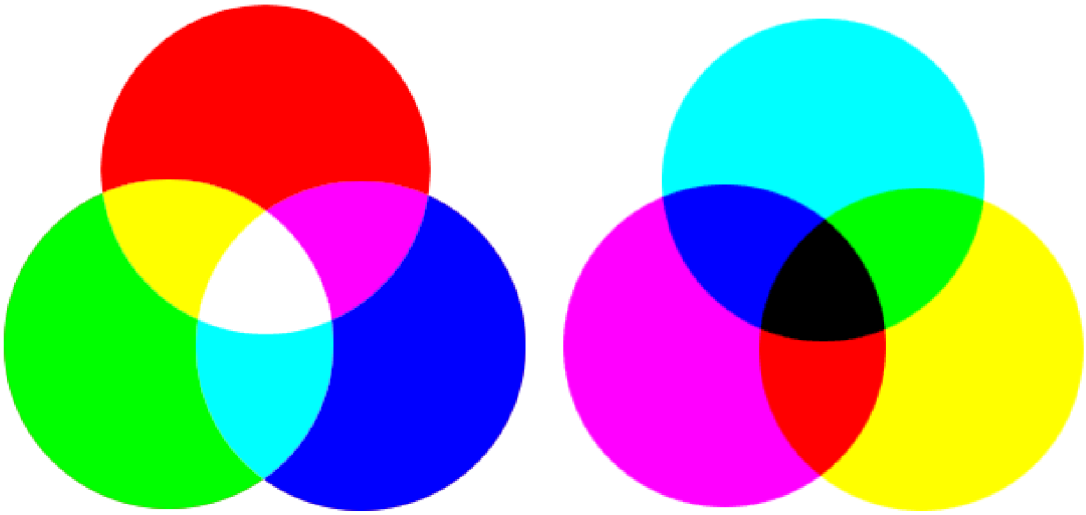
2.1.1 Míchání barev

Abychom mohli v počítačové grafice uchovávat a zpracovávat také barevnou informaci, je potřeba zavést určité zjednodušení. Barva je subjektivní pojem založený na lidském vnímání viditelného spektra elektromagnetického záření. Tento spojitý světelný signál složený z mnoha frekvencí je pro účely počítačové interpretace nutno diskretizovat a modelově popsat, aby bylo možné jej zpětně reprodukovat.

Za těmito účely byly vyvinuty určité barevné modely. Slouží k uchovávání barevné informace a k práci s ní. Tyto modely jsou založeny na existenci tzv. základních barev, jejichž slučováním vznikají další barevné odstíny. Existují dva základní principy míchání barev, aditivní (sčítací) a subtraktivní (odečítací).

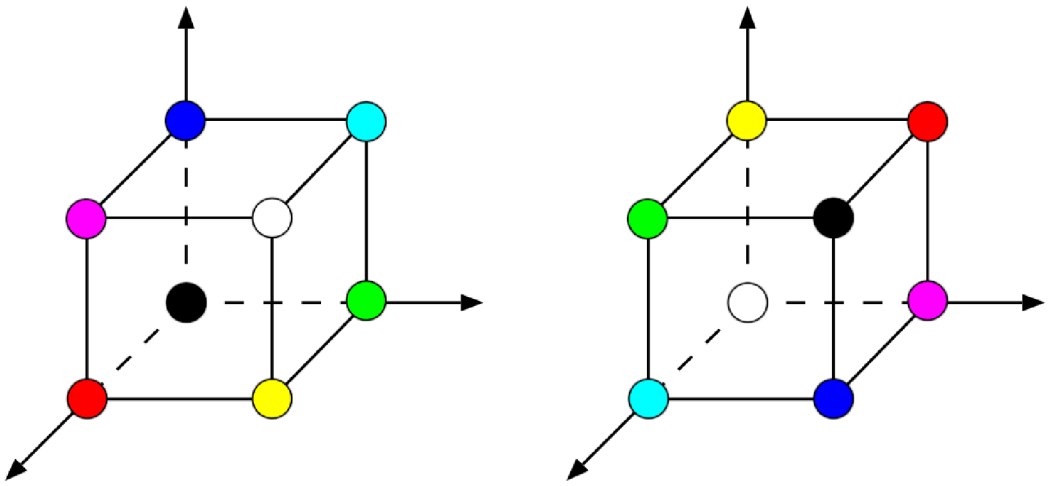
Aditivní barevné míchání

Jedná se o model míchání barev založený na práci se světelnými zdroji (například monitor), které vyzařují světelnou energii. Jeho princip je takový, že složením elementárních barev tato energie stoupá a zvyšuje se celková intenzita výsledné barvy. Použitými základními barvami tu jsou červená (red), zelená (green) a modrá (blue), což vychází z funkce lidského



(a) Aditivní (b) Subtraktivní

Obrázek 2.1: Míchání barev



(a) RGB (b) CMY

Obrázek 2.2: Reprezentace barevných modelů RGB a CMY

oka, kde jsou zvláštní receptory pro každou z těchto tří barev. Smícháním všech základních barev při nejvyšších intenzitách vzniká bílá barva, jak je vidět na obrázku 2.1a.

Základním barevným modelem využívajícím tento princip je RGB. Pro uložení barevné informace používá právě tři základní barevné kanály. Složením světelných intenzit všech těchto tří složek pak vzniká konkrétní odstín. RGB model odpovídá jednotkové krychli, kde osy x, y a z odpovídají červené, zelené a modré barvě (obrázek 2.2a).

Subtraktivní barevné míchání

Tento způsob pracuje s odraženým světlem, typicky například s odrazem pigmentu vytištěného dokumentu. V tomto případě se s každou další složkou celková intenzita světla snižuje vlivem pohlcování světelného paprsku. Základními barvami pro skládání jsou azurová (cyan), purpurová (magenta) a žlutá (yellow). Tyto barvy jsou doplňkové k červené, zelené a modré a jejich složením vzniká barva černá, jak ilustruje obrázek 2.1b.

Na tomto principu pak funguje další z barevných modelů - CMY, případně jeho varianta CMYK, která přidává k základním barvám ještě barvu černou. Podobně jako u RGB se dá tento model vyjádřit pomocí jednotkové krychle (obrázek 2.2b), pouze na jejích třech osách jsou základný barvy CMY.

2.1.2 Barevná paleta

Velmi důležitým pojmem v souvislosti digitálních obrazů je také barevná paleta. Je to v podstatě množina barevných osdtínů, kterých může daný pixel nabývat. Palety mají návaznost na použitý barevný model (RGB, CMYK, atd.), jímž jsou poté uložené hodnoty reprezentovány.

Pro ilustraci se budu věnovat pouze paletám barevného modelu RGB, obsahujícím tři barevné kanály - červený, zelený a modrý. Každý pixel má daný rozsah možných hodnot, do kterých se uloží informace o intenzitě daného barevného kanálu. Pokud je tento rozsah například roven jednomu bitu, pak pro každý z kanálů je možno uchovávat dvě hodnoty 1 a 0. Složením těchto tří kombinací pak vzniká 8 možných barevných odstínů, kterých je tato paleta schopna dosáhnout. Na jeden kanál tedy připadá jeden bit, kanály jsou tři, mluvíme tedy o 3-bitové RGB paletě. Při aplikaci ditheringu pak zpracováváme obraz po jednotlivých barevných složkách r, g a b [11] pomocí zvoleného algoritmu, výsledný obraz je pak zpětně složen z těchto tří složek.

Máme také možnost nepoužít uniformní rozložení prostoru pro všechny kanály, čehož se využívá například v paletě 3-3-2. Zde je použit menší rozsah pro modrý kanál, oproti zbývajícím dvěma. Je to kvůli tomu, aby se uložení jednoho pixelu vešlo do osmi bitů. Na modrý kanál je lidské oko nejméně citlivé, proto je menší rozlišení informace v oblasti modrých barev přijatelné.

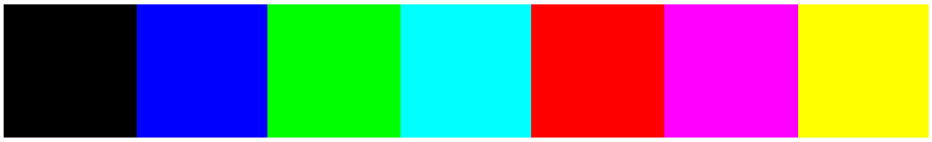
Pro zjednodušení a lepší demonstrativnost se v následujících částech popisujících jednotlivé algoritmy zaměřím zejména na jednobitovou černobílou paletu, protože stejné principy jsou pak aplikovány na jednotlivé barevné složky při použití vícekanálové palety. Podrobněji se budu barevnými paletami a jejich využitím při aplikaci metod ditheringu zabývat v sekci 2.3.

RGB palety používané v této práci jsou na obrázku 2.3.

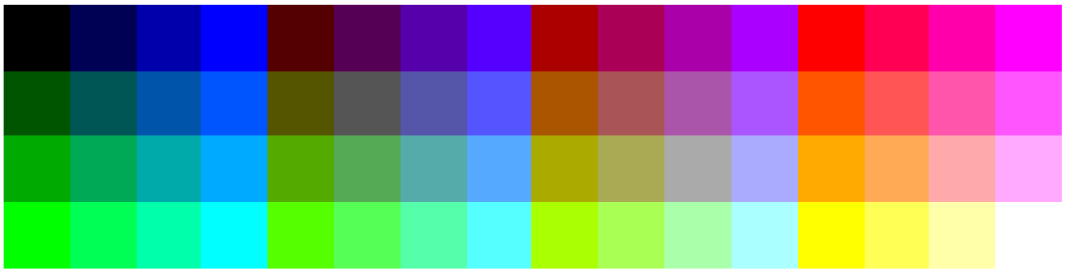
2.1.3 Histogram

Jedním z častých způsobů popisu obrazu je jeho reprezentace ve formě histogramu. Jedná se o statistický útvar zobrazující pravděpodobnostní rozložení určité veličiny ve formě grafu, ve kterém jsou na jedné ose znázorněny jednotlivé diskrétní třídy pravděpodobnosti a na ose druhé jejich relativní četnost.

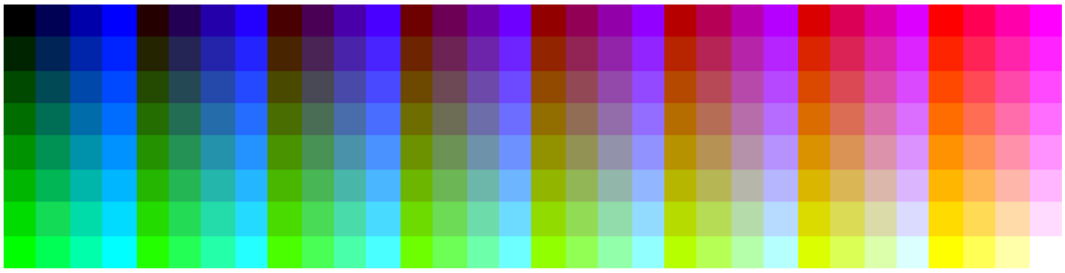
V našem případě budu rozlišovat histogram jasový a histogram jednotlivých barevných složek. Pravděpodobnostní třídy na ose x jsou diskrétní hodnoty jasu (resp. barevného kanálu) od 0 až po maximální hodnotu danou velikostí použité palety (např. pro 8-bitovou paletu hodnota 255). Na ose y je pak vynesen relativní počet pixelů, které mají stejnou



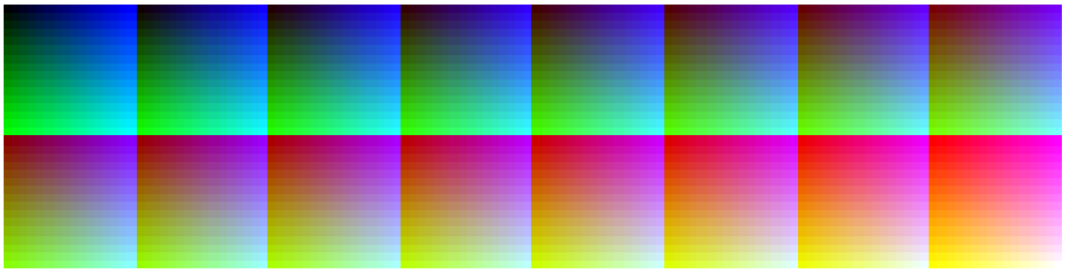
(a) 3-bitová



(b) 6-bitová



(c) 3-3-2



(d) 12-bitová

Obrázek 2.3: Různé druhy RGB palet

hodnotu. Ukázka těchto algoritmů je vidět na obrázku 2.4. Ideální obraz má všechny hodnoty jasu rovnoměrně rozložené, čím více hodnot se blíží levému okraji, tím je v obraze více tmavých prvků a naopak.

2.1.4 Dithering

Nyní přichází na řadu definice pojmu dithering. Jedná se o techniku používanou v počítačové grafice při zpracování digitálního obrazu založenou na redukci jeho barevného prostoru. Cílem je dosáhnout maximální možné věrnosti výstupního obrazu s jeho předlohou při snížených paměťových nárocích (případně při dodržení požadavků na barevný prostor výstupního zobrazovacího zařízení). Pomocí kombinace barev z cílové palety se tak vyvolává

Obrázek 2.4: Jasový histogram a histogramy pro červenou, zelenou a modrou barevnou složku

optický dojem většího barevného rozsahu.

Používají se různé techniky k dosažení optické věrnosti k předloze. Většina z nich je založena na empirické bázi, která využívá způsob vnímání lidského oka. To nerozlišuje jednotlivé dílčí barvy nebo body, ale vytváří si výsledný vjem podle shluků bodů (čehož je využíváno například v monitorech). Proto jsme schopni vyvolat v lidském oku vjem šedé barvy jen pomocí shluku rovnoměrně rozmístěných bílých a černých bodů. Obdobný princip je pak použit u ditheringu barevných obrazů, kdy je barva chybějící ve výstupní paletě vizuálně napodobena určitým rozložením barev jiných.

Důvody používání ditheringu jsou zřejmé. Jedním z nich je komprese vstupního obrazu za pomoci redukce paměťových nároků barevných složek pixelů (kdy například každou barevnou složku ukládáme na dvou bitech namísto osmi). Výsledný obrázek tak umožňuje reprezentovat velmi podobnou množinu informací při znatelně menších paměťových nárocích. Dalším důvodem pro nasazení některého z algoritmů je omezení v podobě výstupního zařízení. Například na monitoru s omezenou barevnou paletou nebo při černobílém tisku potřebujeme interpretovat hodnoty barev vstupního obrazu, které však dané médium není schopno zobrazit. Přistupuje se proto k technikám redukce barevného prostoru za pomoci ditheringu. Velmi významným polem použití zejména prahování (thresholdingu) je také počítačové vidění. Pro rozpoznávání obrazu mnohdy není žádoucí maximální věrnost obrazu, nýbrž co největší míra zjednodušení a zároveň zvýraznění atributů obrazu podstatných pro danou aplikaci.

Kromě paměťových nároků, jejichž snížení je spojeno se zmenšením použité barevné palety, asi nejdůležitějším úkolem ditheringu je maximální kvalitativní přiblížení původnímu obrazu. Právě proto je důležitým faktorem při aplikaci těchto algoritmů zejména co nejvěrnější výstup vůči originálu a právě tímto přístupem by se dal charakterizovat celkový vývoj v této oblasti.

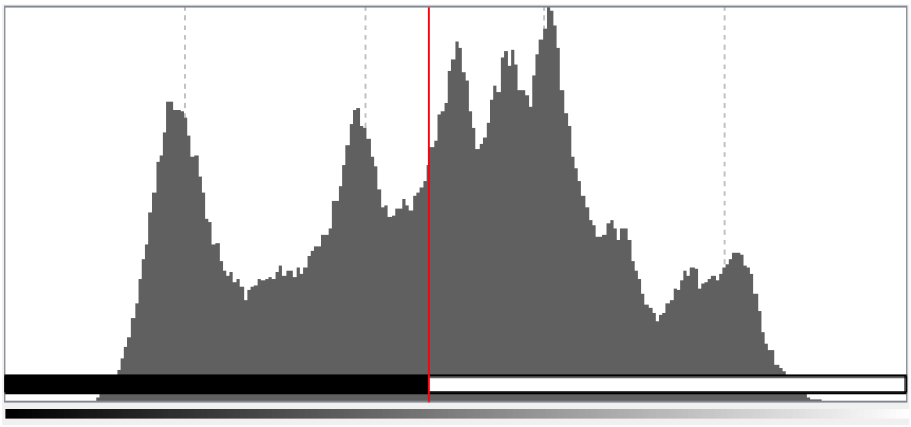
Dithering je typický tím, že pracuje pouze s barevným prostorem obrazu, nemění jeho velikost, resp. rozlišení. Naopak v metodě halftoning, které se budu věnovat později, se používají techniky, při kterých se zvětší rozlišení obrazu.

2.2 Metody ditheringu

V této části jsou analyzovány a vysvětleny všechny známé algoritmy ditheringu.

2.2.1 Prahování

Základní metodou pro redukci barevného prostoru obrazu na jednobitovou paletu je metoda zvaná prahování (thresholding [5]). Tato metoda má široké rozšíření v počítačové grafice, jedná se o nejjednodušší metodu pro převod obrazu na jednobitovou paletu.



Obrázek 2.5: Princip prahování, červená hodnota prahu rozdělí pixely na bílé a černé

Princip prahování se dá vyjádřit tím způsobem, že v histogramu obrazu si vhodně zvolíme určitou tzv. prahovou hodnotu. Všechny pixely, nacházející se v histogramu vlevo od této hodnoty budou mít ve výsledku černou barvu, naopak všechny, které se nacházejí napravo, budou mít barvu bílou (obrázek 2.5).

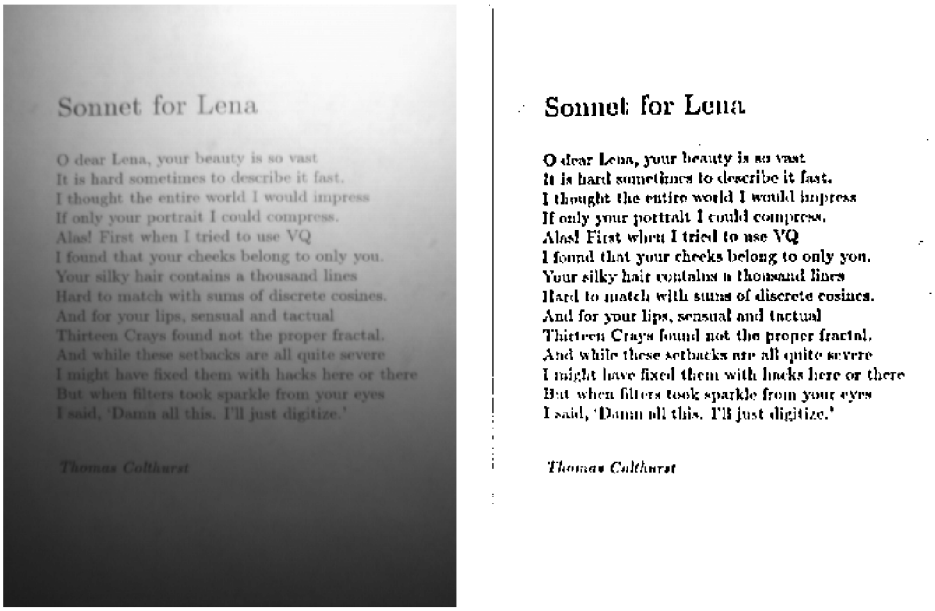
Prahování můžeme vyjádřit jednoduchou rovnicí (2.1). Používá se zde funkce $f(x, y)$, která pro každý pixel o souřadnicích vstupního obrazu x a y určí binární hodnotu příslušného bodu výstupního obrazu, a to buď 0 nebo 1 (černá, resp. bílá barva), následujícím způsobem:

$$f(x,y) = \begin{cases} 1, & g(x,y) \geq T \\ 0, & g(x,y) < T \end{cases} \tag{2.1}$$

Tato funkce porovnává barevnou či jasovou složku pixelu s určitou referenční hodnotou prahu. Pokud je hodnota zkoumaného atributu (typicky jasu) vypočtená pomocí funkce $g(x, y)$ nižší než zvolená mezní hodnota T, výsledkem je černá barva, pokud je naopak vyšší, výsledkem je barva bílá.

Prahování vykazuje obecně velmi dobré výsledky ve vysokých frekvencích, dokáže tak velmi dobře zvýraznit hrany objektů. Naopak v nízkých frekvencích (typicky pozvolné přechody jasu) nepodává příliš dobré výsledky. Využívá se tak hlavně pro segmentaci obrazu, počítačové rozpoznávání objektů, detekce papilárních linií ve snímačích otisků prstů nebo OCR programy pro rozpoznávání písmen (obrázek 2.6). V těchto aplikacích je žádoucí zvýraznit dominantní rysy objektů, kterými často bývají právě jejich hrany. Prahování má také za úkol odfiltrovat některé nežádoucí vlivy, například vrhaný stín.

Klíčovou otázkou v této problematice zůstává metoda nalezení optimální prahové hodnoty. V zásadě existují tři základní způsoby pro určení prahu: manuální, automatický a adaptivní.



(a) Originální obrázek (b) Adaptivní prahování

Obrázek 2.6: Využití adaptivního prahování v OCR programech.

2.2.2 Manuálně zvolený práh

Nejtriviálnějším přístupem je manuální (uživatelské) stanovení pevné hodnoty prahu pro všechny pixely vstupního obrazu. U každého bodu je pak jako referenční práh určena ona fixní hodnota. Výsledky tohoto přístupu jsou ovšem velmi silně závislé na charakteru vstupního obrazu a především na míře vhodnosti stanoveného prahu pro tento obraz (jak je vidět na obrázku 2.7).

Pokud redukujeme obrázek s velkou hustotou informací v nižších hodnotách jasu (například podexponovaná fotografie) a hodnota prahu je příliš vysoká, dojde k velké ztrátě podstatných informací. Stejný problém také nastává, pokud jsou v obraze, se kterým pracujeme, oblasti s různými lokálními jasovými poměry. V takových případech je velmi těžké najít optimální hodnotu prahu.

2.2.3 Automaticky zvolený práh

Vylepšením předchozího principu, zejména problému s mírou vhodnosti daného fixního prahu, spočívá ve vypočítání vhodné hodnoty prahu podle vstupního obrazu. Jedním z postupů, jak tohoto cíle dosáhnout, je tzv. Otsu prahování (Otsu's thresholding). Název má podle svého objevitele Nobuyuki Otsu [3].

Otsu algoritmus

Tento algoritmus je založen na nalezení optimální hodnoty prahu na základě zkoumání rozložení intenzit v obraze (histogramu). Princip spočívá v předpokladu existence dvou tříd pixelů - popředí a pozadí. Cílem je oddělit tyto dvě třídy pro daný konkrétní obraz. Pro každou možnou hodnotu prahu t od 1 až do maximální hodnoty intenzity spočítáme

(a) Originální obrázek

(b) Práh 64

(c) Práh 128

(d) Práh 192

Obrázek 2.7: Manuálně zvolený práh

rozptyl hodnot v rámci dané třídy ($\sigma_f^2(t)$, resp $\sigma_b^2(t)$). Poté provedeme vážený součet rozptylů popředí a pozadí (kde váhy $\omega_f(t)$ resp. $\omega_b(t)$ reprezentují podíl součtu počtu pixelů v dané třídě ku jejich celkového počtu) podle rovnice (2.2). Tím získáme vnitrotřídní rozptyl (within class variance) $\sigma_W^2(t)$. Hodnota intenzity t, která bude vykazovat minimální vnitrotřídní rozptyl, je výslednou hodnotou vypočteného prahu t.

$$\sigma_W^2(t) = \omega_f(t)\sigma_f^2(t) + \omega_b(t)\sigma_b^2(t) \tag{2.2}$$

V praxi se však využívá opačný přístup - zjišťování maximálního mezitřídního rozptylu (between class variance) $\sigma_B^2(t)$. Tento algoritmus popsaný rovnicí (2.3) je znatelně rychlejší, protože umožňuje pracovat s průměrnými hodnotami pixelů v dané třídě ($\mu_f(t)$ resp. $\mu_b(t)$),

(a) Originální obrázek (b) Manuální práh 100 (c) Metoda Otsu

Obrázek 2.8: Porovnání manuálního prahu a algoritmu Otsu

které se dají iterativně upravovat.

$$\sigma_B^2(t) = \omega_f(t)\omega_b(t)\left[\mu_b(t) - \mu_f(t)\right]^2 \tag{2.3}$$

2.2.4 Adaptivní práh

Adaptivní prahování (někdy také udávané jako lokální nebo dynamické) se od předchozích dvou metod liší tím, že nepoužívá pro každý pixel stejnou hodnotu prahu, nýbrž ji pro jednotlivé části obrazu určuje dynamicky. Prozkoumávání menších oblastí je výhodnější z toho důvodu, že je v nich větší pravděpodobnost menších jasových rozdílů. Menší oblasti jsou tedy vhodnější z hlediska prahování. Využívá se statistických metod pro vyhodnocení histogramů jednotlivých částí obrazu k určení nejvhodnější hodnoty prahu pro danou oblast či bod.

Existují dva základní přístupy k tomuto problému. Prvním z nich je determinizace hodnoty prahu na základě rozdělení obrazu. Zástupcem je například Chow a Kaneko algoritmus [1].

Chow a Kaneko algoritmus

Tento algoritmus spočívá v tom, že se vstupní obraz rozdělí na sérii dlaždicovitě překrývajících se oblastí. Pro každou z těchto oblastí můžeme na základě analýzy histogramu (například pomocí výše zmíněného Otsu algoritmu) určit odpovídající práh, který je následně použit pro všechny pixely dané oblasti. Tento přístup je ovšem výpočetně velmi neefektivní, proto se nedá nasadit pro aplikace vyžadující prahování obrazu v reálném čase.

Wellnerův algoritmus

Druhým postupem v oblasti adaptivního prahování je zjišťování hodnoty prahu pro každý bod zvlášť pomocí zkoumání okolí daného bodu - takzvané lokální prahování. Tuto metodu využívá například Wellnerův algoritmus [13]. Ten pracuje na bázi postupného čtení vstupního obrazu po řádcích a vyhodnocování hodnoty prahu na základě určitého počtu již přečtených pixelů. Pro každý bod tedy máme množinu hodnot, podle níž můžeme určit práh pro právě zpracovávaný pixel. Způsob statistického vyhodnocování této množiny do značné míry závisí na charakteru vstupního obrazu. V zásadě máme tři možnosti postupu. První a nejjednodušší z nich je využití průměrné hodnoty jako prahu.

(a) Originální obrázek (b) Redukovaný obrázek

Obrázek 2.9: Chow a Kaneko algoritmus.

$$T = mean$$

Další možností je použití mediánu.

$$T = median$$

Poslední možností je zprůměrovat minimální a maximální hodnotu v množině.

$$T = min + max/2$$

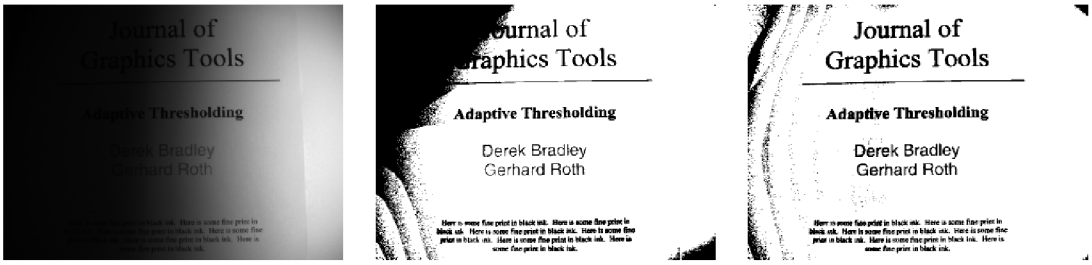
Dalším parametrem, který významně ovlivňuje kvalitu výstupního obrazu, je počet bodů zahrnutých do této statistiky. Ten musí být dostatečně velký, aby mohl pokrýt reprezentativní vzorek bodů popředí i pozadí, ne však příliš velký na to, aby rozložení jasu v dané oblasti přestalo být uniformní.

Výhodou tohoto algoritmu je sekvenční čtení hodnot z paměti řádek po řádku a tudíž výkonnostní urychlení oproti náhodnému přístupu do paměti. Nevýhodou však jsou mírně odlišné výsledky při průchodech obrazu různými směry (zprava doleva, zleva doprava).

Bradleyho algoritmus

Problém s různými směry průchodu vyřešilo rozšíření implementace Wellnerova algoritmu od Dereka Bradleyho a Gerharda Rotha [7]. To namísto počítání s hodnotami předchozích načtených pixelů pracuje se čtvercovým okolím daného bodu. Tento přístup má v praxi mnohem lepší výsledky a jeho kvalita již není závislá na směru průchodu obrazem. Praktické rozdíly v těchto algoritmech jsou viditelné zejména ve tmavých částech na obrázku 2.10.

Určitou nevýhodou oproti Wellnerově algoritmu je o něco pomalejší výpočet, nicméně stále je ještě vhodný i pro aplikace běžící v reálném čase.



(a) Originální obrázek (b) Wellnerův algoritmus (c) Bradleyho algoritmus

Obrázek 2.10: Porovnání Wellnerova a Bradleyho algoritmu.

2.2.5 Náhodný rozptyl

Při jakékoliv redukci barevného prostoru obrazu se dopouštíme určité chyby. Čím menší výslednou paletu použijeme, tím větší ztráty informace se dopouštíme. V případě redukce na jednobitovou paletu je tato ztráta nejznatelnější. Cílem všech metod zmiňovaných v této práci je minimalizovat onu chybu tak, aby výsledný vjem z redukovaného obrazu byl maximálně věrný předloze. Už u prahování vyvstala otázka, jak nalézt nejoptimálnější referenční hodnotu pro každý bod vstupního obrazu. Nejpokročilejší adaptivní algoritmy statisticky vyhodnocovaly okolí zkoumaného bodu a podle něj rozlišovaly, zda aktuální bod náleží spíše do třídy popředí nebo pozadí.

Tento postup je výhodný pro zvýrazňování významných částí obrazu, protože prahování funguje jako vysokofrekvenční 2D filtr. Příliš dobré výsledky tedy nevykazuje v nižších frekvencích, jako jsou například barevné přechody nebo jiné oblasti s pozvolným lokálním poklesem jasu. Pro tyto části obrazu už přestávají stačit běžné metody, přičemž se velmi osvědčuje zavedení určitého nedeterminismu. Využití role náhody bylo historicky prvním pokusem, jak vylepšit algoritmus prostého prahování.

Náhodný rozptyl (dithering) je metoda pro výběr prahové hodnoty pro konkrétní bod vstupního obrazu pomocí výběru náhodného čísla v intervalu $< 0, I_{max} >$, kde I_{max} je maximální hodnota intenzity ze vstupního obrazu (typicky 255 pro 8-bitové RGB). Čím menší světelnou intenzitu I má vstupní pixel, tím větší je pravděpodobnost p, že náhodně zvolená hodnota prahu T bude ležet nad onou intenzitou ($I < T$) a tudíž bude výstupnímu pixelu přiřazena černá barva. Stejně tak, čím je intenzita vstupu vyšší, tím vyšší je také pravděpodobnost, že výsledkem bude barva bílá.

Jistým problémem, který tento postup skýtá, je metodika generování náhodných čísel. Pro tyto účely se používá tzv. generátor náhodných čísel [4]. V oblasti počítačů je velmi obtížné zajistit skutečně náhodně zvolené číslo. Nestačí pouhé výpočetní prostředky procesoru, je zapotřebí určitého fyzikálního jevu, jehož nepředvídatelnost se dá dokázat na atomární či subatomární úrovni za pomoci zákonů kvantové mechaniky. Velmi často se pro skutečnou náhodnost využívá časování pohybů čtecích a zápisových hlav na pevném disku nebo signálů ze síťové karty.

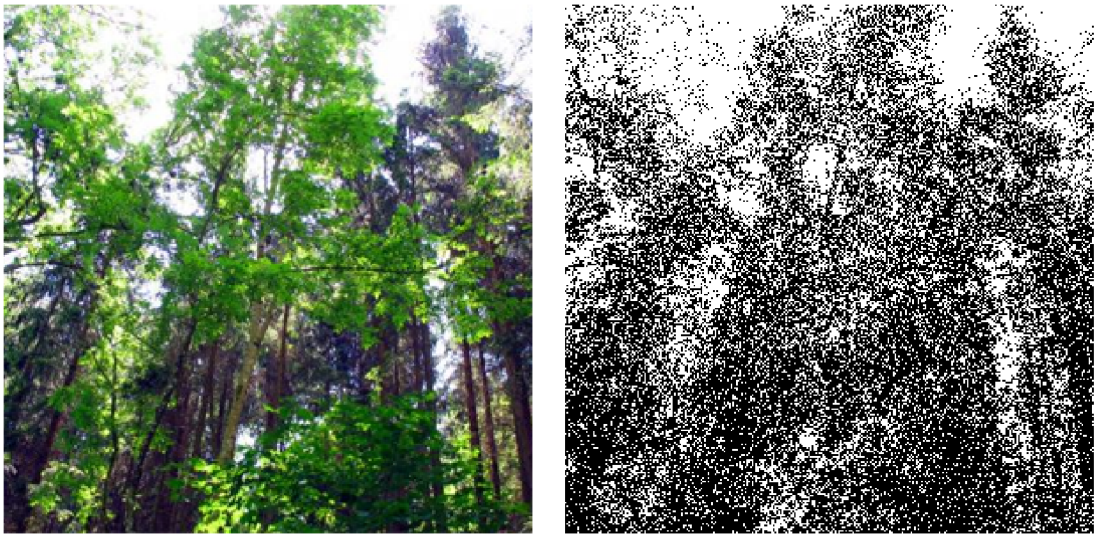
Mnohem běžnější je nicméně použití softwarových generátorů pseudonáhodných čísel. Jedná se o algoritmy (například lineární kongruentní generátor), které generují sérii zdánlivě náhodných čísel. Tato sekvence je ovšem deterministicky závislá na vstupní hodnotě tzv. semínka (seed). Při stejné vstupní hodnotě produkují stejnou sérii čísel.

Výsledkem procesu nahodného rozptylu je značné množství vysokofrekvenčního šumu,

způsobeného pravděpodobnostním modelem. Oblasti o stejné intenzitě ovšem vykazují obdobné spektrum šumu a tudíž si vizuálně odpovídají. Metoda náhodného rozptýlení je díky těmto vlastnostem velmi vhodná na zobrazování již zmíněných nízkofrekvenčních barevných přechodů, jak ukazuje obrázek 2.11. Pozvolný přechod barvy pozadí dokáže náhodný rozptyl velmi věrně zobrazit.

(a) Originální obrázek (b) Náhodný rozptyl

Obrázek 2.11: Příklad vhodnějšího obrazu pro náhodné rozptýlení.



(a) Originální obrázek (b) Náhodný rozptyl

Obrázek 2.12: Příklad nevhodného obrazu pro náhodné rozptýlení.

Určitý problém nastává ve chvíli, kdy je v obrázku mnoho frekvenčních změn, typicky mnoho menších elementů s různým jasem, ostré jasové přechody (hrany) a podobné jevy.

V těchto místech se nejvíce projeví produkce šumu, jak je vidět na obrázku 2.12.

Variantou tohoto algoritmu je přístup, kdy ke každému pixelu připočteme určitou náhodně vygenerovanou hodnotu v uniformním rozložení v rozmezí $< -x, x >$. Tímto způsobem obraz náhodně zašumíme a následně použijeme klasické prahování pro získání finálního obrazu. Tato metoda funguje velmi dobře při redukci barevného obrazu na menší paletu, přidáním náhodné barevné hodnoty ke každému pixelu a následným vyhledáním nejvhodnější barvy z výstupní palety.

2.2.6 Maticový rozptyl

V rámci této sekce popíši další třídu technik pro redukci barevného prostoru na jednobitovou paletu, a to maticový rozptyl. Jak název napovídá, pracuje se zde s maticí prahových hodnot, která se určitým způsobem aplikuje na vstupní obraz.

Uspořádaný maticový rozptyl

Tato technika pracuje s maticí číselných hodnot, které fungují jako koeficienty pro určení prahových hodnot ze vstupního rozsahu intenzit. Matice se dlaždicovitě nanáší na vstupní obraz a pro každý pixel je provedeno prahování s hodnotou na příslušném místě v matici. Výsledný obraz tak tvoří mřížkovité obrazce bodů, jež vizuálně velmi věrně odpovídají různým odstínům šedé barvy. Nejlepších výsledků je dosaženo za pomoci tzv. Bayerovy matice M_{Bayer} o rozměrech 4*4 pixely.

$$M_{Bayer} = \begin{vmatrix} 1 & 9 & 3 & 11 \\ 13 & 5 & 15 & 7 \\ 4 & 12 & 2 & 10 \\ 16 & 8 & 14 & 6 \end{vmatrix}$$

Výhodou tohoto algoritmu je skutečnost, že matice může být libovolně otáčena či transponována bez omezení funkčnosti. Na obrázku 2.13 je vidět porovnání výsledků uspořádaného rozptylu s maticemi M_{Bayer} a $M_{Halftone}$ (ta je popsána níže).

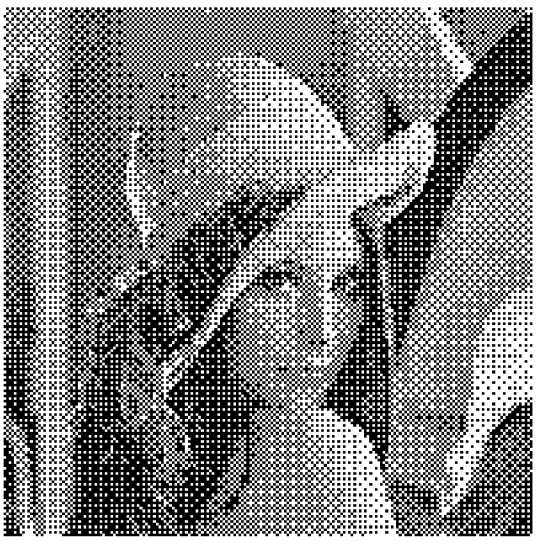
Bayer [6] ukázal, že nejoptimálnější matice pro uspořádaný dithering jsou takové, jejichž rozměr je mocninou dvou. Zde uvedená matice M_{Bayer} je nejpoužívanější z nich.

Můžeme si nadefinovat také matice libovolné a pomocí nich provést distribuci, což může být vhodné pro různé efekty, jako zesvětlení nebo ztmavení obrázku. Příklady jiných matic jsou vidět na obrázku 2.14.

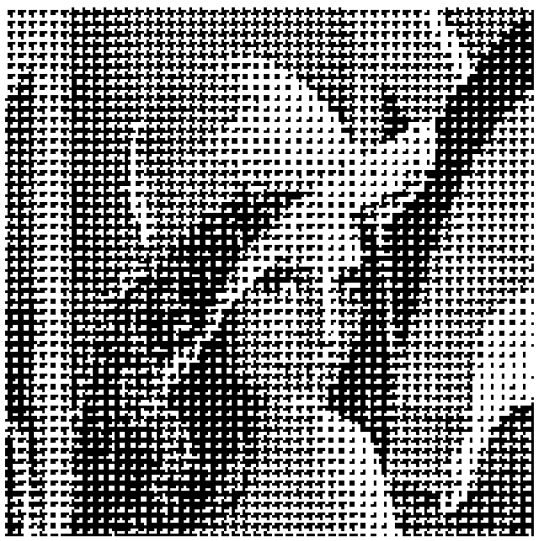
2.2.7 Halftoning

Specifickou metodou související s maticovým rozptylem je halftoning. Je to technika velmi používaná zejména v tisku. Princip halftoningu spočívá v optické simulaci intenzitní škály za pomocí různého rozmístění různě velkých černých skvrn. Každý pixel vstupního obrazu o určité hodnotě světelné intenzity je nahrazen čtvercovým obrazcem pixelů o velikosti odpovídající rozměrům použité matice. Barva jednotlivých pixelů v tomto obrazci odpovídá vždy prahování vstupního bodu s hodnotou na příslušném místě v matici. Matice prahových hodnot je ovšem uspořádaná jinak než v případě Bayerovy matice. Jako příklad uvádím matici $M_{Halftone}$ o rozměrech 4*4 pixely. Větší hodnoty prahů jsou zastoupeny v jejích rozích, z čehož pramení vlastnost vytvářet spíše shluky bodů. Čím je pixel tmavší, tím větší je shluk v rozích obrazce, což se po dlaždicovitém složení s okolními obrazci projeví jako větší černý shluk s menším rozestupem od ostatních. To je velmi žádoucí vzhledem

(a) Originální obrázek



(b) M_{Bayer}



(c) $M_{Halftone}$

Obrázek 2.13: Uspořádaný maticový rozptyl

ke způsobu využití. Tisková hlava zanechává sérii různě velkých černých skvrn s určitými rozestupy a tím simuluje větší rozsah intenzit daný velikostí použité matice.

$$M_{Halftone} = \begin{vmatrix} 2 & 6 & 10 & 3 \\ 9 & 13 & 14 & 7 \\ 5 & 16 & 15 & 11 \\ 1 & 12 & 8 & 4 \end{vmatrix}$$

Důsledkem zpracování obrazu pomocí této metody je zvětšení rozlišení výstupního obrazu, jelikož pro každý bod se rozměr zvětší o rozměr celé matice. Ovšem tiskárny jsou schopny toto omezení kompenzovat díky mnohonásobně vyššímu rozlišení na určitou plochu než u LCD displejů. Čím větší rozměry má použitá matice, tím více úrovní šedé jsme

Obrázek 2.14: Příklady dalších matic

schopni simulovat.

Vzhledem ke zvětšování rozměrů obrazu tato metoda nepatří pod dithering, nicméně její vlastnosti jsou velmi podobné maticovému rozptylu.

2.2.8 Dithering s distribucí chyby

Při jakémkoliv algoritmu redukce barevného prostoru na jednobitovou paletu logicky dochází k velké ztrátě informací. Cílem všech uvedených metod je minimalizovat tuto ztrátu. Zatímco u předešlých principů se této minimalizace dosahuje za pomoci zavedení nedeterminismu (náhodný rozptyl) nebo vizuální simulace daného odstínu aplikací pravidelných obrazců (maticový rozptyl), u této metody se přímo vyčísluje ztracená informace a obraz je upravován, aby tuto ztrátu kompenzoval.

Konkrétně je zde zaveden pojem chyba. Tímto termínem se myslí hodnota rozdílu intenzity vstupního pixelu a námi určená hodnota intenzity na výstupu (černá či bílá). Pokud má například vstupní pixel hodnotu intenzity 100 a použitá metoda (např. prahování) určí, že výstupem bude černý pixel (o intenzitě 0), pak hodnota chyby, které jsme se dopustili, je -100. Tato informace o chybě je pak určitým způsobem rozdistribuována na okolní pixely, což vede k částečné kompenzaci chyby, které se dopustíme při prahování těchto bodů.

Existuje několik různých přístupů, jak přenášet informaci o chybě na okolní body. Všechny z nich předpokládají sekvenční procházení vstupního obrazu po řádcích zleva doprava a modifikují hodnoty intenzit následujících pixelů v okolí napravo a pod nimi. Modifikace spočívá v přičtení určité části hodnoty chyby (ať už kladné či záporné) k okolním pixelům. Takto probíhá určitá korekce chyby způsobené při jejich budoucím zpracování. Čím větší okolí do tohoto procesu zapojíme, tím kvalitnější je výsledek.

V následující části si popíšeme jednotlivé způsoby přenášení informace o chybě na okolní pixely. Liší se tzv. filtrem (schématem) neboli maticí hodnot určující poměr rozložení chyby mezi body. Při ukázkách těchto matic bude značit písmeno X aktuální pixel, na který přiložíme onu matici. Hodnota pixelu I_{new} se spočítá jako $I_{new} = I_{old} + (i * N * E)$, kde I_{old} je původní hodnota intenzity daného bodu, i hodnota uvedená v matici (neboli váha),

E hodnota chyby a N koeficient dané matice udávající převrácenou hodnotu celkového součtu hodnot v ní uvedených (např. 1/16). Tento zápis je pouze pro přehlednost, abychom nemuseli zapisovat do matic zlomky.

Tyto distribuční matice obecně představují prozatím nejvyspělejší techniku pro dithering obrazu.

Na obrázku 2.15 jsou pak vidět rozdíly mezi jednotlivými metodami.

2.2.9 Floyd-Steinberg

První z metod ditheringu s distribucí chyby byl Floyd-Steinbergův filtr představený v roce 1976 [9]:

$$\begin{vmatrix} 0 & X & 7 \\ 3 & 5 & 1 \end{vmatrix}$$

$$N = 1/16$$

Distribuce probíhá na nejbližší okolí bodu, konkrétně 7/16 chyby na následující sousední pixel, 3/16, 5/16 a 1/16 na okolí aktuálního bodu o řádek níže. Toto rozložení platí pro průchod zleva doprava, pro opačný postup stačí převrátit matici. Floyd-Steinberg filtr je velmi často používán díky příznivým výsledkům a výpočetní rychlosti (dělení 16 se dá implementovat jako rychlá bitová operace posunu - dělíme mocninou čísla 2).

Existuje ještě další alternativa, tzv. "falešný"Floyd-Steinberg filtr:

$$\begin{vmatrix} X & 3 \\ 3 & 2 \end{vmatrix}$$

$$N = 1/8$$

Z důvodu menší distribuce vah do okolí má v běžném průchodu výrazně horší vlastnosti, nicméně při speciálním "klikatém"průchodu, tzn. lichý řádek zleva doprava, sudý řádek zprava doleva s převrácenou maticí vykazuje výsledky o poznání lepší.

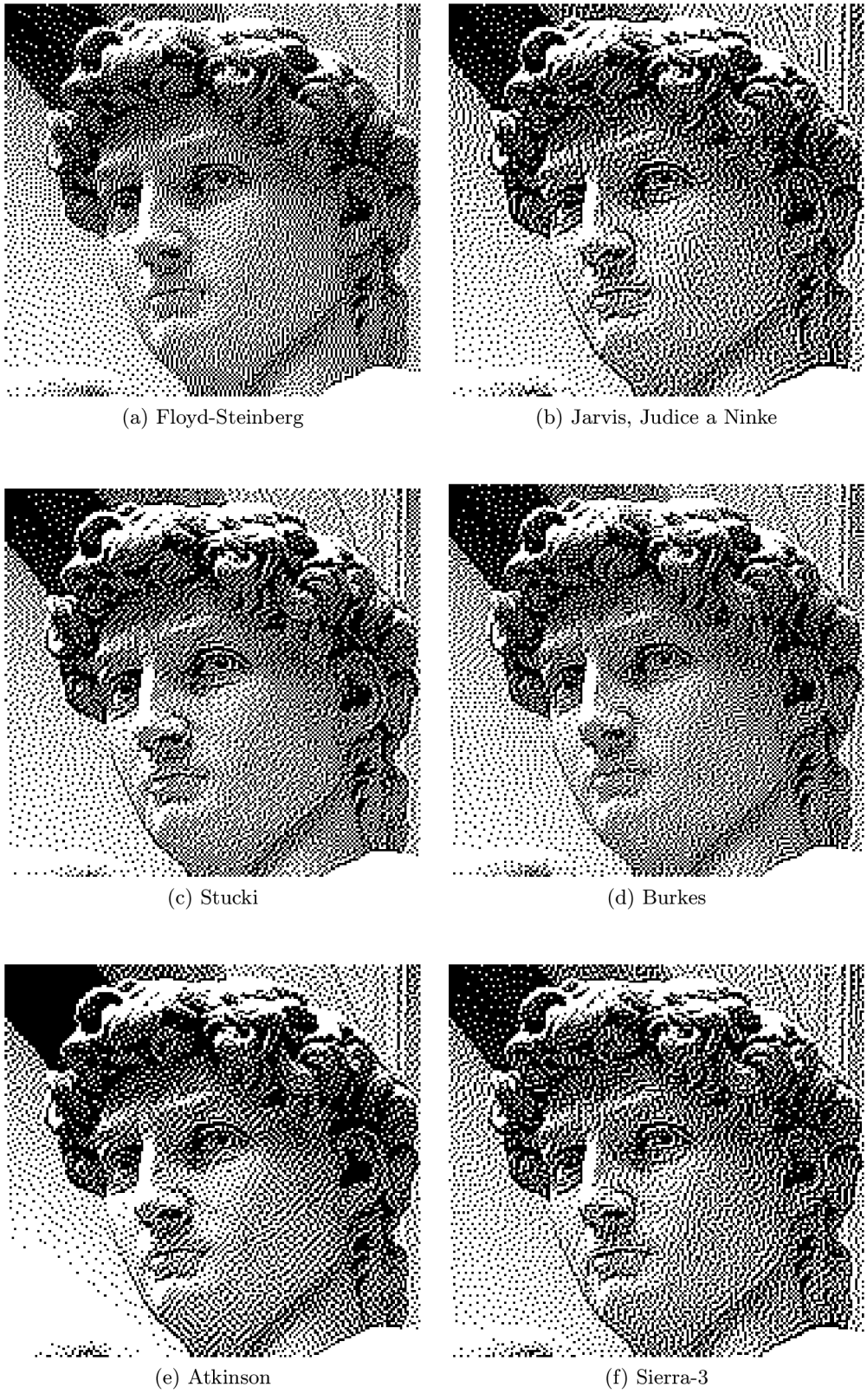
2.2.10 Jarvis, Judice a Ninke

Omezení Floyd-Steinbergova přístupu, které se na některém typu obrazů může více projevit, spočívá ve velmi malém dosahu distribuce chyby pouze na čtyři okolní body. Proto přišli J. F. Jarvis, C. N. Judice a W. H. Ninke s vylepšeným filtrem [10], který předává část chyby 12 pixelům:

$$\begin{vmatrix} 0 & 0 & X & 7 & 5 \\ 3 & 5 & 7 & 5 & 3 \\ 1 & 3 & 5 & 3 & 1 \end{vmatrix}$$

$$N = 1/48$$

Tato metoda produkuje lepší výsledky než původní Floyd-Steinberg, ovšem za cenu vyšší výpočetní náročnosti. Jednak kvůli tomu, že pracujeme se širším okolím, ale hlavně protože operace dělení 48 již není realizovatelná pomocí bitových posunů.



(a) Floyd-Steinberg

(b) Jarvis, Judice a Ninke

(c) Stucki

(d) Burkes

(e) Atkinson

(f) Sierra-3

Obrázek 2.15: Porovnání algoritmů distribuce chyby

2.2.11 Stucki

Tento filtr vychází z výše zmíněného Jarvis, Judice a Ninke filtru, navíc přináší optimalizaci výpočtu [12]:

$$\begin{vmatrix} 0 & 0 & X & 8 & 4 \\ 2 & 4 & 8 & 4 & 2 \\ 1 & 2 & 4 & 2 & 1 \end{vmatrix}$$

$$N = 1/42$$

Tato optimalizace spočítá v redukci operace dělení. Pro každý pixel se jako první spočte hodnota $8 * 1/42 * E$ a všechny následující se z ní počítají bitovými posuny.

2.2.12 Burkes

Další optimalizaci pro rychlost výpočtu představuje filtr prezentovaný Danielem Burksem v roce 1988 [8]:

$$\begin{vmatrix} 0 & 0 & X & 8 & 4 \\ 2 & 4 & 8 & 4 & 2 \end{vmatrix}$$

$$N = 1/32$$

Spočívá v prostém odstranění posledního řádku ze Stucki filtru. To s sebou přináší také výhodu v podobě koeficientu 1/32, díky němuž jsme schopni výpočty provádět opět bitovým posunem.

2.2.13 Atkinson

Filtr navržený Billem Atkinsonem propaguje pouze 75% chyby, což má za následek ztrátu kontrastu ve tmavých a světlých místech (kraje histogramu), ale lepší kontrast ve středních tónech.

$$\begin{vmatrix} 0 & X & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{vmatrix}$$

$$N = 1/8$$

2.2.14 Sierra

Frankie Sierra navrh v letech 1989 - 1990 tři filtry jako vylepšení Floyd-Steinbergova algoritmu.

Třířádkový Sierra filtr:

$$\begin{vmatrix} 0 & 0 & X & 5 & 3 \\ 2 & 4 & 5 & 4 & 2 \\ 0 & 2 & 3 & 2 & 0 \end{vmatrix}$$

$$N = 1/32$$

Dvouřádkový Sierra filtr:

$$\begin{vmatrix} 0 & 0 & X & 4 & 3 \\ 1 & 2 & 3 & 2 & 1 \end{vmatrix}$$

$$N = 1/16$$

Jednoduchý Sierra "filtr lite":

$$\begin{vmatrix} 0 & X & 2 \\ 1 & 1 & 0 \end{vmatrix}$$

$$N = 1/4$$

2.3 Dithering u barevných obrazů

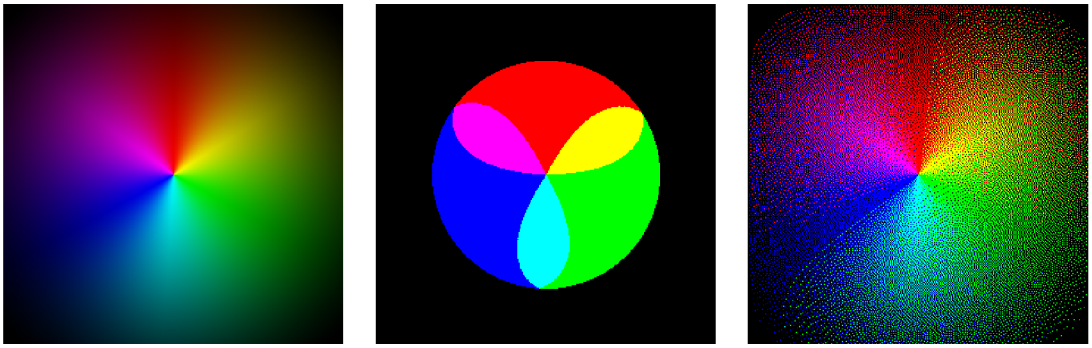
Ve všech výše uvedených případech jsem doposud pro zjednodušení uvažoval pouze převod obrázku v odstínech šedi na jednobitovou paletu. Pracoval jsem tak jen s hodnotou světelné intenzity v jednokanálovém obraze. V praxi se ovšem setkáváme spíše s barevnými obrázky, u nichž potřebujeme redukovat barevnou hloubku, ať už z důvodu paměťových nároků, nebo omezeními zobrazovacího zařízení. U těchto obrázků již nestačí jedna hodnota intenzity, nýbrž je zapotřebí tolik hodnot, kolik kanálů je v použitém barevném modelu. Typicky RGB se třemi složkami nebo CMYK se čtyřmi.

2.3.1 Redukce na n-prvkovou paletu

Jako příklad si uvedeme redukci v barevném prostoru RGB. Obvykle je vstupní obraz v 24-bitové paletě (paleta TrueColor [2]), což znamená, že pro každou složku je k dispozici 8 bitů (čili 256 hodnot). V této paletě jsme schopni zobrazit 16 777 216 odstínů. Tento obraz budeme chtít redukovat do 3-bitové RGB palety, která má pro každou barevnou složku 1 bit, tudíž máme k dispozici pouze 8 odstínů.

Zpracování obrazu probíhá tím způsobem, že se rozdělí na jednotlivé barevné kanály, které obsahují informaci o intenzitě barevných složek. Na každý z těchto kanálů pak aplikujeme libovolnou metodu ditheringu. Výběr odpovídající hodnoty intenzity se provádí jako výběr nejbližší možné hodnoty barevné složky z dané palety. Pokud tedy například pracujeme s paletou, u níž má daný kanál k dispozici jeden bit (dva možné odstíny), postupujeme jako u klasického prahování. Jestliže má k dispozici více bitů, vybere se ten odstín, který nejvíce odpovídá vstupní hodnotě.

Poté, co takto nakvantujeme všechny barevné složky, složíme je zpět do výsledného obrazu s redukovanou paletou.



(a) Originální obrázek (b) 3-bit RGB (c) Floyd-Steinberg

Obrázek 2.16: Redukce na 3-bitovou RGB paletu za pomoci Floyd-Steinbergova filtru.

Při redukci barevného prostoru na n-prvkovou paletu se také dá využít výhod ditheringu pro optickou simulaci barevných odstínů, které nejsou v dané paletě dostupné. A to za pomoci kterékoliv z výše popsaných metod, jako například uspořádaného maticového ditheringu nebo distribuce chyby. Jak vypadá obraz redukovaný na 3-bitovou paletu běžným způsobem a za pomoci Floyd-Steinbergova filtru můžeme vidět na obrázku 2.16. Na obrázku 2.17 pak můžeme porovnat jednotlivé metody při redukci na 3-bitovou RGB paletu (8 barev).

Princip aplikace ditheringu je obdobný jako u monochromatického vstupního obrazu, jen jej aplikujeme podle výše zmíněného principu pro každý barevný kanál zvlášť. Proto kupříkladu při distribuci chyby vypočítáváme a následně i přenášíme hodnotu chyby pro červenou, zelenou i modrou složku.

Při redukci se většinou používají standardní RGB palety, kdy má každá barevná složka k dispozici stejný počet bitů. Používají se však též určité nestandardní palety, a to zejména v případech, kdy potřebujeme co nejvíce minimalizovat paměťové nároky výstupního obrazu.

2.3.2 Paleta RGB 3-3-2

Jednou z nestandardních palet je paleta zvaná 3-3-2. Jedná se o 8-bitovou RGB paletu, kdy máme pro uložení červené a zelené barvy 3 bity a pro uložení modré 2 bity. Vychází se z informace, že lidské oko je na modrou barvu nejméně citlivé, proto je pro tuto barvu vyhrazeno méně paměti. Paleta je tudíž schopná zobrazit méně modrých odstínů, jak je vidět na obrázku 2.18b.

Výsledky jsou velmi uspokojivé s ohledem na třetinovou paměťovou náročnost oproti 24-bitové paletě. Za pomoci ditheringu se dá dosáhnout ještě zajímavějších výsledků (obrázek 2.18c) V tomto případě je použita distribuce chyby s Floyd-Steinbergovým filtrem.

2.3.3 Indexovaná paleta

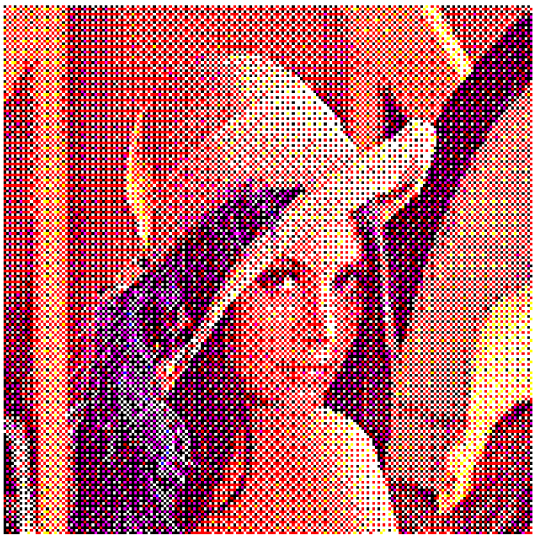
Všechny výše uvedené palety mají z hlediska kvality výsledného obrazu jeden podstatný problém. Vzhledem k pevně stanovenému paměťovému prostoru mají omezený počet barevných odstínů, které jsou schopny reprezentovat. V praxi je běžné, že vstupní obraz obsahuje odstíny, jež nejsou v dané paletě zobrazitelné, pak se musí použít jejich aproximace. Ovšem paleta též zpravidla obsahuje množinu barev, které nejsou v daném obrázku vůbec použity. Dochází tak k neefektivnímu využití paměťového prostoru vyhrazeného pro obrázek.

Nejprve se zaměříme na redukci paměťových nároků daného obrazu. V klasické RGB paletě obsahuje každý pixel informaci o červené, zelené a modré složce. Z těchto tří informací se posléze sestaví výsledná barva. Značné paměťové úspory oproti tomuto modelu se dá dosáhnout za použití tzv. indexované palety. Její princip spočívá v tom, že se nejprve určitým způsobem vygeneruje tabulka barevných odstínů. Toho lze docílit například postupným generováním odstínů RGB palety. Následně se vstupní obrázek pixel po pixelu namapuje na tuto paletu a to tím způsobem, že informace v každém bodě bude obsahovat index do oné předgenerované palety. Tento postup s sebou nese významné snížení použité paměti, kdy je pro každý pixel uložena pouze jediná hodnota. Nevýhodou však zůstává omezená velikost palety s omezeným počtem zobrazitelných odstínů a zejména redundance nepoužitých barev.

(a) Originální obrázek (zmenšeno)

(b) 3-bit RGB

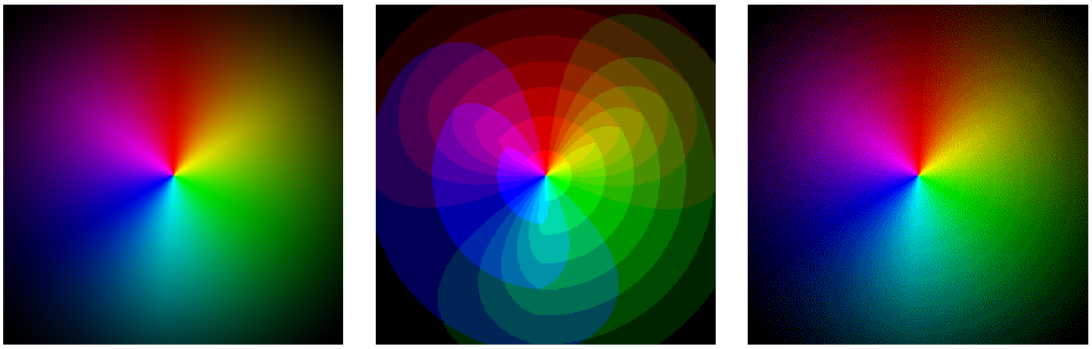
(c) Floyd-Steinberg



(d) Maticový rozptyl

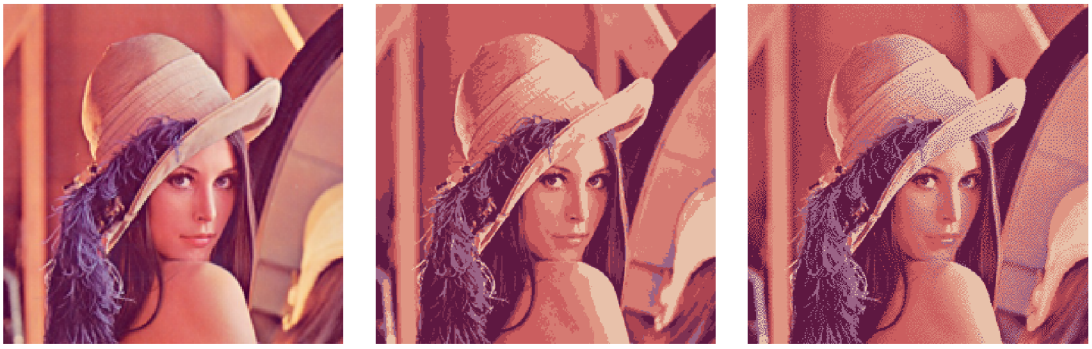
(e) Náhodný rozptyl

Obrázek 2.17: Využití metod ditheringu při redukci na 3-bitovou RGB paletu.



(a) Originální obrázek (b) 3-3-2 RGB (c) Floyd-Steinberg

Obrázek 2.18: Redukce na paletu RGB 3-3-2 za pomoci Floyd-Steinbergova filtru.



(a) Originální obrázek (b) Adaptivní (c) Floyd-Steinberg

Obrázek 2.19: Použití adaptivní palety (8 barev).

2.3.4 Adaptivní paleta

K vyřešení problému nadbytečnosti barev se používá tzv. adaptivních palet. Ty mají stále omezenou velikost, ale proces jejich sestavování je odlišný. Negenerují se všechny barvy, ale pouze ty, které jsou ve vstupním obraze skutečně obsaženy. Za tímto účelem se používají kvantizační algoritmy, které analyzují vstupní obraz a vyberou nejvhodnější barvy pro vložení do palety. Paměťový prostor vyhrazený pro definici palety se tak efektivně zaplní jen těmi odstíny, které jsou skutečně ve výsledku použity. Pro další zlepšení kvality obrázku se dá opět využít ditheringu, s jehož pomocí se dají opticky nasimulovat i ty barvy, které nejsou v paletě obsaženy. Na obrázku 2.19b je demonstrován převod obrázku do osmibarevné adaptivní palety a použití Floyd-Steinbergova filtru (2.19c).

Kapitola 3

Návrh a implementace

V této kapitole se budu podrobněji věnovat návrhu nové metody ditheringu obrazu a také návrhu demonstračního programu. Dále rozeberu návrh porovnávací metodiky.

3.1 Algoritmus

Z teoretického rozboru vyplývá, že nejpokročilejšími metodami z hlediska kvality výsledného obrazu jsou ty, které jsou založené na principu distribuce chyby. Existuje velké množství různých filtrů a algoritmů distribuce chyby mezi jednotlivé pixely, přičemž každý z nich se hodí pro jiný druh obrázků. V oblasti nových kombinací hodnot v maticích filtrů se výzkum zdá být vyčerpán, proto jsem se rozhodl spíše zkombinovat jejich vlastnosti s další metodou ditheringu - náhodným rozptylem.

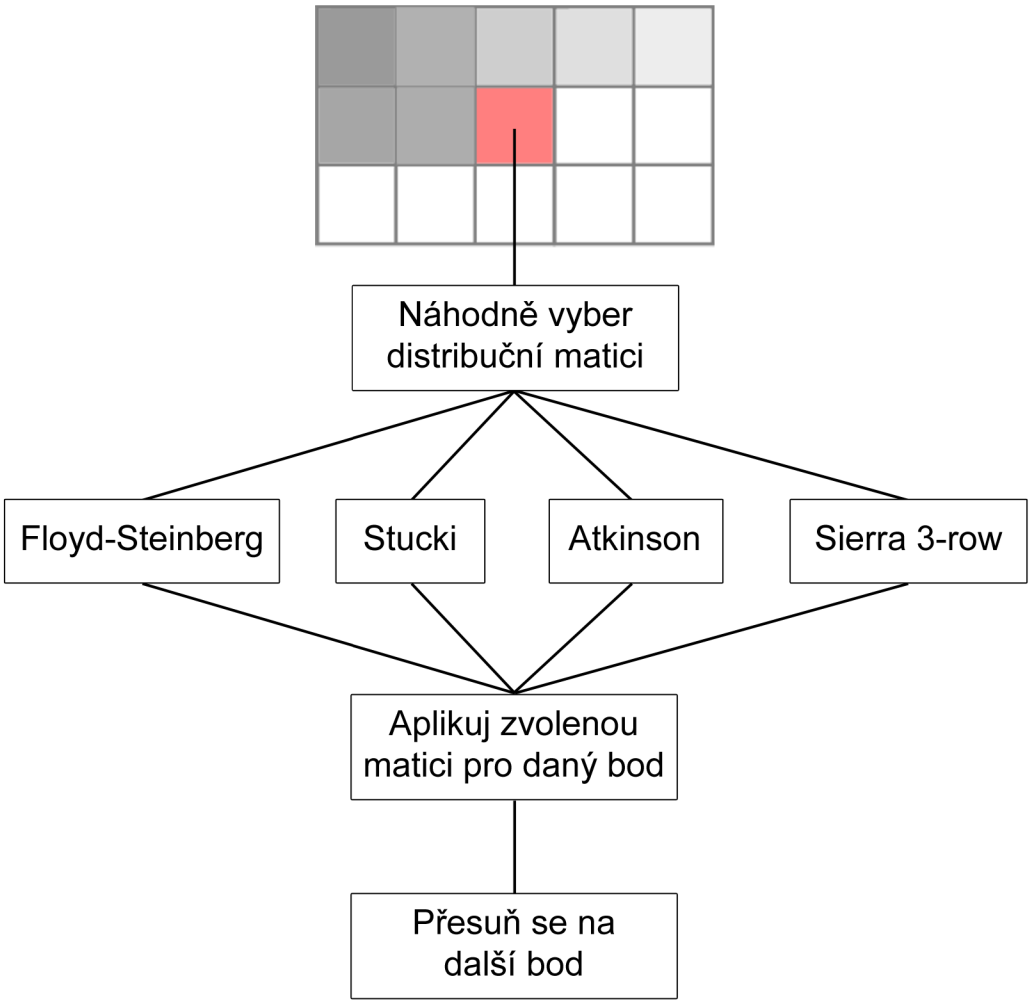
Mnou navržený algoritmus počítá se čtyřmi základními modifikacemi distribuce chyby - Floyd-Steinberg (sekce 2.2.9), Stucki (2.2.11), Atkinson (2.2.13) a Sierra-3 (2.2.14). Vstupní obraz budu procházet po řádcích zleva doprava a pro každý pixel bude náhodně vybrán jeden z těchto čtyř principů, který se aplikuje na výstupní obraz. Schématicky je to znázorněno na obrázku 3.1. Takto se rovnoměrně nedeterministicky využijí všechny distribuční matice, což by mělo vést k větší univerzalitě a kvalitnějšímu výstupu.

Další modifikace tohoto algoritmu spočívá v postupné aplikaci těchto čtyř metod v předem definovaném pořadí, na rozdíl od předešlého stochastického způsobu. Ve fázi praktického testování těchto metod chci zjistit, zda je tento postup lepší než náhodný výběr.

3.2 Porovnávací metodika

Klíčovým problémem celé této práce je nalezení vhodných klíčových faktorů, na základě kterých by se daly s maximální mírou objektivity hodnotit a navzájem porovnávat jednotlivé metody ditheringu.

Jako hlavní kritérium jsem si stanovil kvalitu výsledného obrazu, respektive míru jeho vizuální věrnosti vzhledem k původnímu obrazu při zvolené paletě. Algoritmy jsou tedy srovnávány v kontextu zvolené palety. V praxi jsem zvolil čtyři nejzajímavější palety: 1-bitovou (černobílou), 3-bitovou RGB (jeden bit pro každý kanál), 6-bitovou RGB (dva bity na kanál) a často používanou RGB paletu 3-3-2 (tři bity pro červený a zelený kanál, dva bity pro kanál modrý). Při použití vícebitových palet je vizuální rozdíl mezi výstupem a originálem obtížně rozeznatelný (jako vstupní paletu uvažuji 24-bitovou RGB).

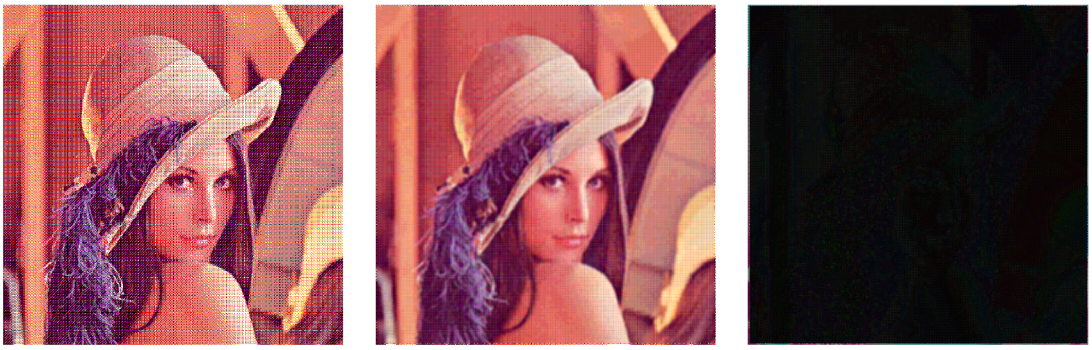


Obrázek 3.1: Schématický diagram funkčnosti vlastního algoritmu náhodné distribuce chyby

Z teoretického rozboru vyplývá, že hlavní úlohou ditheringu je poskytnout možnost opticky interpolovat barvy původního obrazu, které se ovšem nenachází v cílové paletě. Toho se dosahuje shlukem dostupných odstínů, jež se vlivem nedokonalosti lidského oka vizuálně spojí v barvu požadovanou. Právě na tento aspekt jsem se ve své úvaze zaměřil. Hlavním problémem z hlediska porovnávání jsou ale z principu odlišné palety vstupu a výstupu, které znemožňují objektivní srovnání kvality obrazů. Pro odstranění tohoto nedostatku jsem empiricky zkoumal vlivy různých transformačních operací nad výsledným obrazem, za pomocí kterých by se mi povedla odstranit ona nekompatibilita s ohledem na vypovídající hodnotu těchto operací vzhledem k účelu. Nakonec jsem se po sérii experimentů a teoretických úvah vydal níže uvedenou cestou.

Nad obrazem vygenerovaným požadovanou metodou (příklady na obrázcích 3.2a a 3.2d) provedu diskrétní dvourozměrnou konvoluci 3.2a s konvoluční maskou 5x5 s koeficienty $\frac{1}{25}$. Tím se dosáhne rozostření obrazu za použití zprůměrování hodnoty širšího okolí každého pixelu (obrázky 3.2b a 3.2e). Modelově by se dal tento proces přirovnat právě k účinkům metod ditheringu na lidské oko. To také od určité vzdálenosti přestane vnímat barvy jed-

(a) 3-bit RGB bez ditheringu (b) Použití konvoluce (c) Rozdíl oproti originálu



(d) Maticový rozptyl 3-bit RGB (e) Použití konvoluce (f) Rozdíl oproti originálu

Obrázek 3.2: Ukázka porovnávací metodiky

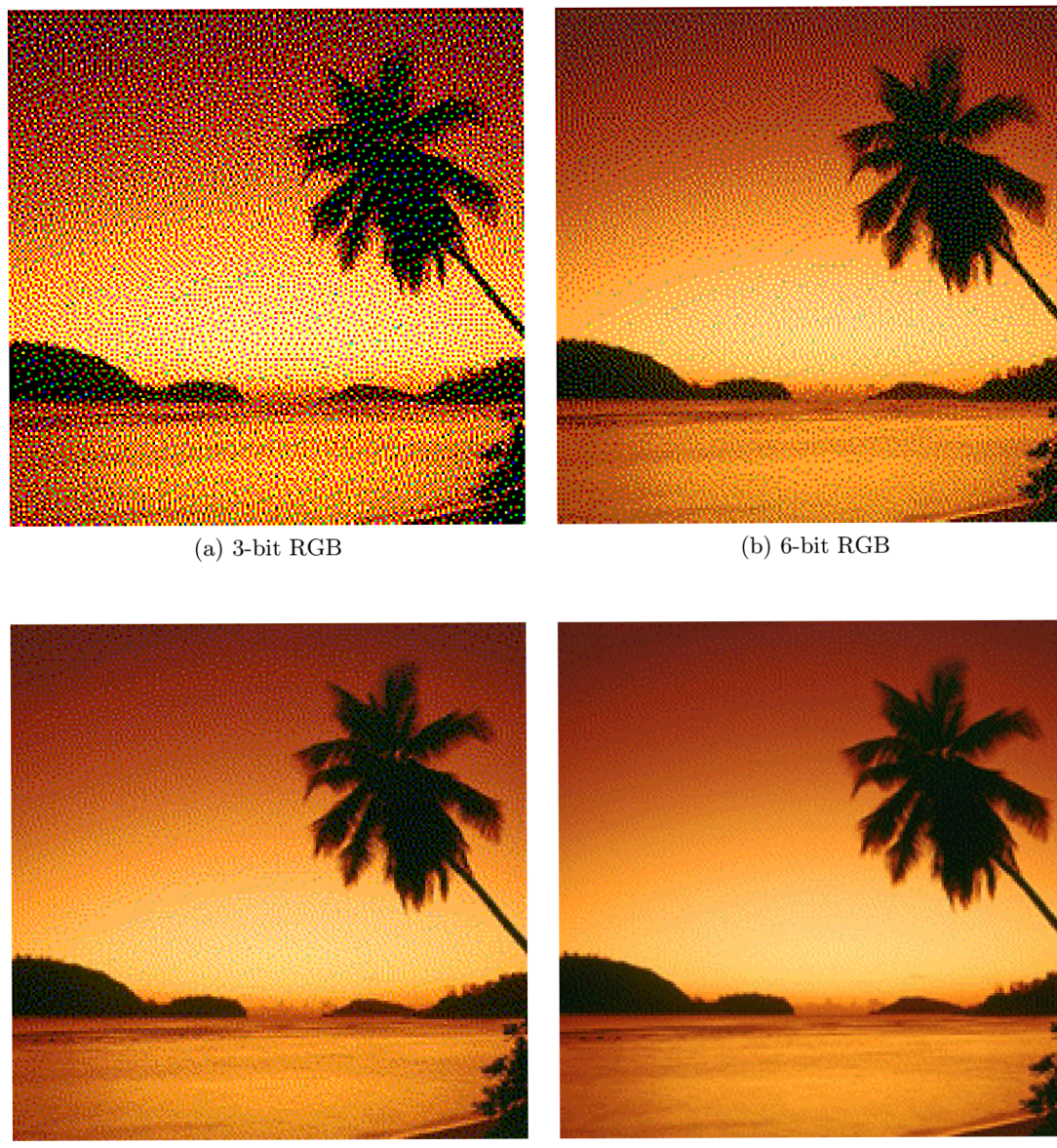
notlivých bodů shluku, ale začne je vidět jako jeden odstín. Do značné míry se přesně tímto způsobem chová právě rozostření obrazu za použití konvoluce.

Všechny metody porovnávám vůči originálnímu obrázku, nad kterým provedu stejnou konvoluci a posléze spočítám jejich rozdíl po jednotlivých barevných kanálech. Vznikne tak jakýsi diferenční obraz (obrázky 3.2c a 3.2f), který vypovídá o míře shody upraveného obrazu s originálem. V ideálním případě je tento obraz černý (rozdíl je roven nule). Čím větší je hodnota intenzity kanálu v určitém bodě, tím více se dva porovnávané obrazy v daném bodě liší.

Pro každou metodu tedy mám hodnotící faktor v podobě tohoto rozdílového obrazu, který již mohu pro jednotlivé barevné kanály statisticky vyhodnotit v porovnání s ostatními metodami ditheringu, za pomoci průměrné hodnoty, mediánu a směrodatné odchylky. Tyto hodnoty pak používám jako hodnotící kritérium kvality jednotlivých algoritmů ditheringu.

3.2.1 Souvislost s barevnými paletami

Jakékoliv porovnání vlastností metod ditheringu má smysl pouze v kontextu zadané barevné palety. Proto budu provádět výše popsanou metodiku zvlášť pro několik základních palet. Těmito paletami jsou jednobitová černobílá, 3-bitová RGB a často používaná 8-bitová RGB (paleta 3-3-2). Je pravděpodobné, že použití vícebitových nebo adaptivních palet pouze utlumí rozdíly mezi jednotlivými metodami, proto se při porovnávání zaměřím pouze na výše zmíněné.



(a) 3-bit RGB

(b) 6-bit RGB

(c) 8-bit RGB (3-3-2)

(d) 12-bit RGB

Obrázek 3.3: Rozdíly mezi barevnými paletami při použití Floyd-Steinbergovy distribuce chyby

Rozdíly mezi barevnými paletami jsou vidět na obrázku 3.3, kde je na stejný vstupní obrázek aplikována metoda Floyd-Steinbergovy distribuce chyby s různými paletami.

3.3 Aplikace

Součástí této práce je také demonstrační aplikace, která implementuje uvedené algoritmy a mechanismy vzájemného kvalitativního porovnání. Jedná se o konzolový skript napsaný

v jazyce Python 2.7[1] za pomoci knihovny Python Imaging Library 1.1.7[2], zapouzdřující práci s obrazovými daty. Primárním cílem první části tohoto programu je implementace představených algoritmů se zaměřením na dávkové zpracování většího množství souborů. Další část slouží ke zpracování výsledků jednotlivých metod za pomoci navržené srovnávací metodiky. Pro vykreslení grafů s výstupními daty je použit program gnuplot ve verzi 4.4.3[3]. Všechny potřebné knihovny v použitých verzích jsou k dispozici na CD v adresáři *program/resources* a to jak instalační balíčky pro Windows, tak ve formě zdrojových kódů pro ostatní platformy. Vývoj probíhal ve vývojovém prostředí NetBeans 6.9.1[4] pod operačním systémem Windows 7 64bit.

Aplikace slouží primárně jako podpůrný a experimentální prostředek pro porovnávání jednotlivých metod ditheringu, přičemž důraz je kladen na modulární strukturu skriptu, budoucí rozšiřitelnost a zejména na možnost snadno přidávat jakékoliv další experimentální algoritmy pro transformaci barevného prostoru vstupního obrazu. Je možné ji též využít jako prostředek pro dávkové zpracování transformace obrazu na určenou paletu za použití libovolné z metod.

Aktivně jsou používány principy objektově orientovaného programování s důrazem na modularitu a univerzalitu. Je možné tak pracovat s libovolně velkou paletou, využívat k transformaci libovolné ostatní moduly a jakkoliv s nimi experimentovat.

Vstupním bodem programu je modul `dither.py`. Zde se zpracují argumenty příkazové řádky a vytvoří se instance třídy `ImageProcessor`, v níž se inicializují objekty pro načítání a ukládání souborů a spustí se zpracovávání. Postupně se volají parametrem nadefinované moduly v adresáři `modules`, které reprezentují jednotlivé metody ditheringu. Každý z těchto skriptů obsahuje funkci `process`. Jako parametry jsou pak předávány objekt vstupního obrázku a argument pro konstruktor třídy `Palette`, reprezentující používanou paletu. Případné další parametry pro funkci `process` se pak dají nadefinovat na začátku souboru `dither.py`, kde jsou uvedeny jednotlivé skupiny metod podle palet, které se přepínají pomocí argumentu `-p`. Mezi takové přídavné parametry patří například vlastní matice pro maticový rozptyl, požadovaná hodnota manuálního prahu a další. Při vlastním experimentování se tak dá daný modul pro transformaci obrazu v inicializaci libovolně parametrizovat a tyto parametry jsou mu při spuštění předány. Tato funkce vrací instanci modifikovaného výstupního obrázku. Za dodržení této konvence je pak velmi snadné přidávat jakoukoliv další metodu, případně experimentovat se stávajícími.

Jednotlivé výstupní obrázky jsou generovány do zvoleného výstupního adresáře. Zde je vytvořena pro každý vstupní obrázek složka s jeho jménem, do ní jsou pak ukládány výstupy jednotlivých metod oddělené mezi sebou názvem metody a případnými parametry.

V modulu `compare.py` je obsažen objekt `Compare` pro přípravu dat na porovnávání a objekt `MethodsAnalyzer` pro jejich analýzu. Pro každý obrázek se provede konvoluční porovnávací mechanismus popsaný v sekci 3.2, včetně rozdílu od originálu a výsledek se uloží do výstupní složky porovnávání. V rámci analýzy výsledků se tamtéž vygenerují zdrojové a datové soubory jako podklady pro program gnuplot. Jsou celkem čtyři - průměrná hodnota, medián, rozptyl a směrodatná odchylka. Poté se zavolá přímo skript gnuplotu, pokud je v počítači nainstalován, pro vygenerování výsledných grafů.

Dalším pomocným modulem je `imageloader.py`, který obsahuje třídy `ImageLoader` a `ImageSaver` pro načítání resp. ukládání obrázků a výsledků. Výčet použitých modulů

[1] http://www.python.org/
[2] http://www.pythonware.com/products/pil/
[3] http://www.gnuplot.info/
[4] http://netbeans.org/

uzavírá `palette.py` s třídou `Palette` pro reprezentaci barevné palety a `error.py` s třídou Error pro výpis chybových a informativních hlášení programu.

Třída `Palette` umožňuje nadefinovat libovolně velkou RGB paletu a to jak s rovnoměrně rozloženým datovým prostorem pro jednotlivé barevné kanály, tak s možností různě velkého prostoru pro různé složky. Příkladem může být paleta 3-3-2. Jako druhý parametr se funkci `process` pro metody ditheringu předává hodnota parametru konstruktoru právě třídy `Palette` a značí počet bitů použitý pro uložení konkrétní hodnoty. Pokud předáme číslo, tak je paleta rovnoměrná (např. 1 pro 3-bitovou RGB paletu), pokud trojici čísel, tak se jedná o počty bitů pro červený, zelený a modrý kanál (čili pro paletu 3-3-2 je tato hodnota (3, 3, 2)).

Z jednotlivých metod ditheringu si pozornost zaslouží modul `error_distribution` reprezentující obecné zpracování obrazu metodou distribuce chyby. Ten byl navržen jako obecný základ pro všechny algoritmy distribuce chyby. Obsahuje třídu `Matrix`, která reprezentuje distribuční matici. Při vytváření instance této třídy se jako parametr předá dvourozměrné pole hodnot, které se použijí při distribuci. Aktuálně zpracovávaný prvek představuje znak X, další hodnoty reprezentují části distribuční chyby, které jsou přičteny k pixelu na místě odpovídajícímu pozici v matici. Předpokládá se přitom postup po řádcích, zleva doprava. Druhým parametrem konstruktoru třídy `Matrix` je číslo, kterým budou všechny hodnoty v matici vyděleny. Tento parametr je nepovinný, implicitně se spočítá jako suma hodnot v matici, ale je využit například v Atkinsonově distribuci, kdy není přenášena celá hodnota chyby. Pomocí tohoto mechanismu se dá nadefinovat libovolně velká a libovolně uspořádaná distribuční matice pro další experimentování.

Jednotlivé metody implementující různé způsoby distribuce chyby si pak vytvoří instanci své charakteristické distribuční matice a používají výše zmíněný obecný model. Specifickými jsou jen moduly `error_random`, resp. `error_random2` a `error_ordered`, reprezentující náhodnou distribuci chyby včetně jejího vylepšení a uspořádanou distribuci chyby. V nich se využívají instance distribučních matic z využívaných metod a aplikují se vlastním způsobem.

3.3.1 Ovládání

Program se spouští pomocí příkazu `python dither.py`. Pokud jej spustíme bez parametrů, spustí se v módu černobílé palety, načte všechny obrázky ze složky `input` a aplikuje na ně příslušné algoritmy, jejichž výstup uloží do složky `output`, rozdělené podle obrázků. Výsledky porovnávání se uloží do složky `output_compare`, včetně vygenerovaných grafů pro gnuplot.

Seznam možných parametrů programu je následující:

- `-h` vypíše nápovědu programu.
- `-d` vypne porovnávání algoritmů (provede se pouze aplikace metod ditheringu).
- `-i adresář` nastaví nový adresář pro vstupní obrázky.
- `-o adresář` nastaví nový adresář pro výstup metod ditheringu.
- `-c adresář` nastaví nový adresář pro výstup porovnávání.
- `-p paleta` nastaví množinu metod ditheringu pro příslušnou parevnou paletu. Možné hodnoty tohoto argumentu jsou: `bw`, `3bit`, `6bit`, `332`, `12bit`.

Kapitola 4

Výsledky

Cílem této kapitoly je sumarizovat výsledky mé práce, zejména co se týče kvalitativního hodnocení vlastností jednotlivých metod ditheringu. Dále shrnu dosažené poznatky získané z návrhu porovnávací metodiky a vlastních algoritmů ditheringu. Na závěr nastíním možnosti dalšího vývoje v této oblasti. Výsledné obrázky všech metod 3-bitové RGB palety je možné pro ilustraci najít v příloze A, vzájemné porovnávací grafy jsou pak v příloze B. Kompletní výsledky pro více obrázků s aplikací více palet jsou pak ke shlédnutí na přiloženém CD.

4.1 Porovnání metod

(a) Obrázek 1 (b) Obrázek 2

Obrázek 4.1: Modelové obrázky pro porovnávání metod.

Pro porovnávání metod ditheringu jsem zvolil dva modelové obrázky (4.1), na nichž budu demonstrovat výsledky. Obrázek 4.1a jako zástupce vysokofrekvenčních obrazů s mnoha barvami a častými změnami intenzity, druhý 4.1b s méně barvami a pozvolným

gradientním přechodem. V přílohách jsou výsledky a grafy pro oba tyto obrázky, výsledky použití dalších palet a vstupů jsou pak na přiloženém CD.

Srovnání vlastností se účastní všechny algoritmy uvedené v teoretickém rozboru s výjimkou specifického halftoningu (2.2.7), který mění rozměry obrázku a také Chow-kaneko adaptivního prahování (2.2.4). To se mi kvůli chybějící podrobnější specifikaci interpolační části algoritmu nepodařilo plně naimplementovat (nicméně verze bez interpolace je součástí testovacího skriptu). Zástupci automatického a adaptivního prahování (Otsu a Wellner - 2.2.3 resp. 2.2.4) jsou vzhledem ke svému zaměření testováni jen na černobílé paletě.

Veškeré techniky prahování (manuální, automatické a adaptivní) technicky nepatří mezi metody ditheringu v pravém slova smyslu, v rámci porovnávání však zůstávají jako zástupci nejtriviálnějších principů redukce barevného prostoru. V praktické aplikaci mají využití spíše v jiné oblasti.

Jednotlivé grafy s výsledky uvedené v příloze B demonstrují uvedené statistické veličiny diferenčního obrazu. Čím menší jsou tedy uvedené hodnoty, tím více se blíží výstup dané metody originálnímu obrazu. U jednobitové se diferenční obraz skládá pouze z jednoho kanálu (šedá barva, světelnost - Luminosity), u ostatních obsahuje graf vyhodnocení pro každý kanál zvlášť - červený, zelený a modrý.

Pro účely hodnocení výsledků popíšu porovnání jednotlivých metod nejprve v souvislosti s černobílou paletou, protože ta je nejvíce demonstrativní z hlediska funkčnosti samotných metod. Pokud totiž provádíme redukci u vícekanálového obrazu, můžeme si každý z těchto kanálů představit jako obraz ve stupních šedi (světelné intenzity dané složky), který převádíme na černobílou paletu (u 3-bitové RGB) nebo na stanovený počet bitů (u vícebitových palet). V druhé části už jen uvedu určitá specifika a poznatky v oblasti barevných obrazů.

4.1.1 Černobílý obraz

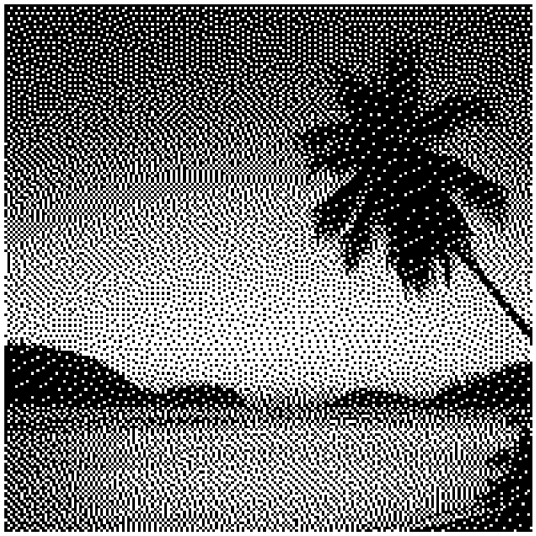
Výsledek ditheringu závisí do značné míry na charakteru obrázku. Použitá metodika však ukázala, že při relativním porovnání metod napříč různými typy obrázků vykazují jednotlivé algoritmy poměrně konstantní výsledky. Rozdíly u technik distribuce chyby jsou statisticky zanedbatelné, nicméně podle subjektivního hodnocení zde určité rozdíly existují, což reflektuje také návrh nového algoritmu.

Nejvyšší odchylky od originálu vykazují dle předpokladů "neditheringové"metody klasického prahování, a to jak manuálního, tak automatického a adaptivního, jak je vidět na grafu průměrných hodnot rozdílu od originálu na grafu 4.3. Jak jsem již ale předeslal, tak tyto metody jsou zde pouze pro srovnání výsledků při nepoužívání ditheringu. Mnohem příznivějších výsledků docílíme už při použití nejjednodušší metody, náhodného rozptylu. Vlivem její neuspořádanosti je ale dopad na kvalitu obrazu, zejména v detailech, oproti ostatním metodám viditelný.

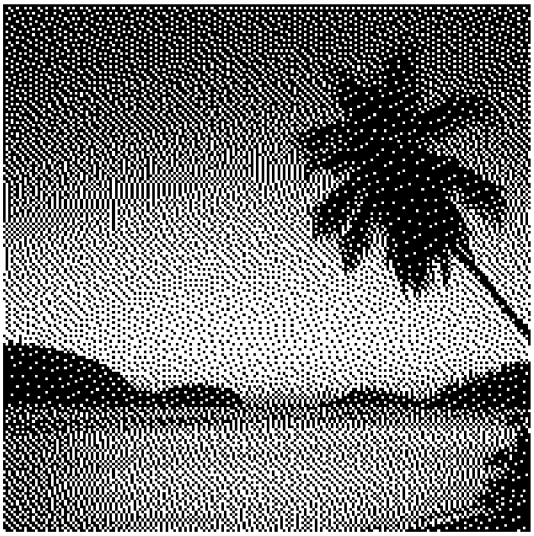
O něco lépe si vede maticový rozptyl. Zde má významný podíl použitá distribuční matice, přičemž nejlepší výsledky v tomto směru vykazuje matice M_{Bayer} uvedená v sekci 2.2.6.

S podobnými výsledky se chovají také metody založené na distribuci chyby, které jsou nejpokročilejší. Jedinou odchylkou je distribuce chyby podle Atkinsona (2.2.13), vlivem zvýšení kontrastu je rozdíl oproti originálnímu obrazu větší. Záleží na konkrétní aplikaci, zda je tento jev žádoucí či nikoliv. Rozdíly mezi ostatními metodami jsou velmi malé, přesto se subjektivně jeví jako mírně lepší tři metody - Floyd-Steinberg (2.2.9), Sierra lite (2.2.14) a Burkes (2.2.12), jejichž ukázky jsou na obrázku 4.2. Této skutečnosti jsem také využil k úpravě algoritmu náhodného chybového rozptylu (4.2e).

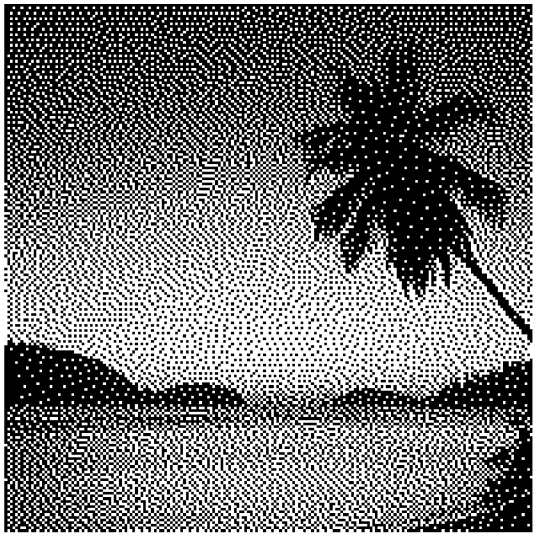
(a) Originální obrázek (zmenšeno)



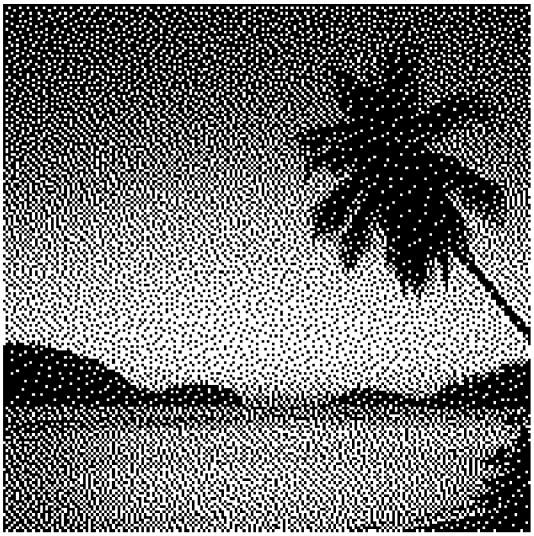
(b) Floyd-Steinberg



(c) Sierra lite

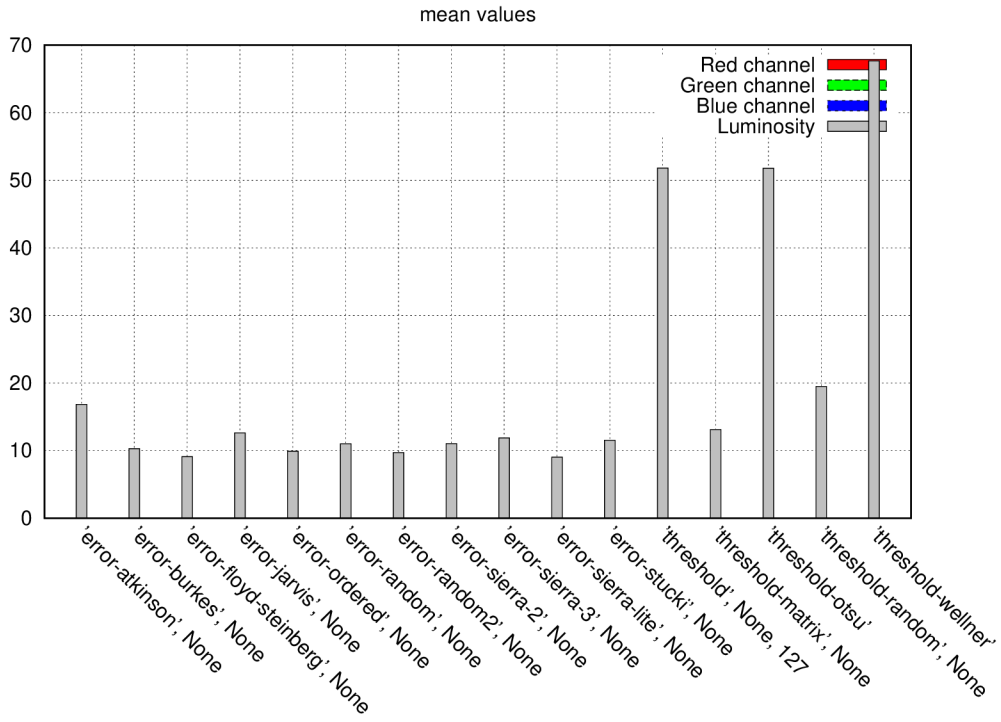


(d) Burkes



(e) Vlastní metoda - náhodná distribuce chyby

Obrázek 4.2: Metody vykazující nejkvalitnější výstup - černobílá paleta



Obrázek 4.3: Průměrné hodnoty rozdílu od originálu po alikaci metod převodu na černobílou paletu

4.1.2 Barevný obraz

Co se barevných obrazů týče, výsledky jsou velmi podobné aplikaci na černobílý obraz. Pro 3-bitovou RGB paletu jsou výsledky průměrného rozdílu od originálu vidět na grafu 4.5. Rozdíly v jednotlivých barevných kanálech jsou zanedbatelné, souvisí s barvami obsaženými v originálním obraze.

Při použití palety 3-3-2 je na výsledném grafu 4.6a jasně viditelný rozdíl modrého kanálu oproti zbývajícím dvěma. Je to dáno menším počtem barev v modrém kanálu, na který připadají pouze dva bity, oproti třem bitům u červené a zelené. Jinak jsou rozdíly mezi metodami v zásadě podobné, jako v předchozích případech.

Čím větší je použitá paleta, tím menší jsou rozdíly oproti originálu, což odpovídá původnímu předpokladu. Při použití palety 3-3-2 a nějaké z forem distribuce chyby je již výsledek velmi dobrý. Pouze u náhodného rozptylu dochází k tvorbě šumu, což při použití vícebitových palet znamená výraznější handicap oproti lépe pracujícím ostatním metodám. Proto vykazuje u některých obrazů horší výsledky než prahování, jak je vidět na obrázku 4.7.

Podobné chování (i když ne tak zřetelné) vykazuje u vyšších palet také maticový rozptyl. Zde je to způsobeno rovnoměrným rozložením prahů v distribuční matici. Pro každý shluk bodů vygeneruje obrazec složený z navzájem kontrastních odstínů, což při použití větších palet s sebou přináší značnou degradaci kvality obrazu.

(a) Originální obrázek (zmenšeno)

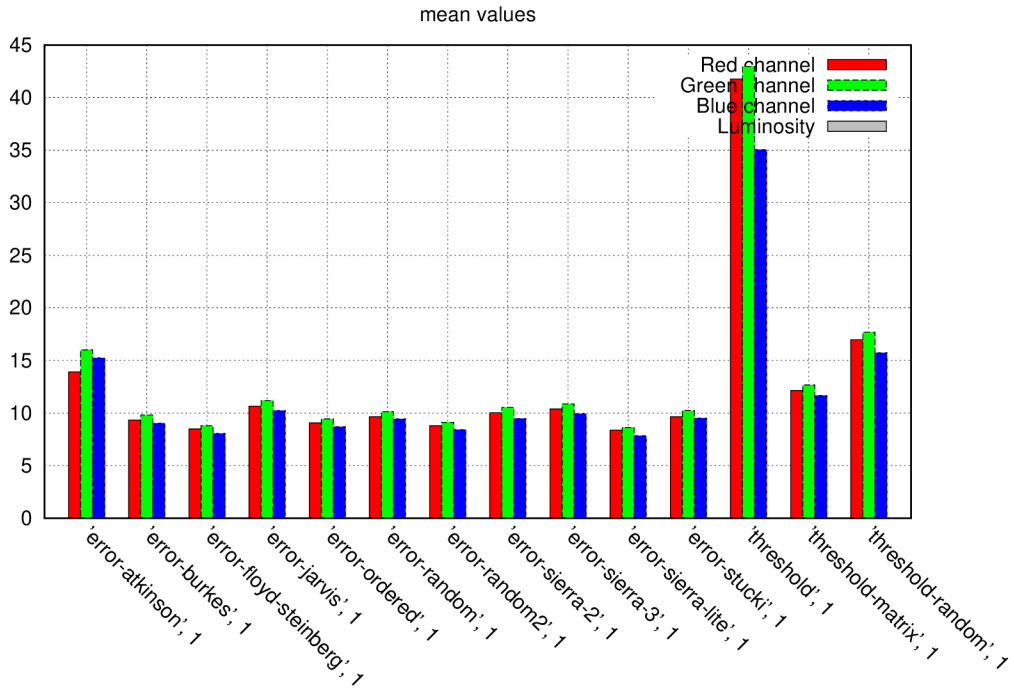
(b) Floyd-Steinberg

(c) Sierra lite

(d) Burkes

(e) Vlastní metoda - náhodná distribuce chyby

Obrázek 4.4: Metody vykazující nejkvalitnější výstup - 3-bitová RGB paleta



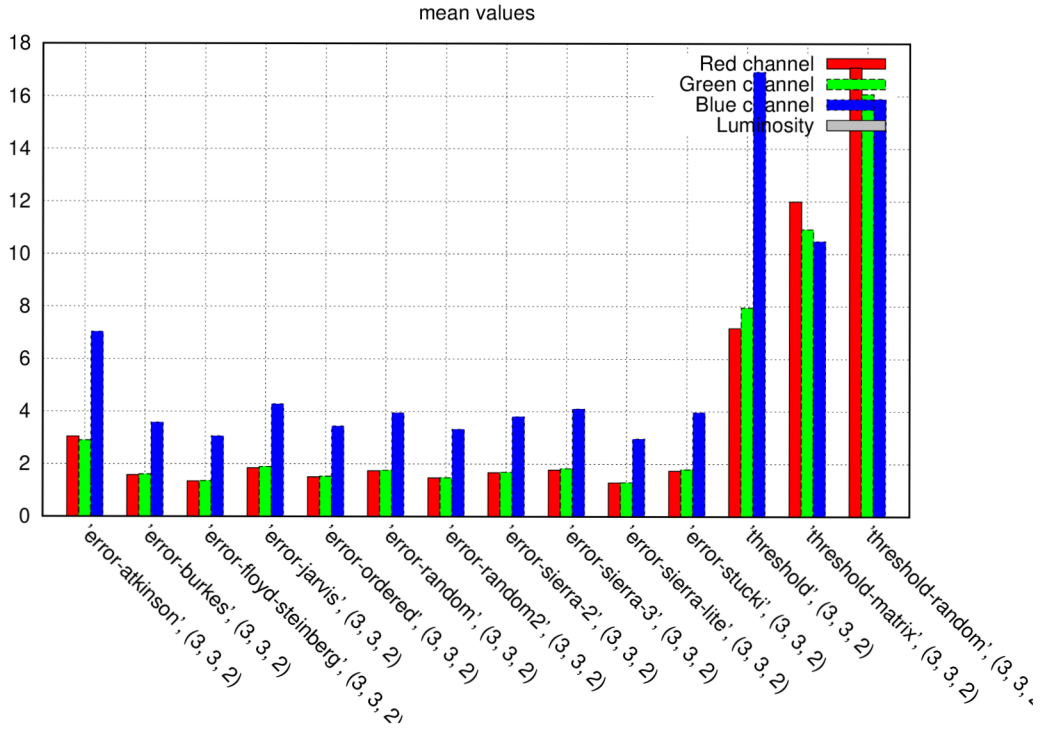
Obrázek 4.5: Průměrné hodnoty rozdílu od originálu po alikaci metod převodu na 3-bitovou RGB paletu

4.2 Porovnávací metodika

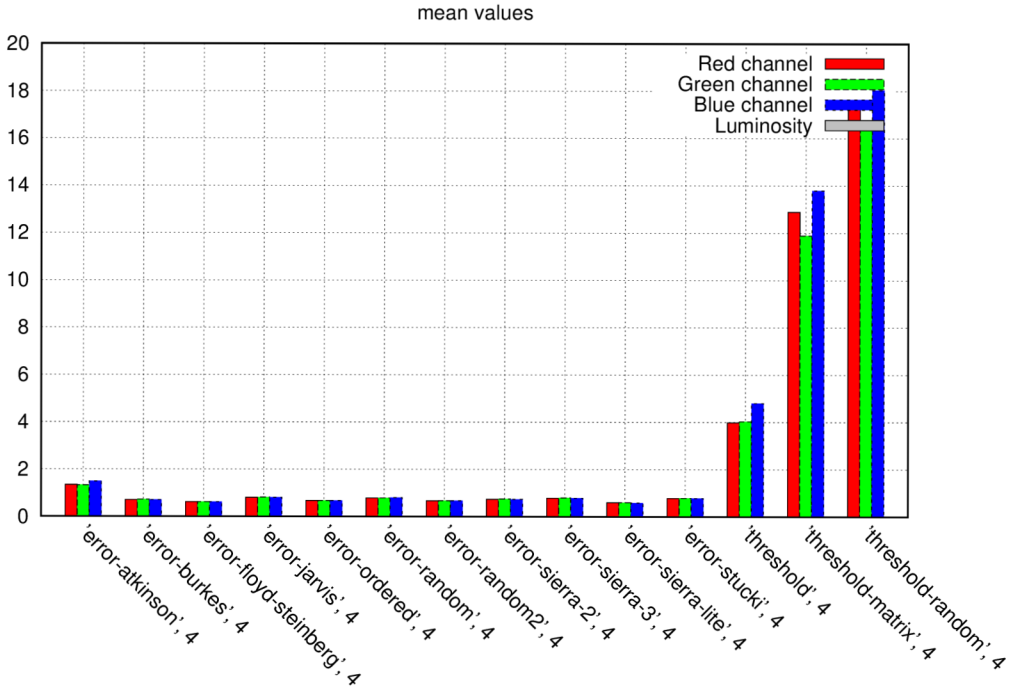
Mým úkolem v této práci bylo mimo jiné porovnat vlastnosti uváděných algoritmů. Po základních analýzách jsem se rozhodl zaměřit se na kvalitu výsledného obrazu, resp. na to, do jaké míry odpovídá obrazu původnímu. Porovnání paměťových nároků metod ditheringu jsem ponechal stranou, jelikož by se jednalo spíše o hodnocení palet jako takových.

Dalším zkoumaným parametrem by mohla být výkonnost jednotlivých algoritmů. Mým cílem bylo ale navrhnout co nejuniverzálnější aplikaci tak, aby šlo snáze experimentovat s parametry všech metod. Proto je například možné použít libovolné rozložení hodnot při distribuci chyby nebo jakoukoliv matici při maticovém rozptylu. Tato univerzalita se ale projevila v neoptimalizovanosti jednotlivých metod, což mi znemožnilo prakticky testovat rychlost jejich zpracování. Poznámky o výkonnosti jsou nicméně uvedeny už v rámci teoretického rozboru.

Hlavním problémem, který jsem v rámci návrhu metodiky řešil, bylo nalezení co nejoptimálnějšího faktoru, který by co nejobjektivněji hodnotil výsledky metod. První úvahou byla snaha o porovnávání histogramů, což ale s ohledem na snižování barevného prostoru nevykazovalo příliš dobré výsledky. Dále jsem experimentoval s mnoha různými operacemi nad obrazem, jako subsampling nebo nakonec použitá konvoluce. Pro výpočet obou těchto metod jsem zkoušel použít různé statistické veličiny, jako minimum, maximum, medián, rozptyl, směrodatnou odchylku a průměr. Nakonec jsem se po úvaze nad podstatou ditheringu rozhodl použít princip konvoluce a zprůměrování hodnoty intenzity z širšího okolí pixelu. Tato metoda se ukázala jako v praxi použitelná a vykazující uspokojivé výsledky.



(a) Paleta 3-3-2



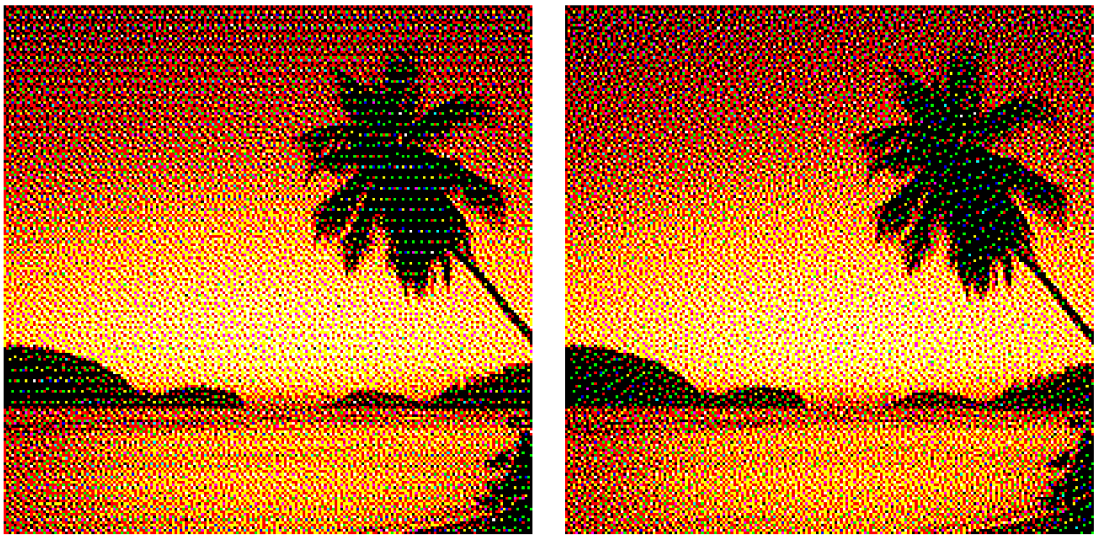
(b) 12-bit RGB

Obrázek 4.6: Průměrné hodnoty rozdílu od originálu po alikaci metod převodu na paletu 3-3-2 a 12-bitovou RGB

(a) Prahování (b) Náhodný rozptyl

Obrázek 4.7: Rozdíl mezi prahováním a náhodným rozptylem při použití palety 3-3-2

4.3 Vlastní implementované algoritmy



(a) Vlastní metoda distribuce chyby - Uspořádaná (b) Vlastní metoda distribuce chyby - Náhodná

Obrázek 4.8: Artefakty v obraze

Na základě provedených experimentů jsem se rozhodl modifikovat mnou navržený algoritmus náhodné distribuce chyby. Vybral jsem tři kandidáty vykazující nejlepší výsledky v praktických testech a to distribuce chyby Floyd-Steinberg, Sierra lite a Burkes. Takto modifikovaná metoda vykazovala o něco lepší vlastnosti než původní návrh náhodné distribuce. Subjektivně hodnotím, že také narušila určité pravidelné artefakty (viditelné u některých

metod, zvláště při plynulých přechodech), čímž vypadá výsledný obraz přirozeněji. Výsledek je vidět na obrázku 4.8b.

Prováděl jsem též pokusy s uspořádanou distribucí chyby, která byla založená na aplikaci jednotlivých distribučních matic v předem stanoveném pořadí. To mělo ale za následek spíše tvorbu výše zmíněných artefaktů (4.8a), tudíž se tato metoda příliš neosvědčila.

Kapitola 5

Závěr

V rámci této diplomové práce jsem provedl sumarizaci a důkladnou analýzu všech nalezených metod ditheringu obrazu, včetně jejich vylepšení a modifikací, se zaměřením na maximalizaci kvality výstupního obrazu. Na základě této analýzy jsem pak navrhl vlastní modifikaci, která využívá klíčových vlastností nejpokročilejších algoritmů s cílem je ještě vylepšit.

Dále jsem navrhnul metodiku pro vzájemné statistické porovnávání výstupní kvality algoritmů obrazového ditheringu a uplatnil ji pro srovnání vlastností uvedených metod. Na základě experimentálních výsledků jsem pak ještě modifikoval mnou navržený algoritmus náhodného rozptylu chyby.

Věřím, že výsledky mé práce mohou sloužit jako znalostní báze, která sumarizuje celkový vývoj v této oblasti. Dosažené poznatky se také dají využít v praktických aplikacích, například v systémech s limitovanou barevnou paletou, požadujících maximální kvalitu obrazu.

5.1 Možnosti budoucího vývoje

Oblast redukce barevného prostoru a využití ditheringu pro zvyšování kvality výstupu je zkoumána již řadu let. Proto je velmi obtížné vnést nějaké převratné změny, či zcela nové algoritmy. Stávající metody jsou velmi vyspělé a značně univerzální. Prostor pro vývoj existuje ve specializovaných aplikacích, například adaptivního prahování. Dalším možným směrem by mohlo být uzpůsobování algoritmu přímo pro konkrétní obraz, případně adaptivní využívání různých metod pro různé části obrazu.

Co se týče metodiky pro porovnání obrazu, dalším krokem by mohlo být nalezení a začlenění více vhodných charakteristik do procesu, například zkoumání ve frekvenční oblasti za pomoci Fourierovy transformace. Dále by se do komplexnějšího procesu porovnávání metod daly zařadit kromě kritérií kvalit i další aspekty, jako například výpočetní rychlost, rozdíly v kvalitě mezi různými barevnými prostory (RGB vs. CMYK) a podobně.

Stejně tak je možné doplnit i čistě subjektivní část hodnocení, v podobě uživatelských dotazníků a anket. Vzhledem k tomu, že kvalita výstupu se nedá objektivně zcela posoudit, dalším prvkem do porovnávacího procesu by se mohl stát určitý koeficient subjektivity, který by vzešel z rozsáhlešího průzkumu, na jehož počátku by byla sada různorodých obrázků a uživatelé by měli za úkol seřadit podle kvality výstupu jednotlivé metody ditheringu, aplikované na vstupní obrázky.

Literatura

[1] Adaptive Thresholding. [online], Naposledy navštíveno prosinec 2010.
URL http://homepages.inf.ed.ac.uk/rbf/HIPR2/adpthrsh.htm

[2] List of palettes. [online], Naposledy navštíveno prosinec 2010.
URL http://en.wikipedia.org/wiki/List_of_palettes

[3] Otsu Thresholding. [online], Naposledy navštíveno prosinec 2010.
URL http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html

[4] Random number generation. [online], Naposledy navštíveno prosinec 2010.
URL http://en.wikipedia.org/wiki/Random_number_generation

[5] Thresholding. [online], Naposledy navštíveno prosinec 2010.
URL http://en.wikipedia.org/wiki/Thresholding_(image_processing)

[6] Bayer, B. E.: An optimum method for two-level rendition of continuous-tone pictures. *IEEE Intl. Conf. on Communications*, ročník 1, 1973: s. 2611–2615.

[7] Bradley, D.; Roth, G.: Adaptive Thresholding using the Integral Image. *journal of graphics, gpu, and game tools*, ročník 12, č. 2, 2007: s. 13–21.

[8] Daniel Burkes: Presentation of the Burkes error filter for use in preparing continuous-tone images for presentation on bi-level devices, září 1988, CIS Graphics Support Forum.

[9] Floyd, R. W.; Steinberg, L.: An Adaptive Algorithm for Spatial Greyscale. *Proceedings of the Society for Information Display*, ročník 17, č. 2, 1976: s. 75–77.

[10] Jarvis, J. F.; Judice, C. N.; Ninke, W. H.: A survey of techniques for the display of continuous tone pictures on bilevel displays. *Computer Graphics and Image Processing*, ročník 5, č. 1, 1976: s. 13–40.

[11] Jiří Žára, J. S. P. F., Bedřich Beneš: *Moderní počítačová grafika*. Computer press, 2004, iSBN 80-251-0454-0.

[12] P. Stucki: MECCA - a multiple-error correcting computation algorithm for bilevel image hardcopy reproduction. Technická Zpráva Research Report RZ1060, IBM Research Laboratory, Zurich, Switzerland, 1981.

[13] Pierre D. Wellner: Adaptive Thresholding for the DigitalDesk. Technická Zpráva EPC-1993-110, Rank Xerox Research Centre, červenec 1993.

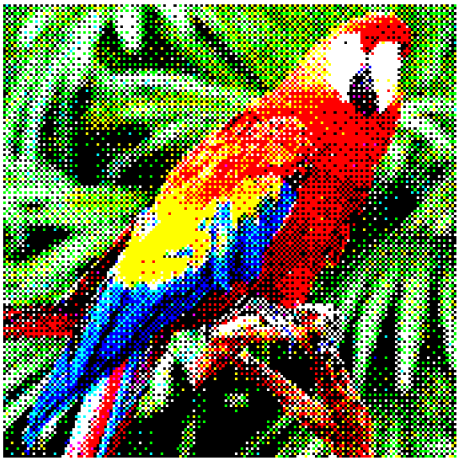
Příloha A

Výsledné obrázky 3-bit RGB

A.1 Obrázek 1

(a) Originální obrázek

(b) Manuální prahování



(c) Maticový rozptyl

(d) Náhodný rozptyl

(e) Distribuce chyby - Atkinson

(f) Distribuce chyby - Burkes

(g) Distribuce chyby - Floyd-Steinberg

(h) Distribuce chyby - Jarvis

(i) Distribuce chyby - Sierra-2

(j) Distribuce chyby - Sierra-3

(k) Distribuce chyby - Sierra lite

(l) Distribuce chyby - Stucki

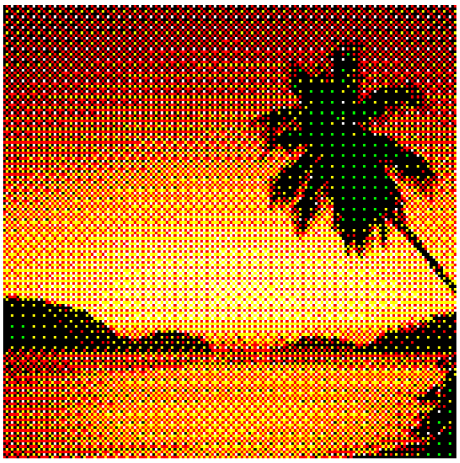
(m) Vlastní distribuce chyby - uspořádaná

(n) Vlastní distribuce chyby - náhodná vylepšená

A.2 Obrázek 2

(a) Originální obrázek

(b) Manuální prahování



(c) Maticový rozptyl

(d) Náhodný rozptyl

(e) Distribuce chyby - Atkinson

(f) Distribuce chyby - Burkes

(g) Distribuce chyby - Floyd-Steinberg

(h) Distribuce chyby - Jarvis

(i) Distribuce chyby - Sierra-2

(j) Distribuce chyby - Sierra-3

(k) Distribuce chyby - Sierra lite

(l) Distribuce chyby - Stucki

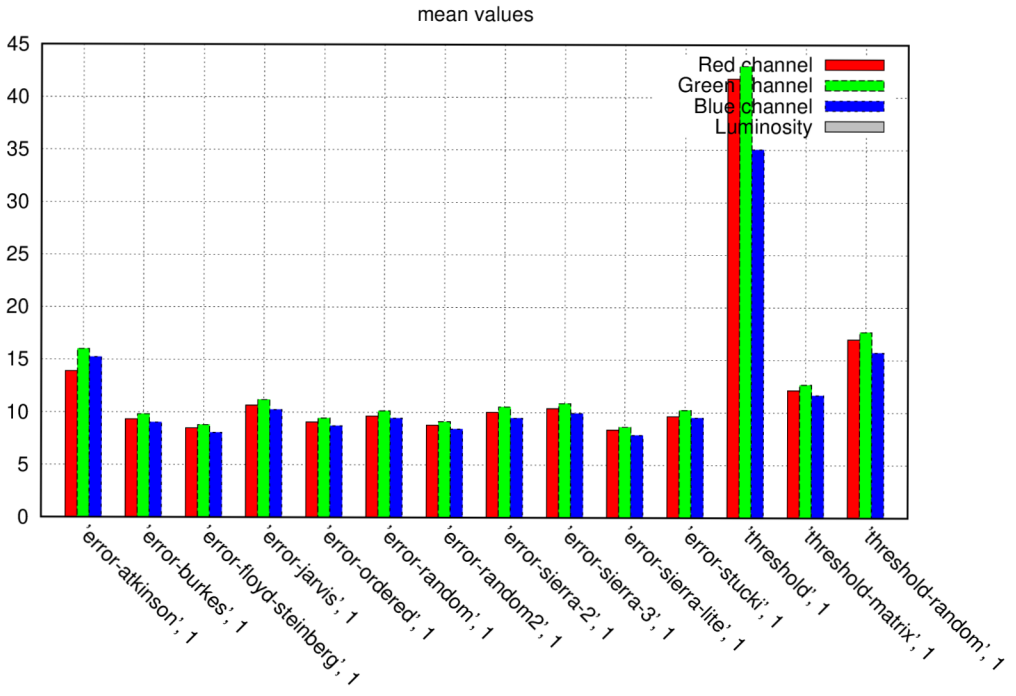
(m) Vlastní distribuce chyby - uspořádaná

(n) Vlastní distribuce chyby - náhodná vylepšená

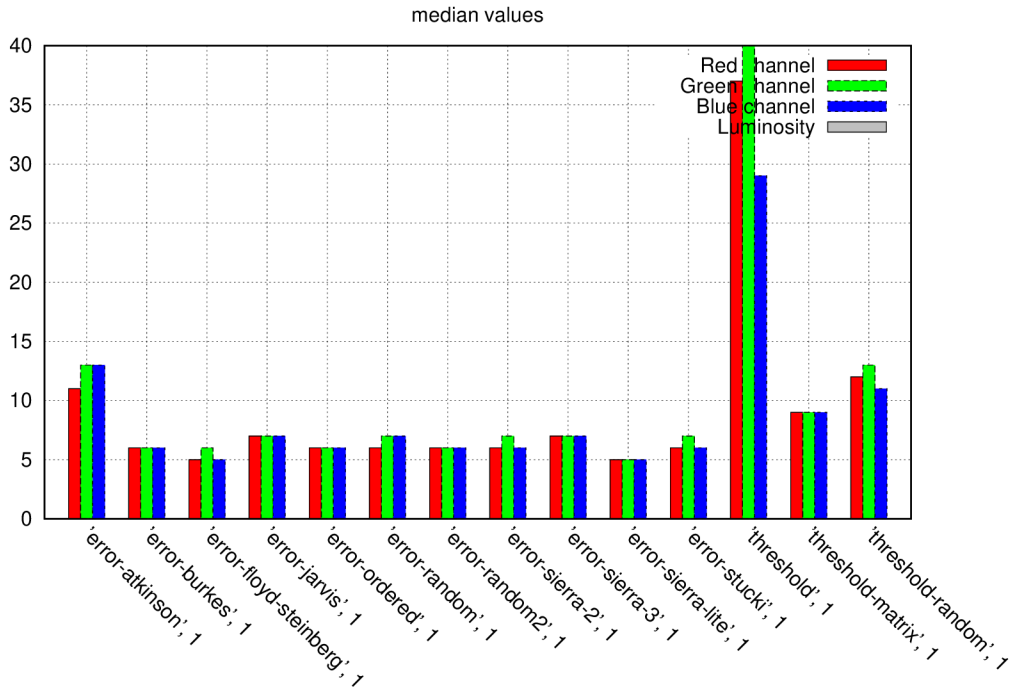
Příloha B

Výsledné grafy

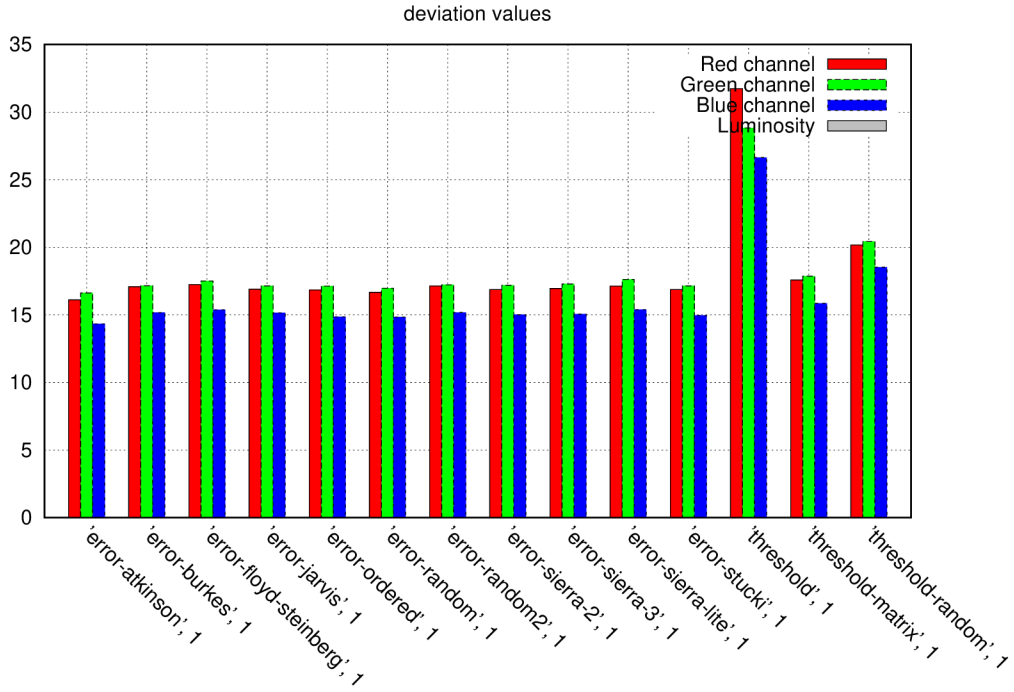
B.1 Obrázek 1



Obrázek B.1: Průměrná hodnota

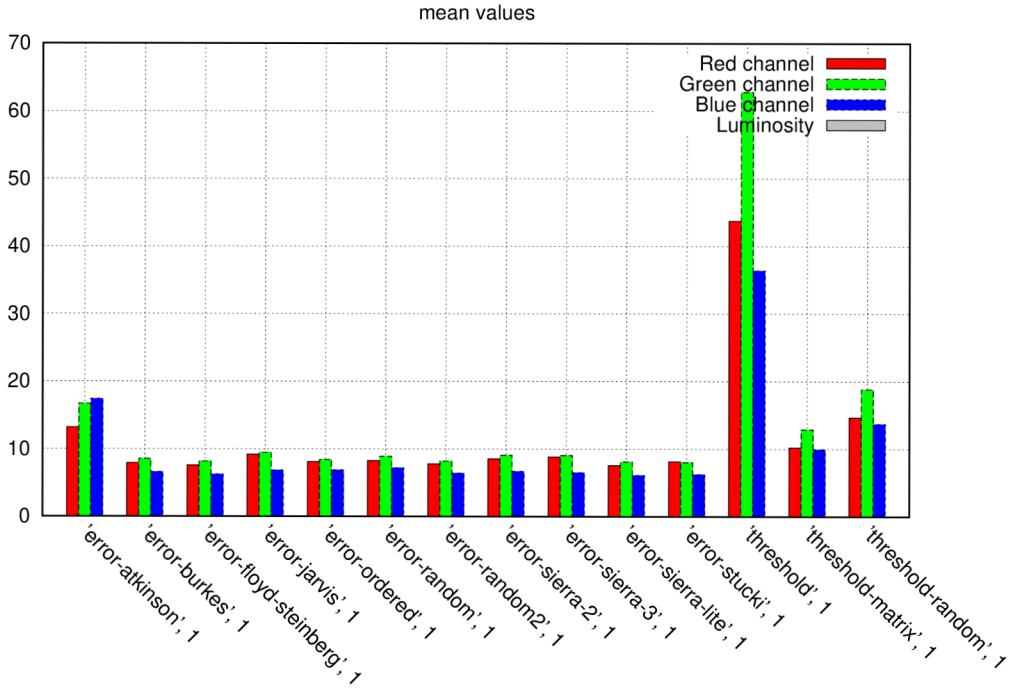


Obrázek B.2: Medián

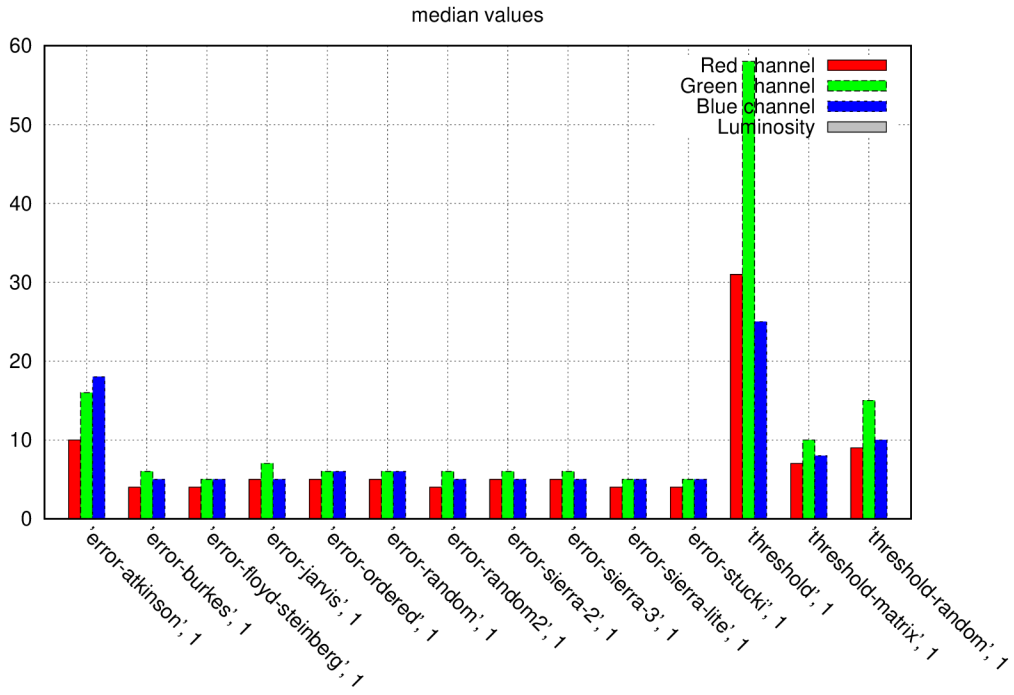


Obrázek B.3: Směrodatná odchylka

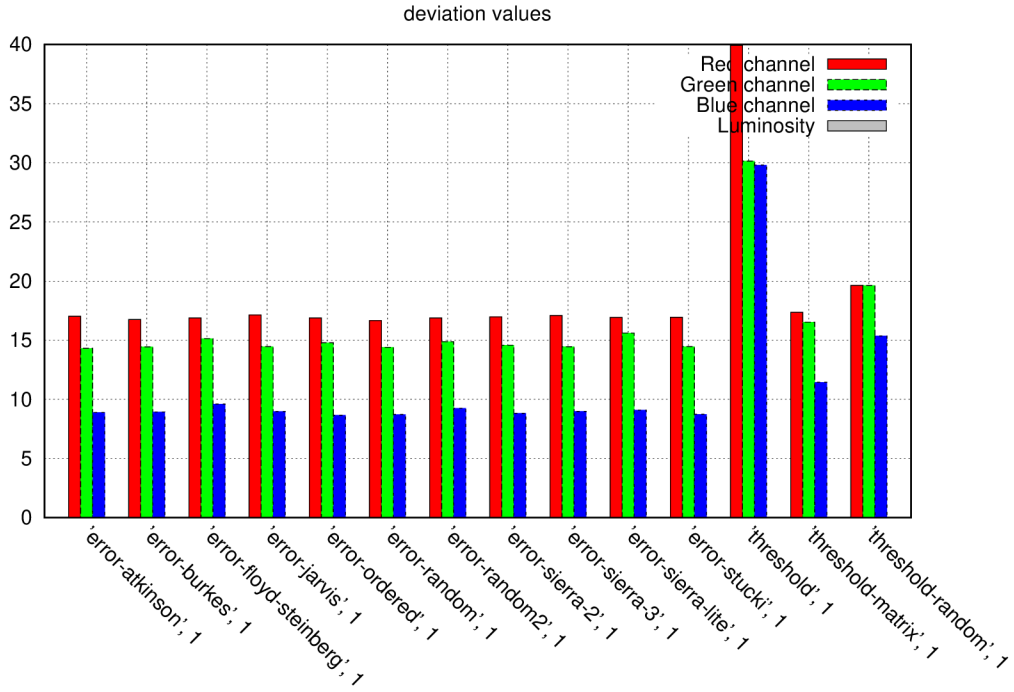
B.2 Obrázek 2



Obrázek B.4: Průměrná hodnota



Obrázek B.5: Medián



Obrázek B.6: Směrodatná odchylka