VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

PŘEVZORKOVÁNÍ A ÚPRAVY OBRAZU

RESAMPLING AND CORRECTION OF IMAGES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **VERONIKA LYSÁKOVÁ**
AUTHOR

VEDOUCÍ PRÁCE **prof. Dr. Ing. PAVEL ZEMČÍK,**
SUPERVISOR

BRNO 2017

Zadání bakalářské práce/19679/2016/xlysak02

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Lysáková Veronika**
Obor: Informační technologie
Téma: **Převzorkování a úpravy obrazu**
Resampling and Correction of Images
Kategorie: Počítačová grafika

Pokyny:

1. Prostudujte literaturu na téma vzorkování signálů a geometrické transformace obrazu.
2. Vytipujte vhodnou sadu funkcí pro řešení reálných úloh vznikajících při fotografování nebo pořizování videa.
3. Implementujte sadu funkcí podle předchozího bodu zadání na vhodné výpočetní platformě.
4. Demonstrujte funkčnost na vhodném příkladu a diskutujte vlastnosti řešení.
5. Vyhodnoťte dosažené výsledky a možnosti pokračování práce.

Literatura:

- Dle pokynů vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3 zadání

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Zemčík Pavel, prof. Dr. Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato bakalářská práce se zaměřuje na návrh a implementaci aplikace pro převzorkování a úpravu obrazu, která umožňuje posouvat pixely fotografie a tím opravovat drobné chyby při fotografování, nebo provádět korekci projekce. V této práci jsou popsány existující nástroje pro úpravu fotografií a geometrické transformace, pomocí nichž jsou tyto úpravy prováděny.

Abstract

This bachelor thesis is focusing on design and implementation of the application for resampling and correction of image which allows movement of the pixels of a photo and that way to fix small mistakes whilst taking photographs or correct the image. One part of the work describes existing tools for adjustments of photographs and geometric transformations which are being used to make these adjustments.

Klíčová slova

Převzorkování, úprava obrazu, geometrické transformace.

Keywords

Resampling, correction of images, geometry transformation.

Citace

LYSÁKOVÁ, Veronika. *Převzorkování a úpravy obrazu.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Zemčík Pavel.

Převzorkování a úpravy obrazu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana prof. Dr. Ing. Pavla Zemčíka. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....
Veronika Lysáková
16. května 2017

Poděkování

Chtěla bych poděkovat svému vedoucímu bakalářské práce prof. Dr. Ing. Pavlu Zemčíkovi za odborné vedení, pomoc a rady při vypracování této bakalářské práce.

Obsah

1 Úvod **2**

2 Obraz a jeho reprezentace **3**
2.1 Základní reprezentace . . . 3
2.2 Vzorkování obrazu . . . 3
2.3 Kvantování obrazu . . . 4
2.4 Barevné modely . . . 4
2.5 Existující řešení . . . 5

3 Geometrické transformace **10**
3.1 Homogenní souřadnice . . . 10
3.2 Dvourozměrné geometrické transformace . . . 11
3.3 Geometrické transformace diskrétního obrazu . . . 12

4 Návrh **15**
4.1 Návrh řešení . . . 15
4.2 Návrh grafického rozhraní . . . 16
4.3 Návrh mapovací funkce . . . 16
4.4 Návrh rekonstrukce obrazu . . . 17

5 Implementace **18**
5.1 Grafické rozhraní . . . 18
5.2 Zpracování obrazu . . . 19
5.3 Převzorkování obrazu . . . 20
5.4 Rekonstrukce obrazu . . . 21
5.5 Příklad použití . . . 21

6 Vyhodnocení **24**

7 Závěr **36**

Literatura **37**

Přílohy **38**

A Obsah přiloženého paměťového média **39**

B Formát souboru pro editaci mřížky **40**

Kapitola 1

Úvod

V dnešní době je moderní zaznamenávat vše, co nás nějak zaujme, nebo okamžiky, které si chceme uchovat na památku, jako obrazová nebo audiovizuální data, proto roste kvalita fotoaparátů, i těch které jsou integrovány do mobilních zařízení. Je také důležité, abychom byli schopni zobrazit představy a myšlenky, toto se ve většině případů vytváří elektronicky. Různým způsobům zobrazení předmětů, jevů a všeho ostatního, co je možné zobrazit, se věnuje počítačová grafika.

Práce s obrazem nebo fotografií a jejich grafické úpravy jsou moderní a velice žádané. Proto i do počítačové grafiky pronikly techniky, pomocí nichž lze dosáhnout různých efektů v obraze. Z tohoto důvodu existuje řada nástrojů a programů pro jejich úpravu. Tyto nástroje ovšem můžou být pro uživatele náročné na manipulaci a pochopení jejich funkcí a vlastností. Většina těchto programů má placenou licenci, a tudíž není pro všechny uživatele dostupná.

Fotografie je možné pořídit v různém rozlišení a s rozdílným přiblížením. Bohužel i přes dnešní techniku neumíme scénu oddálit. Z tohoto důvodu nelze pořídit snímek z jiného pohledu při nedostatku místa. Dále lze vytvořit panoramatické fotografie, z nich udělat výřezy, které ovšem nebudou mít vlastnosti obyčejné fotografie, protože obsahuje všechny úhly pohledu, které jsou v místě pořízení.

Cílem mé bakalářské práce je vytvořit jednoduchou aplikaci, která bude mít za úkol korekci projekce a opravu drobných chyb vzniklých při fotografování. Aplikace toto provede pomocí mapovací funkce, která zajistí převzorkování a úpravu obrazu.

V následující kapitole se budu věnovat popisu informací o obrazu a jeho reprezentaci. Poté budou popsány informace o geometrických transformacích, převzorkování a rekonstrukci obrazu v kapitole 3. Kapitola 4 obsahuje informace o návrhu řešení, jsou zde zmíněny problémy, kterým je nutné se věnovat, a schémata, pomocí nichž budu dále pracovat. V kapitole 5 jsou zmíněny informace týkající se implementace, hlavních problémů vzniklých při implementaci a vyhodnocení funkčnosti dané práce. V závěru je obsaženo shrnutí cílů práce, jak se tyto cíle podařilo naplnit a možnosti pokračování nebo rozšíření práce.

Kapitola 2

Obraz a jeho reprezentace

Tato práce se zabývá úpravou a zpracováním obrazu, proto je důležité říci, co obraz ve skutečnosti je. V této kapitole se budu věnovat nejnutnějším poznatkům o popisu obrazu a jeho reprezentaci, zmíním základní vlastnosti obrazu, proces digitalizace a jeho části, následně popíši barevné modely. Na závěr uvedu existující řešení, která se týkají úpravy obrazu.

2.1 Základní reprezentace

V první řadě je nutné si uvědomit, co znamená pojem obraz. **Obraz** je digitální podoba reálného světa, který vidíme. Tento proces se jmenuje digitalizace, zakládá se na vzorkování a kvantování obrazového signálu. Při digitalizaci dochází k přechodu od spojité funkce k funkci diskrétní, a to v definičním oboru, tak v oboru hodnot.

K formálnímu vymezení obrazu je možné použít matematický model, kterým je spojitá funkce dvou proměnných. [9]

$$f(x, y) \tag{2.1}$$

Definiční obor obrazové funkce lze zapsat jako kartézský součin dvou spojitých intervalů z oboru reálných čísel, ty vymezují rozsah obrazu, protože je možné předpokládat omezené rozměry. Jedná se o zobrazení

$$f : (< x_{min}, x_{max} > \times < y_{min}, y_{max} >) \rightarrow H \tag{2.2}$$

Nejmenší jednotka bitmapové grafiky je **pixel** neboli obrazový bod, charakterizovaný jasem a barvou. Jedná se o bod na obrazovce, který tvoří čtvercovou síť a lze jej jednoznačně identifikovat podle souřadnic. Každému takovému bodu je možné přiřadit digitálně zakódovanou barvu, která se skládá ze tří základních složek, červené, zelené a modré. **Barevná hloubka** obrazu, označována také jako hloubka pixelu, je počet bitů potřebných pro reprezentaci barvy.

2.2 Vzorkování obrazu

Vzorkováním spojité funkce $f(x)$ máme na mysli zaznamenávání hodnot v daných intervalech. Funkci získanou vzorkováním označujeme jako I_i a vzdálenost mezi vzorky Δx. Původní spojitou funkci lze definovat v jakémkoli intervalu $x \in < x_0, x_1 >$. Vzorky indexujeme pomocí

$$I_i = f(x_0 + i\Delta x), i = 0, 1, 2, ... \tag{2.3}$$

Při vzorkování využíváme periodu vzorkovacího signálu, která označuje vzdálenost mezi jednotlivými vzorky, a vzorkovací frekvenci, podle které se dané vzorky zaznamenávají. Čím je vzorkovací frekvence vyšší, tím lépe je obraz zachycen, ale nese to s sebou problémy při výpočetní náročnosti a při velikosti obrazu. Naopak při malé vzorkovací frekvenci ztratíme informaci o detailech obrazu. Při zachování podmínek **Shanonova teorému** nedochází při vzorkování ke ztrátě informace. Shanonův teorém říká, že vzorkovací frekvence musí být minimálně dvakrát větší než maximální frekvence signálu.

Během vzorkování může docházet ke ztrátě informace. Existují dva typy vzorkování bodové a plošné. U **bodového vzorkování** je hodnota daného vzorku sejmuta v jediném bodě. Během **plošného vzorkování** se zaznamenávají hodnoty celého vzorkovaného intervalu Δx.

2.3 Kvantování obrazu

Kvantování patří ke skupině úprav při redukci barev, které mají za úkol omezit paletu barev výstupního obrazu. Jedná se o rozdělení oboru hodnot obrazové funkce na intervaly, kterým je přidělena jediná *zástupná hodnota*. Podle způsobu rozdělení se může jednat o uniformní a neuniformní kvantování.

Uniformní kvantování využívá konstantní délky intervalu. Používá se častěji, protože jeho realizovatelnost v technickém vybavení je jednodušší. **Neuniformní kvantování** používá proměnnou délku intervalu, umožňuje vzít v úvahu nerovnoměrné rozložení dané veličiny. Využívá se při rekvantizaci obrazu, kdy převádíme obraz s vyšším barevným rozlišením na obrazy s nižším počtem barev.

Během procesu kvantování dochází ke ztrátě informace, ta se označuje kvantizační chyba a projevuje se jako náhlý skok barev. Tato chyba působí rušivě, u uniformního kvantování se vyskytuje častěji, protože nebere ohled na hodnoty uvnitř intervalu. [9]

2.4 Barevné modely

V dnešní době existuje široká škála barev a barevných odstínů. Pomocí barevných modelů je možné pracovat s barvami a míchat je. Těchto modelů existuje řada, usilují o napodobení barev co nejpřesněji. Níže uvedu dva modely, které se používají nejčastěji, model RGB, jenž se využívá u digitálních fotografií, a model CMYK, který se používá pro tisk.

2.4.1 Model RGB

Tento model je nejrozšířenější barevný model v počítačové grafice. Využívá aditivního způsobu míchání barev, jedná se o míchání vyzařovaného světla, proto nepotřebujeme světlo vnější. Při aditivním způsobu míchání barev se míchá červené, zelené a modré světlo, ze kterého následně vzniká výsledná barva. Tyto barvy jsou vnímatelné lidským okem. Barvy se dají vyjádřit pomocí desítkové soustavy čísly od 0 do 255. Kombinací těchto čísel určujeme barvu i intenzitu světla.

RGB model lze zobrazit jako krychli, kde kolmá hrana udává škálu barevné složky, potom bod se souřadnicemi (r,g,b) v krychli představuje hodnotu výsledné barvy. Model této krychle je zobrazen na obrázku 2.1[1] .

[1] Obrázek 2.1 je převzat z [3]

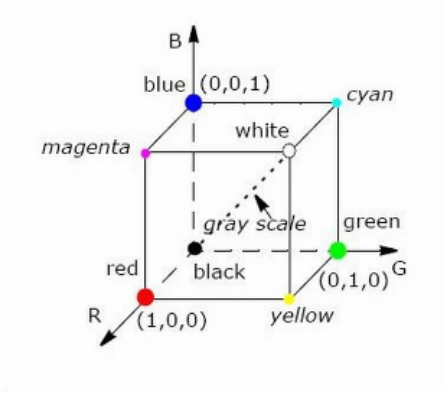
Další variantou RGB je také RGBA, u kterého je přidán alfa kanál, jenž nese informaci o průhlednosti.

Tento barevný model se využívá při vytváření fotografií nebo v různých zařízení, které přidávají barvy a světlo na tmavý podklad, mezi ně patří například televize, projektory a monitory.

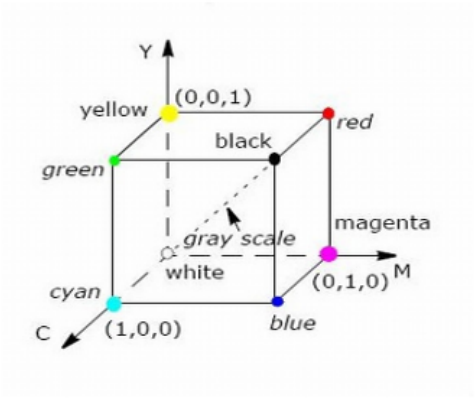
2.4.2 Model CMYK

Model CMYK používá subtraktivním míchání barev, které se využívá především u reprodukčních zařízení. Během subtraktivního způsobu míchání barev se používají tři základní barvy azurová, purpurová a žlutá. Při přidávání sytosti základních barev dostáváme tmavší barvu. Pokud budou základní barvy v nejvyšší intenzitě vznikne černá barva, bohužel ale není příliš kvalitní, z tohoto důvodu lze přidat čtvrtou barvu černou.

Stejně jako u barevného modelu RGB, také barevný model CMY lze zobrazit pomocí krychle. Libovolný bod se souřadnicemi (c,m,y) v krychli představuje hodnotu výsledné barvy.



Obrázek 2.1: Barevný model RGB



Obrázek 2.2: Barevný model CMY

2.5 Existující řešení

Práce s obrazem je velice populární, uživatelé vyžadují různé úpravy obrazu, korekci barev, moderní jsou i deformace obrazu neboli warping. Z tohoto důvodu existuje spoustu editorů a programů, které podporují různé operační systémy a různé funkce pro úpravu fotografií. V této části se budu zabývat nejznámějšími existujícími řešeními, která mají za úkol úpravu a editaci obrazu. Dále zmíním nástroj pro tvorbu uživatelského rozhraní, který využiji při vypracování bakalářské práce.

2.5.1 Existující aplikace

Adobe Photoshop

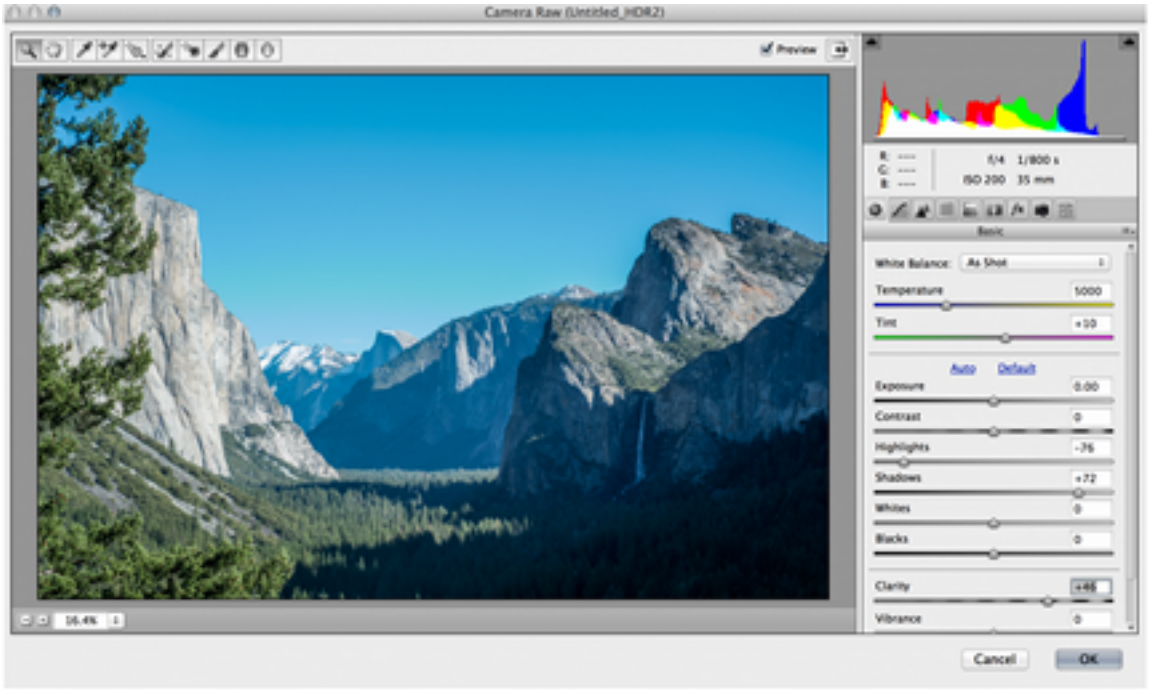
Je jedním z celosvětově nejrozšířenějších programů pro úpravu a editaci fotografií. Jedná se o bitmapový grafický editor od společnosti Adobe Systems. Tato aplikace je jako zkušební verze s následnou placenou verzí dostupná pro operační systémy Mac OS X, Microsoft Windows, Linux.

Adobe Photoshop poskytuje řadu možností grafických úprav a nástrojů pro práci s obrazem od jednoduchých retuší a mazání objektů z obrázku až po komplexní 3D návrhy a ilustrace. Nejnovější verze Adobe Photoshop CC dokonce umožňuje i využívat prostor cloud o velikosti 2 GB, s možností dalšího rozšíření. Tento editor má mnoho funkcí a nástrojů, z toho důvodu existuje i hodně výukových programů a tutorialů.

Navíc obsahuje základní editor pro úpravu videa, který má integrovány tradiční nástroje. Editory jsou rozděleny do dvou prostorů: první na pracovní ploše a druhý v panelech, kde si uživatel může nastavit a vybrat nástroje a upravit je pro svou potřebu. [5]

Program ovšem není pro každého, jak pro svou cenu, tak pro strmé křivky učení.

Screenshot této aplikace je znázorněn na obrázku 2.3[3] .



Obrázek 2.3: Screenshot aplikace Adobe Photoshop

Zoner Photo Studio

Jedná se o český program pro úpravu fotografií, který nabízí širokou škálu možností. Je k dispozici jako zkušební verze s následnou placenou podporou nebo jako placená verze, díky které budete mít neustále aktuální aplikaci dostupnou pro operační systém Microsoft Windows.

Tento program má řadu výhod a funkcí, umožňuje například rychlé stažení fotografií do počítače, rozpoznání a roztřídění fotografií do složek, jejich přejmenování a popsání. Mezi další funkce patří udržování pořádku ve fotoarchivu, pomocí popisků a klíčových slov pro rychlejší nalezení, barevných štítků, prohlížení podle data, katalogu pro rychlejší práci s fotkami.

[3]Obrázek 2.3 je převzat z http://blogs.adobe.com/richardcurtis/2013/05/17/creativefriday-hdr-in-photoshop-cc-and-acr-toning/

Lze také tvořit fotoknihy, kalendáře a obrazy, nebo zasílat pohlednice přímo z programu. Zoner Photo Studio nabízí mnoho nástrojů a filtrů pro fotografie, nedestruktivní úpravy a podporu RAW a bitmapových formátů. [7]

Screenshot této aplikace je znázorněn na obrázku 2.4[4] .

Obrázek 2.4: Screenshot aplikace Zoner Photo Studio

PhotoScape

Jedná se o multifunkční grafický program pro úpravu fotografií, který nabízí řadu možností.Patří mezi ně prohlížení obrázků buď v okně, nebo v režimu celé obrazovky, úprava fotek, prohlížení EXIF metadat, tisk a mnoho dalších. Tento program podporuje spoustu jazyků, mezi ně patří i čeština a slovenština. Dokonce lze používat i české fonty u editačních a manipulačních operací.

Tento editor má centralizovaný přístup k funkcím aplikace a hlavní funkce jsou dostupné pomocí záložek hlavního okna. Mezi další výhody patří kontextové menu, které je dostupné i pro prohlížení obrázku v režimu celé obrazovky. V tomto menu nalezneme například funkce pro otáčení obrázků, odstranění EXIF metadat nebo také automatické korekce barev.

K úpravě fotografií se využívají základní funkce změna velikosti obrázku, úprava barev, zaostření fotografií, automatické nastavení kontrastu a barev, úprava světla. Mezi složitější funkce vytvoření rámečku, mozaiky, přidání textu nebo odstranění červených očí. Dále je tu možnost hromadné úpravy fotek, slučování fotografií do jedné, aj.

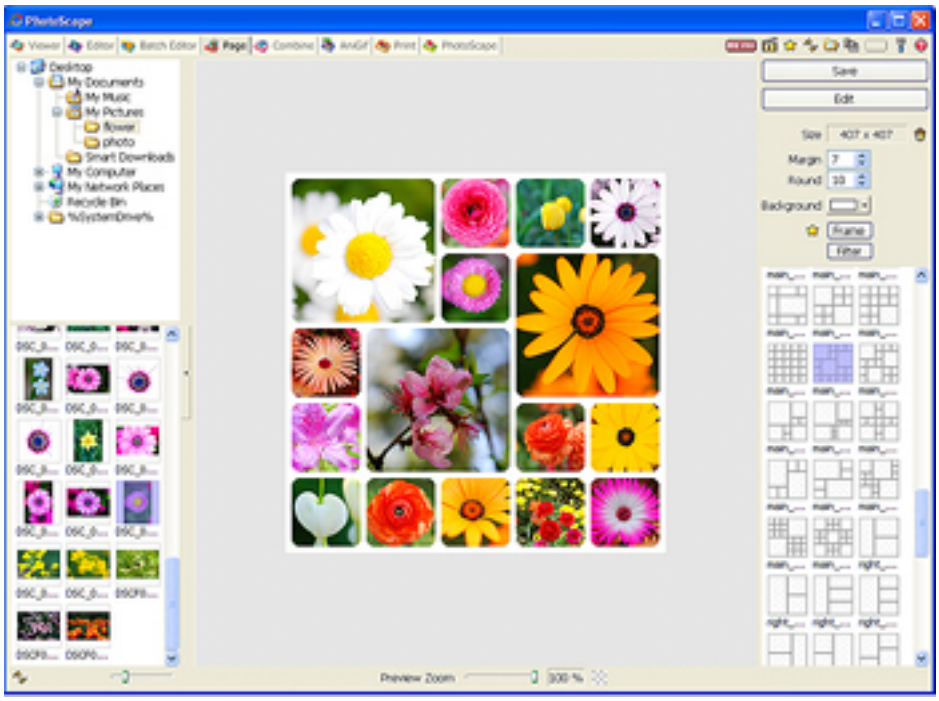
Další funkce umožňují vytvářet animované GIFy, zjišťovat barvy, rozdělovat fotky na menší části, konvertovat fotografie uložené ve formátu RAW do formátu JPG a hromadně přejmenovávat fotografie. Tento program dokáže i vytvářet linkované sešity, kalendáře, notové osnovy nebo rozvrh hodin.

K tomuto programu jsou zajímavé návody, jak upravovat obrázky nebo pracovat s aplikací. Program je distribuován zdarma, a je dostupný pro operační systémy Windows. Vý-

[4]Obrázek 2.4 je převzat z [7]

hodou tohoto editoru je jednoduchá manipulace a nulová cena, ale pro náročnější uživatele a složitější úpravy fotografií není dostačující. [2]

Screenshot této aplikace je znázorněn na obrázku 2.5[5] .



Obrázek 2.5: Screenshot aplikace PhotoScape

2.5.2 Existující nástroje

Knihovna OpenCV

OpenCV je otevřená multiplatformní knihovna, která slouží pro manipulaci s obrazem. Zaměřuje se na zpracování obrazu v reálném čase. Autorem je společnost Intel. Knihovna je vyvíjena v jazyce C a C++, proto tato knihovna může využít vícejádrové zpracování.

Tato knihovna obsahuje více než 2500 optimalizovaných algoritmů, které je možné použít pro detekci a rozpoznání obličeje, identifikaci objektů, sledování pohybů kamery, extrahování 3D objektů, odstranění červených očí a spoustu dalších operací.

Knihovnu lze využít na operačních systémech Linux, Windows, MAC OS a Android, podporuje jazyky C, C++, Python a Java. Tato knihovna je distribuována pod BSD licencí. [4]

Qt

Jedná se o multiplatformní nástroj, který se používá pro návrh a tvorbu grafického uživatelského rozhraní. Tento nástroj byl vytvořen v roce 1999 společností Trolltech. Aplikace je možné distribuovat pod licencí GPL, LGPL, nebo po splnění určitých podmínek komerčně.

Qt umožňuje jednoduchou práci s grafickým rozhraní, jednoduše vytvářet objekty a pak s nimi dále pracovat. Pomocí objektů lze vytvářet tlačítka, informační okna nebo různé

[5]Obrázek 2.5 je převzat z http://www.photoscape.org/ps/main/screenshot.php

předpracované widgety. Objekty mohou být i prvky, které slouží k správnému běhu aplikace. S objekty se pracuje pomocí slotů a signálů, díky kterým je možné specifikovat akci, jež proběhne při použití daného objektu.

Qt je knihovna pro programovací jazyk C++, existuje i pro jazyky Python, Ruby, C, Perl, Pascal, C#, Java a Haskell. Výhodou je kvalitně zpracovaná dokumentace, která zjednodušuje práci s nástrojem. Také vývojové programy Qt Designer nebo Qt Creator. Aplikace využívají nativní vzhled operačního prostředí, proto se přizpůsobí vzhledu daného prostředí.

Podporovanými platformami jsou desktopové, mobilní, a další. Mezi desktopové patří Windows, Linux, macOS, k mobilním Android, iOS, další vestavěné Linux platformy. [6]

Kapitola 3

Geometrické transformace

V předchozí kapitole jsou zmíněny základní informace o obraze a jeho reprezentaci. V této kapitole se budeme zabývat tématem geometrických transformací, převzorkování a rekonstrukce obrazu. Obsahuje přehled informací, které budou dále využity při návrhu a implementaci práce.

Geometrické transformace jsou jedny z nejčastěji používaných operací v počítačové grafice. Lze je rozdělit na dva typy **lineární** a **nelineární**. Mezi lineární transformace patří otáčení, posunutí, změna měřítka, zkosení a operace vzniklé jejich skládáním. S nelineárními transformacemi se setkáváme při složitějších změnách tvaru grafických objektů, mezi ně patří např. deformace prostorových modelů nebo warping obrazu.

Objekty jsou popsány souřadnicemi, které náleží ke zvolenému souřadnicovému systému. Geometrické transformace mohou být aplikovány na jednotlivé souřadnice objektu, který tak může měnit svou polohu. Další možností je podrobit transformaci souřadnicový systém. Tímto dosáhneme výhodnější reprezentace objektu pro jeho další zpracování. Typickým příkladem je umístění objektu do scény s více objekty. Při porovnání polohy různých objektů je nutné přepočítat jejich geometrické údaje, buď mezi jejich souřadnicovými systémy, nebo z lokálních souřadnicových systémů do světového souřadnicového systému společného pro všechny objekty. [9]

3.1 Homogenní souřadnice

Pro zjednodušení výpočtu transformací se používá reprezentace bodů pomocí homogenních souřadnic. Homogenní souřadnice umožňují vyjádření nejčastěji používaných lineárních transformací pomocí jediné matice, to však v nehomogenních kartézských souřadnicích není možné. Další transformací, která lze vyjádřit pomocí homogenních souřadnic, je perspektivní promítání. Tento způsob vyjádření transformací je výhodný, protože pro jejich implementaci je možné využít existující knihovny pro práci s maticemi. Skládání transformací se realizuje jako násobení matic, inverzní transformace je reprezentována inverzní maticí, atd. [9]

3.2 Dvourozměrné geometrické transformace

3.2.1 Posunutí

Posunutí neboli translace bodu P je určeno vektorem posunutí $\overrightarrow{p} = (X_t, Y_t)$ a je dáno vztahem

$$x^{'} = x + X_t, \quad y^{'} = y + Y_t \tag{3.1}$$

Matice transformace T a inverzní matice T^{-1} mají tvar [9] [3]

$$\mathrm{T(X_t, Y_t)} = \begin{bmatrix} 1 & 0 & X_t \\ 0 & 1 & Y_t \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.2}$$

$$\mathrm{T^{-1}(X_t, Y_t)} = \begin{bmatrix} 1 & 0 & X_t \\ 0 & 1 & Y_t \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

3.2.2 Otáčení

Otáčením bodu v rovině s homogenními souřadnicemi P o úhel α kolem počátku souřadnicového systému získáme bod $P^{'}$ se souřadnicemi

$$x^{'} = x\cos\alpha - y\sin\alpha, \quad y^{'} = x\sin\alpha + y\cos\alpha \tag{3.4}$$

Matice transformace otáčení R a inverzní matice R^{-1} mají tvar [9] [3]

$$\mathrm{R} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.5}$$

$$\mathrm{R^{-1}} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

3.2.3 Změna měřítka

Změna měřítka bodu v rovině s homogenními souřadnicemi P a faktory změny měřítka S_x, S_y je dána vztahem

$$x^{'} = x \cdot S_x, \quad y^{'} = y \cdot S_y \tag{3.7}$$

Transformační matice S změny měřítka a inverzní matice S^{-1} je ve [9]

$$\mathrm{S} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.8}$$

$$\mathrm{S}^{-1} = \begin{bmatrix} \frac{1}{S_x} & 0 & 0 \\ 0 & \frac{1}{S_y} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

Je-li faktor změny měřítka $S_{x,y}$ v intervalu $(0, 1)$, pak dochází k zmenšení a přiblížení daného objektu k počátku souřadnic. Je-li $S_{x,y}$ větší než jedna, poté dojde ke zvětšení. Pokud je $S_{x,y}$ záporné, dochází k převrácení daného objektu. [3]

Souměrnost je zvláštní případ změny měřítka, kdy absolutní hodnota faktoru změny měřítka $S_{x,y}$ je rovna jedné. Souměrnost lze ve dvourozměrném případě rozdělit na středovou a osovou souměrnost. Středová souměrnost vznikne změnou měřítka $S_x = -1$ a $S_y = -1$, daný objekt vzniká otáčením o 180° podle počátku souřadnicové soustavy. Pokud osová souměrnost podle osy x má koeficienty $S_x = -1$ a $S_y = 1$, podle osy y $S_x = 1$ a $S_y = -1$, jedná se o překlopení bodu podle jedné ze souřadnicových os. [9]

3.2.4 Zkosení

Zkosení bodu v rovině s homogenními souřadnicemi P a faktory zkosení S_{hx} a S_{hy} je dáno vztahem

$$x' = x + S_{hx} \cdot y, \quad y' = y + S_{hy} \cdot x \tag{3.10}$$

Transformační matice zkosení S_H a inverzní matice S_H^{-1} mají tvar [3]

$$\mathrm{S_H} = \begin{bmatrix} 1 & S_{hx} & 0 \\ S_{hy} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.11}$$

$$\mathrm{S_H^{-1}} = \begin{bmatrix} 1 & -S_{hx} & 0 \\ -S_{hy} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.12}$$

3.3 Geometrické transformace diskrétního obrazu

Geometrické transformace popsané v kapitole 2.2 pracují s bodem, nebo očekávají spojitou reprezentaci obrazu. Diskrétní obraz je jiné povahy, a to s sebou nese problémy. Obecná geometrická transformace přiřazuje pixelu s diskrétními souřadnicemi $[i, j]$ nějaké neceločíselné místo $[x, y]$. V novém obraze tak mohou vznikat „díry“, nebo se může několik pixelů namapovat na jedno místo.

Mějme dvourozměrný obraz složený z bodů $[x, y]$, pak geometrická transformace obrazu má tvar

$$T(u, v) = [x(u, v), y(u, v)] \tag{3.13}$$

Vstupy funkcí $x(u, v)$ a $y(u, v)$ jsou souřadnice u a v. Funkce vrací novou polohu pixelu.

Způsob přiřazení pixelu z jednoho obrazu do druhého se nazývá **mapování**. Je možné jej rozdělit na dopředné mapování a zpětné mapování. Při **dopředném mapování** procházíme pixely vstupního obrazu A a hledáme jejich umístění ve výstupním obraze B. Naopak při **zpětném mapování** procházíme pixely výstupního obrazu B a hledáme odpovídající

oblasti v obraze A. Problém při dopředném mapovaní je, že ve výsledném obraze mohou vznikat volná místa.

Transformace může obraz zvětšovat nebo zmenšovat, tím nemusí zachovat velikost vstupního obrazu. Způsobuje to problém při dopředném i zpětném mapování, kdy jednomu pixelu ve výstupním obraze se přiřadí několik pixelů, nebo žádný ze vstupního obrazu. [9]

3.3.1 Převzorkování a rekonstrukce

Převzorkování je posloupnost operací, kdy se signál nejprve rekonstruuje, poté se s ním pracuje a opět se vzorkuje zpět do diskrétní oblasti. Obraz není nutné rekonstruovat celý najednou, ale pouze oblast, která se používá pro výpočet hodnoty nového pixelu. Rekonstrukce, transformace a nové vzorkování jsou úzce spjaty. Podstatou převzorkování je nalezení spojitého obrazu k obrazu diskrétnímu. [9]

Rekonstrukcí rozumíme přechod od diskrétní funkce I_i ke spojité funkci $f(x)$, kde $i = 1, 2, ...,$ a $x \in R$. Obor hodnot obou funkcí je stejný, jedná se o reálná čísla. Rekonstrukce je poměrně složitá úloha, protože hodnoty mezi vzorky se musí dopočítávat. [9]

Mějme funkci $h(x)$, které budeme říkat konvoluční jádro, nebo také rekonstrukční filtr. Jedinou podmínkou je, že suma koeficientu vymezená touto funkcí musí být jednotková

$$\int_{-\infty}^{+\infty} h(t)dt = 1 \tag{3.14}$$

Pokud je hodnota integrálu větší než jedna, nová funkce byla konvolucí zesílena a naopak. Při rekonstrukci spojité funkce $f(x)$ vynásobíme hodnotu vzorku v bodech I_i konvolučním jádrem, které posuneme do bodu $i\Delta x$. [9]

3.3.2 Interpolace nejbližším sousedem

Jedna z nejjednodušších metod rekonstrukce obrazu. Výpočet spojité funkce $f(x)$ z diskrétní funkce I_i definované v diskrétních bodech $i = 1, \ldots, n$ a konvoluční jádro má tvar

$$f(x) = I_i; \quad i - \tfrac{1}{2} < x \leq i \ + \tfrac{1}{2}, \tag{3.15}$$

$$\mathrm{h(x)} = \begin{cases} 1 & \text{pro } 0 \leq |x| \ < \frac{1}{2} \\ 0 & \text{pro } \frac{1}{2} \ \leq |x| \end{cases} \tag{3.16}$$

Principem této metody je okopírování nejbližší hodnoty v okolí vzorku. Tato metoda je velice rychlá, implementuje se podobně jako zaokrouhlení reálného čísla. Na druhou stranu vede k nežádoucím efektům, např. k poškození čar, které při zmenšení zmizí, nebo zvýraznění skoku u zvětšení. Z důvodu rychlosti se metoda používá k rychlému náhledu.

U této metody se projevuje pixelizace obrazu, obraz je složen z mozaiky čtverců. Je to způsobené aproximací funkce $f(x)$ funkcí obecně nespojitou. [9]

3.3.3 Lineární a bilineární interpolace

Lineární interpolace je metoda prokládání křivek, použitím lineárních mnohočlenů. Jsou-li dány dva body i a $i+1$ s hodnotami I_i a I_{i+1} a hledáme hodnotu $f(x)$ v bodě x, v intervalu $i < x \leq i+1$. Lineární interpolace je úsečka mezi krajními body a hledaná hodnota v bodě x se vypočítá pomocí

$$f(x) = I_i + \frac{x - i}{\Delta x}[I_{i+1} - I_i] \tag{3.17}$$

Konvoluční jádro má tvar jehlanového stanu, nazývá se tent function a má tvar

$$\mathrm{h(x)} = \begin{cases} 1 - |x| & \text{pro } 0 \leq |x| < 1 \\ 0 & \text{pro } 1 \leq |x| \end{cases} \tag{3.18}$$

Bilineární interpolace pro rekonstrukci hodnoty pixelu používá čtyři pixely z jeho okolí. Bilineární interpolace je separabilní, je možné ji složit z jedné operace postupně aplikované na řádky a pak na sloupce obrazu. Z bodů A a B získáme hodnotu v bodě P a stejně pro druhý řádek z bodů C a D vypočteme hodnotu R. Souřadnice x a y se snadno normalizují pomocí [9]

$$P = xA + (1 - x)B, \quad R = xC + (1 - x)D \tag{3.19}$$

Z nových bodů lineární interpolací vypočteme hodnoty výsledku

$$Q = yP + (1 - y)R \tag{3.20}$$

Bilineární interpolace je rychlá, protože se pro výpočet hodnoty pixelu používají čtyři body z jeho okolí. Nevýhodou ovšem je, že rozmazává původní ostré přechody.

3.3.4 Kubická a bikubická interpolace

Kubická metoda potřebuje pro výpočet více vzorků ze svého okolí a z toho důvodu je výpočetně náročnější. Aproximace této metody se provádí pomocí kubické B-spline křivky. Konvoluční jádro aproximace má tvar [9]

$$\mathrm{h(x)} = \frac{1}{6} \begin{cases} 3|x|^3 - 6|x|^2 + 4 & \text{pro } 0 \leq |x| < 1 \\ -|x|^3 + 6|x|^2 - 12|x| + 8 & \text{pro } 1 \leq |x| < 2 \\ 0 & \text{pro } 2 \leq |x| \end{cases} \tag{3.21}$$

Bikubická interpolace je rozšířením kubické interpolace. Bikubická interpolace může být spočtena pomocí Lagrangeových polynomů, kubickými spliny nebo algoritmem kubické konvoluce. Bikubická interpolace používá pro získání nové hodnoty 4×4 okolních pixelů. Tato metoda je odolná proti aliasingu a zachovává jemné přechody.

3.3.5 Lanczosova interpolace

Filtr je pojmenován po Corneliovi Lanczosovi. Jedná se o typ z rodiny oknem omezených *Sinc* funkcí. Tato funkce má následující tvar

$$\mathrm{L(x)} = \begin{cases} \dfrac{\sin(\pi x)}{\pi x} * \dfrac{\sin(\pi x/3)}{\pi x/3} & \text{pro } 0 \leq x < 3 \\ 0 & \text{pro } |x| \geq 3 \end{cases} \tag{3.22}$$

Algoritmus používá 6×6 pixelů ze svého okolí pro výpočet intenzity pixelu cílového obrazu. Provoz tohoto filtru je náročný, protože pro každý výstupní pixel musí provést 42 násobení a 35 sčítání. [8]

Tento filtr lépe zachovává ostrost hran a jemnost tónových přechodů, neprodukuje artefakty. Kvůli použití trigonometrických funkcí je jeho výpočetní náročnost vyšší. Častěji se tedy využívá bikubická interpolace, kvůli malému rozdílu mezi těmito interpolacemi.

Kapitola 4

Návrh

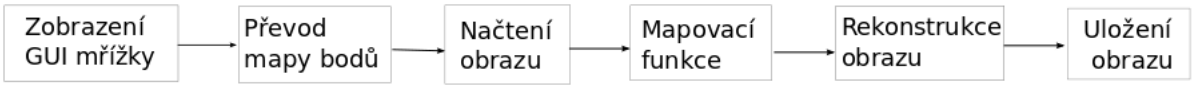
V předchozí kapitole byly zmíněny základní informace o geometrických transformacích a možnostech rekonstrukce, které byly využity během implementace. V této kapitole se budu věnovat specifikaci řešení a návrhu jednotlivých částí.

Existující řešení nabízí spoustu možností, co se týče práce s obrazem, jako jsou deformace nebo různé geometrické transformace. Většina těchto existujících řešení podléhá placené licenci. Na základě těchto poznatků jsem se rozhodla vytvořit aplikaci pro převzorkování a úpravu fotografie, abych si vyzkoušela práci s algoritmy pro převzorkování a rekonstrukci obrazu a porovnala jejich časovou náročnost s kvalitou výsledného obrazu.

Tato aplikace nabízí uživateli možnost úpravy obrazu, pomocí grafického rozhraní bude vykreslena mřížka, kterou je možné posouvat pixely fotografie. Díky tomuto posunu je možné vytvářet různé deformace, upravovat drobné chyby při fotografování nebo provádět korekci projekce fotografie.

4.1 Návrh řešení

V další části textu popíši návrh možného řešení problému. Tuto aplikaci bude tvořit grafické rozhraní, pomocí kterého se bude ovládat. Přes grafické rozhraní se zobrazí mřížka, která bude sloužit pro transformaci a převzorkování obrazu. Dále bude aplikace obsahovat tlačítka pro ovládání. Průchod celou aplikací je znázorněn na obrázku 4.1.



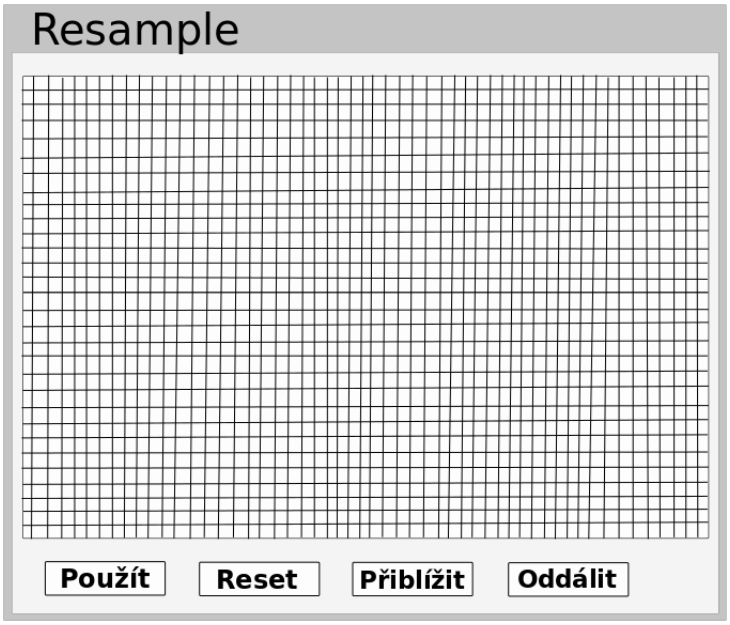
Obrázek 4.1: Schéma návrhu průchodem aplikace

V první řadě zobrazíme grafické rozhraní, pomocí kterého je možné posouvat jednotlivé body mřížky. Ty poté převedeme do mapy bodů. Jakmile máme hotovou mapu bodů, načteme obraz, který i s mapou bodů předáme do mapovací funkce. Touto funkcí provedeme převzorkování a úpravu obrazu, dále je nutné udělat rekonstrukci, která bude provedena pomocí interpolace nejbližším sousedem nebo pomocí bikubické interpolace, a následuje uložení obrazu. Všechny uvedené kroky budou popsány v následujících podkapitolách.

4.2 Návrh grafického rozhraní

Grafické rozhraní nám poskytuje nástroje pro práci s mřížkou a ovládání aplikace.

V následující části se budu zabývat popisem návrhu grafického rozhraní pro tuto aplikaci z obrázku 4.2. Jedná se o grafické okno, které se zobrazí po spuštění aplikace. Obsahuje vykreslenou mřížku pro manipulaci a posouvání jednotlivých bodů a čtyři tlačítka pro ovládání. Na pozadí mřížky bude vykreslený obraz, aby bylo možné správně nastavit jednotlivé body mřížky.



Obrázek 4.2: Schéma GUI aplikace

Pomocí mřížky budou vykresleny jednotlivé body, které bude možné posouvat. Velikost mřížky jsem zvolila 64x64, protože poskytuje dostatečné množství bodů, které je potřeba namapovat na obraz, a je to mocnina dvou. Jednotlivé body budou od sebe vzdáleny podle poměru k obrazu, aby mřížka pokryla celý obraz, což určuje i daný rozsah, který nelze přesáhnout na horizontální ani vertikální ose.

Dále se budeme věnovat tlačítkům. Každé tlačítko má svou úlohu. Tlačítko použít slouží k aplikování a uložení změn v mřížce, zavolání funkcí pro otevření a přemapování obrazu, a k následnému uzavření okna. Tlačítko reset má za úlohu resetovat obsah mřížky do původního stavu před změnou, pokud by došlo k špatnému nastavení mřížky. Protože se obraz nemůže zobrazit v plné velikosti a při práci s mřížkou budeme chtít zobrazit oblast obrazu detailněji další dvě tlačítka se věnují zobrazení obrazu. Tlačítko přiblížit nám obraz přiblíží, a tlačítko oddálit nám naopak obraz oddálí.

4.3 Návrh mapovací funkce

Tato funkce má za úkol převzorkovat daný obraz podle zadané mřížky a umožnit tak uživateli úpravu daného obrazu. Úprava bude spočívat v posouvání pixelu podle definované

mřížky. Bude pracovat na principu zpětného mapování, kdy budeme procházet pixely ve výstupním obraze a budeme se snažit najít jejich odpovídající hodnoty ve vstupním obraze.

Obrázek 4.3: Schéma mapovací funkce

Na obrázku 4.3 je zobrazeno schéma pro mapovací funkci, na kterém je znázorněn průchod obrazu. Na začátku načteme obraz, poté vytvoříme jeho klon, se kterým budeme dále pracovat, jednak aby nedošlo k poškození původního obrazu při chybné manipulaci, jednak abychom mohli pracovat s původními hodnotami obrazu, které budou sloužit pro získání nové hodnoty převzorkovaného obrazu.

V dalším kroku provedeme převzorkování obrazu, ke kterému budeme potřebovat dvě mapy bodů. Původní mapu, v níž jsou uloženy souřadnice needitované mřížky, a novou mapu bodů, kde jsou uloženy změny v mřížce, abychom zjistili velikost posuvu daného bodu. Následně provedeme rekonstrukci obrazu a výsledný obraz uložíme.

4.4 Návrh rekonstrukce obrazu

Pro rekonstrukci obrazu použiji tři různé interpolace, abych mohla porovnat výsledný obraz a náročnost výpočtu. Vybrala jsem interpolaci pomocí nejbližšího souseda, která je jednou z nejjednodušších metod pro rekonstrukci obrazu. Dále bikubickou interpolaci, která zachovává jemné přechody a výpočetně není příliš náročná. Tato interpolace může být spočtena různými způsoby a bilineární interpolaci.

Kapitola 5

Implementace

V minulé kapitole byly popsány mechanismy a návrh aplikace, pomocí které budou implementovány dané funkce. V této kapitole bude zmíněn způsob implementace a vzniklé problémy během implementace. Na závěr budou prezentovány výsledky a vyhodnocení použitých metod a aplikace.

Pro implementaci grafického rozhraní jsem zvolila jeden z existujících nástrojů Qt, který poskytuje nástroje pro vytváření a práci s grafickým rozhraním. Knihovnu OpenCV využiji pro práci s obrazem, protože obsahuje řadu funkcí, které usnadní práci při zpracování obrazu. Aplikaci jsem se rozhodla implementovat v jazyce C++, protože umožňuje práci s těmito nástroji a knihovnami.

5.1 Grafické rozhraní

Podle návrhu z obrázku 4.2 jsem vytvořila výslednou aplikaci, která je zobrazena na obrázku 5.1. Cílem bylo vytvořit jednoduché grafické rozhraní, pomocí kterého se bude aplikace ovládat.

Na pozadí mřížky je vykreslen upravovaný obraz, aby bylo možné přesně nastavit mřížku pro převzorkování. Tento obraz však nemůže být zobrazen v plné velikosti, kvůli omezenému rozlišení monitoru, a je nutné jej upravit. Proto jsem vybrala maximální velikost obrázku 700×600 px, aby byla aplikace zobrazitelná v plné velikosti. Větší obrázky se procentuálně upraví na menší velikost se zachováním poměru stran, aby je bylo možné zobrazit. Podle této velikosti se následně nastaví velikost okna a zobrazí tlačítka.

Aby bylo možné pracovat s mřížkou přesněji, rozhodla jsem se implementovat dvě tlačítka na přiblížení a oddálení obrazu. Při přiblížení nebo oddálení obrazu se přepočítají souřadnice bodů mřížky, aby se s nimi mohlo dále pracovat. Protože je upravovaný obraz na pozadí mřížky zobrazen v procentuální velikosti, rozhodla jsem se jako maximální oddálení použít právě tuto velikost. Mřížka pro editaci bodů se vykresluje podle velikosti obrazu, aby pokryla celý obraz. Při větším oddálení by nemuselo být možné pohybovat s body mřížky a tak provádět transformaci obrazu.

Obrázek 5.1: Výsledná GUI aplikace

Pro zobrazení mřížky jsem se rozhodla implementovat dvě možnosti, podle zadání přepínače $-f$. Pomocí tohoto parametru lze definovat, jak bude mřížka načtena. Pokud bude tento parametr zadán, načtou se souřadnice bodu mřížky ze souboru ve formátu *.cvs*, ty lze dále upravovat v aplikaci. Po dokončení převzorkování obrazu se souřadnice bodů opětovně uloží do tohoto souboru, aby byly zaznamenány změny provedené v aplikaci pro další zpracování. Díky tomuto parametru je možné vytvořit soubor se souřadnicemi dopředu nebo jej editovat mimo aplikaci, tyto souřadnice budou poté načteny a přepočítány podle velikosti obrazu. Pokud tento parametr nebude zadán, vykreslí se mřížka podle velikosti obrazu, tu je možné dále upravovat a po dokončení převzorkování se souřadnice bodů uloží do výchozího souboru, aby je bylo možné použít pro příští zpracování obrazu.

5.2 Zpracování obrazu

U této implementace je důležité správné načtení informace obrazu. Zadání vstupu provedeme pomocí přepínače $-i$. Jako vstup jsem se rozhodla využít dvě možnosti. První možnost vybrat cestu přímo k souboru, který chceme upravit. A jako druhou možnost vybrat cestu ke složce, aby bylo možné upravovat více souborů najednou. U této varianty bude nastavena mřížka pro následné převzorkování obrazu pouze jednou a bude pro všechny obrazy stejná.

Cesta k souboru nebo složce může být zadána absolutní, nebo relativní cestou. Je důležité, aby program pracoval s jednotnou cestou a nedocházelo tak k chybám. Rozhodla jsem se pracovat s absolutní cestou, proto je nutné převést relativní cestu na absolutní.

K uložení obrazu využijeme zadaný výstup pomocí přepínače $-o$. Do tohoto přepínače zadáme cestu ke složce, kde budeme chtít výsledný obraz uložit.

Pro zpracování obrazu jsem se rozhodla použít knihovnu OpenCV, ve které jsou obsaženy potřebné funkce. Pro načtení obrazu je vhodná funkce *imread*, umožňuje načíst obraz v různém barevném módu. Podporuje různé formáty obrazu, jako jsou JPEG, JPEG 2000, Windows bitmaps, Portable Network Graphics a další. Další funkcí této knihovny je funkce *imwrite*, která slouží k uložení obrazu do zadaného souboru.

Pro práci s obrazovými daty je využita třída Mat, pomocí této třídy lze vytvářet klon obrazu, zjistit velikost obrazu nebo změnit velikost a mnoho dalších. Po načtení obrazu jsem vytvořila pomocí funkce *clone* jeho klon , do kterého ukládám nové hodnoty obrazu získané z původního obrazu.

K práci s hodnotou daného pixelu obrazu jsem použila funkci *at*, která vrací hodnotu daného prvku v poli nebo umožňuje zapsat hodnotu do daného prvku. Také je možné definovat, jaký typ hodnoty má funkce vrátit. Pro tuto práci jsem vybrala typ $Vec3b$, který vrací hodnoty BGR obrazu. Prostor BRG je základní barevný prostor pro knihovnu OpenCV, který používá stejné barevné složky jako model RGB. Rozdílem mezi těmito barevnými prostory je uspořádání jednotlivých složek barev.

5.3 Převzorkování obrazu

Převzorkování obrazu má za úkol načíst zpracovávaný obraz, podle zadané mřížky provést převzorkování a úpravu obrazu a tento výsledný obraz poté uložit.

Při převzorkování obrazu je důležité si uvědomit, jaký způsob mapování zvolíme. Rozhodla jsem se použít zpětné mapování bodů obrazu, kdy budu procházet body výstupního obrazu a hledat odpovídající oblast v původním obraze. Tento způsob jsem zvolila, aby nedocházelo k zápisu hodnoty pixelu na reálnou souřadnici bodu, nebo abych nezapisovala hodnotu mimo obraz.

Během převzorkování procházíme pixel po pixelu obrazu a zjišťujeme jeho hodnotu pomocí rekonstrukce obrazu. V první řadě musíme určit, do jakého okna upravené mřížky tento pixel spadá. Okno mřížky je v tvaru obecného čtyřúhelníku, jehož vrcholy jsou určeny souřadnicemi jednotlivých bodů mřížky. Způsobů, jakými je možné zjistit, kam bod patří, je několik, například pomocí obsahu, kdy z daného pixelu vytvoříme trojúhelníky a součet jejich obsahu se musí rovnat obsahu čtyřúhelníku.

Tento způsob má ovšem nevýhodu, pokud počítáme v reálných číslech. V tomto případě může v rámci zaokrouhlení docházet k chybnému určení, což není žádoucí. Proto jsem se rozhodla použít pro tento výpočet barycentrické souřadnice. Daný čtyřúhelník si rozložíme na dva trojúhelníky a zjistíme, zda se pixel nenachází v jednom z nich. Pokud se daný pixel nevyskytuje v žádném čtyřúhelníku mřížky, tak se nebude brát v úvahu a ve výsledném obraze se nezobrazí.

Poté zjistíme pozici tohoto pixelu vůči tomuto čtyřúhelníku a najdeme odpovídající pixel v původním obraze. Podle zvolené metody pro rekonstrukci zjistíme hodnotu pixelu v původním obraze a následně ji uložíme na danou pozici ve výstupním obraze. Celý tento postup je znázorněn na algoritmu 1.

```
1 Načtení obrazu;
2 for každý pixel obrazu do
3 |   Zjistíme odpovídající čtyřúhelník mřížky;
4 |   Zjistíme pozici v daném čtyřúhelníku;
5 |   Najdeme odpovídající pixel v původním obraze;
6 |   Zjistíme hodnotu daného pixelu;
7 |   Hodnotu uložíme na pozici pixelu ve výstupním obraze;
8 end
9 Uložení výsledného obrazu;
```

Algoritmus 1: Převzorkování obrazu

5.4 Rekonstrukce obrazu

Pro rekonstrukci obrazu jsem využila tři různé interpolace, interpolaci nejbližšího souseda, bilineární interpolaci a bikubickou interpolaci. Tyto interpolace jsou rozdílné a každá má jiné vlastnosti.

Interpolace nejbližšího souseda nepoužívá pro svůj výpočet okolí pixelu, ale pouze hodnotu nejbližšího pixelu. Pro tuto aplikaci jsem použila pro získání hodnoty pixelu ve výstupním obraze, zaokrouhlenou hodnotu dané pixelu v původním obraze. Z tohoto pixelu zjistíme hodnoty jednotlivých barevných složek, které následně uložíme do daného pixelu ve výstupním obraze.

Bilineární interpolace používá pro svůj výpočet čtyři body z okolí daného pixelu. U těchto pixelů určíme hodnoty barevných složek, ze kterých vypočteme výslednou hodnotu, jenž poté uložíme do daného pixelu výstupního obrazu.

Bikubická interpolace však pracuje s hodnotami prvku svého okolí. Toto okolí má velikost 4x4, díky čemuž si tato interpolace zachovává jemné přechody a je odolná proti anti-aliasingu. Pro toto okolí jsem vytvořila strukturu, do které ukládám hodnoty jednotlivých barevných složek bodů z okolí. Po uložení hodnot okolí do struktury se provede bikubická interpolace, kterou lze spočítat různými způsoby. Bikubická interpolace je kubická interpolace v 2D prostoru, a lze spočítat pomocí této interpolace pro jednotlivé hodnoty osy x a y. U této interpolace je důležité správně nastavit okolí bodu.

Kubickou interpolaci jsem se rozhodla vypočítat pomocí Catmull-Rom křivky, jedná se o interpolační křivku, která je široce využívaná v grafice. Je dána vztahem [1]

$$\mathrm{Pt} = \frac{1}{2}\begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix}\begin{bmatrix} P_{-1} \\ P_0 \\ P_1 \\ P_2 \end{bmatrix} \tag{5.1}$$

Pomocí tohoto vztahu jsou spočítány hodnoty základních barevných složek pro jednotlivé řádky a pro tyto hodnoty je následně provedena interpolace, díky které získáme výslednou hodnotu pixelu.

5.5 Příklad použití

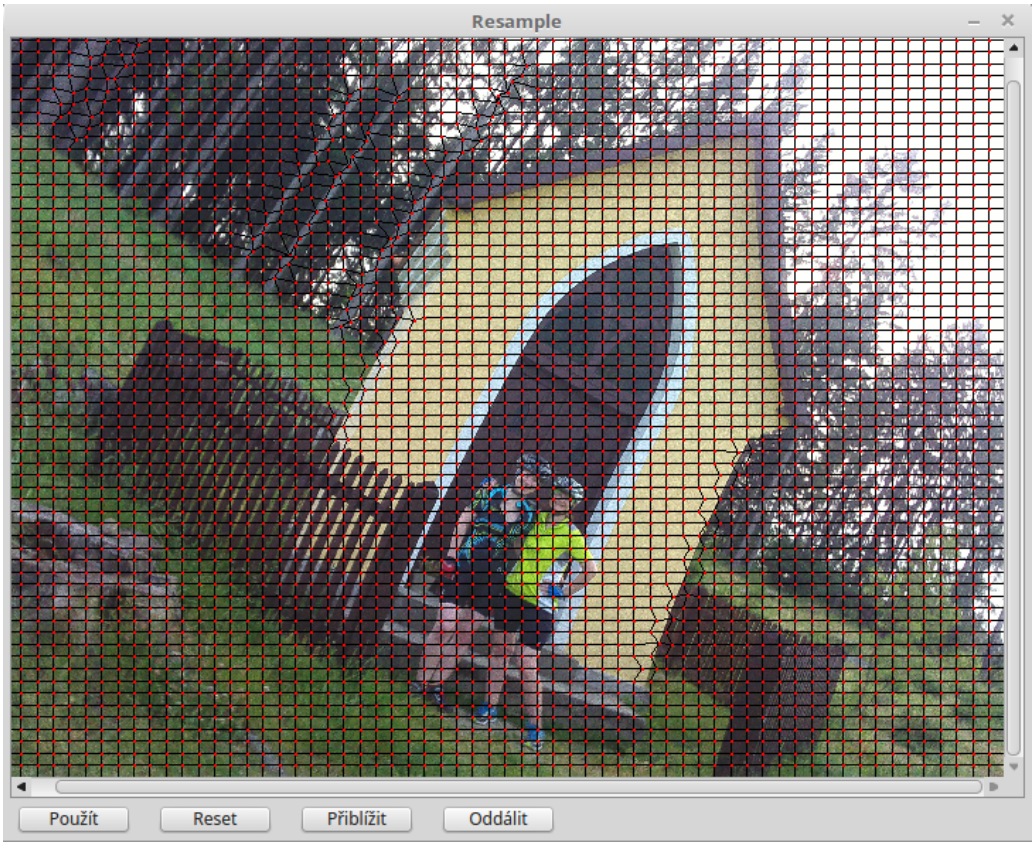
Předpis: **./resample** [**-I soubor/slozka**] [**-O slozka**] [**-f soubor**] [**-i typInterpolace**]

-I Cesta ke zdrojovému souboru/složce (povinný parametr)

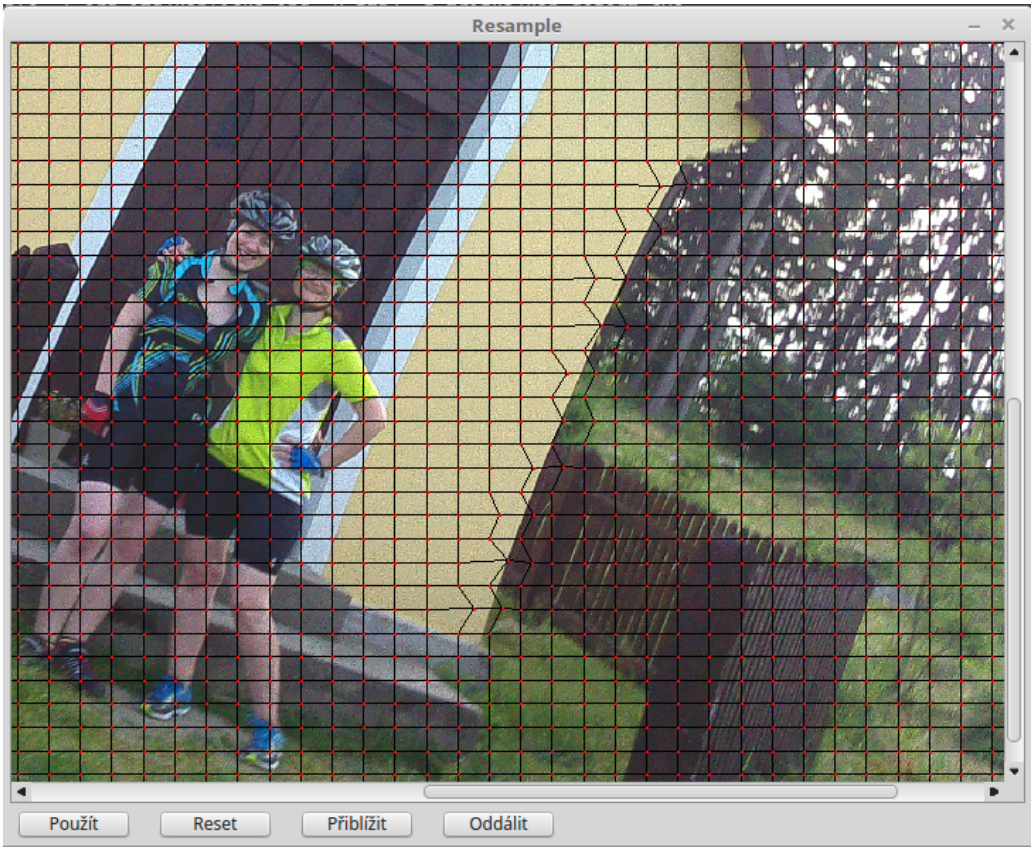
-O Cesta k složce pro uložení výsledku (povinný parametr)
-f Cesta k souboru s uloženými hodnotami mřížky (volitelný parametr)
-i Typ interpolace. Možné hodnoty: NN, BIC, $BILIN$ (povinný parametr)

Podle předpisu spustíme aplikaci, například
$./resample\ -I\ img/IMG\_2015.jpg\ -O\ tmp\ -f\ kap\_deform.cvs\ -i\ NN$

Tento předpis provede spuštění aplikace, kde vstupní soubor je $img/IMG\_2015.jpg$, který bude uložen do složky tmp, souřadnice bodů mřížky budou načteny ze souboru $kap_deform.cvs$. Pokud přepínač $-f$ nebude zadán, provede se spuštění aplikace bez načtení souřadnic bodů ze souboru a tyto body se nastaví podle obrazu. Po dokončení převzorkování se uloží do výchozího souboru $default.cvs$ pro další použití.



Obrázek 5.2: Aplikace spuštěná s danými parametry



Obrázek 5.3: Aplikace spuštěná s danými parametry po přiblížení

Další povinný parametr je přepínač $-i$, který nabývá dvou hodnot, hodnoty NN pro provedení interpolace nejbližšího souseda, hodnoty $BILIN$ pro bilineární interpolaci, nebo hodnoty BIC pro bikubickou interpolaci.

Zobrazení aplikace je ukázáno na obrázku 5.2. Díky možnosti přiblížit je možné lépe nastavovat body mřížky. Tato možnost je ukázána na obrázku 5.3.

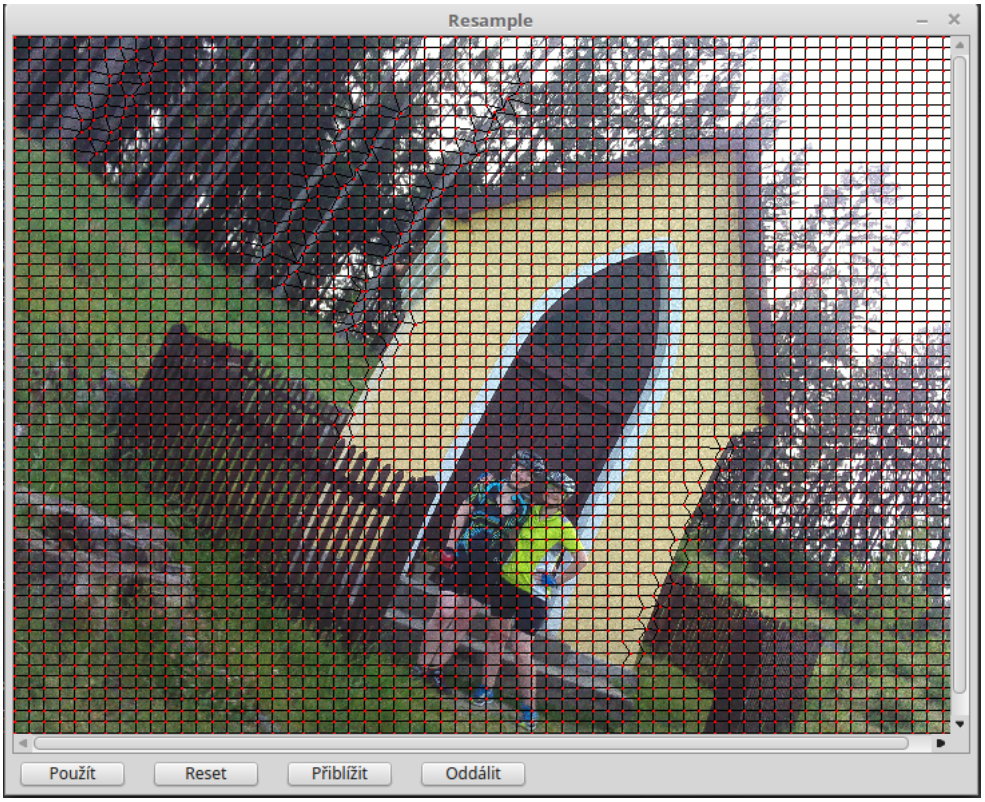
Kapitola 6

Vyhodnocení

V předchozí kapitole byla popsána implementace aplikace. Cílem této aplikace je převzorkování a úprava obrazu. Pod úpravou obrazu si ovšem je možné představit různé efekty v obraze. Tato aplikace převzorkuje obraz na základě definované mřížky, proto jsem pro ověření funkcionality aplikace vybrala deformaci obrazu a korekci projekce.

Pomocí deformace je možné měnit tvary různých objektů nebo části obrazu, například pokřivení čáry, rozmazání objektů a mnoho dalších. U deformace obrazu je velice důležité si předem definovat, jak chceme obraz deformovat. Pro tuto úlohu jsem si vybrala obrázek kapličky. Originální obrázek je zobrazen na obrázku 6.1. Tento obrázek byl pořízen pomocí fotoaparátu integrovaného v mobilním telefonu. Pro převzorkování je načtena mřížka ze souboru *kap_deform.cvs*, viz obrázek 6.2, pomocí této mřížky jsem zdeformovala a pokřivila rohy kapličky a části lesa.

Obrázek 6.1: Originální obrázek s kapličkou



Obrázek 6.2: Obrázek kapličky s vykreslenou mřížkou pro deformaci

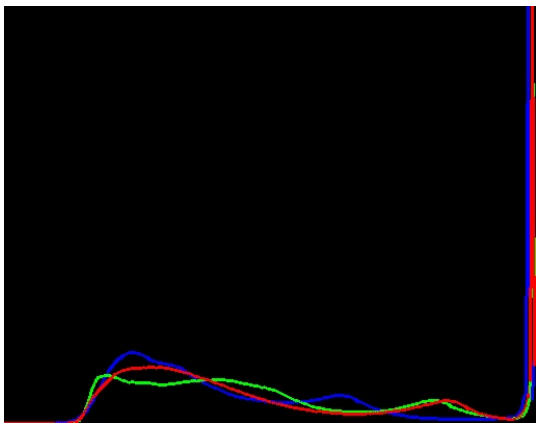
Rekonstrukci obrazu jsem provedla pomocí všech interpolací, abych mohla porovnat výsledky z hlediska rychlosti výpočtu a spektra výsledného obrazu. Převzorkovaný obrázek s rekonstrukcí pomocí bikubické interpolace je znázorněn na obrázku 6.3, s rekonstrukcí pomocí interpolace nejbližšího souseda na obrázku 6.5 a s rekonstrukcí pomocí bilineární interpolace na obrázku 6.4.

Bohužel tento obraz není příliš kvalitní a neobsahuje vysoké frekvence, u kterých dochází k ztrátě hodnoty během převzorkování obrazu. Proto tato hodnota není vypovídající. Jak je na první pohled zřejmé z obrázků 6.6 a 6.7 a 6.8 tato spektra jsou téměř až na velmi malé odchylky totožná. Z tohoto důvodu jsem se rozhodla ověřit výsledné spektrum u obrázku s vyšší kvalitou.

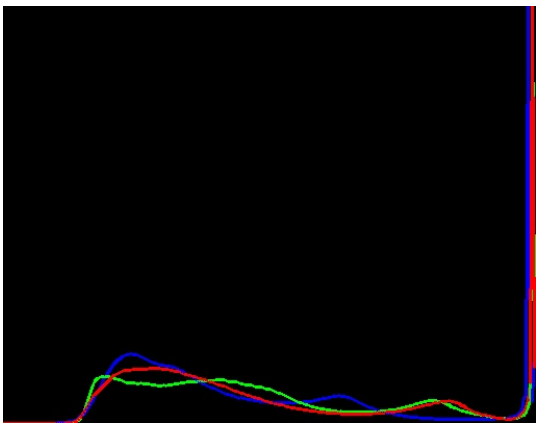
Obrázek 6.3: Obrázek kapličky deformované s bikubickou interpolací

Obrázek 6.4: Obrázek kapličky deformované s bilineární interpolací

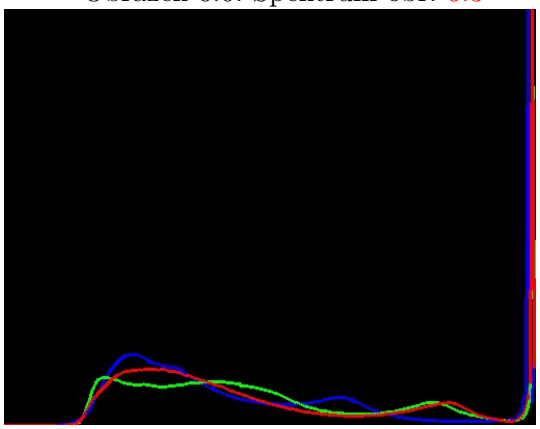
Obrázek 6.5: Obrázek kapličky deformované s interpolací nejbližšího souseda



Obrázek 6.6: Spektrum obr. 6.3



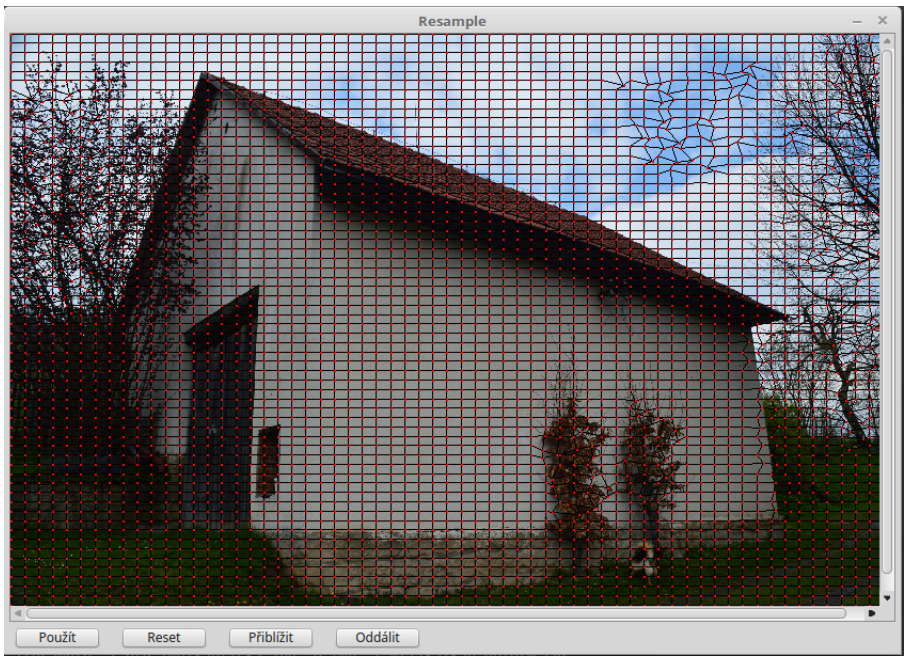
Obrázek 6.7: Spektrum obr. 6.5



Obrázek 6.8: Spektrum obr. 6.4

Pro druhou deformaci jsem si vybrala fotografii, která byla pořízena pomocí zrcadlového fotoaparátu, a provedla jsem deformaci několika objektů na fotografii. Originální snímek je zobrazen na obrázku 6.9. Pro převzorkování je načtena mřížka ze souboru *stod_deform.cvs*, viz obrázek 6.10, pomocí této mřížky jsem deformovala a zakřivila roh stodoly a provedla deformaci dalších objektů na fotografii.

Obrázek 6.9: Originální obrázek se stodolou



Obrázek 6.10: Obrázek stodoly s vykreslenou mřížkou pro deformaci

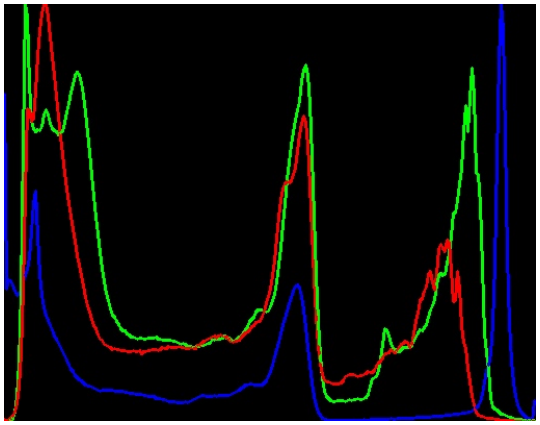
Převzorkovaný obrázek s rekonstrukcí pomocí bikubické interpolace je znázorněn na obrázku 6.11 a s rekonstrukcí pomocí interpolace nejbližšího souseda na obrázku 6.13 a s rekonstrukcí pomocí bilineární rekonstrukce na obrázku 6.12.

Obrázek 6.11: Obrázek deformované stodoly s bikubickou interpolací

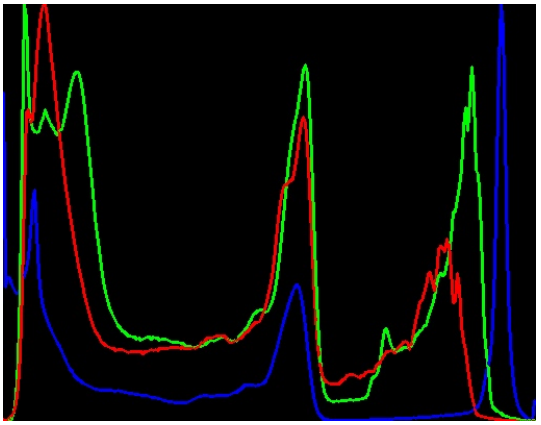
Obrázek 6.12: Obrázek deformované stodoly s bilineární interpolací

Obrázek 6.13: Obrázek deformované stodoly s interpolací nejbližšího souseda

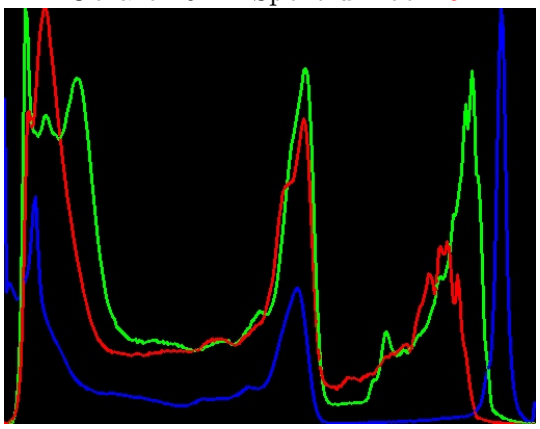
Když se podíváme na obrázky 6.14 a 6.15 a 6.16, zjistíme, že jsou tyto spektra až na velmi minimální odchylky stejné. U tohoto obrázku dokonce došlo k minimální odchylce spektra oproti spektru obrázku původního, na rozdíl od obrázku kapličky, kde byla odchylka znatelnější. Důvodem, kvůli kterému takto spektra vycházejí může být malý počet vysokých frekvencí, u kterých dochází ke ztrátě hodnoty, nebo dále výpočet pozice pixelu, který chceme rekonstruovat, nebo výpočet rekonstruované hodnoty, během kterého nedochází ke ztrátě informace. Další roli může hrát i ukládání obrazu, během kterého může docházet k ztrátě informace.



Obrázek 6.14: Spektrum obr. 6.11



Obrázek 6.15: Spektrum obr. 6.13



Obrázek 6.16: Spektrum obr. 6.12

Během focení kapličky na obrázku 6.1 došlo k pootočení fotoaparátu, což zapříčinilo, že jsou rohy a střecha kapličky zakřivené. Fotografii jsem vybrala právě z tohoto důvodu, abych se pokusila pomocí této aplikace opravit zakřivení střechy a pravého rohu kapličky. Tato oprava je ovšem náročná na nastavení bodů mřížky, protože i při malém nepřesném posunutí se může stát, že na tomto daném místě se může objevit zakřivení nebo nepřesnost. I přes snahu nastavit body přesně, může dojít nepřesnostem, právě z důvodu nepřesnosti v nastavení jednotlivých bodů. Pro převzorkování je načtena mřížka ze souboru *kaplika.cvs*.

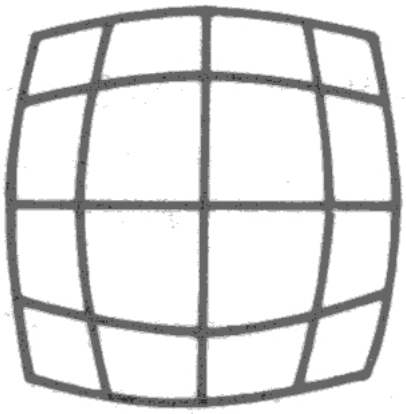
Převzorkovaný obrázek s rekonstrukcí pomocí bikubické interpolace je znázorněn na obrázku 6.17 a s rekonstrukcí pomocí interpolace nejbližšího souseda na obrázku 6.19 a s rekonstrukcí pomocí bilineární interpolace na obrázku 6.18.

Obrázek 6.17: Obrázek kapličky s bikubickou interpolací

Obrázek 6.18: Obrázek kapličky s bilineární interpolací

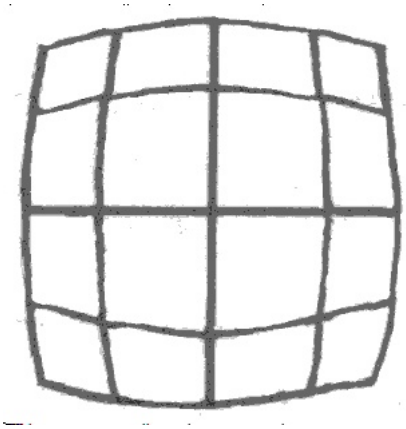
Obrázek 6.19: Obrázek kapličky s interpolací nejbližšího souseda

Mezi další možnosti patří oprava soudkovitého a poduškovitého zkreslení. Tyto zkreslení lze do určité míry upravit, málo kdy se ovšem podaří odstranit úplně toto zkreslení. Rozhodla jsem se vyzkoušet převzorkování na soudkovitém zkreslení. Pro tuto úlohu jsem si vybrala obrázek 6.20, na kterém je znázorněné toto zkreslení. Tento obrázek jsem se pokusila upravit. Bohužel při ručním nastavení mřížky vznikají nepřesnosti, které mají za následek rozostření a zvlnění křivek, toto lze odstranit pomocí nastavení bodů mimo aplikaci, kde tyto body můžeme nastavit přesněji. I přes pokus nastavit body co nejpřesněji, na daném obrázku vznikly nepřesnosti. Pro převzorkování je načtena mřížka ze souboru *soudek.cvs*, tato mřížka je nastavená na tento daný obrázek a nelze jí použít na jiný obrázek se soudkovitým zkreslením. Lze ovšem vytvořit mřížku, kterou bude možné použít na více obrázků s tímto zkreslením.

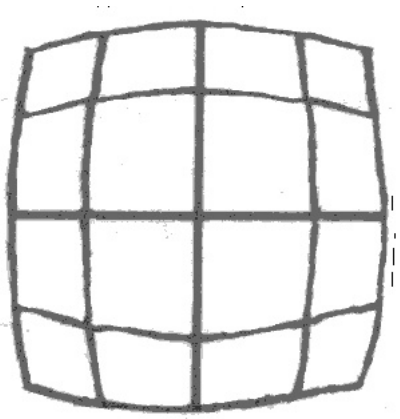


Obrázek 6.20: Originální obrázek se soudkovitým zkreslením

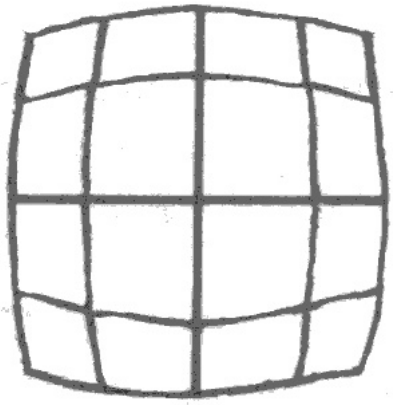
Převzorkovaný obrázek s rekonstrukcí pomocí bikubické interpolace je znázorněn na obrázku 6.21 a s rekonstrukcí pomocí interpolace nejbližšího souseda na obrázku 6.23 a s rekonstrukcí pomocí bilineární interpolace na obrázku 6.22. Obdobným způsobem lze také nastavit mřížku pro poduškovité zkreslení.



Obrázek 6.21: Obrázek soudkovitého zkreslní s bikubickou interpolací



Obrázek 6.22: Obrázek soudkovitého zkreslní s bilineární interpolací



Obrázek 6.23: Obrázek soudkovitého zkreslní s interpolací nejbližšího souseda

Rychlost převzorkování obrazu je ovlivněna jednak rozlišením obrazu, neboli kolik bodů je nutné převzorkovat a také počet změn v obraze. Pokud má obraz malé rozlišení hodnoty obou metod interpolací vycházejí velice podobně. Pokud však bude obraz ve vyšším rozlišení, tak interpolace pomocí nejbližšího souseda bude rychlejší, protože obsahuje menší počet výpočtů.

Protože tato aplikace prochází obraz pixel po pixelu a určuje jeho hodnotu, výpočet je poměrně časově náročný. Z tohoto důvodu je méně vhodný pro obrazy s vysokým rozlišením, protože obsahují příliš vysoké množství pixelů určených k převzorkování.

Kapitola 7

Závěr

Cílem bakalářské práce bylo nastudovat a popsat principy při převzorkování a úpravě obrazu a získat tak dovednosti v této oblasti počítačové grafiky, a v dalším kroku vytvořit aplikaci pro převzorkování a úpravu obrazu, která má za úkol posouvat pixely obrazu na základě definované mřížky. Tento cíl byl splněn.

V první řadě bylo nutné nastudovat dostupnou literaturu a seznámit se s dostupným softwarem a existujícími řešeními. Na základě získaných informací byl vytvořen postup a návrh grafického rozhraní aplikace. Pro tuto aplikaci byla zvolena jedena z nejčastěji používaných technik pro převzorkování a rekonstrukci obrazu, z důvodu jejich výhod při implementaci a reálném řešení úlohy. Rozhodla jsem se implementovat konzolovou verzi podle návrhu pod linuxovou distribucí v jazyce C/C++ s využitím knihoven Qt pro vytvoření grafického rozhraní a OpenCV pro práci s obrazem.

Během této práce byly vyjmenovány a zhodnoceny známé úpravy používané při zpracování fotografií a byla shrnuta existující řešení, které se zabývají touto problematikou. Byly popsány základní operace, které jsou nedílnou součástí úpravy obrazu.

Výsledkem této práce je funkční aplikace, která převzorkuje a upraví obraz podle zadaných parametrů. Výhodou této aplikace je možnost upravovat více fotografií najednou, aniž bychom museli mřížku opětovně nastavovat .

V průběhu vypracovávání této práce jsem získala zajímavé zkušenosti a poznatky o problematice grafického rozhraní, počítačové grafiky a úpravy obrazu. Měla jsem možnost se seznámit s různými metodami týkající se zpracování a úpravu obrazu. Zaujaly mě doprovodné funkce a úpravy, které bylo při použití těchto metod nutné vykonat, aby bylo možné výsledný obraz správně zobrazit a nedocházelo ke ztrátě informací.

K této práci bych se ráda vrátila a pokračovala v její implementaci, aby bylo možné vytvořenou aplikaci jednoduše používat a přidávat další vlastnosti a možnosti využití. Vytvořenou aplikaci je možné dále rozvíjet a přidávat k ní další vlastnosti. Mezi ně patří například přidání volby rekonstrukčního filtru, aby bylo možné porovnat výsledný obraz a jeho kvalitu nebo přidání dalších funkcí pro úpravu obrazu. Vhodným rozšířením by také bylo optimalizace rychlosti výpočtu převzorkování. Dalším vhodným vylepšením je intuitivnější uživatelské rozhraní, s nímž by bylo možné rychleji a lépe pracovat, nebo doplnění volby úpravy obrazu a předdefinování mřížky, aby ji uživatel nemusel nastavovat ručně.

Literatura

[1] Amstrong, J.: *Catmull-Rom Splines*. January 2006, [Online; navštíveno 05.04.2017].
URL http://algorithmist.net/docs/catmullrom.pdf

[2] Grafika.cz: *PhotoScape 3.1: freewarový prohlížeč a editor fotografií*. Grafika.cz - vše o počítačové grafice, [Online; navštíveno 01.04.2017].
URL http://www.grafika.cz/rubriky/software/photoscape-3-1-freewarovy-prohlizec-a-editor-fotografii-136606cz

[3] Ing. Přemysl, K., Ph.D.: *Zákalady počítačové grafiky*. [Online; navštíveno 22.11.2016].
URL https://wis.fit.vutbr.cz/FIT/st/course-files-st.php?file=%2Fcourse%2FIZG-IT%2Ftexts%2Fizg_opora.pdf&cid=10032

[4] team OpenCV: *About - OpenCV library*. OpenCV library, 2017, [Online; navštíveno 31.03.2017].
URL http://opencv.org/about.html

[5] Team Softonic Editorial: *Adobe Photoshop CC - Download*. Adobe Photoshop, [Online; navštíveno 31.03.2017].
URL https://adobe-photoshop.en.softonic.com/

[6] The Qt Company: *Qt - Company | About us*. Qt, 2017, [Online; navštíveno 31.03.2017].
URL https://www.qt.io/about-us/

[7] ZONER software, a.s.: *Zoner Photo Studio X - Jediný software na fotky, který potřebujete*. Zoner Photo Studio X, 2017, [Online; navštíveno 31.03.2017].
URL https://www.zoner.cz/photo-studio/

[8] Ztikhomirov, Y.: *Intel ® AVX Realization of Lanczos Interpolation in Intel ® IPP 2D Resize Transform*. 2015, [Online; navštíveno 16.03.2016].
URL https://software.intel.com/en-us/articles/the-intel-avx-realization-of-lanczos-interpolation-in-intel-ipp-2d-resize-transform/

[9] Žára, J.; Beneš, B.; Sochor, J.; aj.: *Moderní počítačová grafika*. Computer Press, 2004, ISBN 80-251-0454-0.

Přílohy

Příloha A

Obsah přiloženého paměťového média

src - Složka obsahující všechny zdrojové soubory a složku se soubory .cvs pro editaci mřížky

org_img - Složka obsahující testované obrázky pro převzorkování

resample_img - Složka obsahující výsledné převzorkované obrázky

latex_src - Složka obsahující zdrojové kódy písemné práce

opencv-2.4.13.zip - Komprimovaný soubor obsahující zdrojové kódy pro instalaci knihovny OpenCV

readme.txt - Soubor s popisem spouštění a práce s aplikací

Příloha B

Formát souboru pro editaci mřížky

Tato aplikace je navržena tak, aby bylo možné definovanou mřížku opětovně použít. K tomuto účelu slouží soubor *.cvs*, který obsahuje souřadnice jednotlivých bodů mřížky. Pokud nemáme předem tento soubor předem definovaný, můžeme aplikaci spustit bez přepínače $-f$, přičemž dojde k vykreslení needitované mřížky, kterou můžeme dále upravovat a po dokončení úprav a převzorkování se souřadnice bodů mřížky uloží do výchozího souboru $default.cvs$. S tímto souborem můžeme pak dále pracovat.

Jedna z možností je upravit tento soubor mimo aplikaci a pomocí aplikace pak načíst definovanou mřížku, kterou můžeme v aplikaci i nadále upravovat. Druhou možností je upravovat souřadnice pomocí aplikace, kdy tento soubor načteme a provedené změny se opět uloží do tohoto souboru.

Tento soubor musí být v předepsaném formátu, jinak by mohlo dojít k chybnému načtení mřížky. Souřadnice bodu jsou definovány pro osu x a y, kde na prvním místě stojí osa x oddělena mezerou od osy y. Všechny souřadnice jsou od sebe odděleny oddělovačem, za oddělovač jsem zvolila znak čárky. Oddělovačem jednotlivých řádků je pak $\backslash n$.

Formát tohoto souboru je následující:

$$
\begin{gathered}
0\ 0{,}1\ 0{,}2\ 0{,}3\ 0{,}4\ 0{,}.....{,}63\ 0\backslash n \\
0\ 1{,}1\ 1{,}2\ 1{,}3\ 1{,}4\ 1{,}.....{,}63\ 1\backslash n \\
... \\
... \\
... \\
0\ 63{,}1\ 63{,}2\ 63{,}3\ 63{,}.....{,}63\ 63
\end{gathered}
$$

Pokud bychom chtěli tento soubor editovat například pomocí editoru Excel, vypadá tento po načtení následovně:

Tabulka B.1: Náhled souboru editovaného pomocí editoru Excel

0 0	1 0	2 0	3 0	...	62 0	63 0
0 1	1 1	2 1	3 1	...	62 1	63 1
...						
0 63	1 63	2 63	3 63	...	62 63	63 63

I při této editaci je důležité zachovat závislosti a oddělovače jednotlivých znaků, aby došlo ke korektnímu načtení a aplikace proběhla v pořádku.