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

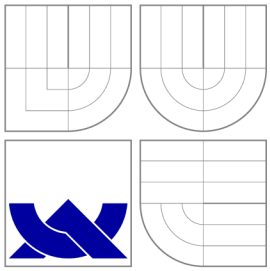
SKLÁDÁNÍ OBRAZŮ A VIDEOSEKVENCÍ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

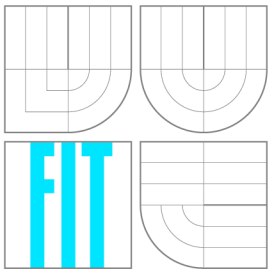
AUTOR PRÁCE
AUTHOR

DAVID KRYM

BRNO 2012



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

SKLÁDÁNÍ OBRAZŮ A VIDEOSEKVENCÍ

MERGING OF IMAGES AND VIDEO SEQUENCES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

DAVID KRYM

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. Dr. Ing. PAVEL ZEMČÍK

BRNO 2012

Abstrakt

Tato bakalářská práce se zabývá tvorbou panoramat z obrázků a snímků videosekvencí pořízených rotací kamery z jednoho místa. Zahrnuje návrh a implementaci aplikace se zaměřením na kvalitu a výkon. Využity jsou moderní techniky a algoritmy, jako je například SURF, ORB, metody k-nejbližších sousedů a vyrovnání svazku. Navržený nástroj zvládne automaticky skládat obrázky bez jakýchkoliv znalostí o scéně či kameře.

Abstract

This bachelor thesis deals with image and video sequence frames stitching when the camera undergoes a pure rotation. It involves design and implementation of application with focus on quality and performance. Modern techniques and algorithms are used, such as SURF, ORB, k-nearest neighbors and bundle adjustment. The application is able to create a panoramic images automatically without any assumptions about the scene or camera.

Klíčová slova

panorama, spojování obrazů, klíčové body, hledání korespondencí, ORB, SURF, homografie, RANSAC, vyrovnání svazku

Keywords

panorama, image stitching, feature points, feature matching, ORB, SURF, homography, RANSAC, bundle adjustment

Citace

David Krym: Skládání obrazů a videosekvencí, bakalářská práce, Brno, FIT VUT v Brně, 2012

Skládání obrazů a videosekvencí

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. Dr. Ing. Pavla Zemčíka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
David Krym
16. května 2012

Poděkování

Rád bych poděkoval vedoucímu této práce, doc. Dr. Ing. Pavlu Zemčíkovi, za odborné rady, užitečné připomínky a celkovou spolupráci na mé bakalářské práci.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Zatímco dříve měly kompaktní fotoaparáty rozlišení pouze v jednotkách megapixelů (MPx), dnes se i ty nejlevnější přístroje běžně prodávají s rozlišením přes 10 MPx. Není ani problém pořídit kvalitní fotografie mobilním telefonem. A přesto, lidské oko i fotoaparát dokáží zachytit pouze omezený úhel pohledu.

Na řadu přicházejí panoramatické fotografie, na kterých lze zachytit úhel až $360° \times 180°$. Lze je vytvořit speciálními fotoaparáty a objektivy. Toto řešení je ale nepraktické a drahé. Zajímavějším způsobem je panorama skládat z jednotlivých snímků. Fotografie tak lze spojovat nejen horizontálně, ale vpodstatě ve všech směrech.

Tato práce se zabývá návrhem a implementací aplikace, které zvládne složit panorama z množiny obrázků či videosekvence pořízené z jednoho místa. Hlavním zaměřením je rychlost a plná automatizace nástroje.

Práce je členěna celkem do osmi kapitol včetně úvodu. V kapitole 2 jsou shrnuty obecné principy, které se pojí se skládáním obrázků. Jsou také představeny používané pojmy, které se budou dále objevovat v celé práci. Kapitola 3 se zabývá detailním popisem některých vybraných technik a algoritmů, které budou následně použity. Zhodnocení současného stavu v oblasti tvorby panoramat se nachází v kapitole 4. Je zde také navrženo obecné řešení, jak by měl celý program fungovat. V kapitole 5 jsou detailněji rozebrány jednotlivé důležité části návrhu. Implementačními detaily se zabývá kapitola 6, která čtenáře seznámí s rozložením návrhu do jednotlivých tříd a s použitými nástroji a knihovnami. V kapitole 7 jsou prezentovány výsledky aplikace a její testování. Závěr práce se nachází v kapitole 8.

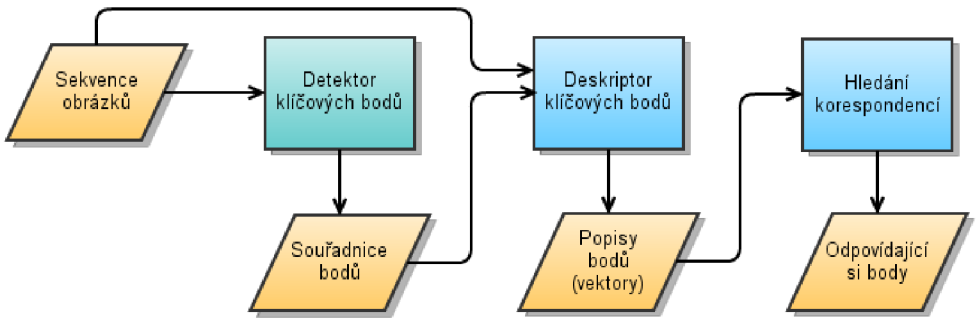
Kapitola 2

Principy skládání fotografií

Kapitola seznámí čtenáře s potřebným minimem pro účely této práce ohledně principu skládání obrázků. Budou postupně představeny některé související pojmy a bude ukázáno, co je vlastně ke složení dvou či více fotografií potřeba a v jakých etapách se vše odehrává.

Proces tvorby panoramat se skládá z několika základních kroků. Na vstupu je vždy množina fotografií, ze kterých bude vytvořeno výsledné panorama. Pro automatizované skládání panoramat je potřeba na snímcích lokalizovat záchytné body, na základě kterých lze k sobě dva snímky přiřadit. Pro bezpečné rozeznání bodů, které si napříč snímky odpovídají, je nutné body nějak rozlišit. Dále se zjistí, které konkrétní dvojice bodů si odpovídají. Posloupnost těchto základních kroků je znázorněna na obrázku 2.1. Na základě těchto informací se fotografie postupně spojí.

Pro nadcházející text je nezbytné definovat pojem **klíčový bod**. Tímto označením je dle [17] myšlen jakýkoliv bod v obraze, u kterého nastává změna signálu ve dvou dimenzích. Ve stejném smyslu bude dále použito i ekvivalentní označení *významný bod*. Toto splňují různé rohy v obrazu, ale také například černá tečka na bílém pozadí nebo konce větví u stromů. Některé významné body ovšem toto nemusí splňovat, proto zde budou jako klíčové obecně označeny ty body, které naleznou detektory k tomu určené.



Obrázek 2.1: Diagram znázorňující nalezení korespondencí mezi snímky. Oranžovou barvou jsou znázorněny vstupy a výstupy, modrou pak jednotlivé procesy.

2.1 Detektory klíčových bodů

Jak už název napovídá, detektor klíčových bodů má za úkol v obrázku nalézt významné body, které daný obraz nějak charakterizují. Nejdůležitější informace, kterou detektor poskytuje, jsou souřadnice klíčového bodu. Často mohou být detekovány i jiné charakteristiky, jako třeba měřítko (*scale*).

Dobrý a správně fungující detektor musí splňovat dva základní požadavky [9] – opakovatelnost a spolehlivost. Opakovatelnost znamená, že stejný klíčový bod bude správně detekován v různých obrázcích. Měl by tedy být invariantní vůči změnám, jako jsou např. rotace, změna měřítka (obě viz kapitola 3.4) či intenzity a podobně. Spolehlivost znamená, že nalezený klíčový bod bude dost charakteristický na to, aby minimalizoval počet kandidátů při hledání stejných klíčových bodů mezi různými snímky.

Detektory lze klasifikovat do tří hlavních skupin [17]:

- **Založené na konturách**. Nejdříve extrahují obrysy ze všech obrázků a poté hledají speciální body. Mohou to být například maximum zakřivení, nebo inflexní body (body, ve kterých se mění zakřivení obrysu). Dále mohou například vytvořit polygonální aproximaci a poté hledat průsečíky.
- **Založené na jasu**. Hledají klíčové body zkoumáním změny jasu okolo bodů. K měření této změny často využívají první a druhou derivaci obrázku.
- **Založené na parametrickém modelu**. Klíčové body jsou nalezeny tak, že odpovídají nějakému modelu nebo šabloně. Poskytují většinou subpixelovou přesnost, ale jsou omezeny pouze na specifické klíčové body.

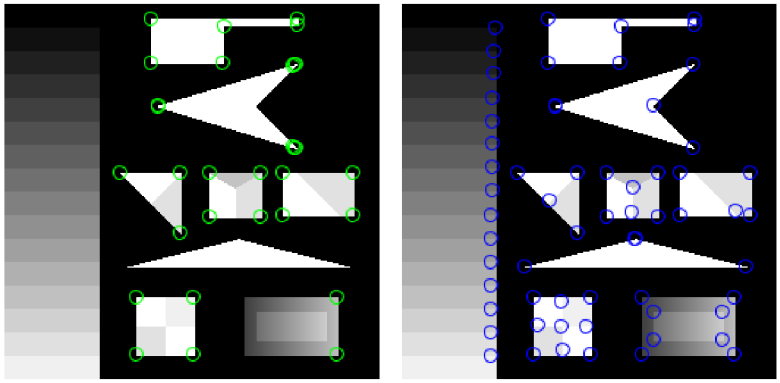
2.1.1 Příklady detektorů

Na povrchu bez jakékoliv struktury (např. nebe bez mraků), je téměř nemožné nalézt klíčový bod. Dobře se naopak lokalizují oblasti, kde nastává veliká změna kontrastu (směr a strmost tohoto růstu určuje gradient).

Moravcův detektor Jeden z prvních detektorů významných bodů založený na zkoumání signálu [17]. Zjišťuje rozdíly jasu mezi zvoleným oknem a oknem posunutým v osmi směrech. Poté je vybrána ta hodnota rozdílu, která je nejmenší. Pokud je větší než zvolená hranice, je bod prohlášen za roh. Nalezené rohy lze vidět na obrázku 2.2a.

Harrisův detektor Na rozdíl od Moravcova detektoru nepoužívá posun oken, ale je založen na autokorelační matici [22]. Tato matice popisuje rozložení gradientu v okolí daného bodu. Používá okno s Gaussovým rozložením. Bod je prohlášen rohem tehdy, pokud se v něm významně mění signál v obou směrech. Ukázka je na obrázku 2.2b.

Detektory skvrn Předchozí dva detektory jsou používány pro nalezení rohů. Pro detekci oblastí se používají detektory skvrn (*blob*). Jsou zaměřeny na hledání bodů či regionů v obraze, kde se mění vlastnosti jako jas či barva, v porovnání s okolím. Do této skupiny patří například detektory založené na DoH (*Determinant of Hessian*), DoG (*Difference of Gaussians*) či LoG (*Laplacian of Gaussian*).



(a) Moravcův (b) Harrisův

Obrázek 2.2: Ukázka nalezených rohů obou detektorů. Vytvořeno webovým appletem [X].

2.2 Deskriptory klíčových bodů

Předpokládejme, že máme dva různé snímky jedné scény a pro každý z nich jsou již získány klíčové body. Aby bylo možné nalézt vzájemně si odpovídající páry klíčových bodů mezi snímky, je potřeba zavést deskriptory těchto bodů [9]. Deskriptor je proces, který použije informaci o klíčových bodech a obrázku, aby vytvořil popis těchto bodů. Jako deskriptor lze označit i tento samotný popis. Nejčastější jsou popisy ve formě vektorů pro každý klíčový bod. Tyto popisy jsou poté použity pro nalezení korespondujících párů klíčových bodů mezi obrázky.

Deskriptor by měl být invariantní vůči rotaci, změně měřítka (obě viz kapitola 3.4) a obecně afinních transformacích. Splněním těchto požadavků se zajistí, že klíčové body na různých snímcích budou mít téměř stejné popisy i při různých deformacích.

Deskriptory lze klasifikovat do čtyř skupin [13]:

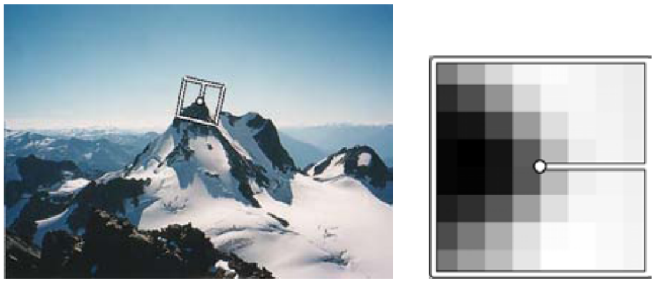
- **Založené na distribuci**. Tyto techniky používají histogramy k reprezentaci různých charakteristik nebo tvarů. Charakteristiky mohou být například jas pixelu, vzdálenost od centrálního bodu nebo gradient.
- **Deskriptory prostorové frekvence**. Techniky popisující frekvenční obsah obrázku. Používají se ke klasifikaci a popisu textur.
- **Diferenciální deskriptory**. Například deskriptor pro vyhodnocení spolehlivosti detektoru. Množina lokálních derivací je použita pro popis významného regionu.
- **Momenty**. Pro popis regionu byly použity momenty. Invariant určuje centrální moment regionu v kombinaci s pořadím momentu a stupněm.

2.2.1 Příklady deskriptorů

V některých případech lze místo deskriptorů použít normalizovanou vzájemnou korelaci (*normalized cross-correlation*) nebo součet čtvercových hodnot rozdílů (*sum of squared differences*) pro přímé porovnání intenzit v malých výřezech kolem klíčových bodů.

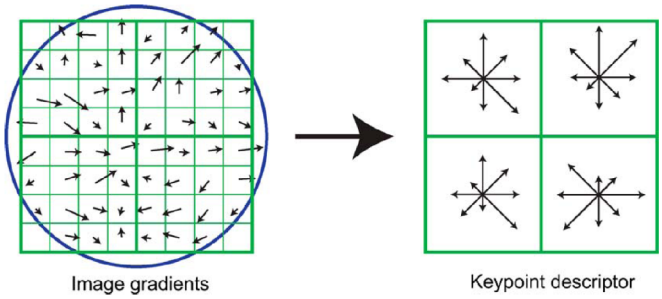
Ve většině případů se ovšem může lokální charakteristika okolí klíčových bodů měnit v orientaci a měřítku, či jinak deformovat. Proto se častěji používá extrahování lokálního měřítka či orientace, následné převzorkování, a poté až tvorba samotného deskriptoru bodu. Tato kapitola čerpá z [20].

MOPS Celým názvem *Multi-scale oriented patches*. Metoda je vhodná pro spojování obrázků. MOPS potřebuje orientovaný klíčový bod, kolem kterého se navzorkuje výřez 8 x 8 pixelů se subpixelovou přesností za použití celkem pěti různých pyramidových úrovní měřítek.



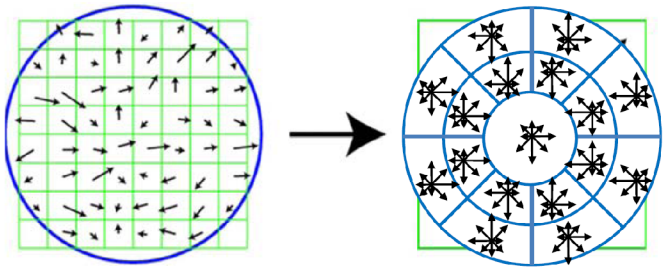
Obrázek 2.3: MOPS deskriptor. Ukázka výřezu 8 x 8 v jedné úrovni. Převzato z [20].

SIFT Je zkratkou pro název *Scale invariant feature transform*. Deskriptor je založen na výpočtu gradientu všech pixelů v okně s velikostí 16 × 16 kolem každého klíčového bodu, za použití patřičné úrovně Gaussovy pyramidy, na které byl bod detekován. Okolí klíčového bodu je rozděleno na 4 × 4 oblasti, kde v každé oblasti je vytvořen histogram orientovaných gradientů. Každý takovýto histogram se sám skládá z osmi binů. Výsledkem pro každý klíčový bod je vektor 128 nezáporných čísel (4 × 4 × 8).



Obrázek 2.4: SIFT deskriptor. Ukázka 2 x 2 pole popisů vypočítaného ze vzorku 8 x 8. Převzato z [12].

GLOH Jedná se o zkratku z názvu *Gradient location-orientation histogram*. Jde o variantu na SIFT deskriptor, která používá speciální strukturu (obrázek 2.5), na rozdíl od rozdělení okolí na čtyři kvadranty, jako u metody SIFT (obrázek 2.4). Výsledkem je také vektor 128 nezáporných čísel.



(a) image gradients (b) keypoint descriptor

Obrázek 2.5: GLOH deskriptor. Ukázka gradientů obrázku a speciální struktury, která je použita k výpočtu orientovaných histogramů. Převzato z [20].

2.3 Hledání korespondencí

Jakmile jsou v obrázcích nalezeny klíčové body a jejich deskriptory, následuje fáze hledání korespondencí *feature matching*. V tomto kroku se předběžně naleznou páry bodů, které si vzájemně odpovídají.

Celý problém lze rozdělit na dvě samostatné komponenty [20]. První z nich je zvolená strategie pro určení korespondencí, které budou použity pro následující zpracování. Druhou komponentou jsou potom použité datové struktury a samotné algoritmy použité k hledání.

2.3.1 Strategie

Mějme dva obrázky, které se určitou částí překrývají. Pro klíčové body prvního obrázku (v pomyslné překrývající se části), bude nalezeno velké množství shod ve druhém obrázku. Některé shody ovšem mohou být špatně určeny, např. kvůli pohybujícímu se předmětu ve scéně. Shody nalezené mimo překryv jsou taktéž nesprávné.

Příklady některých strategií z [13]:

- **Založená na prahu**. Pro dva body je nalezena korespondence, pokud je euklidovská vzdálenost mezi jejich deskriptory menší, než zvolený práh. Pro deskriptor může být nalezeno několik shod a jen některé mohou být správné.
- **Nejbližší soused**. Pro dva body A a B je nalezena korespondence, pokud je deskriptor D_B nejbližší soused D_A, a zároveň pokud je poměr jejich vzdáleností menší, než zvolený práh. Deskriptor tak má pouze jedinou korespondenci.
- **Dva nejbližší sousedé**. Strategie podobná té předchozí. Práh je zde ovšem porovnáván s poměrem vzdáleností mezi prvním a druhým nejbližším sousedem. Pro dva body je nalezena korespondence, pokud $\frac{\|D_A - D_B\|}{\|D_A - D_C\|} < práh$ (D_B a D_C jsou první, respektive druhý nejbližší soused)

2.3.2 Struktury

Nejjednodušší způsob, jak zmíněné strategie použít, je porovnat všechny deskriptory z jednoho obrázku se všemi deskriptory z druhého. Toto má ovšem kvadratickou složitost dle počtu obrázků a pro větší počet obrázků je to nepoužitelné.

Výhodnější je použít některou z indexových struktur, jako například hašovací tabulku či multidimenzionální prohledávací strom. Tyto struktury mohou být vytvořeny zvláště pro každý obrázek, nebo jediná struktura pro všechny dohromady.

Příklady struktur pro hledání nejbližších sousedů [20]:

- **KD-strom**. Přerozděluje multidimenzionální prostor alternativními rovinami rovnoběžnými s příslušnými osami.
- **LSH**. *Locality-sensitive hashing* je metoda, která data na vstupu hašuje tak, aby podobné prvky dat byly s velkou pravděpodobností mapovány na stejné indexy.

2.4 Sledování klíčových bodů mezi snímky

Alternativou k hledání klíčových bodů ve všech snímcích a následnému nalezení jejich vzájemných korespondencí je tzv. *feature tracking* [20]. V prvním snímku jsou nalezeny významné lokace, jejichž pozice se hledá v následujících snímcích. Tento způsob je častěji používán pro sledování objektů ve videu, kde se dá očekávat malé množství pohybu či jiných deformací mezi následujícími snímky.

Pokud jsou lokace sledovány v dlouhé sekvenci snímků, může jejich vzhled procházet mnoha změnami. Variantou pak je, aby sledování lokací neprobíhalo stále jen vůči výřezu z prvního snímku, ale aby se výřez převzorkoval vždy dle aktuálního snímku. Rozšířeným trackerem je např. Kanade-Lucas-Tomasi (KLT).

Kapitola 3

Vybrané techniky používané k tvorbě panoramat

V následující kapitole jsou podrobněji představeny a rozebrány některé techniky a algoritmy používané při tvorbě panoramat. Jsou zde uvedeny popisy vybraných algoritmů, které budou tvořit jádro aplikace navržené v nadcházejících kapitolách.

3.1 SURF

Metoda SURF, celým názvem *Speeded Up Robust Feature*, označuje robustní detektor a deskriptor klíčových bodů. Tuto metodu prezentoval Herbert Bay roku 2006 [1]. Je invariantní vůči rotaci a změně měřítka. Částečně je založena na metodě SIFT, oproti které je ale několikanásobně rychlejší. Nárůst rychlosti byl dosažen využitím integrálních obrazů. Použití tohoto algoritmu ve Spojených státech je patentováno. V kapitole je čerpáno z [1] a [6].

3.1.1 Integrální obraz

Záznam integrálního obrazu $I_\Sigma(\mathbf{x})$ v bodě $\mathbf{x} = (x, y)^\top$ reprezentuje sumu všech pixelů vstupního obrázku I v obdélníku od počátku do bodu $\mathbf{x}$.

$$I_\Sigma(\mathbf{x}) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \tag{3.1}$$

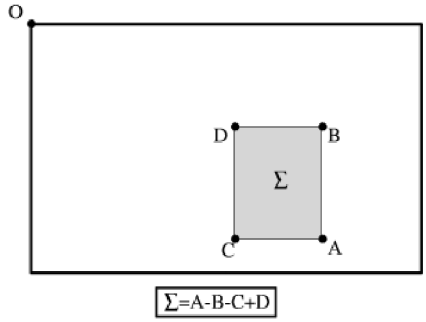
Výhodou je, že pouze tři celočíselné operace stačí ke spočítání jakkoliv veliké obdélníkové oblasti, viz obrázek 3.1.

3.1.2 Detektor

Detektor klíčových bodů je založen na Hessianově matici [1] kvůli dobrým výsledkům v přesnosti a hledají se lokace, kde je determinant maximální.

Mějme bod $\mathbf{x} = (x, y)$ na obrázku I. Hessianova matice $\mathcal{H}(\mathbf{x}, \sigma)$ v $\mathbf{x}$ na měřítku σ je definována následovně:

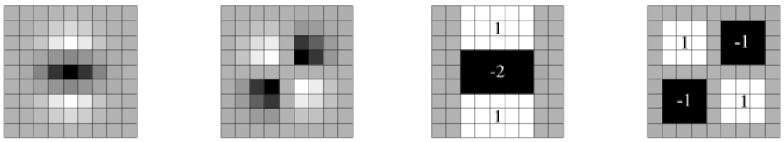
$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix} \tag{3.2}$$



Obrázek 3.1: Ukázka integrálního obrazce a výpočtu jeho plochy. Převzato z [1]

$L_{xx}(\mathbf{x}, \sigma)$ je konvoluce druhé derivace Gaussovy funkce $\frac{\partial^2}{\partial x^2} g(\sigma)$ s obrázkem I v bodě $\mathbf{x}$, podobně potom pro $L_{xy}(\mathbf{x}, \sigma)$ i $L_{yy}(\mathbf{x}, \sigma)$.

Na obrázku 3.2b je vidět použitá aproximace pomocí obdélníkového filtru. Díky integrálním obrazům lze tuto aproximaci velmi efektivně vypočítat bez ohledu na velikost filtru.



(a) Gaussova funkce (b) Aproximace

Obrázek 3.2: Na (a) je vidět druhá derivace Gaussovy funkce ve směrech y (L_{yy}) a xy (L_{xy}). Na (b) jsou zobrazeny odpovídající aproximace D_{yy} a D_{xy}. Převzato z [1].

Zobrazené 9×9 obdélníkové filtry jsou aproximace Gausse s $\sigma = 1,2$. Dále pro ně bude použito označení D_{xx}, D_{yy}, D_{xy}. Dostaneme tedy

$$det(\mathcal{H}_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2 \quad , \tag{3.3}$$

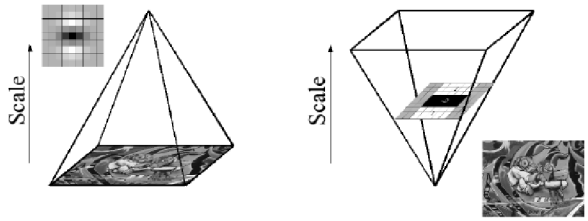
kde relativní váha w je použita k vyvážení Hessianova determinantu. To je potřeba pro zachování energie mezi Gaussovými jádry a aproximovanými Gaussovými jádry.

$$w = \frac{|L_{xy}(1.2)|_F |D_{yy}(9)|_F}{|L_{yy}(1.2)|_F |D_{xy}(9)|_F} = 0,912\ldots \simeq 0,9 \quad , \tag{3.4}$$

kde $|x|_F$ je Frobeniova norma.

Klíčové body je nutné hledat v různých měřítkách zdrojového obrázku. Díky použití obdélníkových filtrů a integrálních obrazů není nutné iterativně používat stejný filtr na výstup předchozí filtrované vrstvy. Místo toho je filtr různých velikostí použit přímo na zdrojový obrázek, viz obrázek 3.3.

Celý prostor měřítek obrázku je rozdělen na oktávy. Oktáva reprezentuje sérii filtrových odezev získaných konvolucí obrázku a filtrů se zvěčující se velikostí.

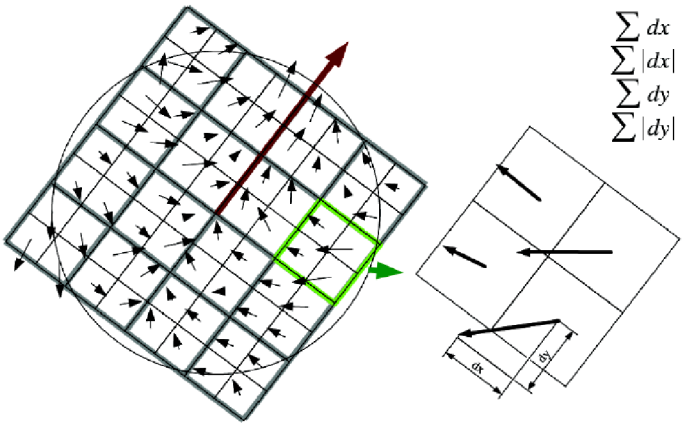


Obrázek 3.3: Místo zmenšování obrázku lze díky integrálním obrazům zvětšovat filtr. Převzato z [1]

3.1.3 Deskriptor

Kvůli invarianci vůči rotaci obrázku je nutné klíčovým bodům přiřadit orientaci. Z tohoto důvodu jsou pro klíčový bod vypočítány odezvy Haarovy vlnky ve směru x a y v kruhovém okolí o poloměru $6s$. Proměnná s zde značí měřítko, na kterém byl bod nalezen. Použity jsou opět integrální obrazce a stačí pouze 6 operací pro výpočet odezev ve směru x a y na jakémkoliv měřítku [1].

Pro samotnou extrakci deskriptoru je nejprve vytvořen čtvercový region o velikosti $20s$ se středem v klíčovém bodě a vypočítanou orientací. Region je rozdělen na 4×4 subregiony. Pro každý subregion jsou vypočítány odezvy Haarovy vlnky. (d_x bude označena Haarova vlnka v horizontálním směru a stejně tak analogicky d_y). Odezvy d_x a d_y ze všech subregionů jsou sečteny a je vytvořena první část popisného vektoru. Každý subregion má čtyřdimenzionální popisný vektor $\mathbf{v}$, kde $\mathbf{v} = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$. Spojením těchto vektorů ze všech subregionů je výsledný vektor o délce 64.



Obrázek 3.4: Orientovaná mřížka se 4×4 regiony (vlevo). Pro každý čtverec jsou vypočítány odezvy vlnky. Zobrazené 2×2 subregiony každého čtverce odpovídají vlastním hodnotám deskriptoru. Převzato z [1]

3.2 ORB

Alternativou k metodě SURF je ORB, celým názvem *Oriented FAST and Rotated BRIEF*. Tato poměrně nová metoda byla roku 2011 prezentována lidmi ze skupiny Willow Garage, kteří momentálně spravují např. i projekt knihovny OpenCV. Kapitola vychází z [16].

Jak už samotný název napovídá, ORB využívá známý detektor klíčových bodů FAST a jako deskriptor je použit BRIEF deskriptor, který také pochází z roku 2011. Byly zvoleny z toho důvodu, že dosahují dobrých výsledků při nízkých výpočetních nárocích. Přesněji, ORB využívá oFAST a rBRIEF, což jsou modifikace výše uvedených, které budou popsány dále. Bylo nutné vyřešit hlavně problém, že BRIEF není invariantní vůči rotaci.

3.2.1 oFAST: Orientace klíčového bodu FAST detektoru

Díky svým nízkým výpočetním nárokům je FAST detektor často používán. Bohužel, nalezené klíčové body nejsou orientované a orientace jim musí být přidána.

Standardní FAST detektor hledá rohy v obrazu. Důležitý je parametr udávající práh mezi intenzitami středového bodu a pixelů v kruhovém okolí tohoto bodu. Zde je zvolen FAST-9, který používá poloměr kruhu 9. Jelikož FAST sám o sobě nemá žádnou míru rohovitosti, byla použita Harrisova míra rohovitosti pro seřazení nalezených klíčových bodů.

Pro nalezení N klíčových bodů je nejprve nastaven nízký práh tak, aby bylo nalezeno více než N bodů. Poté jsou seřazeny na základě Harrisovy míry a je vybráno prvních N bodů. Z důvodu, že FAST nehledá klíčové body na různých měřítkách, byla přidána pyramida měřítek. Klíčové body jsou tak nově hledány na každé úrovni pyramidy.

Orientace z centroidu intenzity

Centroid intenzity předpokládá, že intenzita rohu je ofset z jeho středu a tento vektor může být použit k určení orientace [16].

Momenty výřezu obrázku jsou definován jako

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{3.5}$$

a s těmito momenty lze získat centroid jako

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{3.6}$$

Zkonstruujme vektor ze středu rohu O do centroidu C a značme ho $\overrightarrow{OC}$. Orientace tohoto výřezu potom bude

$$\theta = \text{atan2}(m_{01}, m_{10}) \tag{3.7}$$

3.2.2 rBRIEF: Rotace v BRIEF deskriptoru

BRIEF deskriptor [5] je bitový řetězec popisující výřez obrázku a je vytvořen ze sady binárních testů intenzity.

Mějme výřez vyhlazeného obrázku $\mathbf{p}$. Binární test τ je definován jako

$$\tau(\mathbf{p}; \mathbf{x}; \mathbf{y}) := \begin{cases} 1 & \text{pro } \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & \text{pro } \mathbf{p}(\mathbf{x}) \geq \mathbf{p}(\mathbf{y}) \end{cases}, \tag{3.8}$$

kde $\mathbf{p}(\mathbf{x})$ značí intenzitu výřezu v bodě $\mathbf{x}$. Jeden prvek je definován jako vektor o n binárních testech

$$f_n(\mathbf{p}) := \sum_{1 \leq i \ \leq n} 2^{i-1} \tau(\mathbf{p}; \mathbf{x}; \mathbf{y}) \tag{3.9}$$

Je použito Gaussovo rozložení okolo středu výřezu. Délka vektoru byla zvolena $n = 256$.

Směrovaný BRIEF

Standardní BRIEF se nezvládne vypořádat s rotací větší než pár stupňů. Použité je směrování deskriptorů na základě orientace klíčových bodů. Pro každou sadu prvků n binárních testů na pozici (x_i, y_i) je definována $2 \times n$ matice

$$\mathbf{S} = \begin{pmatrix} \mathbf{x}_1 & \dots & \mathbf{x}_n \\ \mathbf{y}_1 & \dots & \mathbf{y}_n \end{pmatrix} \tag{3.10}$$

Směrovaný BRIEF operátor [16] vznikne jako

$$g_n(\mathbf{p}, \theta) := f_n(\mathbf{p}) | (\mathbf{x}_i, \mathbf{y}_i) \in \mathbf{S}_\theta \tag{3.11}$$

kde θ značí orientaci výřezu a S_θ značí směrovanou verzi předchozí matice S.

Dobrou vlastností BRIEF deskriptoru je, že jednotlivé popisy klíčových bodů se dost liší. To činí klíčové body dobře rozlišovatelnými. Další požadovanou vlastností je mít testy nekorelované, protože se každý test podílí na výsledku. Bohužel, směrovaný BRIEF v obou vlastnostech vychází hůře a bylo potřeba ho dále upravit.

Aby se zredukovala korelace mezi binárními testy a zvýšila rozdílnost jednotlivých popisů, byla vyvinuta metoda strojového učení pro výběr dobré podmnožiny binárních testů. Hledají se mezi všemi možnými binárními testy takové, které mají vysokou rozdílnost a zároveň jsou nekorelované s průměrem kolem 0,5. Výsledek je nazýván jako rBRIEF.

3.3 RANSAC

Pojmenování této metody vzniklo jako zkratka z názvu *RANdom SAmple Consensus*, což lze přeložit jako shodu náhodných vzorků. Tuto metodu již v roce 1981 publikovali M. Fischler a R. Bolles [7]. Přestože se jedná o poměrně starou metodu, je dodnes používaná a existují pro ni různé zrychlující optimalizace. Dále bude nastíněn obecný princip této metody dle [8] a [20].

Ze začátku je důležité definovat dva pojmy:

- inlier – je vzorek dat, který popisuje daný model
- outlier – je vzorek dat, který daný model nepopisuje

RANSAC je iterativní metoda pro odhad parametrů matematického modelu. Předpokládá, že vstupní data obsahují jak inliers, tak outliers, a umí se vypořádat i s vysokým počtem outliers. Princip metody spočívá v opakovaném testování shody hledaného modelu s náhodně vybraným vzorkem dat. Toto pokračuje do té doby, dokud není pro model nalezen zvolený počet inliers nebo proveden zvolený počet pokusů.

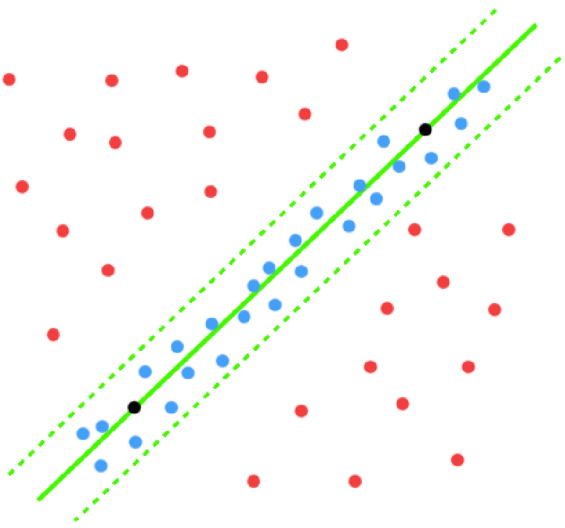
1. Náhodně vyber vzorek s bodů z datové množiny S a instancuj z nich model.
2. Urči množinu všech datových bodů Si, které mají od modelu vzdálenost maximálně t. Nyní se v Si nachází množina všech inliers datové množiny S.
3. Jestliže je počet inliers v množině Si větší než hranice T, odhadni znovu model za použití všech bodů v Si a ukonči algoritmus.
4. Jestliže je počet inliers v množině Si menší než je hranice T, vyber nových s bodů z datové množiny S a proveď znovu předchozí kroky.
5. Po N pokusech vyber Si s nejvyšším počtem inliers, znovu odhadni model za použití všech bodů v Si.

Algoritmus 3.1: RANSAC algoritmus pro obecný model. Přeloženo z [8].

Proměnná t vymezuje hranice modelu, podle které se určí, zda je bod pro daný model inlier nebo outlier a její hodnota je většinou volena na základě experimentů. Z důvodu vysoké výpočetní náročnosti je nevhodné a zbytečné zkoušet všechny možné vzorky z datové množiny S. Místo toho je vybráno pouze N vzorků. Hodnota N musí být dostatečně vysoká tak, aby se s pravděpodobností p zajistilo, že alespoň jeden z vybraných vzorků neobsahuje outliers. Většinou je p zvoleno $0,99$. Pravděpodobnost toho, že jakýkoliv vybraný bod je inliers, zvolme w. Potom pravděpodobnost, že se jedná o outliers lze označit jako $\epsilon = 1 - w$. Bude tedy potřeba minimálně N vzorků (každý vzorek z s bodů), kde $(1 - w^s)^N = 1 - p$. Vhodný počet iterací lze potom určit jako

$$N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^S)} \quad . \tag{3.12}$$

Proměnná T určuje hranici počtu inliers, pro kterou lze model považovat za vyhovující. Pro n bodů by to bylo $T = (1 - \epsilon)n$.



Obrázek 3.5: Vizualizace metody RANSAC. Zde je modelem přímka určená dvěma černými body. Modré body jsou inliers, červené jsou outliers. Zelená přerušovaná čára zobrazuje hranici modelu pro určení inliers a outliers.

3.4 Geometrické transformace souřadnic

Tímto názvem lze souhrnně označit operace, které mění pozice bodů v aktuálním souřadnicovém systému nebo mění souřadnicový systém. Jedná se o jedny z nejpoužívanějších operací v počítačové grafice vůbec a jsou důležité i při skládání obrázků. Vyjadřují se pomocí transformačních rovnic a matic. Celá kapitola vychází z [10].

3.4.1 Afinní transformace ve 2D

Jedná se o lineární transformace, které zachovávají rovnoběžnost. Pokud tedy čáry byly rovné a rovnoběžné, zůstanou takové i po transformaci. Mezi základní afinní transformace lze řadit posun, rotaci, změnu měřítka a zkosení.

Pro jednotnou reprezentaci se používají homogenní souřadnice bodu. Jedná se o jednotnou formu ve tvaru uspořádané trojice $[X, Y, w]$, kde w nazýváme vahou bodu a u lineárních transformací je rovno 1. Příčinou vytvoření těchto souřadnic byl fakt, že posun je vyjádřen jako součet matic, kdežto zbylé tři transformace jsou vyjádřeny jako součin matic. Zavedením homogenních souřadnic lze ke všem čtyřem transformacím přistupovat stejně – jako k násobení matic.

Obecně lze transformaci vyjádřit vztahem $P' = P \cdot \mathbf{A}$, kde P je bodem v homogenních souřadnicích a $\mathbf{A}$ je matice jednotlivých transformací. Pro afinní transformace je tedy výsledný vztah $P' = [X, Y, 1] \cdot \mathbf{A}$.

Posunutí

Přemístění bodu ve směru osy x a y o hodnoty dx a dy. Novou pozici bodů lze vypočítat ze vztahu:

$$x' = x + d_x \ , \quad y' = y + d_y \tag{3.13}$$

Transformační matice má tvar:

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ d_x & d_y & 1 \end{bmatrix} \tag{3.14}$$

Otáčení

Otočení bodu o úhel α se středem otáčení v počátku souřadného systému lze vyjádřit rovnicí:

$$x' = x \cdot cos(a) - y \cdot sin(a) \ , \quad y' = x \cdot sin(a) + y \cdot cos(a) \tag{3.15}$$

Transformační matice:

$$R = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.16}$$

Změna měřítka

Zvětšení či zmenšení je dáno faktorem změny měřítka S_x a S_y. Pokud je $S_{x,y} > 1$, jde o zvětšení. Je-li $0 < S_{x,y} < 1$, jedná se o zmenšení. Při $S_{x,y} < 0$ dochází k zrcadlení. Základní vztah je:

$$x' = x \cdot Sx \ , \quad y' = y \cdot Sy \tag{3.17}$$

Tvar transformační matice:

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.18}$$

Zkosení

Faktor zkosení je označen jako S_{hx} a S_{hy}. Vztah pro zkosení je znázorněn:

$$x' = x + S_{hx} \cdot y \ , \quad y' = y + S_{hy} \cdot x \tag{3.19}$$

Transformační matice:

$$S_H = \begin{bmatrix} 1 & S_{hx} & 0 \\ -S_{hx} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.20}$$

Nežli samostatné, zde uvedené, transformace je častěji nutné použít jejich kombinaci. Důležité je dodržet správné pořadí transformací, neboť u složených transformací na pořadí záleží. Jediná matice, reprezentující složenou transformaci, vznikne postupným vynásobením všech dílčích transformačních matic.

3.5 Homografie

Jiným označením je také projektivita, či projektivní lineární transformace. Jedná se o invertibilní transformaci mezi dvěma projektivními perspektivami, kde se přímky mapují opět na přímky. Celá tato kapitola byla napsána na základě [8] a [3]. Příkladem homografie je středové promítání se středem počátku souřadnic a dvojicí ploch, jež mapuje body z jedné plochy na body druhé plochy $p_i \leftrightarrow p'_i$. Pro body v homogenních souřadnicích pak platí vztah $p'_i = Hp_i$, kde H je transformační matice homografie o rozměrech 3×3:

$$p'_i = Hp_i = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \begin{pmatrix} x'_i \\ y'_i \\ w'_i \end{pmatrix} \tag{3.21}$$

3.5.1 Výpočet homografie

Cílem je získat transformační matici [3], pro niž platí vztah $p'_i = Hp_i$. Z tohoto vztahu je zřejmé, že p'_i a Hp_i si číselně neodpovídají, protože se liší v měřítku daném souřadnicí w'_i. Přesto je možné zapsat $(p'_i)_\times Hp_i = 0$. Nahrazením $(p'_i)_\times$ šikmo symetrickým zápisem a separací neznámých, je výsledkem soustava

$$\begin{pmatrix} 0^\top & -w'_i p_i^\top & y'_i p_i^\top \\ w'_i p_i^\top & 0^\top & -x'_i p_i^\top \\ -y'_i p_i^\top & x'_i p_i^\top & 0^\top \end{pmatrix} h = 0 \tag{3.22}$$

Podoba rovnic je $A_i h = 0$, kde A_i je matice 3×9 a vektor $h = (h_{11}; h12; \ldots; h_{33})^\top$. A_i má hodnost 2 a každá korespondence je tak vyjádřena dvojicí rovnic.

Pro čtyři body vnikne matice A s hodností 8, která tvoří lineární homogenní soustavu rovnic, pro celkem 9 neznámých. Tyto body musí být zvoleny tak, aby žádné tři neležely na stejné přímce. Řešením je potom nulový prostor této matice.

$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 & -y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x'_2 & -y_2x'_2 & -x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y'_2 & -y_2y'_2 & -y'_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & -\vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nx'_n & -y_nx'_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_ny'_n & -y_ny'_n & -y'_n \end{pmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{pmatrix} = 0 \tag{3.23}$$

3.6 Bundle adjustment

Prvním způsobem, jak registrovat více obrázků, je po jednom přidávat nové obrázky a vždy je spojit s předchozími obrázky, které již v panoramatu jsou. Tento přístup ovšem způsobuje akumulování chyby a u 360° panoramat může vést k přítomnosti mezer, či nadměrnému překryvu obou konců panoramatu.

Lepší alternativou je simultánně spojit všechny obrázky dohromady a použít tzv. *bundle adjustment* [19]. Obrázky jsou postupně přidávány do adjusteru tak, že v každém kroku je přidán obrázek s nejvíce korespondencemi. Nově přidaný obrázek je inicializován se stejnou rotací a ohniskovou vzdáleností jako ten, se kterým má nejvíce korespondencí. Následně jsou parametry aktualizovány pomocí algoritmu Levenberg-Marquardt.

Jako účelová funkce je použit robustifikovaný součet kvadratických projekčních chyb [4]. Znamená to, že každý klíčový bod z jednoho obrázku je promítnut do všech ostatních obrázků, se kterými má nějaké korespondence a součet druhých mocnin vzdáleností je zminimalizován v závislosti na parametrech kamery.

Mějme dány korespondence $\mathbf{u}_i^k \leftrightarrow \mathbf{u}_j^l$, kde $\mathbf{u}_i^k$ značí pozici k-tého klíčového bodu v obrázku i. Reziduála je potom:

$$\mathbf{r}_{ij}^k = \mathbf{u}_i^k - \mathbf{p}_i j^k \quad , \tag{3.24}$$

kde $\mathbf{p}_{ij}^k$ je projekce z obrázku j do i bodu odpovídajícího $\mathbf{u}_i^k$

$$\tilde{\mathbf{p}}_{ij}^k = \mathbf{K}_i \mathbf{R}_i \mathbf{R}_j^\top \mathbf{K}_j^{-1} \tilde{\mathbf{u}}_j^l \quad . \tag{3.25}$$

Chybová funkce je součet robustifikovaných reziduálních chyb [19] ze všech obrázků

$$e = \sum_{i=1}^{n} \sum_{j \epsilon \mathcal{I}(i)} \sum_{k \epsilon \mathcal{F}(i,j)} h(\mathbf{r}_{ij}^k) \tag{3.26}$$

kde n je počet obrázků, $\mathcal{I}(i)$ je množina korespondencí do obrázku i, $\mathcal{F}(i,j)$ je množina korespondencí mezi obrázky i a j. Použita je Huberova robustní chybová funkce

$$h(\mathbf{x}) = \begin{cases} |\mathbf{x}|^2 & \text{pro} |\mathbf{x}| < \sigma \\ 2\sigma|\mathbf{x}| - \sigma^2 & \text{pro} |\mathbf{x}| \geq \sigma \end{cases} \quad . \tag{3.27}$$

Pokud je vzdálenost menší než σ, jedná se o inlier. Pokud je vzdálenost větší než σ, jedná se o outlier. Použitá vzdálenost pro outliers je $\sigma = \infty$ během inicializace a $\sigma = 2$ px pro finální řešení.

Algoritmem Levenberg-Marquardt je řešen nelineární problém nejmenších čtverců. Každý krok iterace má tvar

$$\mathbf{\Phi} = (\mathbf{J}^\top \mathbf{J} + \lambda \mathbf{C}_p^{-1})^{-1} \mathbf{J}^\top \mathbf{r} \quad , \tag{3.28}$$

kde $\mathbf{\Phi}$ značí všechny parametry, $\mathbf{r}$ reziduály a $\mathbf{J} = \frac{\partial \mathbf{r}}{\partial \mathbf{\Phi}}$.

3.6.1 Korekce zvlnění

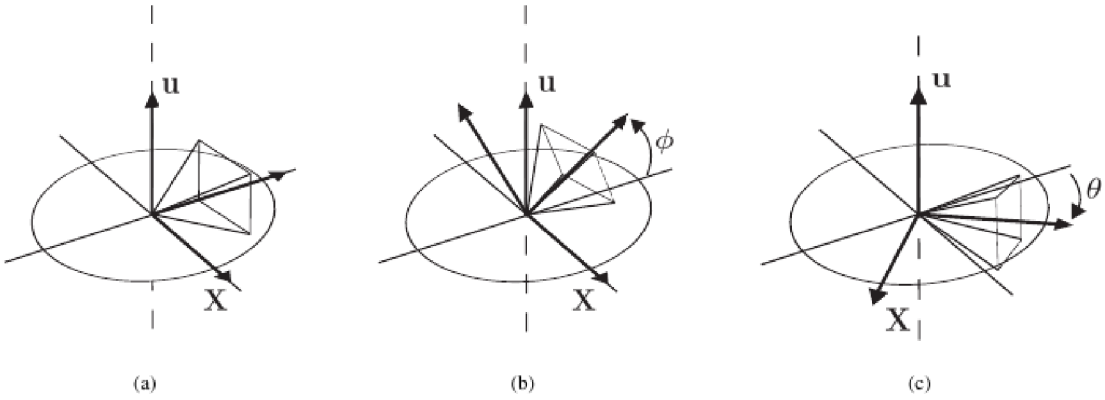
Při skládání obrázků může být výsledné panorama zvlněné. Toto zvlnění lze kompenzovat a celý obrázek vyrovnat heuristikou uvedenou níže [4].

Hlavní myšlenkou je, že lidé jen vzácně otočí kameru relativně vůči horizontu, takže horizontální osa kamery (vektory $\mathbf{X}$) typicky leží v rovině scény. Zároveň vertikální objekty ve scéně, např. stožár, jsou paralelní k okraji fotografie.

Nalezením nulového vektoru kovarianční matice vektorů $\mathbf{X}$ kamery lze získat tzv. *up-vector* $\mathbf{u}$ (kolmý k rovině, která obsahuje střed kamery a horizont)

$$\left(\sum_{i=0}^{n} \mathbf{X}_i \mathbf{X}_i^\top \right) \mathbf{u} = \mathbf{0} \quad . \tag{3.29}$$

Vynásobením všech rotačních matic s tímto vektorem $\mathbf{u}$ nastává korekce zvlnění.



Obrázek 3.6: Zobrazení vektoru $\mathbf{u}$. Při vertikálním náklonu kamery (b) i při horizontální rotaci (c) vektor $\mathbf{X}$ leží v rovině a vektor $\mathbf{u}$ je k této rovině kolmý. Převzato z [4].

3.7 Parametry kamery

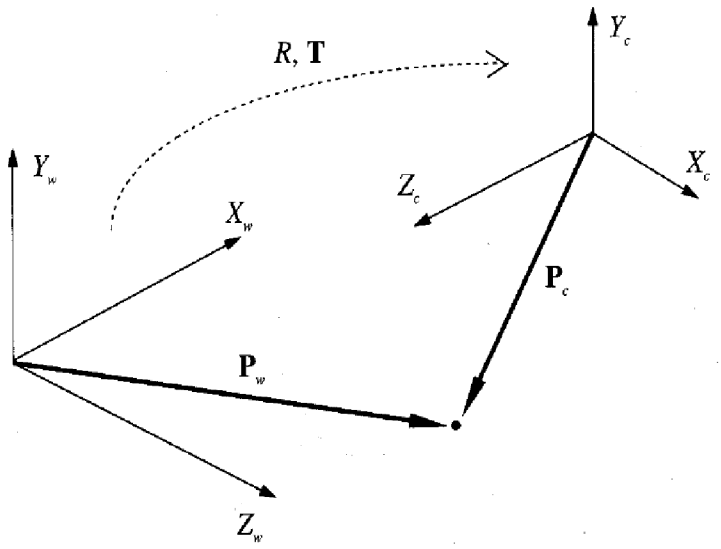
Souřadný systém kamery a souřadný systém světa spolu souvisí podle fyzických parametrů, jako jsou například ohnisková vzdálenost čočky, velikost pixelů, poloha hlavního bodu, poloha a orientace kamery. Lze je rozdělit do dvou skupin na vnitřní a vnější [21].

3.7.1 Vnější parametry kamery

Parametry, které definují polohu a orientaci souřadného systému kamery, v závislosti na souřadném systému obecného světa. Jedinečně nám identifikují transformaci mezi neznámým souřadným systémem kamery a známým souřadným systémem světa.

Určení těchto parametrů typicky znamená [21]:

- Nalezení vektoru posunutí $\mathbf{T}$ mezi relativní pozicí počátků obou souřadných systémů
- Nalezení 3×3 rotační matice R, která by způsobila zarovnání korespondujících os obou souřadných systémů



Obrázek 3.7: Vztah mezi souřadnými systémy kamery a světa. Převzato z [21].

Vztah mezi souřadnicemi bodu $\mathbf{P}$ v souřadném systému světa ($\mathbf{P_w}$) a kamery ($\mathbf{P_c}$) je

$$\mathbf{P_c} = R(\mathbf{P_w} - \mathbf{T}) \tag{3.30}$$

a celková matice vyjadřující vnější parametry kamery je

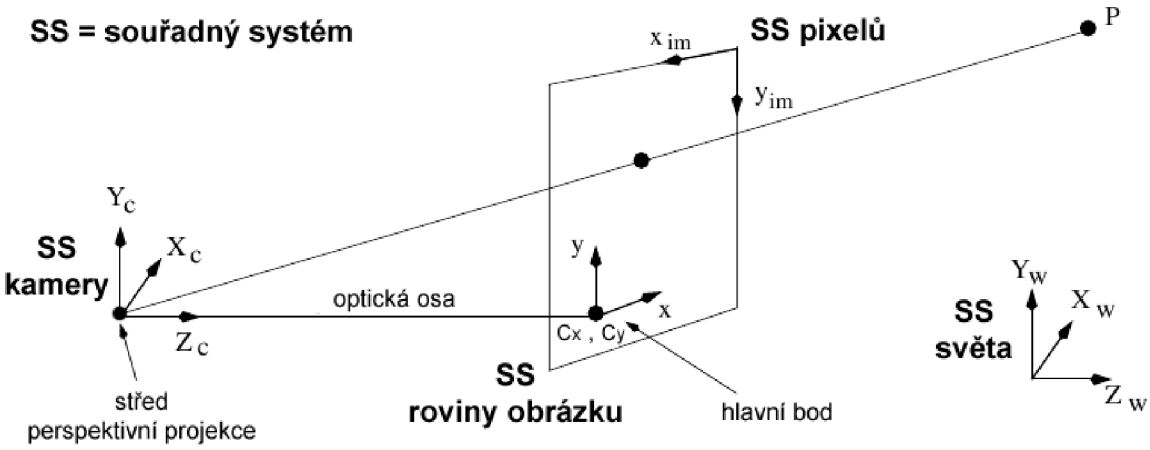
$$M_{ext} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & -\mathbf{R}_1^\top \mathbf{T} \\ r_{21} & r_{22} & r_{23} & -\mathbf{R}_2^\top \mathbf{T} \\ r_{31} & r_{32} & r_{33} & -\mathbf{R}_3^\top \mathbf{T} \end{pmatrix} \tag{3.31}$$

3.7.2 Vnitřní parametry kamery

Parametry udávající vztah mezi souřadnicemi pixelu a souřadným systémem obrázku. Charakterizují optické, geometrické a digitální vlastnosti kamery.

Je potřeba vyjádřit [21]:

- Perspektivní projekce (ohnisková vzdálenost)
- Transformace mezi obrazovou rovinou a souřadnicemi pixelů
- Geometrické zkreslení objektivu



Obrázek 3.8: Vztah mezi pixelem a souřadným systémem obrázku. Přeloženo z [2].

Matice vyjadřující vnitřní parametry kamery má tvar

$$M_{int} = \mathbf{K} = \begin{pmatrix} -f/s_x & 0 & c_x \\ 0 & -f/s_y & c_y \\ 0 & 0 & 1 \end{pmatrix} , \tag{3.32}$$

kde f je ohnisková vzdálenost, s_x a s_y značí efektivní velikost pixelů v obou směrech, c_x a c_y jsou souřadnice hlavního bodu. Matice $\mathbf{K}$ se někdy také označuje jako kalibrační matice.

3.8 Nepravidelné přechody metodou Graphcut

Jedná se o algoritmus pro nalezení nepravidelného přechodu mezi snímky. Z důvodu, že se jedná o oblast teorie grafů, bude dále uveden jen krátký popis. Více informací a podrobností lze nalézt v originálním dokumentu [11].

Tento algoritmus bere obrázek jako propojený graf, který má hrany ohodnoceny na základě rozdílů mezi sousedními pixely. Celý graf je poté řešen jako problém *max-flow/min-cut*[1], kde jako vrcholy *sources* jsou označeny pixely z prvního obrázku a jako vrcholy *sinks* pixely z druhého obrázku.

[1] Tento vztah uvádí Fordova-Fulkersonova věta

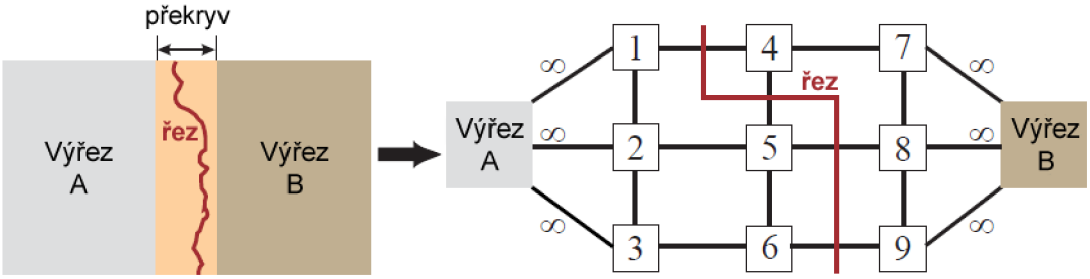
Graf obrázku lze vyjádřit maticí sousednosti, kde je každá položka tok. V [11] je navrženo použít součet sumy čtverců rozdílů mezi prvním a druhým obrázkem a sumou čtverců rozdílů mezi prvním a druhým obrázkem v sousedních pixelech.

Nejdříve se musí zvolit cenová funkce pro pixely z prvního do druhého výřezu. Vybraná cesta povede mezi dvojicí pixelů. Nejjednodušší způsob je měřit změny barvy mezi oběma pixely. Kvalita je definována jako cenová funkce M mezi dvěma přilehlými pixely

$$M(s,t,\mathbf{A},\mathbf{B}) = ||\mathbf{A}(s) - \mathbf{B}(s)|| + ||\mathbf{A}(t) - \mathbf{B}(t)|| \quad , \tag{3.33}$$

kde s značí aktuální pixel a t značí sousední pixel. **A** je první a **B** druhý obrázek.

Na obrázku 3.9 je vidět graf, který zobrazuje jeden uzel na pixel v překryvu dvou výřezů. Cílem je nalézt nejkratší cestu ze shora dolů řešením problému *graph cut*. Pouze demonstrativně se jedná jen o 3×3 region.



Obrázek 3.9: Algoritmus Graphcut. Vlevo zobrazen překryv dvou výřezů. Vpravo grafové vyjádření se zobrazením řezu. Přeloženo z [11].

3.9 Korekce expozice v překryvu

Pro odstranění chyb expozice musí být aplikována přenosová funkce na vstupní obrázek tak, aby vypadal více jako jeho sousedé. Situaci značně ztěžuje, pokud má vstupní obrázek překryv s více sousedy najednou. Proto byla vyvinuta technika pro vyrovnání expozice na základě bloků [23]. Tato technika umožňuje měnit přenosovou funkci na základě překryvů s různými sousedy.

Každý obrázek je rozdělen do bloků. V každém bloku je spočítána kvadratická přenosová funkce, která ve smyslu nejmenších čtverců nejlépe mapuje tento blok do kompozice všech překrývajících se bloků. Iterací lze tuto již vyrovnanou kompozici použít jako referenční pro následující krok. Pro dobré výsledky stačí tři iterace.

Použitím bloků jsou ale ve výsledku vidět hrany těchto bloků. Je potřeba tedy použít vyhlazení. Nejprve se zprůměrují funkce ve všech blocích s jejich sousedy. Použito bylo jádro $(\frac{1}{4}, \frac{1}{2}, \frac{1}{4})$. Pro každý pixel byla dále aplikována přenosová funkce z okolních bloků. Lze použít bilineární či bikvadratické interpolace (rovnice (3.34)).

$$p(x,y) = \sum_{v=-1}^{v=1} C_v(\frac{y}{N}) \sum_{u=-1}^{u=1} C_u(\frac{x}{N}) f_{m+u,n+v}(p(x,y)) \quad , \tag{3.34}$$

kde $p(x,y)$ je hodnota pixelu, N značí velikost bloku, $C_n(x)$ bilineární či bikubický koeficient a $f_{u,v}(x)$ funkci přenosu pro blok u,v.

Kapitola 4

Analýza stavu a návrh řešení

Pokus o složení fotografií do jednoho celku je v počítačovém vidění dosti stará záležitost. Například článek [14] pochází z roku 1975, a to určitě není nejstarší. Za tuto dobu vznikla spousta technik a způsobů jak řešit jednotlivé kroky a etapy.

4.1 Existující nástroje pro tvorbu panoramat

Pro názornost byly vybrány tři nástroje, které se zabývají skládáním obrázků do panoramat. Přestože lze nalézt mnoho nástrojů pro skládání obrázků, nástrojů pro složení panoramatu z videa je velmi málo.

Hugin

http://hugin.sourceforge.net

Jedná se o multiplatformní nástroj pro tvorbu panoramat s otevřeným kódem. Přesněji řečeno, jedná se o grafické rozhraní nástroje *Panorama Tools*. Sám se skládá z několika nezávislých aplikací. Podporuje různé jazyky, včetně češtiny.

Obsahuje veliké množství nastavení, ale nástroj je na první dojem poměrně složitý. Zvládá například úpravu expozice, vyvážení bílé barvy či práci s *High Dynamic Range* obrázky. Velice užitečné je zobrazení náhledu v reálném čase ve 3D a nástroj pro ruční korekci nalezených klíčových bodů.

PTGui

http://www.ptgui.com

Nástroj pro tvorbu panoramat pod systémem Windows a Mac OS X. Jedná se o placený nástroj, který umožňuje stažení omezené verze na vyzkoušení. Tato zkušební verze přidává do generovaných panoramat opakovaně vodoznak. Stejně jako Hugin, zvládá skládat obrázky ve více řadách a sloupcích. Jedná se o výkonný nástroj, se kterým lze s minimálním úsilím vytvořit kvalitní panorama.

Obsahuje asistenta, který plní funkci průvodce, a tak i bez jakýchkoliv znalostí o skládání obrázků lze získat v pár krocích výsledné panorama. Samozřejmostí jsou různé optimalizace pro dosáhnutí co nejlepšího výsledku a všemožná volitelná manuální nastavení.

Image Composite Editor

http://research.microsoft.com/en-us/um/redmond/groups/ivm/ice/

Nástroj firmy Microsoft, který je pro nekomerční použití zdarma. Oproti oběma předchozím nástrojům má minimum funkcí a tím pádem je i intuitivnější co se ovládání týče. Jako jediný zde uvedený nástroj umí skládat panorama přímo z video souboru. Jedná se o jediný nástroj zdarma, který jsem našel, co toto vůbec umí.

4.2 Shrnutí a zhodnocení současného stavu

V kapitole 3 byly představeny pouze konkrétní techniky a algoritmy, které budou dále využity. Je to z důvodu, že nějaký komplexnější rozbor technik a algoritmů by vydal na samostatné dílo. K zasazení těchto vybraných technik do širšího kontextu proto poslouží následující kapitola.

Nevhodným způsobem při skládání snímků je vzít každý obrazový bod z jednoho snímku a hledat ho ve druhém snímku. Tato metoda vyžaduje velký výpočetní výkon a stejně se vyskytne mnoho chyb. Lepší řešení je z obrazu extrahovat významné body, kdy potom celý obraz budou reprezentovat jen tyto vybrané.

4.2.1 Detektor a deskriptor

Existuje velké množství různých detektorů pro detekci rohů, hran či skvrn. Detektory jako Harrisův, FAST, SUSAN, pouze naleznou souřadnice klíčových bodů. Deskriptor se tak musí dopočítat zvlášť nebo se vůbec nepoužije. Příklad samostatného deskriptoru je BRIEF [5]. Místo deskriptorů se může zvolit korelace (respektive normalizovaná křížová korelace) pro srovnávání okolí bodů. Tak či tak, dostáváme dva různé problémy k řešení.

Dále jsou robustní detektory, které přímo obsahují výpočet deskriptoru pro snadnější hledání korespondencí. Jsou jimi často používaný SIFT a SURF i méně známý ORB. U těchto kompletních detektorů se přímo předpokládá využití vzájemného hledání shod.

Výhodou první skupiny detektorů je možnost přesně nakombinovat detektor a deskriptor dle požadovaného využití. Například u metody sledování klíčových bodů se deskriptory vůbec nepoužívají a bylo by tedy zbytečné je počítat. Naopak, pro skládání snímků je vhodné spíše využít některou z komplexních metod z druhé skupiny.

4.2.2 Hledání shod versus sledování bodů

U hledání shod se nejdříve extrahují všechny klíčové body a následně na základě deskriptoru jsou hledány korespondence. Při sledování jsou klíčové body extrahovány pouze z prvního snímku a následně jsou tyto body lokalizovány v dalším snímku.

Naivním řešením pro hledání shod je porovnávat hrubou silou všechny deskriptory navzájem, proto se ani nepoužívá. Zase zde totiž narazíme na výpočetní náročnost. Proto se používají speciální multidimenzionální struktury a strategie pro rychlé prohledávání prostoru.

4.2.3 Optimalizace

Je spousta optimalizací, které sice nesouvisí přímo s procesem skládání obrázků, ale výrazně vylepšují vzhled výsledku. Za poměrně zásadní posun ve skládání lze považovat použití

algoritmu *bundle adjustment* (kap. 3.6) vůbec pro účely počítačového vidění, neboť jeho původní využití bylo pro fotogrammetrii, což je zjednodušeně věda, zabývající se měřením rozměrů a polohy objektů na snímku.

Panorama může být mapováno na různé povrchy, místo prostého mapování do roviny. Žádanou je korekce expozice pro úpravu světelných změn mezi jednotlivými snímky. I pro odstranění neovlivnitelných vad objektivů existují zmírňující techniky. Například kvůli vinětaci pouhým spojením dvou obrázků vznikne viditelný přechod mezi oběma snímky. Ten lze zmírnit vzájemným prolnutím snímků. Pro zamezení ostrého přechodu v místě spojení může být nalezena nepravidelná spojnice, která nahradí běžně rovný okraj obrázku. Pokud se pohybující objekt vyskytne pouze v jednom z obrázků zrovna v místě překryvu, vznikne z něho duch, což lze odstranit metodou *deghosting*.

4.2.4 Výpočetní rychlost

Dalo by se říct, že čehokoliv chceme při skládání dosáhnout, máme několik odlišných způsobů, jak se toho dobrat. Existuje spousta dobrých a kvalitních algoritmů, které na to lze použít. Co se výpočetního výkonu týče, problém je spíše ve vstupních datech. Obrázky či snímky videa obsahují tisíce bodů, které je potřeba zpracovat. U videa, které má například 30 snímků za vteřinu, je to ještě markantnější.

Pro zrychlení algoritmů lze využít možnosti paralelizace. Zatímco se procesor skládá třeba z 8 jader, grafický procesor obsahuje stovky menších jader. Vše stále obstarává procesor, jenom výpočetně náročné operace se vypočítají na grafické kartě a procesor jen převezme výsledek a dále ho využívá. Firma nVidia například poskytuje platformu CUDA pro paralelní výpočty na svých grafických kartách [15].

Je ovšem na zvážení, kdy se GPU výpočty opravdu vyplatí použít. Před prvním zavoláním jakékoliv funkce z knihovny CUDA totiž vždy probíhá inicializace, která trvá i kolem 10-15 vteřin. Stojí za zvážení, zda při jednodušších operacích tento inicializační čas nepřekoná celkovou dobu, kterou by se operace prováděla na procesoru.

Přesto nelze zrychlování přes výpočty na grafických kartách považovat za jediný správný směr, neboť někdy nemusí grafické karty výpočty podporovat, či je potřeba skládat obrázky jen na vestavěných počítačích.

4.3 Cíle návrhu

Osobně si nemyslím, že pro složení kvalitního panoramatu je nezbytně nutné vymýšlet nové algoritmy a tato práce se o to ani nebude pokoušet. Celý proces skládání je tvořen několika samostatnými algoritmy, což znamená, že různou kombinací postupů lze získat různé výsledky. V jedné kombinaci postupů lze eliminovat různé chyby, stejně tak ale můžeme zanést další.

Zaměřit bych se chtěl hlavně na rychlost a kvalitu. Podle mého názoru je největší prostor pro zrychlení na samém začátku, v etapě detekce klíčových bodů a hledání korespondencí. Zde je důležitá volba samotných algoritmů, stejně jako datových struktur. Bude potřeba vybrat takové algoritmy, které jsou rychlé, ale zároveň jejich zrychlení není na úkor kvality či přesnosti.

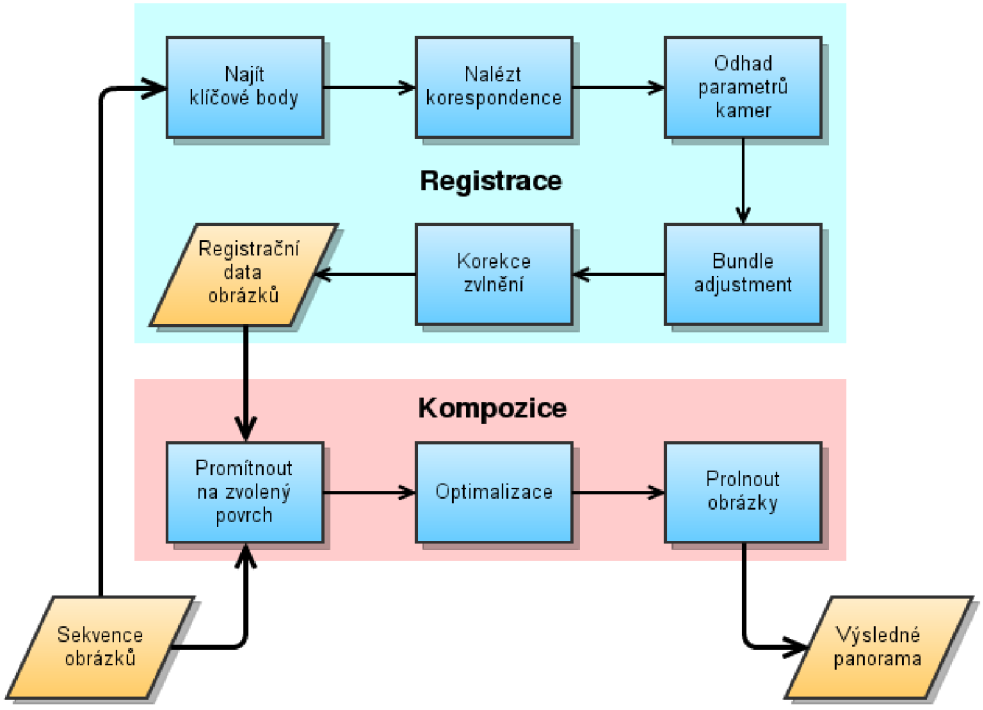
4.4 Návrh řešení práce

Celý návrh řešení skládání obrázků do panoramat je tvořen z několika pevně za sebou následujících kroků, které se ale mohou dost podstatně lišit v samotném provedení. Návrh bude pro tvorbu panoramat využívat klíčových bodů a bude rozdělen do dvou hlavních částí. Diagram obou částí i jednotlivých kroků je zobrazen na obrázku 4.1.

První z nich je **registrace** obrazových dat. Registrace je vůbec tou nejdůležitější částí, protože při nesprávných registračních datech nelze složit dobré panorama, ať už pak mluvíme o drobné chybě ve výsledku, či chybě několika desítek pixelů. Úvodními kroky jsou nalezení klíčových bodů ve všech snímcích a zjištění korespondencí. Základní schéma znázorňující zmíněné kroky bylo již představeno, a to v kapitole 2 na obrázku 2.1. Po nalezení korespondencí je nutné zjistit perspektivní transformaci mezi dvěma snímky – homografii. Na jejím základě lze odhadnout parametry kamery. Získáme tak informace, jak zhruba byly snímky před kamerou rozmístěny při jejich pořizování.

Druhou částí je **kompozice** obrázků. Zahrnuje v sobě různé techniky a optimalizace související s tvorbou výsledného panoramatu. Prosté nakopírování obrázků přes sebe by dalo vyniknout všem nepřesnostem. Tyto optimalizace nejsou nepostradatelné, neboť se nezaměřují na samotné skládání a správný výsledek bychom mohli dostat i bez nich. Na druhou stranu nám poskytují vylepšení po kvalitativní stránce.

Kostra tohoto návrhu je rámcově inspirována technikami a postupy navrženými v [19] a [4]. Bude ale kladen větší důraz na rychlost a kvalitu výsledku, proto je potřeba při řešení některé kroky a techniky obměnit.



Obrázek 4.1: Diagram znázorňující funkci nástroje. Oranžovou barvou jsou znázorněny vstupy a výstupy, modrou barvou pak procesy vedoucí k výsledku.

Kapitola 5

Nástroj pro tvorbu rotačních panoramat

Tato kapitola pojednává o teoretickém návrhu programu pro tvorbu rotačních panoramat. Budou postupně navrženy a diskutovány jednotlivé etapy vedoucí k vytvoření panoramatického obrázku.

Zdrojové snímky či videosekvence jsou snímány z jednoho místa rotací kamery kolem vertikální osy, která prochází optickým středem této kamery. Toto je pouze teoretický předpoklad a ve skutečnosti se předpokládají drobné odchylky, které vzniknou přirozeným, nemechanickým pohybem kamery v ruce. Návrh i následná implementace jsou provedeny s myšlenkou na dobrý poměr kvality a výkonu, což bude diskutováno v kapitole 7.

Nejdříve, bez jakéhokoliv uvažování informací o obrázku uveďme, jak by probíhalo ruční skládání fotografií. Vzali bychom dva obrázky a druhý bychom pouhým posunem zkoušeli umístit na první jen tak, aby okraj druhého obrázku pasoval do scény obrázku prvního. Záhy bychom zjistili, že v tomto způsobu je příliš mnoho prostoru pro chyby a že je potřeba využít toho, co obrázek zobrazuje. Najdeme výrazný společný prvek obou obrázků a na jeho základě provedeme spojení. Nebude nám ovšem stačit pouze horizontální či vertikální posun. Z toho plynou tři nutné podmínky pro automatické skládání:

1. Snímky se určitou částí překrývají.
2. V překryvu existuje něco (objekt, textura), na základě čeho lze snímky k sobě jednoznačně přiřadit.
3. Máme k dispozici transformace, jejichž aplikováním lze snímky spojit.

5.1 Detektor a deskriptor klíčových bodů

V této práci bylo zvoleno použití kombinace detektoru a deskriptoru v jednom celku. Zde se přímo nabízí často používaná metoda SIFT, která je invariantní vůči měřítku a rotaci. Vlastnostmi a výsledky vyhovující, výkonem bohužel nikoliv. Byla tedy vybrána metoda SURF, která z ní vychází, má podobné výsledky, ale je mnohem rychlejší, viz 3.1.

I přes znatelné zrychlení oproti SIFT, výpočetní doba metody SURF na jeden snímek je relativně vysoká. Záleží ale spíše na konkrétním použití. Pokud panorama skládáme z obrázků v jednotkách kusů, jedná se o robustní detektor a deskriptor s výsledky v přijatelném čase. Pokud jsou ovšem snímků desítky či stovky, může se uživateli výpočetní doba zdát dlouhá, a to i přesto, že se většinou jedná o lineární nárůst času.

Bylo potřeba najít takovou, která bude rychlejší a zároveň bude mít stále dobré výsledky v porovnání s metodou SURF. Vybrána byla nakonec metoda ORB, která se skládá z FAST detektoru a BRIEF deskriptoru, viz kapitola 3.2. Na rozdíl od SURF není ORB invariantní vůči měřítku, hledá konstantní počet klíčových bodů a je značně rychlejší.

Obě metody mají pro i proti a při testech nešlo určit jednoznačného vítěze. Místo toho budou použity obě a v programu půjde zvolit, který detektor použít. Myšlenkou je preferovat ORB kvůli rychlosti, ale mít možnosti použít alternativu, SURF, pokud by byl například výstup s velkým množstvím defektů. Detekce vždy probíhá na snímkách ve stupních šedi.

(a) SURF, 3339 bodů (b) ORB, 1530 bodů

Obrázek 5.1: Oba detektory v praxi. Detekováno na snímku města Třebíče v rozlišení 0,7 Mpx.

5.2 Hledání korespondencí a homografie

Jak vyplývá z předchozí kapitoly, hledání shod mezi snímky bude probíhat na základě porovnávání deskriptorů nalezených klíčových bodů. Hledání významných bodů a následných korespondencí jsou klíčové kroky, které jsou časově nejnáročnější a zároveň i nejdůležitější pro správnost výsledného panoramatu.

Při návrhu byly uvažovány dvě různé možnosti. Vstupní obrázky nemusí být správně seřazeny. Respektive, pokud nebude explicitně zadáno, že jsou obrázky již seřazeny, musíme předpokládat, že jsou v náhodném pořadí. To znamená, že se musí shody hledat mezi všemi snímky navzájem. Vstupní obrázky ale mohou být seřazeny, a tak fakticky stačí hledat korespondence pouze mezi sousedícími snímky. Hledáním shod systémem každý s každým, v již seřazené posloupnosti obrázků, zbytečně roste výpočetní doba bez jakýchkoliv výsledků. Je to ale přesto možné bez negativního dopadu na výsledek. Naopak, testovat pouze sousední snímky v neseřazené posloupnosti, ztrácí smysl.

Jako výchozí možnost se tedy bude testovat systémem každý obrázek s každým, ale půjde zvolit i testování pouze s dvěma následujícími snímky. Dvěma proto, že se při testování algoritmu stávalo, že ze tří snímků měl první a třetí lepší shody, než první a druhý.

5.2.1 K-nejbližších sousedů

Pro hledání shodných párů byl použit algoritmus k-nejbližších sousedů na základě euklidovské metriky, kde k bylo zvoleno 2. Tento postup již byl zmíněný v kapitole 2.3.1. Pro každý klíčový bod v obrázku A (levý) jsou nalezeni dva nejbližší sousedé z obrázku B (pravý). Následně je proveden test poměru vzdáleností mezi prvním a druhým sousedem (D_1 a D_2). Pokud vzorec (5.1) platí, máme potencionální shodný pár pro aktuální klíčový bod a jeho nejbližšího souseda.

$$\frac{D_1}{D_2} < 1 - T \tag{5.1}$$

Aby bylo nalezeno více potenciálních shod, je celý postup použit i v opačném směru, tedy z obrázku B do A. Pokud platí podmínka (5.1) a zároveň takový pár ještě nemáme, uložíme si ho.

Jako práh T byly zvoleny doporučené hodnoty 0,65 pro SURF a 0,3 pro ORB. Při hledání správné hodnoty T jsem zjistil, že nižší hodnota prahu způsobila nalezení velkého počtu korespondencí, ale u obrázků způsobila buď velmi podobné spojení, jako nakonec zvolené hodnoty, nebo dokonce i lepší.

Další poznatek byl, že u většiny obrázků je nejvíce korespondencí v prostředku obrázku ve vertikálním směru. Vychází to z předpokladu, že při pořizování fotografie se člověk snaží snímat scénu tak, aby se například budovy či horizont krajiny nacházely uprostřed. Méně korespondencí na krajích obrázků občas způsobovalo špatné spojení. Byl aplikován jednoduchý způsob, jak zvýšit počet korespondencí při okrajích obrázků a zároveň ho nezvyšovat v jeho středu. Jak lze vidět na obrázku 5.2, byly snímky rozděleny na třetiny a pro krajní třetiny je práh T jiný, než pro středovou.

pro SURF	pro ORB
T = 0,55	T = 0,25
T = 0,65	T = 0,3
T = 0,55	T = 0,25

Obrázek 5.2: Maska zobrazující rozdělení obrázku na třetíny. Práh T se liší dle použitého detektoru a podle umístění třetiny v obrázku.

5.2.2 Nalezení homografie a určení inliers

Jakmile nalezneme odhady korespondujících si bodů, musíme vyfiltrovat ty, které jsou odhadnuty špatně. Tento krok je esenciální, neboť nalezená perspektivní transformace z velké části určuje kvalitu a správnost výsledného panoramatu.

Nejjednodušší využití homografie je pro získání transformace jednoho obrázku do druhého. Zde ale bude homografie využita více. Použije se i pro odhad ohniskových vzdáleností a odhad rotace.

Homografie mezi dvěma snímky je hledána tak, aby chyba zpětné projekce (5.2) byla co nejmenší.

$$\sum_i \left(x'_i - \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 + \left(y'_i - \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 \tag{5.2}$$

Pro odfiltrování nesprávných korespondencí je počítána homografie s využitím iterativní metody RANSAC. V každé iteraci jsou vybrány čtyři korespondence, ze kterých se vypočítá homografie. Následně se spočítají inliers, které vypočtené homografii odpovídají. Iterativně se testují náhodná řešení a to s největším počtem inliers je uloženo. Počítat homografii pouze ze čtyř vzorků by nemuselo mít uspokojivé výsledky. Proto je homografie spočítána ještě znovu, nyní pouze pro inliers. Až tato matice homografie je brána jako výsledná.

5.2.3 Verifikace shod

Pro verifikaci nalezených korespondencí jako celku jsem použil podmínku zveřejněnou v [4]

$$n_i > \alpha + \beta n_f \quad , \tag{5.3}$$

kde n_f značí počet nalezených shod a n_i počet pouze inliers. Zbylé dvě proměnné jsou konstanty, $\alpha = 0,8$ a $\beta = 0,3$.

Vzorec je pro lepší použití upraven na rovnici (5.4), kde c značí konstantu, kterou bude každá homografie ohodnocena.

$$c = \frac{n_i}{\alpha + \beta n_f} \tag{5.4}$$

Při skládání se budou uvažovat pouze homografie, pro které bude platit $1,0 < c < 3,0$. Nebudou se tak uvažovat korespondence snímků, které mají příliš málo či hodně inliers (malý, respektive moc velký překryv obrázků). Pokud podmínka (5.3) neplatí, tak usuzuji, že dané dva snímky spolu žádné korespondence nemají.

5.3 Předzpracování videosouboru

Požadavkem na nástroj byla možnost získat panorama jak z jednotlivých obrázků, tak také z videosekvence. Zatímco počet obrázků na vstupu lze předpokládat maximálně v desítkách, videosekvence má stovky či tisíce snímků.

Pro spojování snímků z videa by bylo nejlepší použít techniku *feature tracking*, kdy se naleznou klíčové body v prvním snímku, a poté se tyto body sledují. Při sledování optického toku videa zjistíme vektory, o které se body posunuly, což lze dále využít i při tvorbě panoramat.

Použil jsem ovšem jinou možnost, a to z videa extrahovat jednotlivé snímky a dále se k nim chovat stejně, jako by se skládaly běžné fotografie. Hned na první pohled lze vidět, že se jedná o podstatně horší možnost, neboť jsme se dobrovolně vzdali všech informací, které videosekvence nabízí oproti běžným fotkám. Vzhledem k tomu, že jsem se v navrhovaném nástroji zaměřil na rychlé hledání shod mezi snímky pro zrychlení celého procesu skládání, volil jsem tuto horší možnost, neboť lépe zapadá do celkového konceptu této práce. Zvolením první možnosti by se totiž musel použít jiný detektor klíčových bodů a celý krok hledání korespondencí by rázem odpadl.

5.3.1 Výběr klíčového snímku

Pouze extrahovat všechny snímky a přímo je skládat by nebylo možné. Zpracování by trvalo příliš dlouho a kvůli podobnosti sousedních snímků by většina korespondencí nemusela splňovat podmínku uvedenou v kapitole 5.2.3 a byly by tak zamítnuty. Místo toho je videosoubor předzpracován tak, že je vybráno pouze několik klíčových snímků.

Výběr se skládá z nutného minima již představených kroků, a to detekce klíčových bodů a nalezení korespondencí. V každém snímku jsou nalezeny ORB detektorem klíčové body. Následuje fáze hledání korespondencí, která nyní probíhá mezi klíčovým snímkem a snímky následujícími. Fáze je podobná té, kterou používám pro obrázky. Nyní ale není důležitá ani tak kvalita, ale pouze rychlost, protože se nalezené korespondence nebudou používat přímo k samotnému složení panoramatu, ale jen pro určení klíčových snímků. Shody jsou tak hledány v jednom směru, a to z klíčového snímku. Nalezne se homografie s využitím algoritmu RANSAC, která se ale nyní neukládá, pouze slouží k spočítání inliers. Původně jsem kvůli rychlosti inliers nehledal a bral jsem všechny shody. Ukázalo se ovšem, že vyfiltrování outliers výpočetní čas zvýší jen nepatrně při dosáhnutí mnohem lepších výsledků.

Pro klíčový snímek je nejprve zjištěn počet inliers s přímo následujícím snímkem. Tato hodnota, upravená o koeficient, slouží jako referenční pro určení nového klíčového snímku. Testováním byla zvolena hodnota koeficientu $0,15$. Hodnota se ale může lišit dle natáčené scény, a proto bude možnost koeficient měnit. Při testech se hodnota pohybovala mezi $0,1$ a $0,2$. Pokud počet inliers aktuálního snímku s posledním klíčovým klesne pod referenční hranici, snímek je prohlášen za klíčový.

Jak již bylo uvedeno, vše bylo prováděno s důrazem na rychlost. Klíčové snímky jsou výstupem, který je nástrojem zpracován stejně, jako by se jednalo o obrázky uložené na disku. Při skládání se tak pro klíčové snímky znovu hledají klíčové body a korespondence. Je to hlavně z důvodu, aby mohl být použit i SURF detektor a kvalitnější způsob hledání korespondencí z kapitoly 5.2.

5.4 Odhad parametrů kamery

Jelikož je panorama tvořeno rotací kamery, je potřeba pracovat s jejím modelem, tj. zjistit parametry kamery. Pro určení kalibrační matice vnitřních parametrů je nutné zjistit ohniskovou vzdálenost a hlavní bod.

Hlavní bod je označení pro bod, který leží v průsečíku optické osy a roviny obrazu. Z tohoto bodu se měří ohnisková vzdálenost. Pro zjednodušení umístíme hlavní bod do středu obrazu. Tento bod často ve středu obrazu opravdu je, přestože to nemusí platit vždy. Ohniskovou vzdálenost můžeme zjistit z EXIF *Exchangeable image file format* metadat fotky. Tyto metadata do fotek vkládají přímo fotoaparáty. Bohužel je zcela na výrobcích, jestli fotoaparát EXIF data uloží. Někdy se může stát, že ačkoliv jsou metadata k dispozici, nejsou správná. Druhou možností by bylo nechat na uživatelovi, aby ohniskovou vzdálenost zadal. Předpokládám, že většina uživatelů tuto hodnotu neví a proto by je zadávání obtěžovalo či odradilo. V rámci co největší automatizace aplikace tedy bude ohnisková vzdálenost odhadnuta z homografií. Odhadnuté ohniskové vzdálenosti jsou seřazeny a jako výsledná se vezme medián. Upravením matice vnitřních parametrů z kapitoly 3.7.2 získáme inicializační

matici pro kamery všech obrázků

$$\mathbf{K} = \begin{pmatrix} f_o & 0 & w/2 \\ 0 & f_o & h/2 \\ 0 & 0 & 1 \end{pmatrix} \quad , \tag{5.5}$$

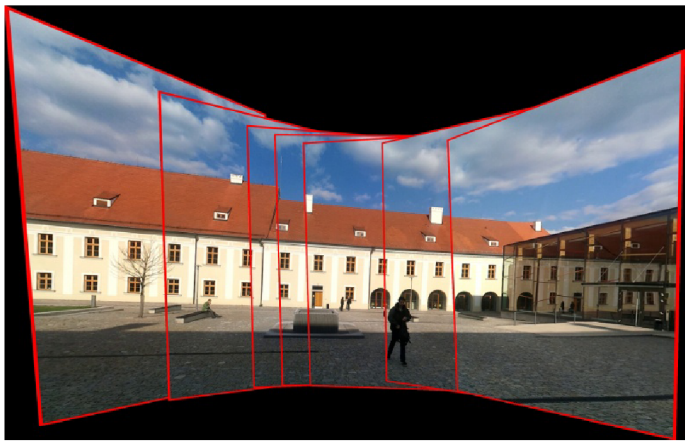
kde f_o je odhadnutá ohnisková vzdálenost, $w/2$ a $h/2$ značí střed obrázku.

Ohnisková vzdálenost je pouze odhadnutá a tudíž nepřesná. Pro zpřesnění se použije *bundle adjustment*. Ohniskové vzdálenosti jsou znovu seřazeny a medián bude použit jako hlavní ohnisková vzdálenost, která bude využita při mapování na kouli nebo válec. Pro všechny rotační matice je ještě nalezen a aplikován up-vektor pro korekci zvlnění výsledného panoramatu (kapitola 3.6.1).

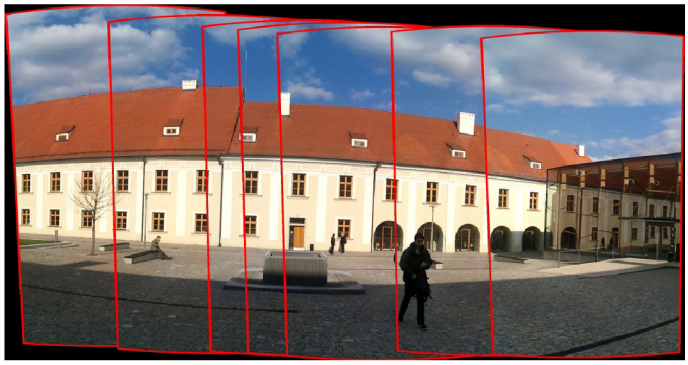
5.5 Povrch pro mapování

Pokud bychom pořídili snímky paralelně, tj. posunem kamery, mohli bychom je naskládat vedle sebe do roviny bez většího zkreslení. U snímků, pořízených rotací kamery, skládáme-li menší množství, bylo by přirozené zvolit některý obrázek jako referenční a ostatní mapovat do jeho souřadnicového systému (obr. 5.3a). Pro zobrazení většího zorného úhlu je to ovšem nepraktické, protože se po krajích začnou pixely obrázků nepříjemně natahovat.

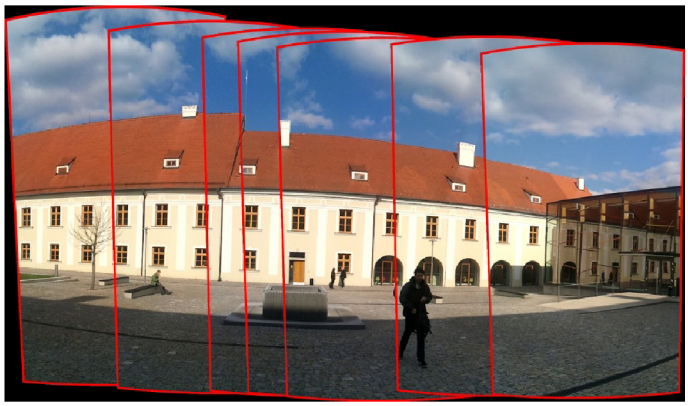
Volil jsem proto dva často používané povrchy a mapování do roviny vůbec neuvažoval. Jako referenční snímek je zvolen ten, který má nejvíce shod s ostatními obrázky. Preferovaný povrch je koule (obr. 5.3b), případnou alternativou pak válec (obr. 5.3c). Musíme každý pixel výsledného panoramatu převést na paprsek a poté ho mapovat zpět do každého obrázku na základě rovnic zvolené projekce.



(a) Rovina



(b) Koule



(c) Válec

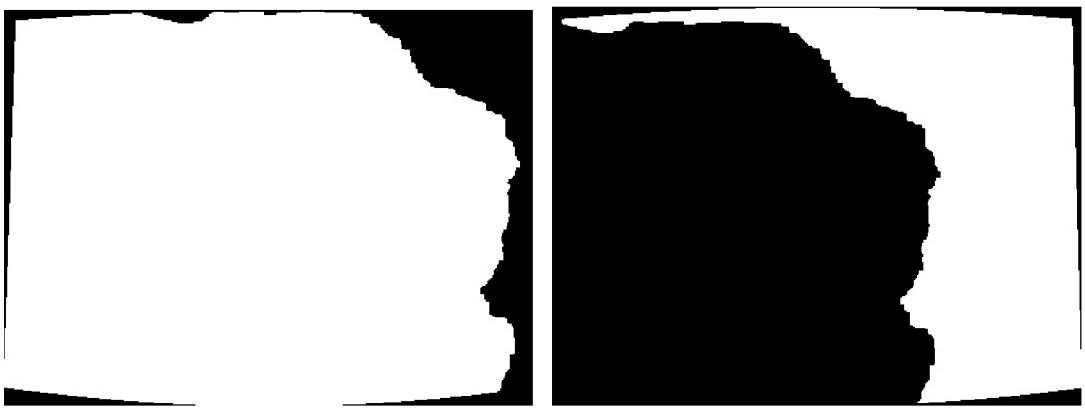
Obrázek 5.3: Ukázka pěti fotografií areálu FIT VUT v Brně, jak jsou mapovány na různé povrchy.

5.6 Optimalizace

Jako optimalizace budou postupně využity tři techniky. Hlavně u fotografií pořízených ve venkovních prostorách je běžné, že přestože fotíme z jednoho místa stejnou scénu, mění se světelné podmínky. Výsledné snímky v panoramatu by tak byly různě světlé či tmavé, což by způsobilo nepěkný výsledek. Byla použita bloková korekce expozice v místě překryvu (kapitola 3.9). Lepším řešením by bylo zvolit některou z komplexnějších metod pro celkovou korekci expozice snímku.

Jako druhá optimalizace byla zvolena metoda Graphcut představená v kapitole 3.8. Nalezením nepravidelného přechodu (řezu) mezi snímky lze dosáhnout kvalitnějšího spojení. Navíc se částečně eliminuje chyba způsobená pohybujícím se objektem v překryvu.

Poslední úpravou, kterou můžeme zařadit do optimalizací, je míchání, neboli prolnutí. I když se použijí obě předchozí metody, stále nemusí být snímek dokonalý. Na snímcích se projevuje vinětace, což je vada objektivu, způsobující nižší jas na okrajích fotografie. Tuto a jiné vady lze do značné míry korigovat mícháním snímků, konkrétně bude použita metoda Multi-Band Blending [4].



(a) Maska 1 (b) Maska 2

(c) Snímek 1 (d) Snímek 2

Obrázek 5.4: Ukázka nalezeného řezu metodou Graphcut, který poslouží pro spojení. Pro každý obrázek je nalezena maska ořezu.

Kapitola 6

Implementace nástroje

K dalším cílům této práce patřilo postupy navržené v předchozí kapitole také implementovat do funkčního celku. Implementovaný program zvládne spojovat sadu jednotlivých obrázků či snímky videosekvence do výsledného panoramatu.

Není jednoduché vytvořit nástroj, který bude automatický a zároveň bude vždy fungovat správně. Vstupní sady snímků se mohou dost podstatně lišit. Některé jsou světlé, jiné tmavé. Scéna může být příliš pravidelná nebo málo členitá. Nástroj proto funguje ve dvou režimech. V prvním, plně automatickém, je mu pouze předána sada obrázků či jeden video soubor a program vyprodukuje výsledek. V druhém režimu má uživatel možnost některá klíčová nastavení programu přenastavit pro dosažení lepších výsledků a lze tak korigovat případné nedostatky, které by produkoval automatický režim. Možná nastavení programu jsou zobrazena v příloze A.

Program je napsán v jazyce C/C++ a využívá knihovnu počítačového vidění OpenCV a dále knihovnu Qt. Jak plyne z kapitol 3 a 5, implementovat kompletně všechny navržené algoritmy by byl poměrně nelehký úkol. S aktuální verzí knihovny OpenCV, nyní ve verzi 2.3.1, byl zveřejněn Stitching modul. Tento modul je zatím z velké části nekomentovaný, nezdokumentovaný, obsahuje množství ladících výstupů a lze odhadnout, že se zdaleka nejedná o finální verzi. Přesto již nyní obsahuje množství tříd, které souvisí se skládáním obrázků. Implementovaný program se snaží maximálně využít funkce OpenCV knihovny a tohoto modulu.

6.1 Vstup a výstup programu

Program pracuje s množinou obrázků nebo s jedním video souborem. Jejich formát je přímo ovlivněn podporou v knihovně OpenCV. Vstupní obrázky či snímky videa jsou načítány v plném rozlišení i barvách. Při samotném zpracování se ovšem používají pouze zmenšeniny ve stupních šedi, hlavně kvůli paměťovým nárokům. Detekce klíčových bodů obrázků či snímků videa probíhá na rozlišení maximálně 0,7 Mpx. Pokud má vstupní snímek vyšší rozlišení, je zmenšen. Není použitý pevný rozměr, zmenšuje se poměrově tak, aby plocha obrázku odpovídala rozlišení 0,7 Mpx. Hodnota byla nalezena experimentálně a oběma detektorům toto rozlišení postačuje.

Knihovna podporuje různé formáty obrázků, které se ale mohou lišit použitým operačním systémem. Měly by jít načíst obrázky v nejvíce používaných formátech jako jsou BMP, JPG, PNG a TIF. U videa je situace složitější, neboť se podporované formáty liší také podle operačního systému, ale i podle kodeků a kontejnerů, které jsou v systému k dispozici.

U komprimovaných videí nastával problém při získávání konkrétních snímků. Některé komprimované kodeky ukládají pouze změny mezi snímky a OpenCV funkce se s tím neuměla vypořádat a vracela snímky s artefakty. Z tohoto důvodu byla testovací videa převedena do nekomprimovaného formátu RAW I420, který umí OpenCV zpracovat na všech systémech. Bohužel, video soubor v tomto formátu má velikost přes 600MB u sekvence s délkou 20 vteřin.

Ve finální části skládání, v kompozici panoramatu, je obrázek znovu načten v originálním rozlišení. Nedochází již k žádné korekci rozlišení a panorama je tak skládáno z obrázků v původních rozměrech.

6.2 Třída předzpracování video souboru

Třída `StitcherVid` má na vstupu video soubor a jejím výstupem je množina snímků z videa. Účelem je vybrat klíčové snímky, které budou následovně použity při samotné tvorbě panoramatu. Jedná se tedy pouze o jakýsi předstupeň samotného skládání.

V třídě jsou obsaženy pouze čtyři metody. Metoda `loadVid()` slouží k inicializaci videa pro získávání jednotlivých snímků a uložení základních parametrů, jako je například rozlišení a celkový počet snímků. V jednotlivých snímcích jsou postupně hledány klíčové body v metodě `findFramesFeatures()`.

Jádrem třídy je metoda `selectKeyFrames()`, která se stará o výběr klíčových snímků. Pro každý klíčový snímek je nalezen počet inliers s přímo následujícím sousedním snímkem, což označíme jako T. Dále zavedeme váhu w pro určení hraničního počtu inliers. Aktuální klíčový snímek je postupně porovnáván se všemi nadcházejícími snímky a jakmile klesne počet inliers při hledání korespondencí pod hodnotu wT, je nalezen nový klíčový snímek. Počet inliers je vracen z metody `matchCnt()`. Postup je znázorněn algoritmem 6.1.

Vstup: Videosoubor

1. Nalezni klíčové body v každém snímku.
2. První snímek je automaticky klíčový.
3. Dle počtu inliers s bezprostředně následujícím snímkem urči hranici T.
4. Pro každý následující snímek až po poslední proveď:
  4.1 Najdi počet inliers mezi posledním klíčovým snímkem a aktuálním snímkem.
  4.2 Pokud je počet menší než wT, snímek je klíčový. Pokračuj na krok 3.
  4.3 Pokud je počet větší než wT, pokračuj na krok 4.
5. Poslední snímek je také automaticky klíčovým.

Výstup: Množina klíčových snímků.

Algoritmus 6.1: Postup pro získání klíčových snímků z videosekvence.

6.3 Třída pro skládání obrázků

Třída `StitcherImg` je srdcem navrhovaného nástroje a odehrává se v ní celý proces skládání snímků do panoramatu. Vstupem jsou obrázky z disku, nebo klíčové snímky videa. Výstupem je neořezaný panoramatický obrázek. Třídu rozdělíme na dvě části dle návrhu v kapitole 4.4.

Registrací obrazových dat se zabývají následující tři metody. Hledání klíčových bodů na obrázcích v maximálním rozlišení 0,7 Mpx probíhá v metodě `findFeatures()`. Nalezení korespondencí mezi snímky a následné případné vyřazení korespondencí, které neodpovídají verifikaci z kapitoly 5.2.3 je provedeno v `matchFeatures()`. Registraci zakončuje metoda `estimateCamParam()`, ve které probíhá odhad rotačních matic obrázků, ohniskové vzdálenosti a korekce zvlnění výsledného panoramatu.

Kompozice probíhá v metodě `composePano()`. Jsou z ní volány instance tříd ze Stitcher modulu `BlocksGainCompensator` a `GraphCutSeamFinder` pro korekci expozice snímků a nalezení přechodů mezi snímky. Závěrečné prolnutí všech snímků má na starost třída `MultiBandBlender`.

Vstup: Množina obrázků

1. Nalezni klíčové body ve všech obázcích.
2. Nalezni 2 nejbližší sousedy za pomoci LSH a vypočítej homografii
   a) mezi všemi obrázky.
   b) pouze mezi následujícími dvěmi obrázky.
3. Vyřaď obrázky, které do panoramatu nepatří.
4. Z homografie zjisti rotační matici a ohniskovou vzdálenost pro každý snímek.
5. Použij bundle adjustment pro zpřesnění rotačních matic.
6. Proveď korekci rotační matice pro odstranění zvlnění výsledku.
7. Promítni fotky na kouli nebo válec.
8. Nalezni nepravidelný přechod, přes který do sebe snímky pasují.
9. Uprav expozici snímků.
10. Prolni fotky metodou Multi-band Blending.

Výstup: Výsledné panorama

Algoritmus 6.2: Jednotlivé kroky vedoucí k výslednému panoramatu.

6.4 Třídy pro hledání shod

Třídy pro samotné hledání korespondencí jsou implementovány v souboru `matchers.cpp`. Původní soubor byl převzat ze Stitcher modulu a dále upravován. Liší se tak od předchozích dvou tříd v tom, že se skládá z několika menších tříd. Kostru jsem ponechal a využívám tam i původní jmenné prostory a některé původní názvy metod.

Předzpracování klíčových bodů je provedeno ve třídě `FeaturesMatcher`. Podle zvoleného způsobu hledání shod (každý snímek s dvěma sousedy nebo každý snímek s každým) je vytvořena matice párů indexů pro příslušné deskriptory. Matice je vytvořena předem, aby samotné hledání mohlo probíhat paralelně. Paralelní cyklus zajišťuje OpenCV funkce `parallel_for()` a následující třídy, respektive metody, jsou volány paralelně přes celou matici párů indexů.

V třídě `CpuMatcher` je implementováno samotné hledání korespondencí. Pro urychlení hledání korespondencí bylo zvoleno použití *Locality-sensitive hashing* z OpenCV rozhraní knihovny FLANN (*Fast Approximate Nearest Neighbor Search*).

Z nalezených korespondencí jsou ve třídě `BestOf2NearestMatcher` určeny inliers a následně zjištěna homografie mezi snímky.

6.5 Nevyřešené problémy

Dále uvedené problémy jsem původně řešil, ovšem zdárně nevyřešil. Chyby jsou nejspíše v samotné knihovně OpenCV. Svou aplikaci jsem zkoušel ladit a chyby odhalit, ale díky zkompilovaným knihovnám jsem se nikdy v ladícím nástroji nedostal až na potřebnou úroveň zanoření.

6.5.1 Grafické rozhraní

Součástí mělo být původně i jednoduché uživatelské rozhraní. Bohužel jsem při implementaci narazil na velký problém. OpenCV metoda pro hledání nejbližších sousedů, `knnmatch()`, by měla pro dvě konkrétní sady deskriptorů vracet stabilně stejné výsledky.

Neočekávané chování nastalo, když byla tato metoda zavolána z jakékoliv třídy, která dědí z třídy `QObject`, což je hlavní Qt třída a musí být použita pro grafické aplikace. Metoda v tomto případě vracela vždy různé výsledky s každým zavoláním. I přes usilovnou snahu se mi nepodařilo přesný problém identifikovat, a proto jsem nakonec grafické rozhraní zavrhl. Aplikace je tedy pouze konzolová.

6.5.2 Zrychlení přes grafickou kartu

Jelikož knihovna OpenCV i Stitcher modul podporují některé výpočty provádět na GPU, zvažoval jsem tento přístup jako alternativu pro celkové zrychlení, hlavně u zpracování videa. Zamýšlena byla možnost výběru, zda spojování proběhne na procesoru nebo na grafické kartě. Musím konstatovat, že při testování se ukázalo, že jen některé metody fungují bez problému.

Například třída detektoru ORB ani nespustila výpočet. Používá totiž matice ukazatelů na různé další funkce, ale při spuštění výpočtu v této matici ukazovala na hodnotu `NULL` a celý program se tak nečekaně ukončil. Kvůli nestabilitě jsem proto toto zajímavé řešení nevyužil.

Kapitola 7

Testování a výsledky nástroje

Všechny testovací snímky byly pořízeny s automatickým nastavením. U fotoaparátu nebyly ručně nastavovány žádné parametry (expoziční čas, ISO citlivost, clona) a byl použit automatický režim bez blesku. Videa byla natáčena na mobilní telefon [1], který zvládá natáčet HD video v rozlišení 1280×720 . Fotografie testovací sady v kapitole 7.2.2 byly převzaty z [18]. Tyto fotografie byly pořízeny digitální zrcadlovkou [2].

Obrázek 7.1: Areál FIT VUT v Brně. Část ořízlého panoramatu složeného z videa.

[1] Samsung S8500 Wave
[2] Nikon D80

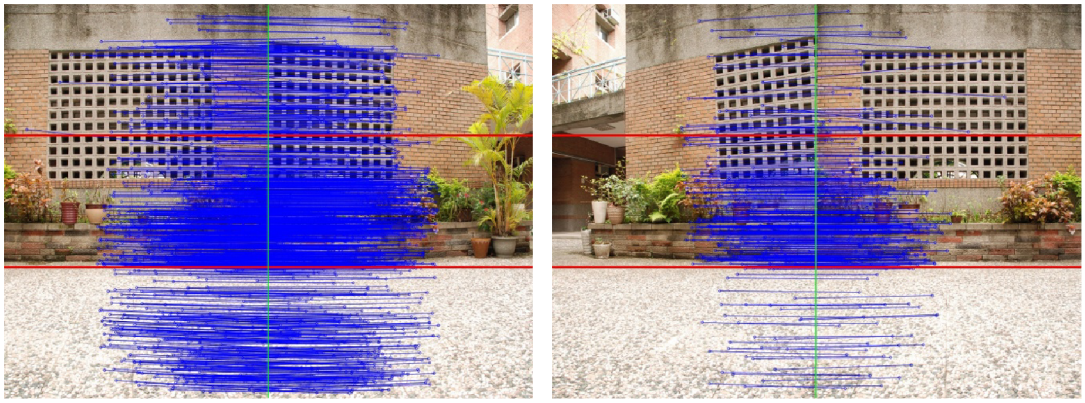
7.1 Nalezené korespondence

V kapitole 5.2.1 byl již představen použitý systém hledání shod. Vyhledané shody jsou filtrovány podle poměru vzdáleností mezi prvním a druhým sousedem vůči prahu.

Na obrázku 7.2a lze vidět hranici $T = 0,45$, která má za následek velký počet shod. Hlavně v prostřední třetině obrázku je příliš mnoho shod, které nemají žádnou další vypovídající hodnotu.

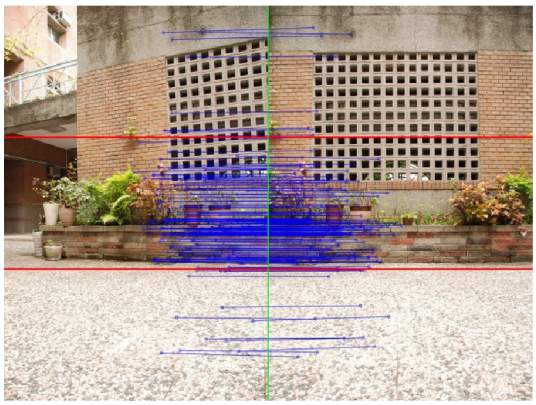
U obrázků 7.2b a 7.2c lze vidět hlavní rozdíl v počtu korespondencí v krajních dvou třetinách obrázku. Jejich kombinací vznikne obrázek 7.2d, který má krajní třetiny z obrázku 7.2b a prostřední třetinu z 7.2c.

Obrázky jsou rozděleny na třetiny pouze vertikálně, což předpokládá spojování v horizontálním směru zleva doprava. Pro spojování ve vertikálním směru by se mohl obrázek rozdělit ještě na tři horizontální třetiny, respektive na devět obdélníků. Podle směru, ve kterém by se obrázky překrývaly, by se zvolily hodnoty prahu T.

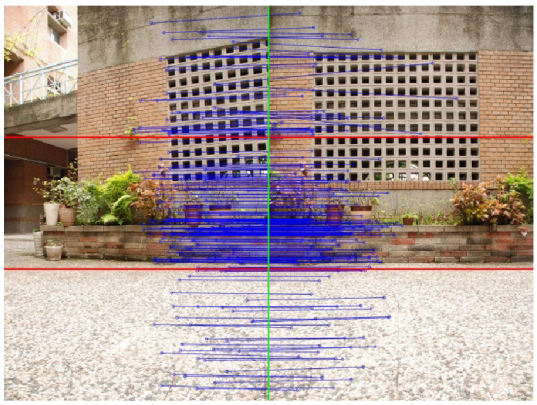


(a) $T = 0,45$ (1103 inliers)

(b) $T = 0,55$ (344 inliers)



(c) $T = 0,65$ (163 inliers)



(d) $T_2 = 0,65$ a $T_{1,3} = 0,55$ (212 inliers)

Obrázek 7.2: Ukázka nalezených shod metodou SURF pro dva sousední obrázky (odděleny zelenou čárou). Červené čáry slouží k označení třetin obrázku. Modrou jsou označeny nalezené korespondence. Hodnota T odpovídá prahu popsaném v kapitole 5.2.1.

7.2 Rychlost skládání

V této kapitole budou změřeny časy, ve kterých navržená aplikace zvládne složit panorama z testovacích sad obrázků a videosekvencí. Testy byly prováděny na počítači s následující konfigurací

- procesor Core2Duo 2 GHz
- 3 GB paměť ram
- operační systém Ubuntu 10.10 32b

Vyvíjeno a testováno bylo pouze na systému Linux. Díky multiplatformnosti použitých knihoven by ale po drobných úpravách měla aplikace fungovat i na systému Windows.

Rychlost byla měřena jako reálný čas, po který jednotlivé funkce běžely. Využity byly OpenCV funkce pro měření času, viz algoritmus 7.1. Tento čas nelze brát jako objektivní měření, je zde uveden pouze pro představu a srovnání trvání výpočtů.

```
double t = (double) getTickCount();
// provádění výpočtů
t = ((double) getTickCount() - t)/getTickFrequency();
```

Algoritmus 7.1: Způsob měření času v sekundách.

7.2.1 Vstup videosekvence

Výsledky pro videosekvenci jako vstup jsou prezentovány v tabulce 7.1 pro dvě vybraná videa areálu FIT VUT v Brně. Je zde zobrazen pouze výběr klíčového snímku, neboť samotné složení klíčových snímků je shodné se skládáním obrázků, což je rozebráno v následující kapitole 7.2.3.

Sekvence **FIT1** je 13 vteřin trvající video o rozlišení 1280x720 a snímkovací frekvencí 25 fps. Video **FIT2** je 19 vteřin dlouhé s rozlišením 720x1280 a snímkovací frekvencí 30 fps. Obě videa jsou snímána pouze horizontální rotací kamery.

Klíčové body jsou nalezeny detektorem ORB a z tabulky lze vyčíst průměrnou dobu detekce bodů v jednom snímku na 0,07s.

Videosekvence	**FIT1**	**FIT2**
Hledání klíč. bodů	23,6693	42,8616
Nalezení klíč. snímků	29,9155	44,308
Počet snímků celkem [ks]	318	583
Počet klíčových snímků [ks]	11	25

Tabulka 7.1: Tabulka rychlosti výběru klíčového snímku. Uvedené hodnoty jsou v sekundách

7.2.2 Testovací sada obrázků

Sada	Info	Obrázky
A1	10 snímků, každý 551x768 (0,42 Mpx)	
A2	10 snímků, každý 1296x1936 (2,5 Mpx)	
A3	10 snímků, každý 1296x1936 (2,5 Mpx)	

Tabulka 7.2: Vstupní sady obrázků. Obrázky byly převzaty od [18].

7.2.3 Složení testovacích sad obrázků

Pro otestování byly vybrány tři sady po deseti obrázcích a časové výstupy programu zapsány do tabulky 7.3. Aplikace hledá klíčové na rozlišení maximálně 0,7 Mpx. Kvůli šetření paměti jsou obrázky pro další zpracování zmenšeny na rozlišení 0,2 Mpx. V původním rozlišení jsou znovu načteny až při samotném skládání.

Obrázky ze sady **A1** jsou v rozlišení 0,45 Mpx. Obrázky z **A2** a **A3** jsou v rozlišení 2,5 Mpx a tím pádem byly zmenšeny na 0,7 Mpx. V první části tabulky, u nalezení klíčových bodů a shod, si lze všimnout zřetelného rychlostního rozdílu mezi detektorem ORB a SURF. Detektor ORB je zde více než 20x rychlejší. Korespondence byly hledány vždy z aktuálního snímku do dvou následujících, jelikož byla sada seřazená, a proto je zbytečné hledat shody se všemi obrázky.

V druhé části tabulky si lze všimnout velice podobných časů, neboť již mají všechny tři sady rozlišení 0,2 Mpx a také stejný počet obrázků. Drobné časové odchylky nastávají u metody bundle adjustment. Jelikož se jedná o iterační metodu, může tuto odchylku způsobit různý počet iterací, nebo je to pouze zkreslení dané způsobem měření.

V poslední, třetí části, jsou časově nejnáročnější úkony. Kompenzace expozice i hledání řezu přechodu se stále ještě provádějí na zmenšeninách a jejich čas je tedy dosti podobný. Kompozice už probíhá se znovu načtenými originálními obrázky a proto je čas závislý hlavně na rozlišení. Zde je potřeba upřesnit, že pojem kompozice zde znamená označení konečného kroku skládání, a ne celou část skládání, jak bylo použito v kapitole 4.4. Zahrnuje závěrečné spojení obrazů dle nalezených řezů a multipásmové míchání obrázků.

Položka „Celkový čas“ značí dobu od spuštění aplikace až do zápisu výsledného panoramatu na disk a ukončení programu. Tento čas tak nelze brát jako pouhý součet všech předchozích. Časy uvedené v tabulce zobrazují jen měřené úseky hlavních částí. Neobsahují ovšem různé mezivýpočty, které jsou zahrnuty právě až v poslední položce tabulky.

Detektor	**ORB**			**SURF**		
Sady obrázků	**A1**	**A2**	**A3**	**A1**	**A2**	**A3**
Nalezení klíč. bodů	0,4898	0,9261	0,9009	10,7335	20,7976	24,1280
Nalezení shod	2,8080	3,4782	2,9743	3,5579	8,2390	9,3115
Odhad rotace	0,0003	0,0003	0,0003	0,0003	0,0003	0,0003
Bundle adjustment	0,6764	0,5899	1,5120	0,3138	0,6360	1,2728
Korekce zvlnění	0,0001	0,0002	0,0001	0,0001	0,0001	0,0001
Warping zmenšenin	1,2639	1,2962	1,2899	1,2898	1,3247	1,2853
Kompenz. expozice	9,4079	10,8069	10,3514	9,5657	10,7247	10,3822
Hledání přechodu	6,5044	8,0192	7,1000	7,8451	6,5872	6,8896
Kompozice	4,5546	27,2985	26,9122	4,5900	27,5424	26,7181
Celkový čas	26,6728	56,8445	55,0675	38,8631	80,3535	84,0808

Tabulka 7.3: Tabulka rychlostí jednotlivých částí. Uvedené hodnoty jsou v sekundách

(a) Sada A1, 2715 × 743 (2,0 MPx)

(b) Sada A2, 6703 × 1834 (12,3 MPx)

(c) Sada A3, 5755 × 1872 (10,8 MPx)

Obrázek 7.3: Výsledná panoramata z testovacích sad. Použit byl detektor ORB.

Kapitola 8

Závěr

Práce se zabývala skládáním obrázků a videosekvencí. Cílem bylo nastudovat, navrhnout a implementovat aplikaci pro tvorbu panoramat s ohledem na potřebný výpočetní čas a kvalitu výsledku.

Po nastudování potřebných principů byly vybrány konkrétní techniky a návrh byl rozdělen na dvě hlavní části – registraci obrazových dat a kompozici. Zaměřil jsem se na rychlé vyhodnocení první části. Celá registrace obrazových dat při použití ORB detektoru trvá pouze $10 - 15\%$ a s detektorem SURF $40 - 45\%$ celkového času tvorby panoramatu. Dále byla navržena optimalizace, aby nalezené korespondence mezi snímky byly lépe rozloženy v obrázku při minimálním zvýšení výpočetní doby nalezení shod. Nejvíce času spotřebuje kompozice, která zde ale byla použita pouze pro zkvalitnění výstupu bez ohledu na výkon.

Možnosti pokračování práce vidím v dvou hlavních směrech. Bylo by dobré dodělat nadstavbu jednoduchého grafického uživatelského rozhraní, které se mi z důvodů uvedených v 6.5.1 nepodařilo vytvořit. Přece jenom, dnes už konzolové aplikace nejsou tak populární a dobrým konzolovým aplikacím jsou zpětně dodělávány grafická rozhraní.

Druhým směrem je rozšíření po stránce výkonu a kvality. Pro tvorbu panoramat z videosekvencí by bylo velice žádané využít co nejvíce informací, které nám video poskytuje oproti fotografiím. Využitím principů *feature tracking*, místo *feature matching*, by se dosáhlo lepších kvalitativních výsledků. Přestože navržený nástroj s testovacími videosekvencemi fungoval správně a relativně rychle, nevyužívá informace o pohybu kamery, které by mohli dosti podstatně zpřesnit samotný proces skládání.

Literatura

[1] BAY, H.; ESS, A.; TUYTELAARS, T.; aj.: Speeded-Up Robust Features (SURF) *Comput. Vis. Image Underst.*, ročník 110, č. 3, Červen 2008: s. 346–359, ISSN 1077-3142.

[2] BEBIS, G.: Geometric Camera Parameters. [online], [cit 11.5.2012]. Dostupné na WWW: http://www.cse.unr.edu/~bebis/CS791E/Notes/CameraParam%eters.pdf

[3] BENEDA, M.: Homografie a epipolární geometrie [online]. 31.10.2010. Dostupné na WWW: trilobit.fai.utb.cz/homografie-a-epipolarni-geometrie

[4] BROWN, M.; LOWE, D. G.: Automatic Panoramic Image Stitching using Invariant Features *Int. J. Comput. Vision*, ročník 74, č. 1, Srpen 2007: s. 59–73, ISSN 0920-5691.

[5] CALONDER, M.; LEPETIT, V.; ÖZUYSAL, M.; aj.: BRIEF: Computing a Local Binary Descriptor Very Fast *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011, ISSN 0162-8828.

[6] EVANS, C.: Notes on the OpenSURF Library. Technická Zpráva CSTR-09-001, University of Bristol, January 2009. Dostupné na WWW: www.cs.bris.ac.uk/Publications/Papers/2000970.pdf

[7] FISCHLER, M. A.; BOLLES, R. C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography *Commun. ACM*, ročník 24, č. 6, Červen 1981: s. 381–395, ISSN 0001-0782.

[8] HARTLEY, R.; ZISSERMAN, A.: *Multiple view geometry in computer vision.* Cambridge University Press, druhé vydání, 2003, ISBN 05-215-4051-8.

[9] KIEN, D. T.: A review of 3D reconstruction from video sequences. Technická zpráva, University of Amsterdam, 2005.

[10] KRŠEK, P.: Základy počítačové grafiky: Studijní opora předmětu IZG.

[11] KWATRA, V.; SCHÖDL, A.; ESSA, I.; aj.: Graphcut Textures: Image and Video Synthesis Using Graph Cuts *ACM Transactions on Graphics, SIGGRAPH 2003*, ročník 22, č. 3, July 2003: s. 277–286.

[12] LOWE, D. G.: Distinctive Image Features from Scale-Invariant Keypoints *International Journal of Computer Vision*, ročník 60, 2004: s. 91–110, ISSN 0920-5691.

[13] MIKOLAJCZYK, K.; SCHMID, C.: A Performance Evaluation of Local Descriptors *IEEE Trans. Pattern Anal. Mach. Intell.*, ročník 27, č. 10, Říjen 2005: s. 1615–1630, ISSN 0162-8828.

[14] MILGRAM, D. L.: Computer Methods for Creating Photomosaics *IEEE Trans. Comput.*, ročník 24, č. 11, Listopad 1975: s. 1113–1119, ISSN 0018-9340.

[15] nVidia: What is GPU Computing? [online], [cit 10.5.2012].
Dostupné na WWW:
http://www.nvidia.com/object/what-is-gpu-computing.htm%l

[16] RUBLEE, E.; RABAUD, V.; KONOLIGE, K.; aj.: ORB: An Efficient Alternative to SIFT or SURF. V *International Conference on Computer Vision*, Barcelona, 11/2011 2011.

[17] SCHMID, C.; MOHR, R.; BAUCKHAGE, C.: Evaluation of Interest Point Detectors *Int. J. Comput. Vision*, ročník 37, č. 2, Červen 2000: s. 151–172, ISSN 0920-5691.

[18] SU, Y.-F.: Image stitching. [online], [cit 12.5.2012].
Dostupné na WWW:
http://mpac.ee.ntu.edu.tw/~sutony/vfx_stitching/pano.h%tm

[19] SZELISKI, R.: Image alignment and stitching: a tutorial *Found. Trends. Comput. Graph. Vis.*, ročník 2, č. 1, Leden 2006: s. 1–104, ISSN 1572-2740.

[20] SZELISKI, R.: *Computer Vision: Algorithms and Applications*. Springer, 2010, ISBN 978-184-8829-34.

[21] TRUCCO, E.; VERRI, A.: *Introductory techniques for 3-D computer vision*. Prentice Hall, 1998, ISBN 978-0132611084.

[22] TUYTELAARS, T.; MIKOLAJCZYK, K.: Local invariant feature detectors: a survey *Found. Trends. Comput. Graph. Vis.*, ročník 3, č. 3, Červenec 2008: s. 177–280, ISSN 1572-2740.

[23] UYTTENDAELE, M.; EDEN, A.; SZELISKI, R.: Eliminating Ghosting and Exposure Artifacts in Image Mosaics. V *CVPR (2)*, IEEE Computer Society, 2001, s. 509–516, ISBN 0-7695-1272-0.

Příloha A

Manuál

Program je konzolová aplikace, která skládá množinu obrázků či jeden videosoubor do výsledného panoramatu. Aplikace funguje zcela automaticky, kdy se obrázky či video předají jako parametry. Dále lze nepovinnými parametry ovlivňovat kvalitu a rychlost výsledku. Výsledné panorama je uloženo do aktuální složky jako **pano.jpg**.

Spuštění aplikace:
`stitcher_cli [parametry...] [obrazky | -v video]`

Parametry aplikace:

`-h, --help`	Zobrazí nápovědu.
`-f, --finder [surf|orb]`	Detektor klíčových bodů. Výchozí: `ORB`.
`-w, --warper [koule|valec]`	Povrch, na který promítnout. Výchozí: `koule`.
`-pt, --pair_type [kazdy|soused]`	Systém hledání shod mezi snímky. Výchozí: `kazdy`.
`-v, --video`	Bude načteno video místo obrázků.
`-kfc, --keyframe_coef cislo`	Práh určení nového klíčováho snímku. Výchozí: `0.15`.

Příklad výstupu v konzoli:

```
dkrym@ntb$ ./stitcher_cli -pt soused o1.jpg o2.jpg o3.jpg
Hledam klicove body (3 obr.):
[========================] 100%     > Nalezeno za: 0.155174 s
Hledam korespondence mezi snimky:
[========================] 100%     > Nalezeno za: 0.583104 s
Parametry kamery:
  Odhad rotaci kamery...        0.000113355 s
  Bundle adjustment...          0.0503999 s
  Korekce zvlneni...            0.00051006 s
  Warping zmensenin...          0.396123 s
Optimalizace:
  Kompenzace expozice...        0.339479 s
  Hledani prechodu...           0.867015 s
Kompozice panoramatu:
[========================] 100%     > Slozeno za: 2.83802 s
Doba celkem: 5.84655 s
```

Příloha B

Obsah DVD

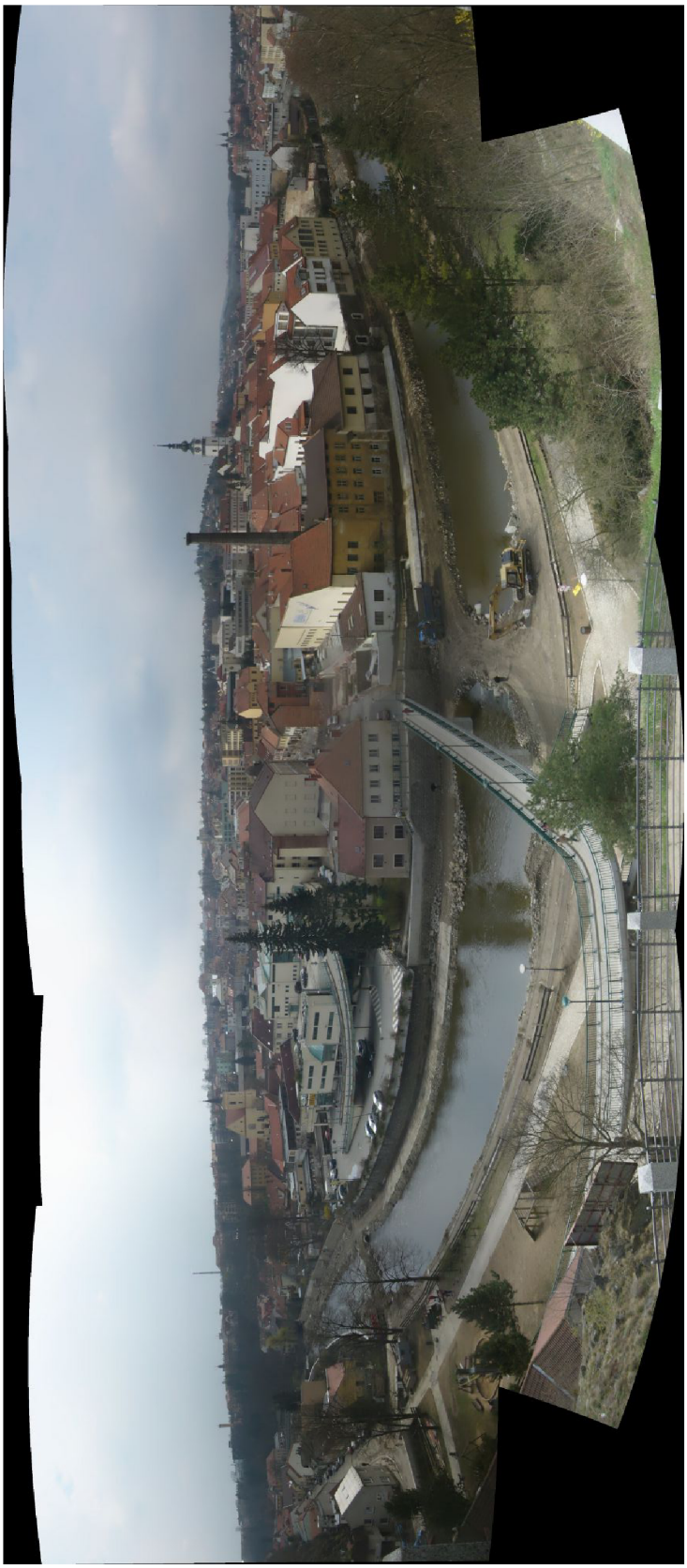
Na přiloženém DVD se nachází zdrojové kódy aplikace, včetně knihovny OpenCV ve verzi 2.3.1. Dále je tam obsažena tato práce, včetně jejího zdrojového kódu. Jsou obsažena testovací data, na kterých lze vidět funkci. Součástí jsou i složené panoramata z testovacích sad kapitoly 7.2.2.

- **/doc** Tato práce včetně zdrojových kódů.
- **/install** Složka s knihovnou OpenCV.
- **/program** Zdrojové kódy aplikace. Obsahuje Makefile.
  - **/test_data** Obrázky a videa. Pro každou sadu je uložen i výsledek.
- **/testovaci_sady_vysledky** Výsledná panoramata z testovacích sad.

Příloha C

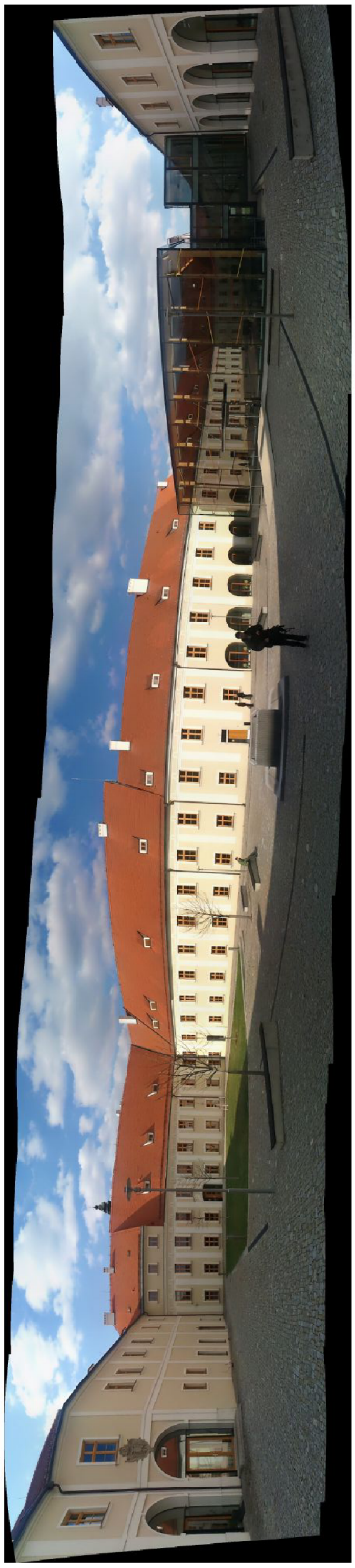
Ukázky výsledných panoramat

Ukázky výsledných panoramat, které jsou výstupem navrženého programu. Obrázky jsou schválně neořezány, aby byly přibližně vidět okraje zdrojových obrázků a ořezáním na obdélník bychom mohli přijít i o velké části obrázku, což by nebylo tak demonstrativní. Pro přehlednost se každé panorama nachází na nové stránce.



Obrázek C.1: Zdroj: 11 snímků. Použitý detektor SURF. Korespondence: každý s každým.

Obrázek C.2: Zdroj: 15 snímků. Použitý detektor SURF. Korespondence: každý s každým.



(a) 26 klíčových snímků (z 583)

(b) 11 klíčových snímků (z 318)

Obrázek C.3: Zdroj: video. Použitý detektor SURF. Korespondence: dva následující. Obrázky pocházejí každý z jiného videa.