

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

PANORAMATICKÉ SNÍMKY AUTOMATICKY

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

VLADIMÍR MOTÁČEK

BRNO 2008



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

PANORAMATICKÉ SNÍMKY AUTOMATICKY

AUTOMATIC PANORAMIC IMAGE MOSAICING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VEDOUCÍ PRÁCE
SUPERVISOR

VLADIMÍR MOTÁČEK

Ing. VÍTĚZSLAV BERAN

BRNO 2008

Abstrakt

Tato práce se zabývá automatickým skládáním panoramatických snímků. Snímky nemusí být pořízeny pouze v horizontálním směru ani nemusí být seřazeny. Využívá základních technik jako detekce bodů v obraze Harrisovým rohovým detektorem, nalezení korespondencí těchto bodů pomocí korelace a počítání homografie metodou RANSAC. Obrázky jsou mapovány do roviny dané referenčním snímkem.

Klíčová slova

panoramatický snímek, skládání panoramat, počítačové vidění (CV), RANSAC, detekce rohů, homografie, význačný bod, detekce význačných bodů, hledání shod význačných bodů

Abstract

This paper concerns automatic panoramic image mosaicing. Images can be taken in any direction and in any order. This work uses basic technics such as Harris corner detection, correlation of image patches for finding correspondences and computing homography using RANSAC. The images are mapped to the reference image plane.

Keywords

panoramic image, image stitching, image mosaicing, computer vision (CV), RANSAC, corner detection, homography, feature point, feature point extraction, feature point matching

Citace

Vladimír Motáček: Panoramatické snímky automaticky, bakalářská práce, Brno, FIT VUT v Brně, 2008

Panoramatické snímky automaticky

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vítězslava Berana

.....
Vladimír Motáček
5. května 2008

Poděkování

Rád bych poděkoval svému vedoucímu bakalářské práce Ing. Vítězslavu Beranovi za cenné rady, vstřícnost, drahocenný čas a poutavý výklad

© Vladimír Motáček, 2008.

Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1	Úvod	2
2	Teorie	4
2.1	Detekce hran	4
2.2	Detekce důležitých bodů v obraze	4
2.3	Hledání korespondencí	7
2.4	Geometrické transformace	8
2.5	Homografie	9
2.6	RANSAC	11
3	Automatický generátor panoramat	13
3.1	Návrh řešení	13
3.2	Detekce důležitých bodů v obraze	15
3.3	Určení korespondencí mezi snímky	16
3.4	Homografie mezi snímky	17
3.5	Vzájemná pozice snímků v panoramatu	19
3.6	Korekce panoramatu	22
4	Výsledky	23
5	Závěr	27

Kapitola 1

Úvod

Panorama je obecně širší pohled na fyzický prostor. Klasický fotoaparát zachycuje úhel pohledu zhruba $50 \times 35^\circ$. Lidský úhel vnímání je asi $200 \times 135^\circ$. Naproti tomu panoramatický snímek nám dokáže poskytnout daleko více informací [4] a to až v úhlu $360 \times 180^\circ$.

Panoramatický snímek můžeme vytvořit [8] čtyřmi způsoby:

- Skládáním jednotlivých snímků
- Pomocí zrcadel
- Použitím širokouhlého formátu
- Skenováním otočnou kamerou

První metoda skládá posloupnost klasických snímků s určitým přesahem. Je to tradiční a nízkonákladová metoda. To je však vyváženo vyšší výpočetní náročností. Druhá metoda využívá přímé nebo nepřímé odrazy na zrcadle. V principu rozdělujeme tyto techniky na dvě skupiny. První využívá jedno konvexní zrcadlo (parabolické, kulové, kónické, hyperbolické). Druhá dvě všesměrové zrcadla (kónické a/nebo konkávní a parabolicko-konkávní). Výhoda tohoto druhu metod je rychlé pořízení celého panoramatu. Nevýhodou je menší rozlišení. Třetí metoda využívá kameru s dlouhým a širokým formátem. Komerčním standardem se stal formát 6×17 cm. K zobrazení většího zorného úhlu musíme pořídit více fotografií. Poslední metoda je určena převážně na plná panoramata 360° . Zapotřebí je speciální panoramatická kamera, která v jednom skenování zachytí celé okolí. Kamera pracuje otáčením kolem své vertikální osy.

Tato práce se bude zabývat první metodou – skládáním panoramat z jednotlivých snímků. Výhodou této metody je její obecnost. Nejsme omezeni pouze horizontálním pohybem, ale můžeme skládat fotografie ve všech směrech. Detailnějším nafocením objektu více fotografiemi dostáváme větší rozlišení.

Pro člověka je vcelku nesnadným úkolem ručně si spojit své fotografie do panoramatu. Když pomíneme odlišné úrovně jasu, spojení pouze dvou fotografií zachycující blízké objekty může být složité kvůli značnému perspektivnímu zkreslení, zkreslení fotoaparátu apod. Člověk by těžko hledal správnou rotaci, zkosení a případně změnu měřítka tak, aby obrázky „seděly“ na sobě. Fotografie můžeme naštěstí skládat automaticky.

Tvorbě panoramat ze sekvence fotografií se věnuje celá řada aplikací jak komerčních, tak volně dostupných. Některé programy vyžadují pro svou správnou činnost inicializaci. Například zadání pořadí snímků v sekvenci, hrubé složení snímků v editoru nebo určení kontrolních bodů ve snímcích. Cílem bude omezit uživatelův vstup na minimum.

Metody na automatické skládání snímků lze ze široka rozdělit [3] do dvou skupin. Dělíme je na přímé metody a metody založené na význačných bodech. Přímé metody mají za výhodu, že pracují ze všemi dostupnými daty a proto nám poskytují velmi přesné výsledky. Na druhou stranu potřebují počáteční inicializaci. Jejich využití je při skládání snímků s velkým přesahem a malou vzájemnou rotací či malým posunutím. Metody pracující na rozpoznávání význačných bodů nepotřebují žádný proces předzpracování a poradí si i s menší překrývající se oblastí. S jejich pomocí docílíme přesných výsledků avšak za cenu větší výpočetní složitosti. Skládáním panoramatických snímků se zabývá jedno odvětví výzkumu v počítačovém vidění (computer vision).

Tato práce se skládá včetně úvodu z pěti kapitol. V druhé kapitole najdete přehled technik uplatňovaných ve skládání snímků. Jednotlivé postupy jsou zde obecně podrobněji rozepsané a na začátku každého pododdílu se čtenář seznámí s jejich konkrétní aplikací právě vzhledem k dané problematice. Vše je celkově shrnuto na začátku třetí kapitoly. Tato kapitola taktéž obsahuje postup vlastního řešení a použité techniky. Dosažené výsledky a ukázky řešení jsou prezentovány ve čtvrté kapitole. Poslední kapitolou je závěr celé práce.

Program byl napsán v jazyce C/C++ s využitím knihovny pro počítačové vidění OpenCV. Dílo je vysázeno typografickým systémem \LaTeX .

Kapitola 2

Teorie

Tato kapitola shrnuje základní uplatňované techniky ve skládání snímků. Jednotlivé postupy jsou zde obecně podrobněji rozepsané. Začátek každého pododdílu se věnuje konkrétní aplikaci právě vzhledem k dané problematice.

2.1 Detekce hran

Základním kamenem rozpoznávání dvou fotografií je určení význačných bodů. Detekce těchto bodů úzce souvisí s detekcí hran. Nastíňme alespoň základní princip rozpoznávání oblastí [6] v obrázcích.

Ukázalo se, že člověkem jsou intenzivně vnímány místa, kde se náhle mění hodnota jasu. Takováto místa nazýváme hrany. Hrana v obraze je dána vlastnostmi obrazového elementu a jeho okolí. Je určena tím, jak se náhle mění hodnota obrazové funkce $f(x, y)$. Tyto změny můžeme matematicky popsat parciálními derivacemi. Změnu funkce udává její gradient. Gradient určuje směr největšího růstu funkce a strmost tohoto růstu (velikost, modul gradientu). Pixely s velkým modulem gradientu nazýváme hrany. Hrany můžeme popsat gradientními operátory. Často se používá všesměrový lineární Laplaceův operátor–Laplacián, který vychází z druhých parciálních derivací. Pro monotónně rostoucí jasovou funkci v příslušném okolí je Laplacián nulový tam, kde je velikost gradientu maximální.

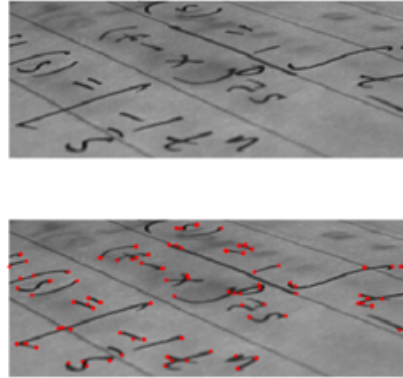
K detekci hran nám poslouží konvoluční masky aproximující derivace obrazové funkce. Operátorů, lišící se svými vlastnostmi, je celá řada. K nejznámějším patří Robertsův nebo Laplaceův. Dalšími operátory jako Prewittové, Sobelův, Kirschův a Robinsonův aproximujeme první derivaci.

2.2 Detekce důležitých bodů v obraze

Při tvorbě panoramatických fotografií musíme určit vzájemně si odpovídající body v jednotlivých snímcích. Z pochopitelných důvodů nelze hledat odpovídající si pixely. Množinu bodů ve snímku můžeme získat za pomoci nějakého níže uvedeného rohového detektoru.

Detekci významných bodů [11] často používáme na sledování pohybu bodů (tracking), skládání panoramatických obrázků, detekci pohybu, 3D modelování a rozpoznávání objektů. Ve snímku mají tyto body dobře definovanou pozici a můžeme je robustně detekovat. Mezi významné body patří například rohy. Roh definujeme jako průsečík dvou hran. V tomto bodě mají obrazové funkce velký gradient ve směrech x a y . Výhodnější je hledat rohy právě za pomoci gradientu, než z obrazu hran. K dalším význačným bodům, které lze

spolehlivě detekovat, můžeme zařadit také izolovaný bod s lokálním minimem nebo maximum, konec čáry, popřípadě bod na křivce v místě největšího zakřivení. Různé význačné body ilustruje obrázek 2.1.



Obrázek 2.1: Detekované rohy

O rozpoznávání těchto bodů se starají rohové detektory. I když už víme, že to není úplně přesné označení. V praxi to znamená, že detekci rohů myslíme detekci význačných bodů, než jen samotných rohů. Kvalita rohových detektorů se posuzuje na schopnosti určit ty samé význačné body ve snímcích, které jsou si podobné, ale ne identické. Například mají rozdílnou světlost, jsou posunuté, otočené nebo jinak transformované. Kvalitu také můžeme posuzovat podle efektivity výčtu, odolnosti vůči šumu apod. Jednoduchý přístup k detekci rohů je prostá korelace, ale ta je velmi výpočetně náročná a neoptimální. Další možností je využít nějaký rohový detektor. Jejich výkonnost je samozřejmě daná způsobem detekce hran, apod. Rohových detektorů je celá řada. Mezi nejznámější patří asi Harrisův detektor.

Rohové detektory nám dají pouze informaci o souřadnicích význačných bodů (rohů). Zvláštní skupinou detektorů tvoří detektory, které nám kromě zjištění souřadnic bodu tento bod zároveň popíše nějakým deskriptorem. To může být výhodné zejména pro další zpracování z pohledu hledání korespondencí. Tyto algoritmy jsou už většinou robustní a mají lepší využití v praxi. Mezi takové patří například SIFT, SURF a další.

2.2.1 Moravcův detektor rohů

Tento algoritmus je jeden z nejjednodušších a nejčastějších metod na detekci rohů. Moravcův operátor detekuje významné body jako body, kde je velká změna rozptylů jasů ve všech směrech. Okno o určitých rozměrech, například 3×3 se posouvá po pixelu v osmi základních směrech. Rozptyl pro daný posun získáme jako sumu čtverců rozdílů odpovídajících si pixelů ve dvou oknech. Rozptyl intenzit v centrálním bodě je určen jako minimum z osmi základních směrů.

Jeden ze základních problémů Moravcova detektoru je, že není izotropní. Jestliže není roh ve směru souseda, nebude detekován.

2.2.2 Detektor SUSAN

SUSAN je zkratka pro Smallest Univalve Segment Assimilating Nucleus. K výpočtu jasu dochází v kruhovém okolí (nucleon) kolem zkoumaného bodu a porovnáním se zkoumaným bodem porovnávací funkcí. Postupně se generuje kruhová maska kolem daného bodu a porovnávají se jasové hodnoty uvnitř oblasti. Algoritmus se opakuje pro všechny body v obrázku. Oblasti s podobnou jasovou hodnotou nazýváme USAN.

2.2.3 Detekce pomocí LoG, DoG, DoH

Význam jednotlivých zkratk je LoG (Laplacian of Gaussian), DoG (Difference of Gaussian), DoH (Determinan of Hessian). Tyto techniky se využívají k detekci bodů mimo jiné v takzvaném SIFT [3] detektoru. Algoritmy jsou většinou velmi robustní a dávají dobré výsledky.

2.2.4 Harrisův a Stephensův operátor

Harrisův operátor je vylepšením Moravcova rohového detektoru. Na rozdíl od něj uvažuje směr posunutí snímku. Pracuje na podobném principu posouvání snímků ve všech základních směrech a počítá gradienty v jednotlivých bodech snímku. Základem je výpočet rozdílů čtverců jasu (SSD) ve všech bodech snímku. Výrazné omezení je, že porovnávané snímky musí mít stejný jas a kontrast. Pro každý bod snímku se počítá autokorelační matice druhých parciálních derivací.

$$A = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix}$$

Zda je daný bod významný či není, určíme z vlastních čísel po výpočtu matice. Pro vlastní čísla platí [5]

1. Jestliže $\lambda_1 \approx 0 \wedge \lambda_2 \approx 0$ nebyl nalezen žádný význačný bod
2. Jestliže $\lambda_1 \approx 0 \wedge \lambda_2$ je velké kladné číslo, byla nalezena hrana
3. Jestliže λ_1 i λ_2 jsou velká kladná čísla, byl nalezen roh

Harris a Stephens poznamenali, že výpočet vlastních čísel je výpočetně náročný, kvůli počítání odmocniny. Místo toho navrhli funkci M_c , kde κ je nastavitelný parametr.

$$M_c = \lambda_1 \lambda_2 - \kappa(\lambda_1 + \lambda_2)^2 = \det(A) - \kappa \text{Trace}^2(A)$$

Hodnota κ byla určena experimentálně a literatura uvádí přijatelné rozmezí od 0.04 do 0.15. Před samotnou Harrisovou detekcí je vhodné odstranit šum pomocí Gaussova filtru.

2.2.5 SIFT

SIFT je patentovaný algoritmus jehož název je odvozený z anglického Scale Invariant Feature Transform. Jak napovídá název, algoritmus si dokáže poradit se změnou měřítka. Mezi jeho další přednosti patří nezávislost na rotaci, osvětlení a šumu. Rozpoznávání klíčových bodů se děje ve čtyřech krocích [14] a každý bod je poté popsán deskriptorem (128-rozměrný vektor celých kladných čísel).

Algoritmus je velice robustní a je v něm aplikovaná myšlenka, kterou rozvíjí další navazující detektory. Ty se snaží hlavně algoritmus zrychlit zanedbáním méně podstatných vlivů,

avšak ne na úkor kvality a robustnosti. Konkrétně byly snahy některé výpočty nepočítat přesně, ale pouze aproximovat s vysokou pravděpodobností a také zjednodušit deskriptor na méně údajů a tím zefektivnit zjišťování jejich korespondencí. To se povedlo, podle tvrzení autorů, detektoru SURF (Speeded Up Robust Features).

2.3 Hledání korespondencí

Hledání korespondencí je další významný krok celé procedury spojování fotografií. Po určení rohů v jednotlivých snímcích musíme nějak zjistit, jestli si tyto rohy náhodou neodpovídají a tudíž je možnost, že se jedná o jedny a ty samé význačné body ve dvou fotografiích. Hledání korespondencí záleží na způsobu detekce rohů. Pokud jsme použili nějaký pokročilý detektor, který nám zároveň vygeneroval deskriptor rohu, pak navzájem porovnáváme tyto deskriptory. Jestliže máme k dispozici pouze souřadnice rohu, například z Harrisova detektoru, určujeme podobnosti pomocí okolí tohoto rohu. Možností jak to udělat, je celá řada. Nejjednodušším způsobem by mohlo být z tohoto okolí vypočítat histogramy a ty potom porovnat. Porovnání můžeme mimo jiné provést i korelací.

Korelace slouží [15] k vzájemnému porovnávání dvou navzorkovaných signálů. Jako výsledek získáme posloupnost čísel udávající podobnost signálů, jejich posunutí, periodu apod. Zvláštní případ je autokorelace, kdy se porovnávají dva shodné signály. Ta je z našeho pohledu zajímavější. Algoritmus je založen na vzájemném násobení vzorků signálu a počítání sumy z těchto vzorku.

$$R(m) = \sum_{n=0}^{N-m-1} s(n)s(n+m)$$

Normalized cross-correlation má v určitých případech lepší uplatnění než klasická korelace. Nevýhoda standardní autokorelační funkce je postupné zkracování oblasti, ze které autokorelační koeficienty počítáme. Je proto výhodnější přejít na *cross-korelační* funkci. Začátek srovnávání označme zr .

$$CCF(m) = \sum_{n=zr}^{zr+N-1} s(n)s(n+m)$$

Při výpočtu CCF můžeme narazit na problém příliš velké energie jednoho ze signálů, která přebije podobnost dvou rámců. Rozdílnost energií originálního a posunutého rámce řešíme pomocí normalizace.

Tím dostáváme *normalizovanou cross-korelaci*:

$$NCCF(m) = \frac{\sum_{n=zr}^{zr+N-1} s(n)s(n+m)}{\sqrt{E_1 E_2}}$$

kde E_1 a E_2 jsou energie originálního a posunutého rámce:

$$E_1 = \sum_{n=zr}^{zr+N-1} s^2(n) E_2 = \sum_{n=zr}^{zr+N-1} s^2(n-m)$$

Template matching je technika hledání vzoru v obraze. Základní myšlenka spočívá v porovnávání všech pixelů v obrázku se vzorem. Vzor je malý výřez z obrázku, který po něm posouváme pixel po pixelu. Pro středový pixel výřezu dostáváme koeficient podobnosti. Způsob jakým se mezi sebou výřez a obrázek porovnává, může být libovolný.

2.4 Geometrické transformace

Bez geometrických transformací by se počítačová grafika nemohla obejít. Konkrétně pro tvorbu panoramat je pochopení této problematiky naprosto zásadní. Transformace nám dovolují upravovat obraz a dál s ním pracovat. Například skládat panoramata nebo odstraňovat geometrické zkreslení apod. Geometrické transformace mapují bod v obraze (x_1, y_1) na novou pozici (x_2, y_2) . To vyjádříme jako:

$$\begin{vmatrix} x_2 \\ y_2 \end{vmatrix} = A \begin{vmatrix} x_1 \\ y_1 \end{vmatrix} + B$$

Transformace můžeme rozlišit na:

- lineární (posuv, otočení, změna měřítka, zkosení a jejich kombinace)
- nelineární (warping - obraz se může různě jinak deformovat)
- projekce (zobrazování 3D objektů na 2D výstup)

Někdy se setkáváme s pojmem afinní transformace. Tyto transformace zachovávají rovnoběžnost, tj. vzájemně rovnoběžné čáry budou rovnoběžné i po transformaci.

Při počítání transformací se často využívá homogenních souřadnic. Bod v homogenních souřadnicích má ve vektorovém prostoru o jednu dimenzi navíc. Důvodem pro zavedení bylo sjednocení operací posunu, změny měřítka, zkosení a rotace. Bodu $[x, y]^T$ odpovídá v homogenních souřadnicích bod $[x, y, w]^T$. Třetí dimenzí w můžeme nazvat váhou. Tímto parametrem totiž ostatní dimenze dělíme. Pro homogenní souřadnice je $w = 1$, tedy bod zapíšeme ve tvar $[x, y, 1]^T$.

Posunutí přemístí každý bod ve směru osy x a y o X_T a Y_T . Nové souřadnice dostaneme jako:

$$\begin{aligned} X_2 &= X_1 + X_T \\ Y_2 &= Y_1 + Y_T \end{aligned}$$

Transformační matice:

$$A_T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ X_T & Y_T & 1 \end{bmatrix}$$

Otočení provádí rotaci *kolem souřadnicového systému* o daný orientovaný úhel α . Nové souřadnice dostaneme jako:

$$\begin{aligned} X_2 &= X_1 \cos \alpha - X_T \sin \alpha \\ Y_2 &= X_1 \sin \alpha + X_T \cos \alpha \end{aligned}$$

Transformační matice:

$$A_R = \begin{bmatrix} \cos \alpha & \sin \alpha & 0 \\ -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Změna měřítka se docílí vynásobením složek X a Y faktorem S . Pro S platí:

$S > 1$ zvětšení
 $0 < S < 1$ zmenšení
 $S < 0$ převrácení

Nové souřadnice dostaneme jako:

$$\begin{aligned} X_2 &= X_1 S_X \\ Y_2 &= Y_1 S_Y \end{aligned}$$

Transformační matice:

$$A_S = \begin{bmatrix} S_X & 0 & 0 \\ 0 & S_Y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Zkosení obraz zkosí ve směrech os x a y . Míru zkosení udává S_{HX} a S_{HY} . Nové souřadnice dostaneme jako:

$$\begin{aligned} X_2 &= X_1 + Y_1 S_{HX} \\ Y_2 &= Y_1 + X_1 S_{HY} \end{aligned}$$

Transformační matice:

$$A_{Sh} = \begin{bmatrix} 1 & S_{HY} & 0 \\ S_{HX} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

V praxi je většinou potřeba na objekt aplikovat více transformací. Výslednou transformaci lze vyjádřit jednou transformační maticí, která vznikne postupným skládáním jednotlivých transformací. Skládáním rozumíme násobení transformačních matic, přičemž záleží na pořadí aplikování transformací, tedy záleží na pořadí násobení matic.

2.5 Homografie

Homografie je definována [12] jako relace mezi dvěma útvary, kde každému bodu jednoho přísluší určitý bod druhého a naopak. Homografie je v počítačovém vidění chápána jako mapování bodů z jednoho obrazu do druhého a vyjadřuje transformaci mezi obrazy. Při skládání panoramatických snímků využíváme homografii k určení transformací mezi překrývajícími se snímky.

Matematický popis ve 2D

$$p_a = \begin{bmatrix} x_a \\ x_a \\ 1 \end{bmatrix}, p_b = \begin{bmatrix} x_b \\ x_b \\ 1 \end{bmatrix}, \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

$$\begin{aligned}
p_a &= H_{ab}p_b \\
p_b &= H_{ba}p_a \\
H_{ba} &= H_{ab}^{-1}
\end{aligned}$$

2.5.1 Výpočet homografie z korespondujících bodů

Pro každou dvojici korespondujících bodů lze psát [2]

$$wp' = Hp$$

kde w je neznámý parametr.

Pokud bude $(x, y, 1)$ a $(x', y', 1)$ dvojice korespondujících bodů v obrazech, dostaneme

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

po roznásobení matic a eliminací třetí rovnice dosazením w do předešlých dvou rovnic získáme dvě lineární algebraické rovnice.

$$\begin{aligned}
h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx' - x' &= 0 \\
h_{21}x + h_{22}y + h_{23} - h_{31}xx' - h_{32}yx' - x' &= 0
\end{aligned}$$

Pro n korespondencí ve dvou obrazech získáme systém $2n$ homogenních lineárních algebraických rovnic pro 9 neznámých. Tento lineární systém můžeme označit jako

$$Ah = 0$$

kde A je matice $2n \times 9$ a h vektor 9×1 obsahující všechny složky homografie.

$$A = \begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 & -y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x'_2 & -y_2x'_2 & -x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y'_2 & -y_2y'_2 & -y'_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nx'_n & -y_nx'_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_ny'_n & -y_ny'_n & -y'_n \end{pmatrix}$$

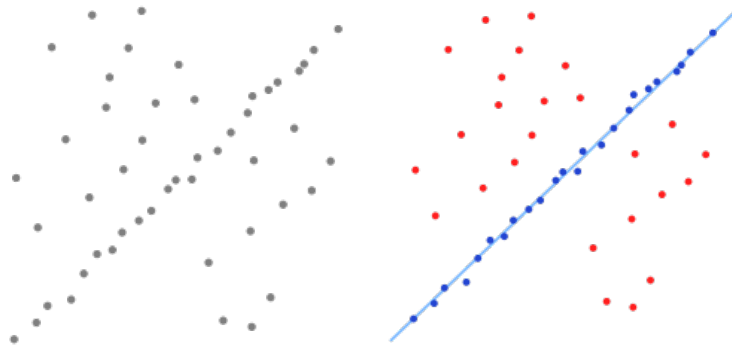
Protože používáme homogenní souřadnice je homografie H určena až na skalární násobek w jednoznačně [7]. Matice A má potom 8 stupňů volnosti. Homografii pak můžeme vypočítat z osmi lineárně nezávislých řádků. Jelikož z každé dvojice korespondujících bodů dostaneme dvě rovnice, je minimální počet korespondencí $n = 4$. Tyto body nesmějí ležet na přímkách.

Řešení soustavy rovnic matice A vypočítáme pomocí jejího SVD rozkladu.

2.6 RANSAC

RANSAC [13, 9] je zkratka pro RANdom SAMple Consensus - shoda náhodných vzorků. Je to iterativní metoda na odhadování parametrů matematického modelu. Model je popsán daty, které do modelu patří tzv. „inliers“, ale i daty, které nezapadají do správného modelu tzv. „outliers“ neboli šum. RANSAC slouží k vybrání těch správných dat (inliers). Metoda je vhodná pro data s velkým množstvím správných i nesprávných vzorků a můžeme ji tak použít například k počítání homografie v panoramatických snímcích. Algoritmus je často využíván v počítačovém vidění.

Příklad obecné aplikace algoritmu vidíme na obrázcích 2.2. Úkolem je najít takovou přímku, aby s ní korespondovalo co nejvíce bodů. Náhodně vždy vybereme dva body, proložíme jimi přímku a otestujeme kolik ostatních bodů leží poblíž této přímky.



Obrázek 2.2: Množina bodů v obraze

Algoritmus iterativně vybírá náhodné body z množiny všech hypotetických inliers bodů a ověřuje jestli jsou tyto body opravdu inliers. Všechny ostatní data jsou testovány tímto modelem tzn., že pro každý zbývající bod se počítá, jak moc daný bod koresponduje s hypotézou. Když dostatek bodů souhlasí s odhadnutým modelem hypotetických inliers, získáváme přijatelný model. Současně odhadujeme chybu jaké jsme se dopustili vzhledem k relativnímu modelu. Tato procedura je opakována předem daným počtem iterací. Pro každé opakování se vytváří jeden model, který je buď zamítnut, protože příliš málo bodů je klasifikováno jako inliers nebo potvrzen s určitou chybou. Pokud je chyba modelu menší než u předchozího potvrzeného modelu, uchováваме si vzniklý model s menší chybou. Slovní popis můžeme vyjádřit pseudokódem.

```
iterations = 0
bestModel = nil
bestConsensusSet = nil
bestError = infinity
while iterations < k do
  maybeInliers = n // nahodne vybrane hodnoty z dat
  maybeModel = model // parametry k maybeInliers
  consensusSet = maybeInliers
  for kazdy bod v datech ktery neni v maybeInliers do
    if bod sedi maybeModelu s chybou mensi nez t then
      pridej bod od consensusSet
```

```

    end if
end for
if pocetelementuvconsensusSet > d then
    betterModel = model
    thisError = a // meri jak moc betterModel souhlasí s temito body
    if thisErr < bestErr then
        bestModel = betterModel // nasli jsme zatim nejlepsi model, tak si ho uchovame
        bestConsensusSet = consensusSet
        bestError = thisError
    end if
end if
increment iterations
end while
return bestModel, bestConsensusSet, bestError

```

Hodnota parametrů t a d závisí na konkrétní aplikaci algoritmu. Parametr k je počet iterací. Počet iterací může být odvozen ze vzorce pravděpodobnosti 2.2. Označme si p jako pravděpodobnost, že v některé iteraci bude ze sady dat vybráno n bodů a všechny budou inliers. Pravděpodobnost p značí užitečné výsledky. Proměnná w je pravděpodobnost vybrání inlier bodů pokaždé, když je vybrán nějaký jeden bod. Neboli

$$w = \text{pocetInliers} / \text{pocetBoduVDatech} \quad (2.1)$$

Pravděpodobnost w^n popisuje, že všechny body n jsou inliers a $1 - w^n$ je pravděpodobnost, že nejméně jeden z n bodů je outlier. Pravděpodobnost, že algoritmus nikdy nevybere sadu bodů, které by všechny byli inliers je $1 - p$.

$$1 - p = (1 - w^n)^k \quad (2.2)$$

Výpočet počtu iterací k pro dosažení úspěchu můžeme spočítat následovně:

$$k = \frac{\log 1 - p}{\log (1 - w^n)} \quad (2.3)$$

Výhoda RANSACu je jeho schopnost robustně odhadnout parametry matematického modelu s vysokou přesností, dokonce i přes zastoupení outliers v sadě dat. Jeho nevýhoda je, že model může být získán až po velkém počtu iterací. Pokud se počet iterací omezí, získané řešení nemusí být optimální.

Kapitola 3

Automatický generátor panoramat

Program aplikuje jednotlivé základní postupy, tak jak je deklarované v následující kapitole 3.1. Rohovým detektorem najde pouze souřadnice bodů, na základě jejich okolí určí korespondence a pak snímky spojí.

Algoritmus je napsán v C/C++ s použitím knihovny OpenCV [1]. Protože knihovna OpenCV podporuje celou řadu funkcí na zpracování obrazu a navíc jsou tyto funkce značně optimalizované, využívám je pro úsporu času i výpočetního výkonu.

3.1 Návrh řešení

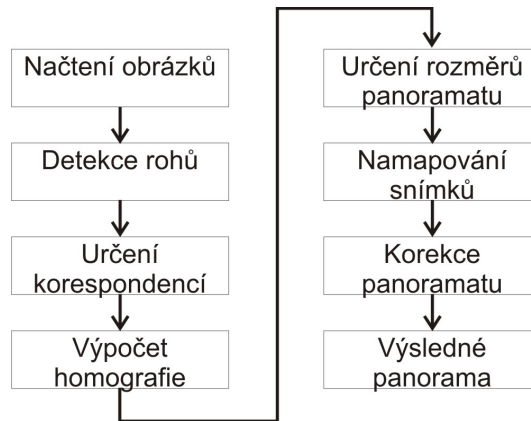
Základem je určit, které dvě fotografie k sobě patří. Program se zjednodušeně chová jako člověk, který by měl za úkol fotografie poskládat. Člověk se podívá jestli je na fotografiích nějaký společný prvek, třeba budova v dálce a podle ní pozná, že ony dvě fotografie patří k sobě, protože zachycují jednu a tu samou skutečnost. Lidé jsou navíc schopni pracovat s jistou úrovní představitosti, takže určí i vzájemnost takových obrázků, které se nijak nepřekrývají a pouze na sebe navazují. To by byl pro počítačové vidění velmi obtížný úkol. Předpokládáme, že je mezi fotografiemi určitý překryv.

Automatickou tvorbu panoramat můžeme realizovat ve čtyřech krocích

1. Detekce důležitých bodů v obraze (Feature Points Extraction)
2. Určení korespondencí mezi snímky
3. Výpočet homografie mezi snímky
4. Výpočet vzájemné pozice snímků v panoramatu, proložení snímků
5. Korekce panoramatu, blending, korekce soudkovitosti, narovnání, ...

Jako první se u všech vstupních dat provede detekce význačných bodů. Tento krok je spolu s určením korespondence nejdůležitější, protože vlastní kvalita detektoru významně ovlivňuje schopnost snímky posléze spojit. Vzájemné korespondence můžeme určit dvěma způsoby. Prvním je použití nějakého robustního detektoru na určení rohů a zároveň popsat detekované význačné body deskriptory, které pak budeme srovnávat. Druhým způsobem je použití jednotlivě rohového detektoru dávajícího pouze souřadnice rohu a metody na určení korespondencí na základě okolí rohu.

Program musí umět spojovat panoramatické snímky bez předchozí inicializace uživatelem. Tudíž nemáme k dispozici žádnou informaci o sousedství snímků, jejich překryvu a zařazení



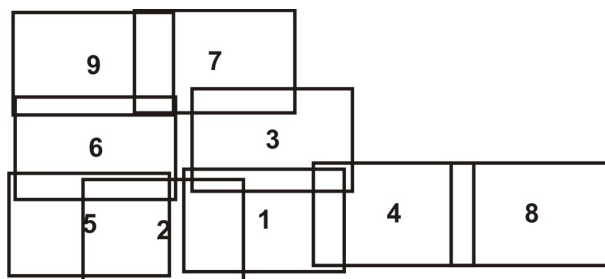
Obrázek 3.1: Zjednodušený diagram řešení

v celkovém obraze. To celé řešení dosti komplikuje. V důsledku to znamená, že se hledání korespondencí musí provést vždy pro dva snímky systémem každý s každým.

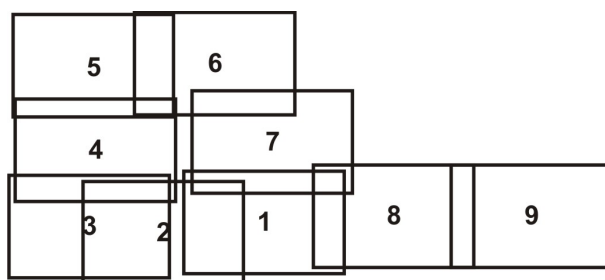
Jakmile máme vyřešenou korespondenci bodů, přichází na řadu výpočet homografie, tedy jak je jeden snímek promítán do druhého. V úvahu přichází posunutí, rotace, zkosení a změna měřítko, případně další nelineární transformace. Homografii můžeme například určit z korespondujících bodů metodou nejmenších čtverců, ale kvůli vysokému množství chybných shod je tento přístup naprosto nevhodný. Pro jeden bod v prvním snímku může být několik kandidátů ve snímku druhém nebo mohou být spolu spojeny body, které jsou si sice podobné, ale v obrázcích mezi sebou nikterak nesouvisí. Zde lze s úspěchem využít algoritmus RANSAC, který právě eliminuje chybně určené body (outliers) a výsledek počítá jen s potenciálními inliers body. Ověření homografie můžeme provést pomocí „transfer error“, „symetric transfer error“ nebo Sampsonovi metody.

Na základě homografie mezi dvěma snímky jsme schopni vypočítat celkovou velikost panoramatu a následně pozice jednotlivých snímků ve výsledném obrázku. Pozice snímků se odvíjejí od tzv. referenčního snímku. To je první snímek zasazený do výsledného zatím prázdného obrázku, na který se na základě homografie namapují jeho korespondující sousedé. Na jeho sousedy se opět spočtenou homografií mapují jejich sousedé a takto rekurzivně, až má každý snímek své místo. Snímky skládám po úrovních. V každé úrovni rozvinu sousedy snímků v předchozí úrovni. Teprve až je celá úroveň zpracovaná, začínají se rozvíjet snímky v další úrovni. Kdyby se tak nečinilo, může se stát, že neustálým rozvíjením sousedů jednoho snímku se oklikou dostaneme ke snímku, který leží hned vedle. Rozdíl demonstrují obrázky 3.2 a 3.3. Dalším důvodem proč se nejprve rozvíjejí sousední snímky, je akumulace chyby násobením transformačních matic. Jelikož se všechny snímky mapují vzhledem k referenčnímu snímku, akumuluje se chyba homografie. Proto je vhodné jako referenční snímek vzít prostřední snímek.

Kapitola sama pro sebe je finální úprava výsledného panoramatu, aby uživatel nepostřehl přechody mezi snímky či jiné deformace výsledného obrázku. Nejčastějším způsobem jak toho docílit, je míchání snímků (blending). Blending se u tvorby panoramat využívá pro vyhlazení přechodů mezi snímky. Jeho nevýhodou je, že může zapříčinit rozmazání snímku nebo efekt duchů. Další postupy pro korekci panoramatu jsou automatické narovnání snímku, oříznutí snímků, korekce soudkovitosti, vylepšení homografie pomocí newtonovy metody, gain compensation atd.



Obrázek 3.2: Rozvíjení snímků v úrovních



Obrázek 3.3: Postupné rozvíjení snímků

3.2 Detekce důležitých bodů v obraze

K detekci bodů jsem si vybral jeden z nejznámějších detektorů, a to Harrisův rohový detektor. Ten dává dobré výsledky a pro účel této práce naprosto dostačuje. Jeho nevýhodou je vyšší citlivost na šum v obraze, proto před samotnou detekcí musíme snímky předzpracovat.

Nejprve si načteme všechny snímky do paměti a zároveň vytvoříme jejich kopie, které převedeme do úrovně šedé. Další zpracování snímků se bude provádět jen s šedotónovým obrazem. Převod se dá realizovat velmi jednoduše na základě zkušenosti vnímání barev lidským okem.

$$I = 0.299R + 0.587G + 0.114B$$

Vzorec se aplikujeme na každý pixel, kde R , G , B jsou jednotlivé barevné kanály. O načtení obrázků se stará funkce *cvLoadImage*. Funkce má podporu pro různé formáty obrázků (BMP, DIB, JPEG, JPG, JPE, PNG, PBM, PGM, PPM, SR, RAS, TIFF, TIF).

Jednokanálový obrázek rozmážeme Gaussovým filtrem a tím ho zbavíme šumu. Aplikování filtru nám přinese viditelné výsledky. Pomůžeme tím jak lepší detekci Harrisovým detektorem, tak i kvalitnějšímu porovnávání bodů pomocí korelace. Obrázek 4.11 demonstruje výsledek s Gaussovým filtrem a bez použití filtru. Bez filtrace bylo detekováno 304 rohů, se sousedním obrázkem bylo nalezeno 57 korespondencí z toho 33 inliers označených červenou čarou. S použitým filtrem bylo nalezeno 247 rohů, 104 korespondencí z toho 65 inliers. Rohů se našlo méně, což urychluje výpočet korespondencí, navíc jsme získali dvojnásobný počet správných korespondencí (inliers).

V takto předzpracovaných obrázcích detekujeme rohy Harrisovým detektorem. Na základě mého experimentování (přesnost/rychlost) jsem zvolil maximální počet rohů v jednom obraze na 400. Kolik rohů se nakonec nadetekuje však záleží na typu obrázku a nastavení minimální kvality přijatých rohů. Při blízkém snímku budovy se detekuje třeba jen polovina

rohů z maximálního přípustného počtu. Naopak u vzdálené členité krajiny není problém při stejné kvalitě rohovosti detekovat dvakrát tolik bodů než je maximum.



Obrázek 3.4: Detekované rohy ve dvou obrázcích (247 resp. 390 bodů)

Pro dosažení co nejlepšího výsledku počítáme s takzvanou subpixelovou přesností. Čili souřadnice rohu nemáme v celých, ale v desetinných číslech typu float.

3.3 Určení korespondencí mezi snímky

Z Harrisova detektoru dostaneme souřadnice se subpixelovou přesností. Srovnávat mezi sebou jednotlivě pixely je z pochopitelných důvodů nemožné. Protože každý roh je určen svým okolím, budou se navzájem srovnávat právě tyto okolí. Čím větší okolí pixelu, tím větší je jeho unikátnost. Na druhou stranu musíme navzájem srovnávat větší celky a tím nám roste výpočetní složitost. Zanedbatelný není ani fakt, že díky všemožným zkreslením a posunutí kamery nemusí být příliš velká okolí identická. Malému okolí bodu zase vyhovuje hodně kandidátů. Potom je procento spojených bodů, které mezi sebou nemají žádnou souvislost bohužel větší.

V mém projektu jsem se rozhodl pro čtvercové okolí 10 pixelů, tzn. okénko 20 x 20 pixelů s rohem uprostřed. Je to mnou experimentálně ověřený kompromis mezi rychlostí a přesností. Velikost okolí vyhovuje i s ohledem na rozlišení obrázků. Svět v okénku obrázku s velkým rozlišením se logicky tolik nemění jako okénko téhož zmenšeného obrázku. Možnou úpravou by mohlo být přepočítávání korelačního okénka v závislosti na rozlišení obrázku tak, aby pokaždé zobrazovalo stejný výřez.

Jak již bylo naznačeno, srovnání okolí provádím korelací. Přesněji řečeno normalizovanou cross-korelací, která je invariantní vůči rozdílným jasům v obrázcích. Okolí se srovnávají hrubou silou, systémem každý s každým. Postupně vezmeme první roh z prvního obrázku a srovnáme jej se všemi ostatními rohy s druhého obrázku a zapamatujeme si nejlepší shodu. Pak vezmeme druhý roh, opět provedeme srovnání atd. Na konci každého srovnání porovnáme nejlepší shodu s prahovou hodnotou. Výstupem korelace je desetinné číslo udávající podobnost dvou vzorků. Výsledek leží v intervalu $< -1, 1 >$, kde hodnota -1 znamená nejmenší shodu a hodnota 1 identitu. Prahování z dalšího zpracování vyloučí body nemající žádnou korespondenci v druhém obrázku. Práh se musí zvolit citlivě, aby eliminoval co možná nejvíc špatných bodů a přitom nevyloučit potenciální správné korespondence.

Normalizovanou korelací jsem nejprve napsal podle vzorce v kapitole 2.3, ale nakonec bylo výhodnější přejít k již optimalizovanému řešení v knihovně OpenCV. Zde je imple-

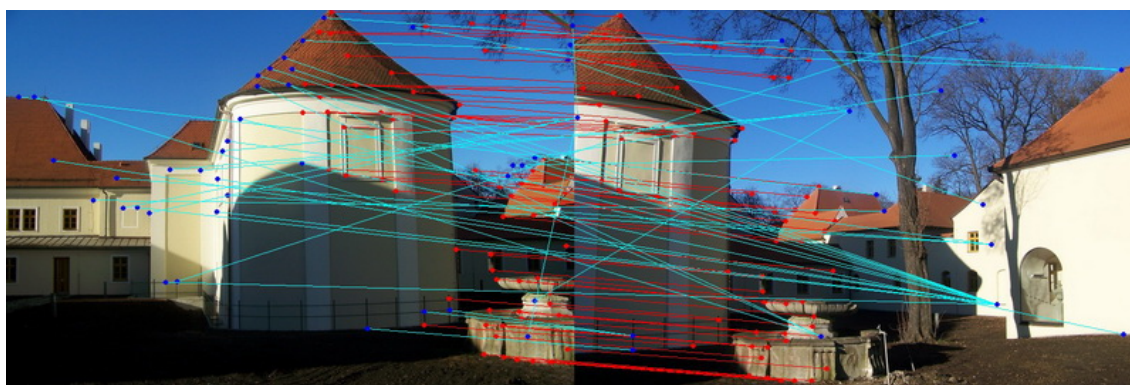
mentováno řešení pro template matching. Není to úplně přesně to co potřebujeme, protože už víme, které dvě oblasti máme srovnávat. Nemusí se tudíž procházet celý obrázek. Aplikujeme ovšem podobnou myšlenku template matchingu. Abychom nemuseli pořád dokola určovat korelační okénko kolem rohu, „vyřízneme“ jej ze subpixelovou přesností a za sebe uložíme do pomocné datové struktury. Vznikne dlouhý pás výřezů z obrázku jako na obr. 3.5. Srovnání pak probíhá postupným posunováním po okénkách.



Obrázek 3.5: Pomocná struktura výřezů z obrázku

Porovnávání provádím pouze prvního obrázku s druhým a nikoli naopak. Kdybychom obrázky srovnávali i obráceně, dostaneme i z části jiné korespondence a jiný počet korespondencí. Mohlo by být výhodné takto korespondující body přesněji detekovat a eliminovat množství špatně určených shod, ale vzhledem k velkému nárůstu výpočetní náročnosti tak nečiním.

Hledání korespondencí je úzké místo celého programu. Je to dáno neustálým opětovným počítáním velkého množství korelací. Obzvláště citelně to lze pozorovat při větším množství fotografií. Protože neznáme vzájemnou polohu fotografií, musí se takřka každá fotografie srovnávat s každou. Pro praxi je jistě výhodnější pouze jednou popsat okolí rohu deskriptorem a posléze srovnávat jen tyto deskriptory. Pro představu u SIFTu to jsou 128 bitové vektory, které se navíc uspořádají do k-dimenzionálního stromu, a tak se jejich srovnávání výrazně urychlí [3].



Obrázek 3.6: Korespondující si body (65 inlier-červeně, 39 outlier-modře)

3.4 Homografie mezi snímky

Kvůli velkému množství nesprávně určených korespondencí počítám homografii s využitím algoritmu RANSAC. RANSAC je obecný algoritmus, a proto se musí modifikovat na výpočet homografie. Bohužel v knihovně OpenCV je algoritmus napsán jen k nalezení fundamentální matice. Toho se využívá v epipolární geometrii při tvorbě modelu ze zobrazovaných dat u dvou stereometrických snímků.

Při každé iteraci algoritmu vybereme náhodné vzorky dat a z nich vypočítáme homografii. Zjistíme kolik těmto vzorkům odpovídá korespondencí (inliers). Řešení s větším počtem inliers si zapamatujeme. Algoritmus se provádí tak dlouho, dokud počet jeho iterací nepřekročí předem stanovenou mez nebo dokud nedosáhneme vysoké pravděpodobnosti správného řešení.

V části 2.5.1 je ukázáno, že minimální počet korespondencí na výpočet homografie je čtyři. U výběru bodů kontrolujeme, zdali se nám do sady náhodně nevybral dvakrát či vícekrát jeden a ten samý bod a zda vybrané body neleží v řadě. Z takto vybraných bodů v jednom obrázku spočteme jejich namapování do druhého obrázku. Získáme potenciální homografii mezi obrázky.

Homografii testuji pomocí tzv. „Symetric transfer error“ [9].

$$d_{transfer}^2 = d(x, H^{-1}x')^2 + d(x', Hx)^2$$

Vezmeme bod v prvním obrázku a potenciální homografií spočítáme jeho souřadnice mapované do druhého obrázku. Určíme odchylku takto vzniklých souřadnic od souřadnic jeho skutečného korespondujícího bodu. Stejný postup se provede i obráceně s mapování bodu z druhého obrázku do prvního. Pokud je symetrická chyba v normě, získali jsme inlier. Test provedeme postupně pro všechny korespondující dvojice. Tím zjistíme kolik inliers odpovídá potenciální homografii. Když je počet inliers v celé historii RANSACu největší, uložíme si potenciální homografii jako zatím nejlepší řešení a upravíme počet iterací algoritmu podle nové pravděpodobnosti.

Po ukončení RANSACu je výhodné homografii znovu přepočítat tentokrát se všemi inliers. Homografie se tím bude průměrovat ze všech inliers a ne jenom z náhodných čtyřech nebo více vzorků. Výsledek bude buď stejný nebo lepší (znázorněno na obrázku 3.8). Homografii nepočítám ručně, ale získávám ji funkcí z knihovny OpenCV. Funkce matici rovnou normalizuje vůči zvětšení ($h_{33} = 1$). Vzhledem k dalšímu zpracování si ke každé fotografii ukládám všechny její homografie k ostatním fotkám.



Obrázek 3.7: Spojení dvou fotografií



Obrázek 3.8: Homografie počítaná z 5 vzorků–vlevo a z 65 vzorků–vpravo

I se čtyřmi inliers lze napojit snímky přesně, zvláště když mají dobrou distribuci v obraze. Pracování s tak malým počtem inliers však v sobě skrývá rizika. Výpočet není moc spolehlivý. S menším počtem inliers se často stává, že se najde transformace, která v reálu vůbec nevyhovuje vzájemné pozici snímku. Homografie různě zdeformuje navazující obraz a i k takovéto homografii se může najít více odpovídajících korespondencí. Přičemž stačí, aby mezi správně určenými shodami byla jediná nesprávně určená a výsledek je nepoužitelný. Řešením je počítat potenciální homografii z více počátečních bodů a zároveň brát jako správnou tu transformaci, které vyhoví více korespondencím než stanovená mez. Za správný výsledek se například bere transformační matice počítaná z pěti bodů a vyhovující pro 10 a více korespondencí.

Zvednutí počtu náhodně vybíraných bodů není zadarmo. Počet kombinací jak můžeme vybrat n bodů z celkového počtu m bodů je $\binom{m}{n}$. Máme-li $m = 200$ korespondencí tak $n = 4$ body můžeme vybrat $\binom{200}{4} = 64.7$ milionu kombinací. Pro 5 bodů už je to 2.5 miliard kombinací. S více jak čtyřmi body mi RANSAC přestal hledat shody v rozumném počtu iterací. Uvažujme, že fotografie mají překryv 50%. Pravděpodobnost, že vybereme všech n bodů v jedné polovině obrázku je $p = \frac{\binom{m/2}{n}}{\binom{m}{n}}$. Pro $n = 4$ je pravděpodobnost $p = 0.06$ a pro $n = 5$ je $p = 0.03$. Dostáváme tak pouze 6% respektive 3% potenciálně úspěšných iterací. Vzniklý problém jsem vyřešil vybíráním bodů střídavě vždy z jedné poloviny obrázku (horní polovina, dolní polovina, levá polovina, pravá polovina). Obrázek jsem rozdělil na 4 kvadranty a n bodů, pak vybírám ze dvou sousedních kvadrantů. Tyto čtyři poloviny nestřídám pravidelně, ale rozhoduji se podle počtu korespondencí v dílčí polovině. Procento potenciálně úspěšných iterací je vyšší. Výpočet je únosný i pro více jak 4 body a je méně náchylný k chybám.

Určení inliers v korespondencích zachycuje obrázek 3.6. Výsledek spojení je v obr. 3.7.

3.5 Vzájemná pozice snímků v panoramatu

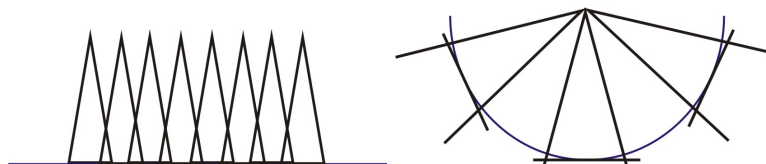
Samotné spojení fotografií není kvůli perspektivě a zkreslení kamery triviální. Situaci zachycuje obrázek 3.9. Vzdálenější objekty díky perspektivě a zkreslení svoji polohu moc nemění, zatímco bližší objekty se „pohybují“ rychleji. Snímky pořízeny paralelně mohou být mapovány do roviny a žádné zkreslení se neprojeví. Naopak u snímků pořízených horizontální rotací kamery se zkreslení projevuje, a proto by měly být mapovány na válec. Projekci na plochu a válec ilustrují obrázky 3.10 a 3.14.



Obrázek 3.9: Projev zkreslení u sloupku plotu

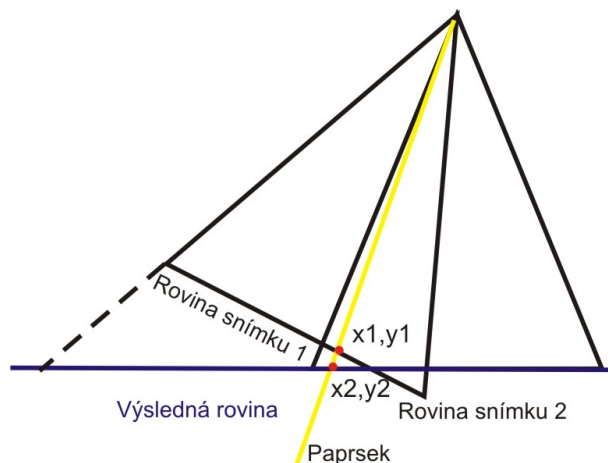
Z předchozího kroku jsme získali dílčí homografie mezi dvěma obrázky. Nyní se musí určit velikost výsledného obrázku a do něj namapovat všechny fotografie. Před tím by bylo ideální fotografie upravit. Hlavně zrušit zkreslení čočky kamery a fotografie mapovat na válec. Nic takového ale nedělám a obrázky mapuji pouze do roviny dané referenčním snímkem. Protože jsou všechny obrázky mapovány vůči referenčnímu snímku, s každým další snímkem se postupně akumuluje chyba projevující se nepříjemným roztažováním výsledného panoramatu. Viz. obrázek 3.11. Tuto vlastnost se alespoň snažím zmírnit vybráním referenčního snímku uprostřed panorama.

Velikost výsledného panorama nemůžeme díky zkreslení jen tak počítat s náhodně zvoleným referenčním snímkem. Obrázek 3.12 zachycuje skutečnost, že pro každý jiný snímek zvolený jako referenční dostáváme rozdílné rozměry výsledného panorama. Referenční snímek určuji následovně tak, že z počátku náhodně zvolím dočasný referenční snímek, ale mapování ostatních snímků proběhne pouze jako posunutí po osách x a y (z homografie беру v potaz pouze vektor posunutí). Přesněji řečeno dočasný referenční snímek беру jako bod $0,0$ a tento bod se vektorem posunutí dostává na další souřadnice. Zapamatuji si maximální získané souřadnice ve všech světových stranách. Tím získám čtvercovou obálku všech mapovaných snímků. U ní určím střed a euklidovskou vzdáleností zjistím snímek nejbližší středu. Postup popisuje obrázek 3.13. Získali jsme konečný referenční snímek. Nyní už můžeme vypočítat výslednou velikost panorama. Čtvercovou obálku počítáme podobně. Rozdílem je, že v úvahu bereme celou homografii a nemapujeme pouze jeden bod, ale čtyři rohy snímku, u kterých zaznamenáváme maximální souřadnice ve světových stranách.

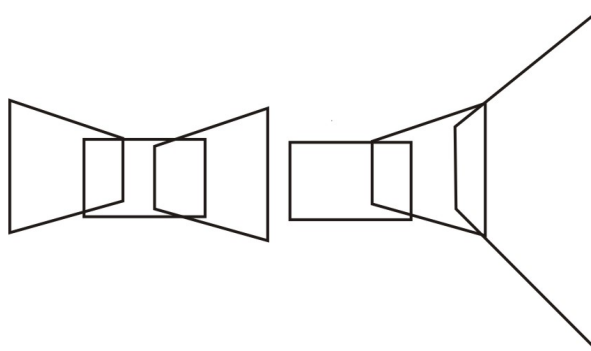


Obrázek 3.10: Snímání prostředí paralelně – vlevo a rotací kolem své osy – vpravo. Kamera je umístěna na špičce trojúhelníku

Skládání snímků probíhá od referenčního snímku postupným rozvíjením po úrovních jak je naznačeno v kapitole 3.1. Vycházel jsem z předpokladu, že každým dalším mapováním



Obrázek 3.11: Roztahování snímků při projekci do roviny. Snímky jsou pořízených rotující kamerou kolem své osy

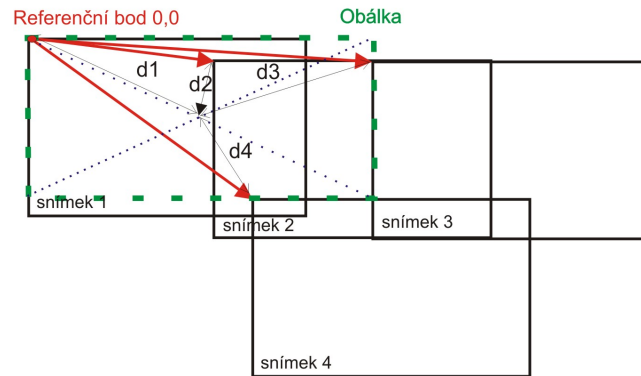


Obrázek 3.12: Jiné rozměry čtvercové obálky pro druhý a první referenční snímek

se akumuluje chyba a proto pokud se od jednoho snímku mohou dostat k druhému v první úrovni a od dalšího k tomu samému v druhé úrovni bude se preferovat první případ. Pokud nastane, že se ke druhému snímku můžeme dostat dvěma způsoby v první úrovni, vybere se homografie s větším počtem inliers. Ukázalo se, že tato myšlenka není ideální. Snímky na sebe kvůli zkreslení stejně čistě nenavazovaly. Rozhodl jsem se celý postup přehodnotit a zjednodušit. Korespondence obrázků již nehledám systémem každý s každým, ale jakmile se k jednomu obrázku nalezne korespondující obrázek, další korespondence se už k tomuto obrázku nehledají, byť by teoreticky mohli mít větší počet inliers. Dá se tím trochu ušetřit počet všech vzájemných srovnání a výsledek je prakticky totožný. Neoptimálnější řešení hledání vzájemné pozice obrázků by snad mohlo být zohlednění všech homografií, které má snímek s ostatními snímky a její zprůměrování na základě počtu inliers.

Mapování snímků různě transformuje, mění jeho velikost potažmo rozlišení. Nově vzniklé pixely dostaneme interpolací z originálních pixelů.

Pro přehlednost mapuji referenční snímek jako poslední. V panoramatech ho poznáte jako prostřední nezdeformovaný snímek ležící před všemi ostatními.



Obrázek 3.13: Výpočet referenčního snímku. Jako referenční bude zvolen snímek č.2, protože vzdálenost d_2 od středu obálky je nejmenší



Obrázek 3.14: Fotografie mapované na válec o různém poloměru

3.6 Korekce panoramatu

Tato kapitola se úzce prolíná s předchozí kapitolou. Některé korekce se totiž provádí zároveň s mapováním snímků. Řeč je především o míchání (blending) a úpravě jasu. Jiné lze provést i s výsledným panoramatem. Například narovnání či oříznutí. Postupy byly nastíněny v kapitole 3.1. Zde se jimi nebudu zabývat, protože jsou nad rámec mé práce.

Kapitola 4

Výsledky

Snímky ve výsledcích byly cíleně pořízeny bez manuálního nastavení parametrů fotoaparátu (čas, clona, ISO, ...) a bez použití stativu. Roztahování panoramat je dáno mapováním do roviny referenčního snímku a nikoli na válec. Projekce na válec a zajištění bezešvých přechodů může být předmětem další práce.

Ryze pro demonstrační účel jsem vytvořil obrázek 4.2. Jedná se stejnou sekvencí snímků jako na obrázku 4.1, ale mapovanou na válec. Poloměr válce byl experimentálně odhadnut. Černé pruhy znázorňují okraje snímků. K převodu do cylindrických souřadnic je potřeba znát ohniskovou vzdálenost nebo úhel pohledu. Ohnisková vzdálenost se dá zjistit například ze dvou snímků, kalibrací fotoaparátu nebo z metadat uložených u obrázku, pokud jsou tyto data k dispozici. Více o dané problematice naleznete v [10].



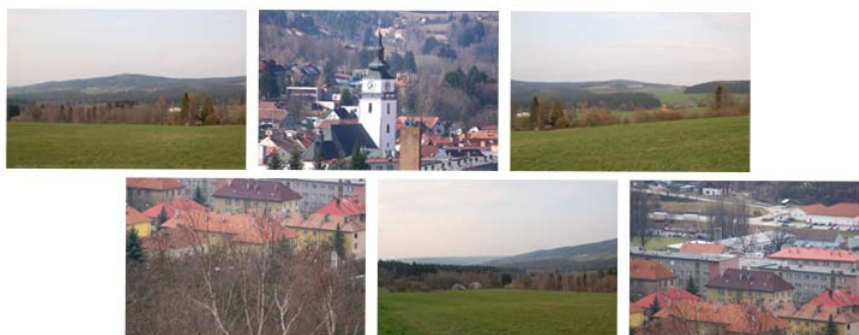
Obrázek 4.1: Krajina na vysočině (7 fotografií)



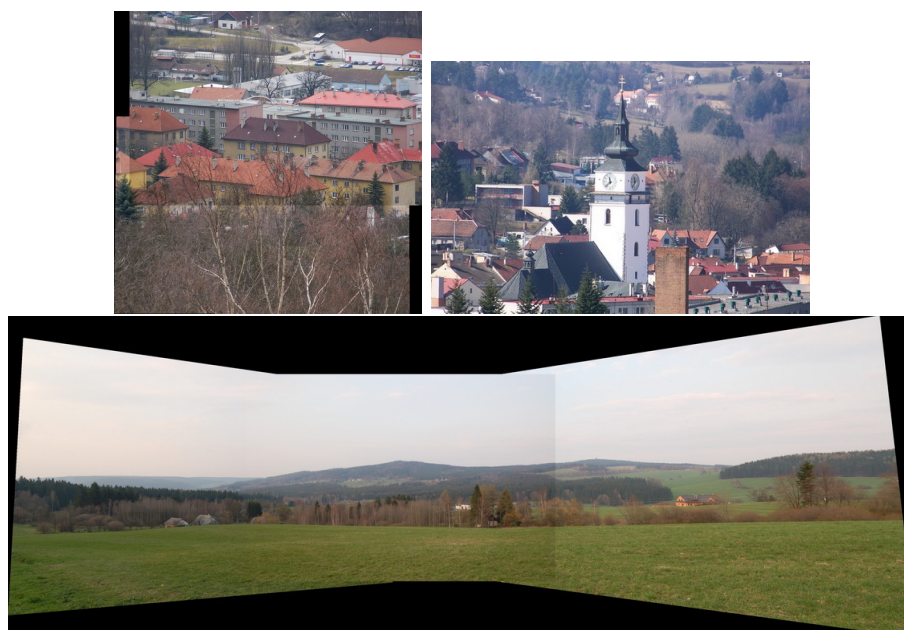
Obrázek 4.2: Krajina mapovaná na válec

V programu je nastaven maximální počet rohů na 400, korelační okénko je 20 x 20 pixelů a maximální počet iterací algoritmu RANSAC byl zvolen na 1000. Procentuální

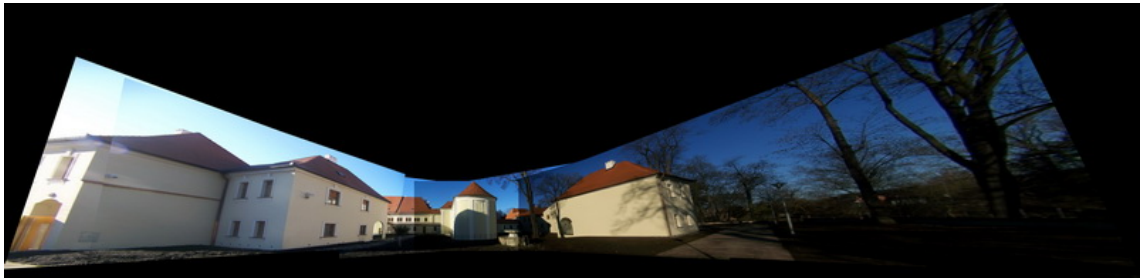
rozložení výkonu pro tyto parametry je zhruba následovné: hledání rohů 1.5 %, korelace 84 % , RANSAC 12 %, mapování 2.5 %. Výpočetní složitost významně neovlivňuje ani větší rozlišení obrázků. Vyšší rozlišení se projeví nepatrným zvednutím času u hledání rohů a mírným nárůstem procentuálního rozložení výkonu při mapování. Nárůst je v jednotkách procent a v porovnání s korelací je zanedbatelný. Zvednutí počtu iterací u RANSACu za účelem přesnějších výsledků nemělo smysl. U překrývajících se snímků většinou algoritmus skončí dříve a u nesouvisajících snímků, kterých je více, by se zbytečně počítalo až do konce zarážky. Jediným způsobem jak regulovat rychlost programu je snížení počtu detekovaných rohů. Vždy pro dva obrázky se korelací srovnává každý roh s každým, tzn. provede se 400 x 400 korelací. Složitost roste kvadraticky. Pokud jsem snížil počet rohů na 300, výpočet se urychlil téměř dvojnásobně. Naproti tomu počet inliers klesnul v průměru o 30 %. To může být kritické pro snímky s menším překryvem.



Obrázek 4.3: Sekvence fotografií



Obrázek 4.4: Výstup ze sekvence 4.3



Obrázek 4.5: Klášter FIT (7 fotografií)



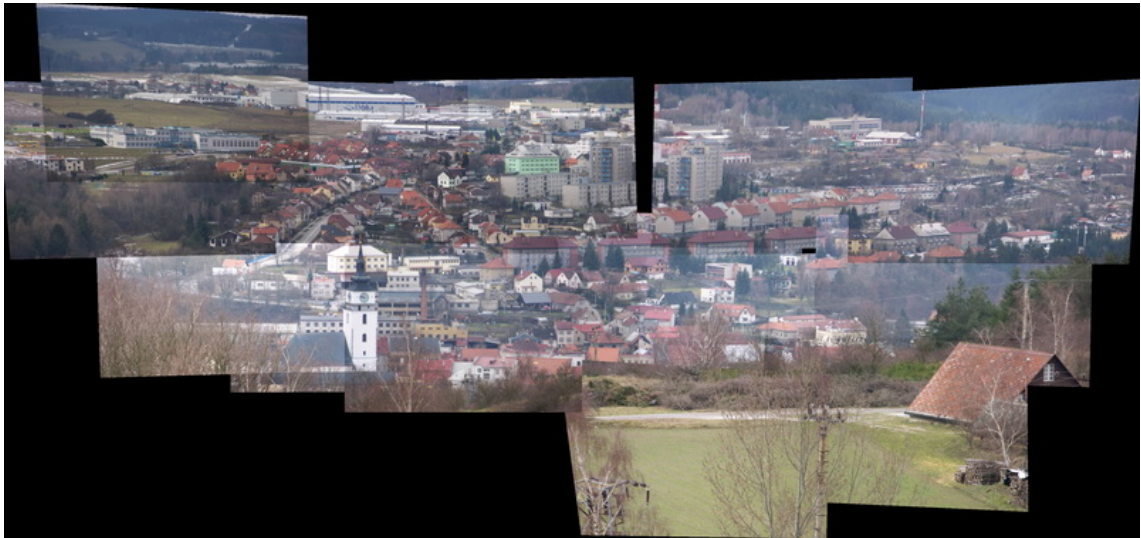
Obrázek 4.6: Nový komplex FIT (4 fotografie)



Obrázek 4.7: Panorama Velkého Meziříčí (11 fotografií)



Obrázek 4.8: Panorama Velkého Meziříčí (9 fotografií)



Obrázek 4.9: Velké Meziříčí (18 fotografií)



Obrázek 4.10: Velké Meziříčí (7 fotografií)



Obrázek 4.11: Rozdílný výsledek s Gaussovým filtrem a bez něj

Kapitola 5

Závěr

Počítačové vidění je v dnešní době velmi dynamicky se vyvíjející odvětví. Při psaní práce jsem měl možnost se s ním blíže seznámit. Konkrétně s jeho základními technikami jako je detekce bodů v obraze, hledání korespondujících bodů, počítání geometrických transformací a celkovou techniku práce s obrazem.

Po prostudování jednotlivých vlastností rohových detektorů jsem se rozhodl pro aplikaci Harrisova detektoru, jelikož splňuje účely této práce. Korespondenci rohů ve dvou fotografiích zjistím srovnáváním čtvercových okolí rohů normalizovanou cross-korelací. Na výpočet homografie jsem aplikoval algoritmus RANSAC, kvůli jeho vlastnosti eliminovat nesprávné korespondence. Řešení jsem se snažil navrhnout tak, aby bylo co nejobecnější. Snímky mohou být pořízeny v libovolném směru, nemusí být seřazené a ve vstupních datech může být přítomno více spolu nesouvisejících sekvencí nebo jednotlivých snímků. Ve finálním panoramatu jsou fotografie mapovány do roviny dané referenčním snímkem a vrstveny pouze na sebe. Všechny výpočty jsou prováděny ze subpixelovou přesností.

V rámci další budoucí práce můžeme použít k přesnějšímu hledání rohů robustnější detektor než je Harris-Stephensův. Místo korelace bychom mohli srovnávat rohy pokročilejším algoritmem, který je více invariantní vůči geometrickým transformacím, změně osvětlení, šumu, apod. Z tohoto pohledu by bylo zajímavé vyzkoušení algoritmů SIFT nebo SURF. Sliboval bych si od toho především zpřesnění výsledků a snížení výpočetní náročnosti. Nejvýraznějším viditelným vylepšením by určitě bylo zrušení zkreslení čočky kamery a mapování snímků na válec. Obrázky by na sebe lépe navazovali a neprojevovalo by se nepříjemné roztahování výsledného snímku do stran. Lepšího vnímání panoramatu bychom docílili vyrovnáním jasů jednotlivých obrázků a mícháním snímků na jejich přechodech. Finálním krokem by bylo vytvoření grafického uživatelského rozhraní na tvorbu panoramat.

Literatura

- [1] Open Source Computer Vision Library – OpenCV.
URL <<http://www.intel.com/technology/computing/opencv/index.htm>>
- [2] *Assignment 4: RANSAC for fitting translations and homographies*. [online], 23.10.2004, naposledy navštíveno 2008-04-01.
URL <http://www-static.cc.gatech.edu/classes/AY2005/cs4495_fall/assignment4.pdf>
- [3] Brown, M.; Lowe, D. G.: Automatic Panoramic Image Stitching using Invariant Features. Technická zpráva, University of British Columbia, Vancouver, 1996.
- [4] Brown, M.; Lowe, D. G.: Recognising Panoramas. [online], [cit. 2008-05-01], powerpoint presentation in Proceedings of the 9th International Conference on Computer Vision (ICCV2003).
URL <<http://www.cs.ubc.ca/~mbrown/panorama/iccv2003.ppt>>
- [5] Harris, C.; Stephens, M.: A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, 1988, s. 147–151.
- [6] Hlaváč, V.: *Zpracování signálů a obrazů*. Praha: ČVUT, 2005, ISBN 80-01-03110-1.
- [7] Krížek, P.: PVI-Panorama. [online], [cit. 2008-04-01].
URL <<http://cmp.felk.cvut.cz/cmp/courses/pvi2003/Projects/Uloha1/krizekp1/index.html>>
- [8] Parian, J. A.; Gruen, A.: A Sensor model for panoramatic cameras. In *6th Conference on Optical 3D Measurement Techniques*, 2003, s. 130–141.
- [9] Potůček, I.: Přednášky FIT pro předmět počítačové vidění POV (Detekce a parametrizace objektů v obraze, transformace, RANSAC). [online], [cit. 2008-04-11], dostupnost na privátních stránkách předmětu POV.
- [10] Shum, H.-Y.; Szeliski, R.: Panoramic Image Mosaics. Technická zpráva, Microsoft Research, Redmond, 1997.
- [11] Wikipedia: Corner detection. [online], [cit. 2008-04-11].
URL <http://en.wikipedia.org/wiki/Corner_detection>
- [12] Wikipedia: Homography. [online], [cit. 2008-04-11].
URL <<http://en.wikipedia.org/wiki/Homography>>
- [13] Wikipedia: RANSAC. [online], [cit. 2008-04-11].
URL <<http://en.wikipedia.org/wiki/RANSAC>>

- [14] Wikipedia: Scale invariant feature transform. [online], [cit. 2008-04-11].
URL <http://en.wikipedia.org/wiki/Scale-invariant_feature_transform>
- [15] Černocký, J.: Zpracování řečových signálů – studijní opora FIT. [online], 6.12.2006,
[cit. 2008-04-13], dostupnost na privátních stránkách předmětu ZRE.

Seznam příloh

1. Základní manuál programu
2. CD/DVD

1. Základní manuál programu

Jedná se o konzolovou aplikaci. Vstupní obrázky se zadávají jako parametry oddělené mezerou. Příklad spuštění:

```
pano.exe obrazek1.jpg obrazek2.jpg obrazek3.jpg
```

Program načítá obrázky ze svého kořenového adresáře. Pokud se načítání nezdaří, program se ukončí. Výstup je zobrazen v novém okně a uložen opět do kořenového adresáře ve tvaru outputX.jpg, kde X značí pořadí vygenerovaného panoramatu. Číslování začíná od jedničky. Aby uživatel nemusel zadávat názvy obrázků ručně, je v souboru readme.txt vytvořen seznam obrázků obsažených ve složce images.

2. CD/DVD

Nosič má následující strukturu:

/images	testovací obrázky a spustitelný program
/source	zdrojové soubory programu
/text	zdrojové soubory pro L ^A T _E X
readme.txt	informace k projektu