VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

POKUS O NÁVRH NOVÉ METODY PRO STEREOVIDĚNÍ

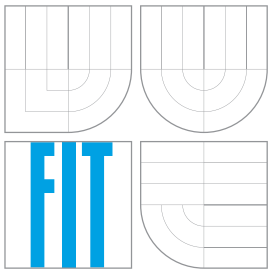
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JOSEF KOPEČNÝ

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

POKUS O NÁVRH NOVÉ METODY PRO STEREOVIDĚNÍ

THESIS TITLE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JOSEF KOPEČNÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. FILIP ORSÁG, Ph.D.

BRNO 2008

Abstrakt

Tato diplomová práce se zabývá problematikou fotogrammetrie. Popisuje nástroje, teoretické podklady pro postupy při získávání, předzpracování, segmentaci vstupního obrazu a pro výpočet hloubkové mapy. Hlavní náplní této práce je popis nové metody pro stereovidění. Její algoritmus, implementace a zhodnocení experimentů. Popisovaná metoda se řadí mezi metody založené na korelaci. Hlavní důraz je kladen na segmentaci s jejíž pomocí se počítá hloubková mapa.

Klíčová slova

Stereo vision, Photogrammetry, metody založené na korelaci, metody založené na příznacích, hloubková mapa, mapa disparity, konvoluce, korelace, diskrétní Fourierova transformace, rychlá Fourierova transformace, segmentace, metoda narůstání oblastí, invarianty, momenty, Sobelův operátor, OpenCV, FFTW, JMF.

Abstract

This thesis covers with the problems of photogrammetry. It describes the instruments, theoretical background and procedures of acquiring, preprocessing, segmentation of input images and of the depth map calculating. The main content of this thesis is the description of the new method of stereovision. Its algorithm, implementation and evaluation of experiments. The covered method belongs to correlation based methods. The main emphasis lies in the segmentation, which supports the depth map calculation.

Keywords

Stereo vision, Photogrammetry, Correlation-based methods, Feature-based Methods, depth map, disparity map, convolution, correlation, discrete Fourier transform, fast Fourier transform, segmentation, region Growing, invariants, moments, Sobel operator, OpenCV, FFTW, JMF.

Citace

Josef Kopečný: Pokus o návrh nové metody pro
stereovidění, diplomová práce, Brno, FIT VUT v Brně, 2008

Pokus o návrh nové metody pro stereovidění

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Filipa Orsága Ph.D.

.....
Josef Kopečný
15. května 2008

Poděkování

Rád bych poděkoval Ing. Filipu Orságovi Ph.D. za trpělivost při vedení mé diplomové práce.

Obsah

Kapitola 1

Úvod

Svět, ve kterém žijeme, je tří dimenzionální, ale obrazy zachycené pomocí fotografického přístroje jsou dvou dimenzionální. Jedna dimenze je ztracena během procesu zvaném perspektivní projekce. Proto jedním z důležitých úkolů počítačového vidění je znovunabytí třetí dimenze ze samostatného obrázku nebo z více obrázků. V tomto dokumentu nejprve vysvětlíme, jak člověk vnímá prostor pomocí jednoho oka (monokulární podněty), pojem perspektivní projekce, poté ukážeme vnímání pomocí obou očí a vysvětlíme základní metodu zabývající se zobrazením scény prostřednictvím dvou obrazů simulujících prostorový vjem. Odtud už je jenom krůček k fotogrammetrii (opak stereoskopie). Fotogrammetrie se snaží ze dvou snímků zjistit, jak vzdálené jsou objekty. V tomto dokumentu porovnáme dvě hlavní metody řešící problém fotogrammetrie. Prvním typem metod jsou metody založené na korelaci a druhým typem metod jsou metody založené na příznacích. K hlavním částem této diplomové práce patří kapitola vysvětlující všechny algoritmy, které byly použity při návrhu nové metody. Mezi tyto metody patří konvoluce, invarianty, momenty, korelace a další. Dále budou nastíněné možnosti získávání dat a odkud pochází testovací obrázky. Nyní jsme se dostali k samotnému problému celé práce a tím je návrh nové metody pro stereovidění. Bude zde popsán algoritmus a naznačeny podstatné informace o implementaci. Algoritmus je možné rozdělit do několika hlavních částí. Těmito částmi je předzpracování obrázků, segmentace, hledání korespondujících segmentů a výpočet disparitní mapy. Každá část algoritmu bude detailně popsána. Neméně důležitou částí dokumentu jsou experimenty s nově navrženou metodou. Experimenty je možné opět rozdělit do několika částí podobně jak tomu bylo u algoritmu. Závěrem celé práce je shrnutí dosažených výsledků a možná rozšíření a nedostatky algoritmu.

V Semestrálním projektu byla vypracována kapitola 2 a kapitola 3, na které diplomová práce navazuje.

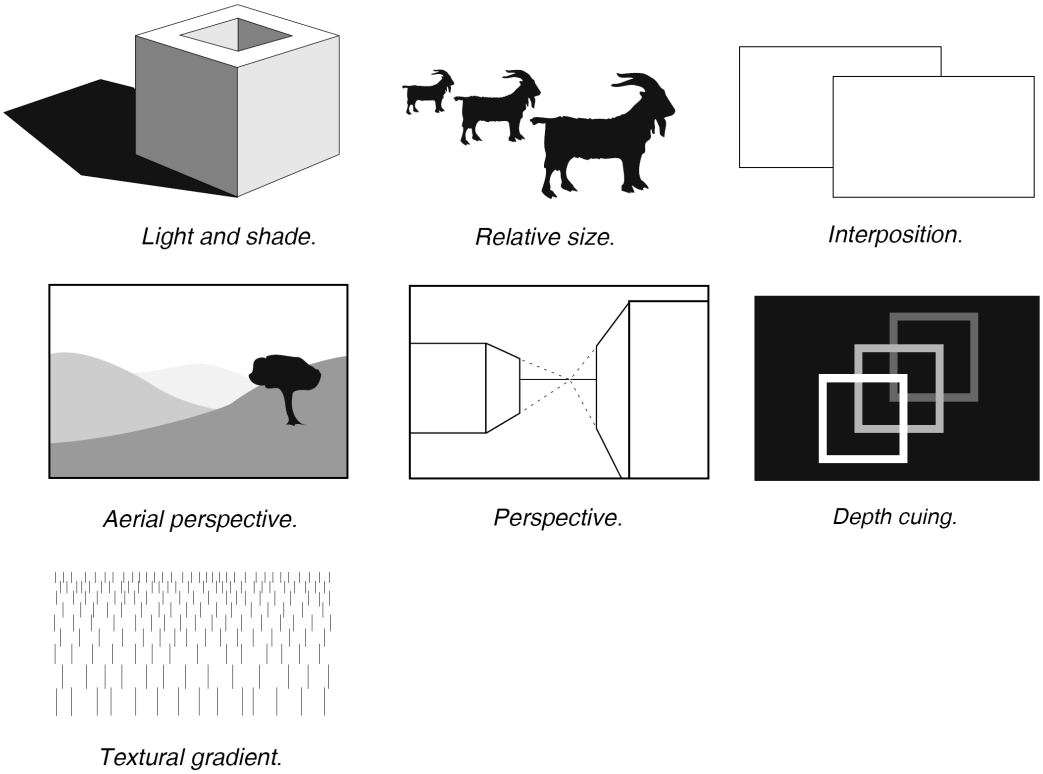
Kapitola 2

Vnímání prostoru

2.1 Vnímání jedním okem

Při vnímání jedním okem nevidíme prostorově, přesto jsme schopni odhadnout vzdálenosti předmětů a určit, který objekt leží před kterým. Faktory, díky nímž jsme toho schopni, se nazývají monokulární podněty.

2.1.1 Monokulární podněty

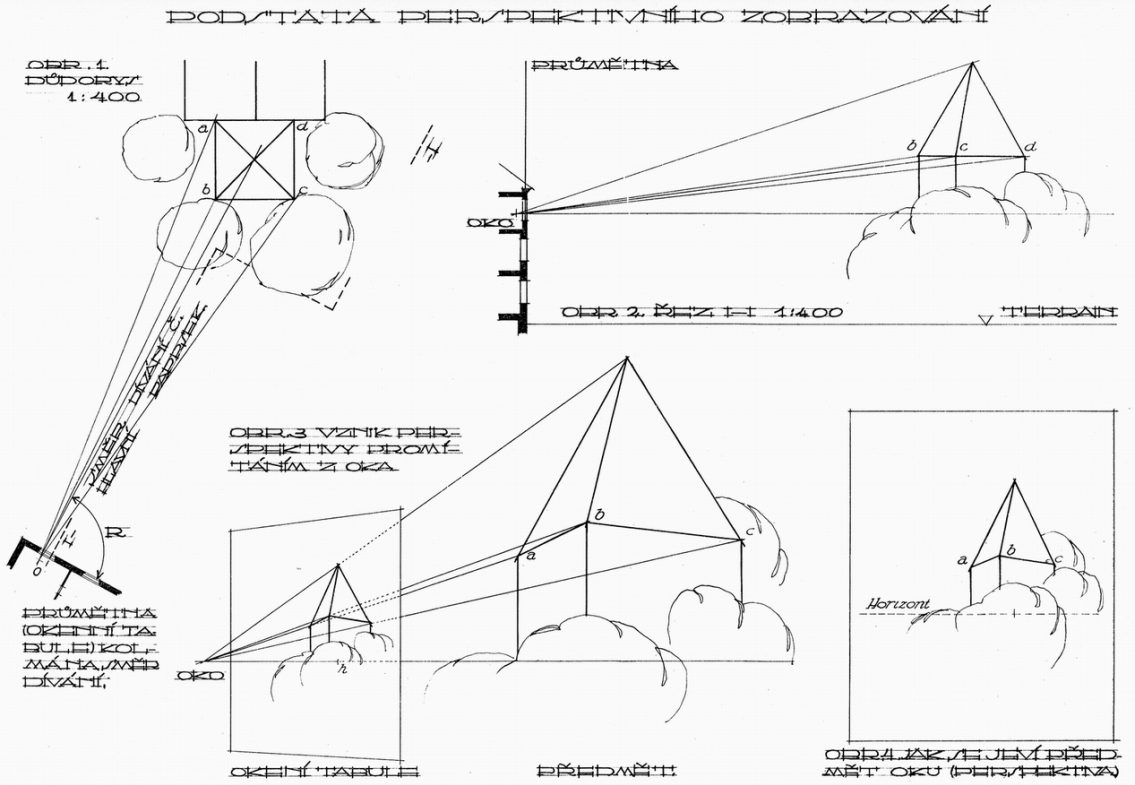


Obrázek 2.1: Monokulární podněty [2]

Jsou doplňující informace, které pomáhají vnímat prostor i jedním okem. [10]

Perspektiva je nejdůležitější hloubkový podnět nepřímo související s prostorovým vnímáním. Z výtvarného hlediska je vysvětlena perspektiva takto:

"Z latinského perspectus = průhled. Metoda zobrazení trojrozměrného prostoru v ploše. Přitom je v takto vytvořeném díle dosaženo iluze prostoru, v němž se rozměry směrem do dálky zmenšují a rovnoběžky na horizontu sbíhají. Perspektivní zákony známé už z antiky zdokonalili umělci renesance (Masaccio, P. Della Francesca, P. Uccello, A. Dürer). Tuto lineární perspektivu obohatili malíři 15. století (bratři Eyckové) o perspektivu barevnou (barvy studené vyvolávají dojem dálky, barvy teplé mají tendenci vystupovat do popředí)." [1]



Obrázek 2.2: Podstata perspektivního zobrazení. [9]

Skutečné předměty vidíme prostorově, protože se díváme oběma očima, zatímco perspektivní obraz odpovídá pohledu jedním okem. Oko vidí totéž, dívá-li se na předmět samo, nebo na obraz, který se s ním v pohledu úplně kryje. Perspektivní obraz vzniká protínáním paprsků, které jsou vedeny z oka k předmětu s nějakou průmětnou. Různá vzdálenost průmětny při stejné poloze oka má vliv pouze na velikost obrazu. Na obrázku 2.2 je ukázána podstata perspektivního zobrazení při pohledu jedním okem skrz okenní tabuli.

Světlo a stín poskytuje základní informaci o hloubce. Stíny informují o vzájemné pozici různých objektů. Světlejší objekty se jeví blíže než objekty tmavé.

Relativní velikost podle velikosti objektu jsme schopni určit vzdálenost. Bližší objekty se jeví větší než objekty vzdálenější.

Vložené objekty jsou objekty, které leží mezi pozorovaným objektem a námi.

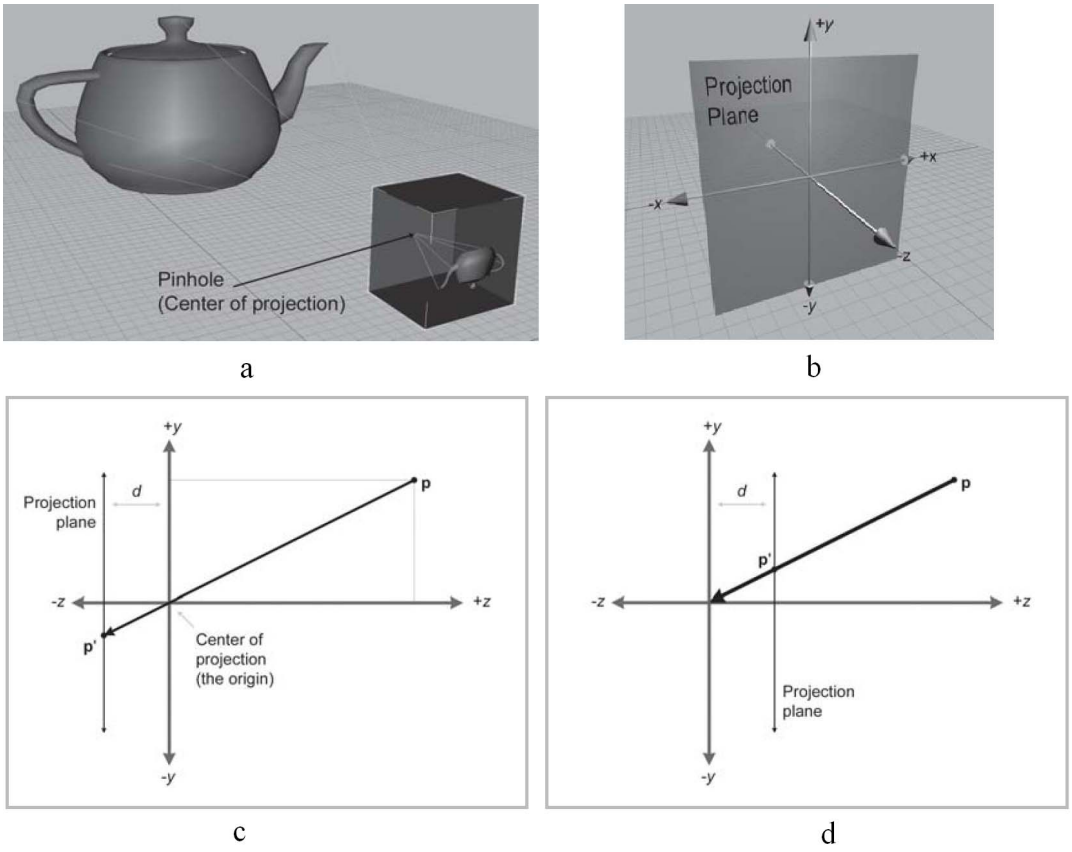
Gradient textury materiály ve formě textury (například trávník, rozlehlý les ...) poskytují informaci o hloubce, protože textura je více patrná u bližších objektů.

Barevné zkreslení je snížení viditelnosti vzdálených objektů způsobených ležícím oparem. Velmi vzdálené objekty získávají často modravý nádech způsobený rozptylem červeného světla.

Pohybová paralaxa je informace o hloubce poskytovaná pohybujícím se obrazem (buď svět pohybující se kolem nás nebo my při pohybu). Pohybové paralaxy lze dosáhnout například rotací objektu. Při pohybu se objekty blízko nás pohybují mnohem rychleji než objekty vzdálené.

Jas objekty vzdálenější jsou méně jasné než objekty bližší. Na toto má vliv počasí, ale i vzdálenost.

2.1.2 Perspektivní promítání



Obrázek 2.3: *a* - dírková kamera, *b* - definování os vůči projekční rovině, *c* - pohled na scénu *a* z boku, *d* - projekční rovina je před dírkou [5]

Perspektivní promítání zobrazuje 3D svět na 2D plochu. Je odstraněna souřadnice hloubky, která je nahrazena perspektivou. Nejlépe se perspektivní promítání ukazuje na dírkové kameře [5] (obrázek 2.3 a).

Dírková kamera

Paprsek vstupuje otvorem a zasahuje protější stěnu krabice, která se tak stává projekční rovinou (obrázek 2.3 a). Obraz je zde, ale převrácen, jelikož střed projekce je v otvoru (paprsky světla se zde kříží).

Nyní popíšeme geometrii perspektivní projekce pro dírkovou kameru. Obrázek 2.3 b ukazuje, jak umístíme osy vzhledem k projekční rovině. Abychom mohli spočítat projekci libovolného bodu $\vec{p}$ skrz otvor na bod $\vec{p'}$ projekční roviny, potřebujeme znát vzdálenost otvoru od projekční roviny. Tuto vzdálenost budeme značit d. V našem případě (obrázek 2.3 c) platí $z = -d$. Nyní se zaměříme na výpočet y - souřadnice bodu $\vec{p'}$. Pomocí podobnosti trojúhelníků dostaneme vztah: [5]

$$\frac{-\vec{p'_y}}{d} = \frac{\vec{p_y}}{z} \Rightarrow \vec{p'_y} = \frac{-d\vec{p_y}}{z} \tag{2.1}$$

Výpočet $\vec{p'_x}$ je totožný

$$\vec{p'_x} = \frac{-d\vec{p_x}}{z} \tag{2.2}$$

$$\vec{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \vec{p'} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \frac{-dx}{z} \\ \frac{-dy}{z} \\ -d \end{bmatrix} \tag{2.3}$$

Pro případ, že projekční rovina je před otvorem (obrázek 2.3 d) a místo otvoru si představíme oko a místo projekční roviny například okno (jak bylo ukázáno na obrázku 2.2), platí vztah: [5]

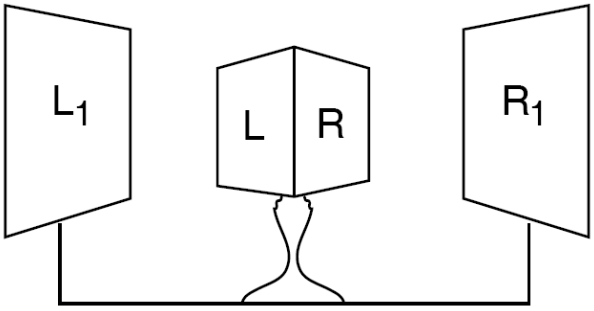
$$\vec{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \vec{p'} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \frac{dx}{z} \\ \frac{dy}{z} \\ d \end{bmatrix} \tag{2.4}$$

2.2 Vnímání prostoru dvěma očima

Prostorově vidíme proto, že máme oči umístěné několik centimetrů od sebe (u dospělých jedinců cca 64 mm). Levé oko vidí svět z trochu jiného úhlu než oko pravé. Naše mysl porovnává to, co vidí levé oko, s obrazem, který vidí oko pravé, určí rozdíly mezi oběma perspektivami a ty pak interpretuje jako hloubku.

2.2.1 Historie

V roce 1838 sir Charles Wheatstone ukázal, že existuje unikátní způsob vnímaní hloubky, a to tak, že nakreslil na listy papíru jednoduché obrazce a pozoroval je pomocí vynálezu složeného ze zrcadel a posuvných desek (obrázek 2.4). Nakreslil dva obrazce znázorňující dva různé pohledy, jak je vidí naše oči, pozorují-li nějaký předmět. Když se díval na tyto obrazce skrz svůj vynález, zrcadla napomohla vizuálnímu splynutí dvou velkých kreseb, a tak viděl obrázky trojrozměrně.

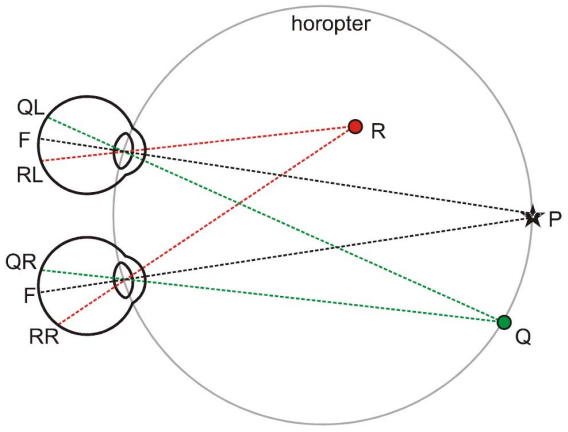


Obrázek 2.4: Wheatstoneův stereoskop - nos člověka je umístěn na spojnici zrcadel L a R a oči vnímají obrazy L_1 a R_1. Výsledkem je jediná prostorová scéna (jsou-li obrazy správně vytvořené). [10]

2.2.2 Binokulární podněty

Tak jako monokulární podněty pomáhají vnímat hloubku prostoru jedním okem, binokulární podněty pomáhají vnímat hloubku oběma očima.

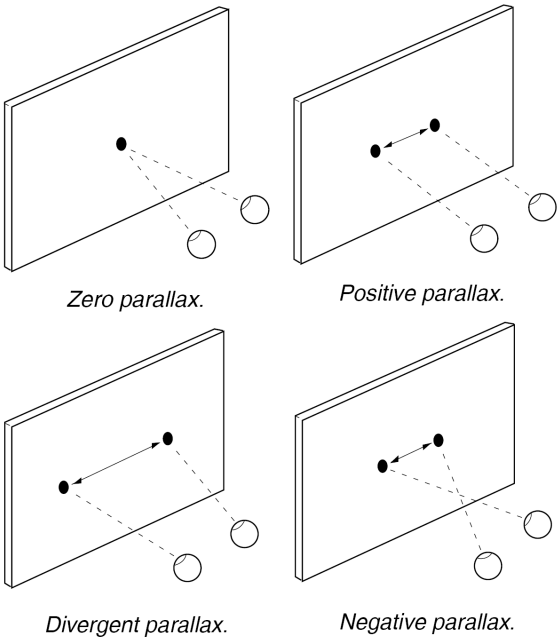
Binokulární disparita [8] Binokulární disparita je klíčovým binokulárním vodítkem pro prostorové vidění ve středních vzdálenostech. Na obrázku 2.5 je pohled zaměřen na bod P, jehož obraz dopadá do středu sítnice F. Bod Q ležící v jiném místě prostoru má příslušné obrazy QL a QR. Tyto dva body jsou od bodu F stejně vzdáleny, mají nulovou disparitu a bod Q vnímáme stejně vzdálený jako P. Říkáme, že všechny body stejně vzdálené jako P leží na horopteru a vidíme je ve stejné "hloubce". Tvar horopteru záleží na našem mozku, na tom, jak vnímáme vzdálenosti, není to rovina, ani kulová plocha [3]. Sítnicová disparita je způsobena rozestupem očí.



Obrázek 2.5: Binokulární disparita

Paralaxa (obrázek 2.6) je horizontální vzdálenost mezi souhlasnými body levého a pravého

obrazu [8]. Je měřena jako vzdálenost na promítací rovině a ne na sítnici. Pocit hloubky je možné vytvořit i pokud nejsou použitá žádná jiná prostorová vodítka.



Obrázek 2.6: Základní typy paralax: vlevo nahoře - nulová paralaxa, vpravo nahoře pozitivní paralaxa, vlevo dole - divergentní paralaxa, vpravo dole - negativní paralaxa.

Nulová paralaxa pokud souhlasné body levého a pravého obrazu jsou totožné, vytvářejí nulovou paralaxu. Při stereoskopické projekci se tyto body zdají být v rovině promítací plochy.

Pozitivní paralaxa při sledování objektu s pozitivní paralaxou se tento objekt zdá být za rovinou promítání.

Divergentní paralaxa v reálném světě tento případ nenastává, protože optické osy se při divergentní paralaxe vzdalují.

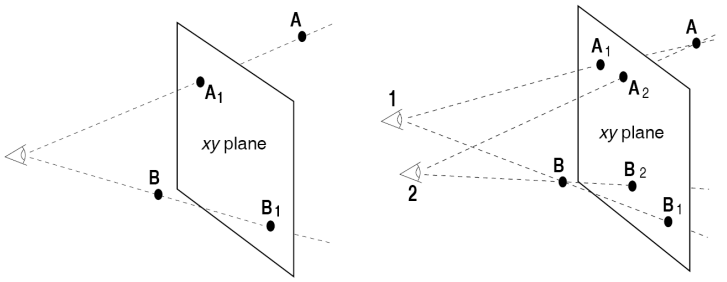
Negativní paralaxa nastává při překřížení optických os. Objekty s touto paralaxou se zdají být před promítací rovinou.

2.2.3 Stereo perspektivní promítání

Některé metody vytvářející dojem prostoru:

Autostereogram (obrázek 2.8 a) nejjednodušším typem autostereogramu je horizontálně opakující se vzor. Tyto autostereogramy se nazývají wallpaper autostereogram. Prostor je zde vytvořen změnou vzdálenosti mezi vzory. Dalším typem autostereogramu je random dot autostereogram, který je generován z výškové mapy. Pro každý bod výškové mapy jsou vygenerovány 2 souhlasné body (obrázek 2.7).

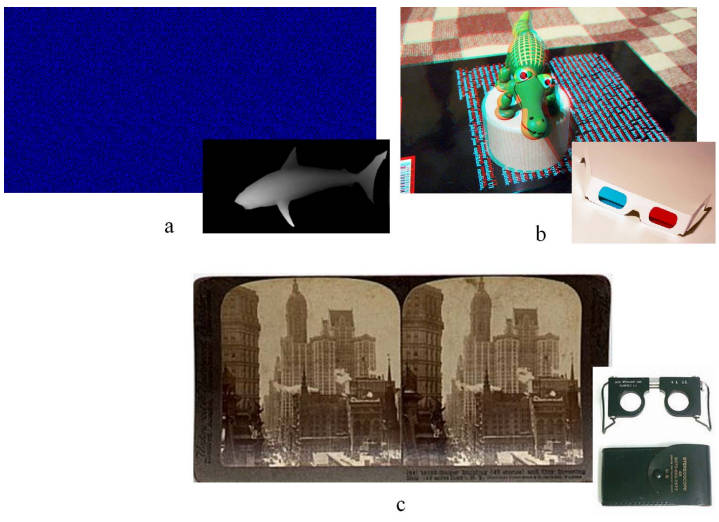
Anaglyph (obrázek 2.8 b) je vytvořen ze dvou obrázků, které jsou vůči sobě posunuté. Například při fotografování se druhý (pravý) snímaný obrázek získá tak, že se fotoaparát posune o pár centimetrů doprava. Rozdíly obrázků jsou vyjádřeny barvou



Obrázek 2.7: Průmět bodů A a B do roviny zobrazení pro jedno oko (vlevo) a pro dvě oči (vpravo). Průmět do roviny respektuje zákonitosti prostorového vnímaní.

(nejčastěji červená-modrozelená, ale používají se i jiné barvy). Pro prohlížení jsou zapotřebí brýle s barevnými filtry, které způsobují, že každé oko vidí jiný obrázek a tím je vytvořen prostorový dojem. Nevýhodou je, že přicházíme touto metodou o realistické barvy.

Stereographic card (obrázek 2.8 c) je obdoba Anaglyph, s tím rozdílem, že dva obrázky nejsou v sobě zakódovány barvou, ale jsou vytisknuté vedle sebe. K prohlížení se používají speciální brýle (stereoskop).



Obrázek 2.8: a - Autostereogram (random dot) obrázku žraloka s hloubkovou mapou, b - Anaglyph a brýle na prohlížení, c - Stereographic card a stereoscope na prohlížení [17]

Kapitola 3

Fotogrammetrie

"Fotogrammetrie je způsob získání informace o prostorové struktuře a vzdálenosti scény ze dvou nebo více obrázků pořízeních z rozdílných pozic". [22] Fotogrammetrie má široké uplatnění. Využívá se například v průmyslu, robotice, k tvorbě prostorových map a v armádě.

Fotogrammetrie se dá rozdělit na dvě části:

Problém korespondence [15] určuje, který bod v levém obrázku koresponduje s bodem pravého obrázku (obrázek 3.1 a). Tento úkol je nejsložitější částí fotogrammetrie. Úkol dále ztěžuje fakt, že některé body nemusí být na obou obrázcích viditelné.

Metody založené na korelaci používají informace o rozložení jasových úrovní snímků, přímo k nalezení korespondence mezi párem bodů. Elementy, mezi kterými je hledaná shoda výpočtem korelace, jsou tvořeny okénky bodů ve dvou snímcích.

Metody založené na příznacích srovnávají určité vlastnosti a rysy vyčleněné z páru snímků, například hrany, čáry, vrcholy a další běžné tvary. Během tohoto postupu lze nalézt nejlepší shodu mezi párem snímků. Obvykle metoda "token detection" nachází korespondenci s větší přesností.

Hybridní algoritmy spojují obě předchozí metody.

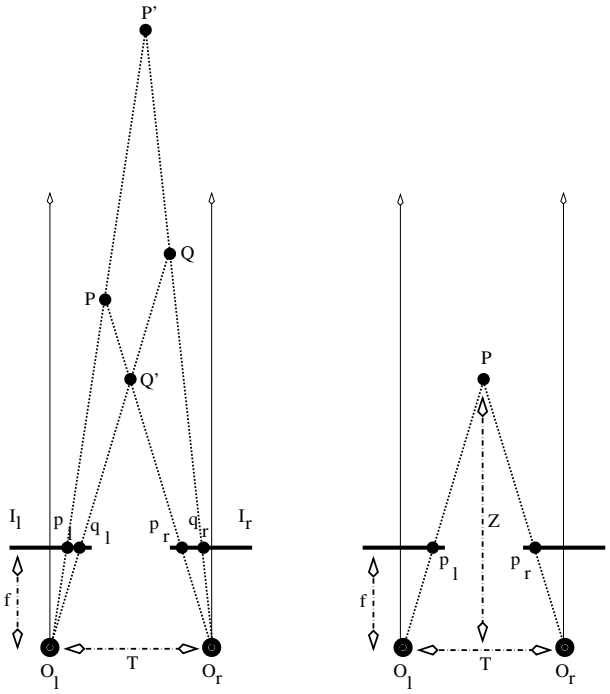
Problém rekonstrukce z disparity korespondujících bodů je možné pomocí triangulace vypočítat vzdálenost scény (obrázek 3.1).

3.1 Korespondence

3.1.1 Epipolární geometrie

Pro určení, kde se budou nalézat korespondující body pozorovaného bodu P na obrazových rovinách, lze využít epipolární geometrie (obrázek 3.2). Body C_1 a C_2 jsou kamery a bod P je pozorovaný bod. Při pohybu bodu P v prostoru scény se pohybuje i epipolární rovina, a to tak, že rotuje okolo úsečky C_1C_2. Epipolární linie jsou průsečíky obrazových rovin a epipolární roviny. Bod v epipolární rovině se promítá do obrazu na epipolární linie. Díky tomuto faktu se problém hledání korespondujícího bodu zužuje na hledání epipolární linie.

Pokud je vzdálenost mezi kamerami malá, je hledání korespondujících bodů snazší, naproti tomu při větším vzdálenosti kamer je hledání korespondujících bodů obtížnější. Přesnost závisí na rozestupu kamer a na úhlu, který svírají. S rostoucí vzdáleností a úhlem se přesnost zvyšuje.



Obrázek 3.1: Jednoduchý stereo systém. a - 3D rekonstrukce hloubky založená na řešení problému korespondence, b - hloubka je odhadnutá ze vzdálenosti korespondujících bodů [22]

3.1.2 Metody založené na korelaci

Tato metoda [12] je časově méně náročná než ostatní metody. Obraz je popsán obrazovou funkcí dvou souřadnic $g(x, y)$ v obrazové rovině. Obor hodnot obrazové funkce (jasu) je omezený. Nejnižší hodnota odpovídá v monochromatickém obrazu černé a nejvyšší bíle. Mezilehlé hodnoty odpovídají různým stupňům šedi.

Nechť x_0 a y_0 jsou složky posunutí (disparity) mezi dvěma korespondujícími body v páru snímků. Nechť $g(x, y)$ vyjadřuje rozložení stupňů šedi v pravém snímku v maticovém okolí $(n \times m)$ bodu (x, y). Pak $\tilde{g}(x + x_0, y + y_0)$ lze definovat jako funkci C

$$C = C(g(x, y), \tilde{g}(x + x_0, y + y_0)) \tag{3.1}$$

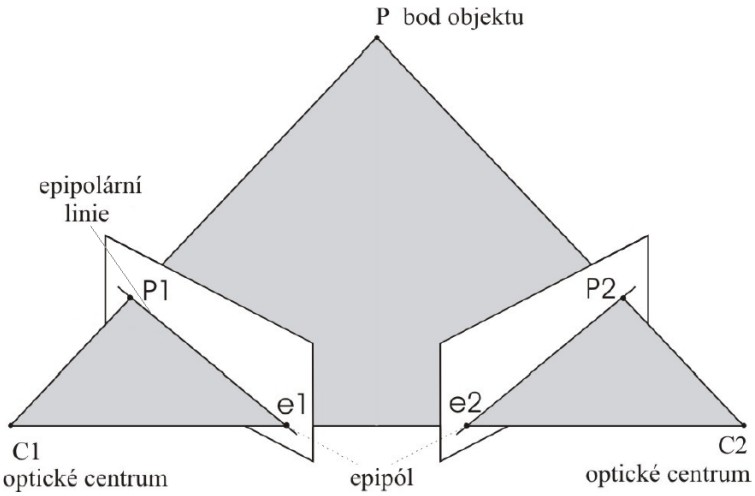
kde $g(x, y)$ a $\tilde{g}(x + x_0, y + y_0)$ jsou rozložení úrovní šedi snímků v okolí bodů (x, y) a $(x + x_0, y + y_0)$ ohraničených obdélníkem o počtu $(n \times m)$ bodů. C znamená vzájemný vztah definující míru korelace mezi g a $\tilde{g}$.

Je-li pro stanovení podobnosti mezi g a $\tilde{g}$ použito Euklidovské míry (normy), je formální zápis diskrétní korelační funkce $S(x_0, y_0)$

$$C(x_0, y_0) = \sum_{i}^{n} \sum_{j}^{m} |g_{ij}(x, y) - \tilde{g}_{ij}(x + x_0, y + y_0)| \tag{3.2}$$

po normalizaci lze diskrétní korelační funkci zapsat

$$C(x_0, y_0) = \frac{\sum_i^n \sum_j^m g_{ij}(x, y)\tilde{g}_{ij}(x + x_0, y + y_0)}{\sqrt{\sum_i^n \sum_j^m g_{ij}^2(x, y) \sum_i^n \sum_j^m \tilde{g}_{ij}^2(x + x_0, y + y_0)}} \tag{3.3}$$



Obrázek 3.2: Epipolární geometrie [12]

Uvažovaná oblast $(n \times m)$ v kterékoli z těchto diskrétních korelačních funkcí je podmnožinou šedých stupňů vybranou ze dvou snímků. Bod $(x + x_0, y + y_0)$, který koresponduje s bodem (x, y) v prvním snímku, je získán nalezením maximální hodnoty zmíněné korelační funkce (3.3). Disparita mezi prvním a druhým snímkem je (x_0, y_0). Z této hodnoty je vypočtena potřebná souřadnice Z.

Hledání korespondujících bodů neprobíhá v celé ploše snímků, ale pouze ve vybraném výřezu.

Okénkový algoritmus používá v levém snímku stojací okénko velikosti $(n \times m)$ a bod k němuž se hledá odpovídající na pravém snímku, je ve středu tohoto okénka. V pravém snímku je okénko pohyblivé a posouvá se po řádcích v celém výřezu s přírůstkem souřadnic jeden bod. V každé pozici je stanovena míra korespondence mezi stojacím a pohyblivým okénkem hodnotou korelační funkce.

3.1.3 Metody založené na příznacích

Vstupem je pár stereoobázků I_l a I_r a dvě korespondující množiny popisujících znaků. Nechť $R(f_l)$ je hledaný region v pravém obrázku asociovaný s popisujícími znaky f_l a $d(f_l, f_r)$ je disparita mezi korespondujícími znaky f_l a f_r.

Pro každý f_l v levém obrázku:

- vypočítáme podobnost mezi f_l a každým znakem v $R(f_l)$
- vybereme znak pravého obrázku, f_r, který je nejvíce podobný
- uložíme shodu a disparitu f_l

Výstupem je seznam korespondujících si znaků a mapa disparity. [22]

Algoritmus	hodnocení
AdaptingBP	1.9
DoubleBP	2.4
SymBP+occ	5.6
Segm+visib	6.0
C-SemiGlob	6.8
RegionTreeDP	7.7

Tabulka 3.1: Tabulka metod

3.2 Rekonstrukce

Při rekonstrukci vypočítáváme z nalezených korespondujících bodů hloubku scény, a to s využitím triangulace. K výpočtu budeme potřebovat znát korespondující body p_1 a p_2, meziosovou separaci T (vzdálenost mezi soustavami čoček) a ohniskovou vzdálenost f.

3.2.1 Triangulace

Obrázek 3.1 zobrazuje vrchní pohled stereo systému složeného ze dvou kamer. Kamery jsou položeny rovnoběžně vedle sebe ve vzdálenosti T a jsou zaměřeny na nekonečně vzdálený bod. Levý a pravý obraz je reprezentován segmenty I_l, I_r a O_l, O_r reprezentují středy projekce.

Pro zjištěné vzdálenosti bodu P potřebujeme korespondující body. Ty získáme pomocí metod popsaných výše. Mějme tedy dva korespondující body p_l a p_r. Bod P získáme tak, že vyšleme paprsek z bodů O_l a O_r. Paprsek vycházející z O_l protíná bod p_l a pokračuje dále, dokud se neprotne s paprskem vycházejícím z bodu O_r, který prochází bodem p_r. Na místě, na kterém se střetly, se nachází hledaný bod P. V případě, že špatně určíme korespondující body, může být výsledek velmi odlišný. Například, pokud místo p_r vyhodnotíme, že korespondujícím bodem je q_r, dostaneme velmi odlišný výsledek, a to bod P' (obrázek 3.1).

Nechť x_l souřadnice pro bod p_l a x_r je souřadnice pro bod p_r vztahující se k c_l a c_r. Potom platí vztah: [22]

$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z} \tag{3.4}$$

po úpravě dostaneme: [22]

$$Z = f\frac{T}{x_r - x_l} \tag{3.5}$$

3.3 Reálné metody fotogrammetrie

Existují internetové stránky [20] zabývající se srovnáním metod fotogrammetrie. Je zde uveden seznam různých metod a jejich porovnání. Porovnává se, kolik bodů určily metody špatně. Tabulka 3.1 vypisuje několik metod a ukazuje výsledky měření. Čím menší číslo, tím je metoda přesnější. Všechny metody se testovaly na stejných datech a stejným způsobem.

Kapitola 4

Použité algoritmy

Nejprve zde popíšeme již existující algoritmy s jejichž pomocí je algoritmus vybudován. Na tyto bude odkazováno v následujících kapitolách a bude vysvětleno jejich konkrétní použití.

Navržený algoritmus lze rozdělit do 4 částí:

1. předzpracování obrázků
2. segmentace
3. hledání korespondujících oblastí
4. s využitím informace o podobných oblastech hledaní korespondujících si bodů

4.1 Předzpracování

4.1.1 Konvoluce

Konvoluce je matematický způsob kombinace dvou signálu, čímž získáme výsledný třetí signál. Konvoluce je široce používaná technika při zpracování obrazu, patří k nejdůležitějším technikám v digitálním zpracování signálu (DSP) a je důležitá i v dalších oblastech. [4] [16]

Delta funkce

V DSP technice se *delta funkce* ($\delta[n]$) definuje jako normalizovaný impuls, který má ve vzorku 0 hodnotu 1, zatímco ostatní vzorky mají hodnotu 0. Delta funkce je také zvaná *jednotkový impulz*. Libovolný impulz může být reprezentován jako posunutá a škálovaná *delta funkce*. [4] [16]

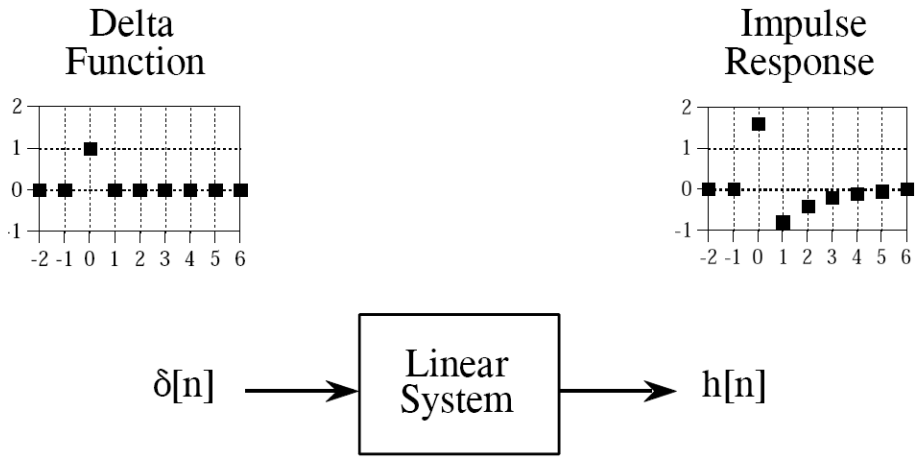
Konvoluce v DSP

Získání výstupního signálu ze vstupního: [16]

1. Vstupní signál je rozložen na množinu impulzů. Každý impulz je posunutá a škálovaná *delta funkce*.
2. Výsledný výstup získaný z každého impulzu je posunutá a škálovaná verze *impulzní odezvy*.
3. Celkový výstupní signál je získán přidáním posunuté a škálované *impulzní odezvy*($h[n]$).

4. Znalost impulzní odezvy systému umožňuje stanovit výstup pro libovolný vstupní signál

5. Pokud je lineární systém koncipován jako *filtr*, potom se *Impulzní odezva* nazývá *filtr*, *konvoluční jádro* nebo jednoduše *jádro*.



Obrázek 4.1: Definice *delta funkce* a *impulzní odezvy*. [16]

Při konvoluci ze dvou signálů získáme třetí signál. V lineárních systémech se konvoluce využívá k popsání vztahů mezi vstupním signálem, impulzní odezvou a výstupním signálem. Obrázek 4.3 ukazuje vstup vstupního signálu $x[n]$ do lineárního systému s *impulzní odezvou* $h[n]$. Výsledkem je výstupní signál $y[n]$. Formálně zapsáno: [16]

$$x[n] * h[n] = y[n] \tag{4.1}$$

Konvoluce se značí znakem $*$.
Matematický zápis výpočtu výstupního signálu: [16]

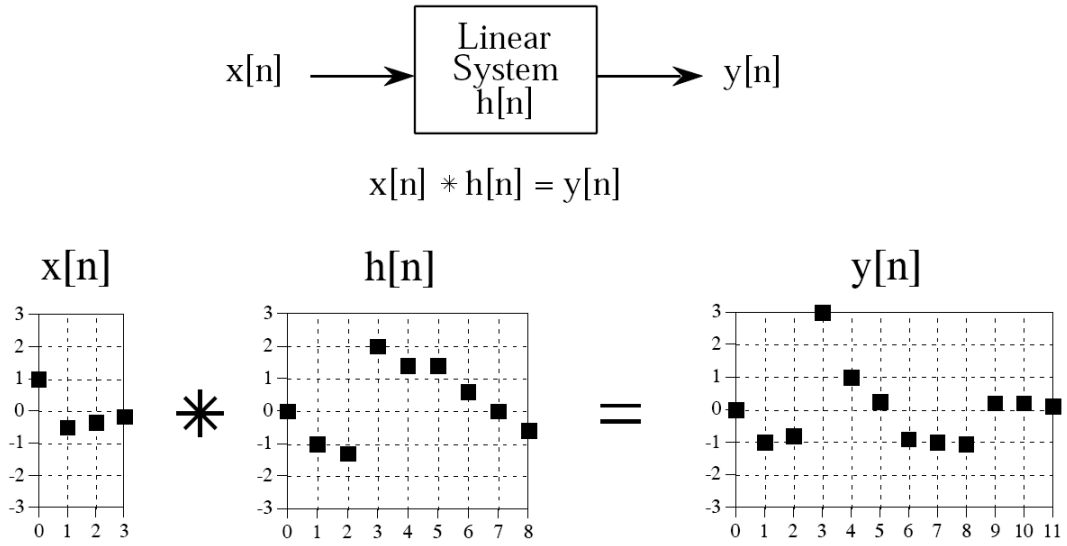
$$y[i] = \sum_{j=0}^{M-1} h[j]x[i-j] \tag{4.2}$$

Kde $x[n]$ je vstupní signál složeny z N vzorků. První vzorek má index 0 a poslední $N-1$. $h[n]$ je konvoluční jádro o M vzorcích s indexy od 0 do $M-1$. Výsledný signál bude obsahovat $N+M-1$ vzorků indexovaných od 0 do $N+M-2$. Každý výstupní vzorek je počítán nezávisle vůči ostatním výstupním vzorkům. Index i určuje, který vzorek výstupního signálu je počítán.

Vztah 4.2 se nazývá *konvoluční suma*.

Konvoluce obrazu

$$y[r,c] = \sum_{k=0}^{M-1}\sum_{j=0}^{M-1} h[k,j]x[r-k,c-j] \tag{4.3}$$



Obrázek 4.2: Příklad konvoluce vstupního signálu ($x[n]$) s lineárním systémem ($h[n]$). [16]

Vztah 4.3 popisuje výpočet konvoluce obrazu pro výstupní bod $y[r, c]$. $x[,]$ je vstupní obraz a $h[,]$ je konvoluční jádro. Velikost konvolučního jádra je $M \times M$ bodů s indexy v rozsahu od 0 do $M - 1x$

Konvoluční jádro bývá málokdy větší než 15 bodový čtverec. Nejčastější rozměr je 3x3 bodů.

V případě diskrétní konvoluce můžeme říct, že konvoluční masku položíme na příslušné místo obrazu a každý bod obrazu, který se nachází pod konvoluční maskou vynásobíme koeficientem v příslušné buňce konvoluční masky a poté provedeme součet všech těchto hodnot. Tímto získáme jeden nový bod. [4]

Pomocí konvolučních masek lze definovat filtry provádějící:

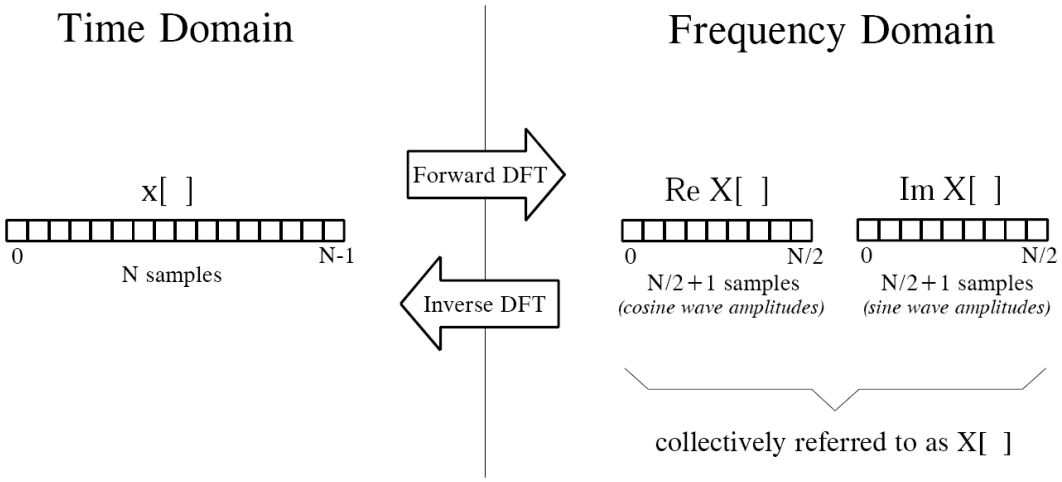
- vyhlazování
- doostřování
- detekce hran

4.1.2 FFT konvoluce

Diskrétní Fourierova transformace (DFT)

Diskrétní Fourierova transformace [16] převádí N vzorkový vstupní signál na dva $N/2 + 1$ vzorkové signály (Obrázek 4.3). Jinak řečeno, *DFT* převádí signál z *časové oblasti* do *oblasti frekvenční*. *Frekvenční oblast* obsahuje tytéž informace jako *oblast časová*, pouze v rozdílné formě. *Frekvenční oblast* popisuje vstupní signál pomocí kosinusových ($ReX[]$) a sinusových ($ImX[]$) amplitud. Z jedné oblasti je možné převádět data do druhé. Pokud převádíme z *časové oblasti* do *oblasti frekvenční*, nazývá se proces převodu *DFT*. Při převodu opačném (z frekvenční oblasti do oblasti časové), nazýváme proces *inverzní DFT*.

Při převodu signálu z časové oblasti do oblasti frekvenční je vstupní signál o šířce N vzorků je převeden na $N/2 + 1$ sinusových a kosinusových vln, přičemž frekvence každé



Obrázek 4.3: Převod mezi časovou oblastí a frekvenční oblastí [16]

sinusové a kosinusové vlny jsou přesně dány. První sinusová a kosinusová vlna má 0 kompletních cyklů za N vzorků, druhá sinusová a kosinusová vlna má 1 kompletní cyklus za N vzorků, ... , $N/2+1$ sinusová a kosinusová vlna má $N/2$ kompletních cyklů za N vzorků.

Následující vztahy (4.4, 4.5) vypočítají reálnou ($ReX[k]$) a imaginární ($ImX[k]$) část frekvenční oblasti pro zvolenou frekvenci k (0 až $N/2$) ze vstupního signálu x, který je složen z N vzorků.

$$ReX[k] = \sum_{i=0}^{N-1} x[i]\cos(2\pi ki/N) \tag{4.4}$$

$$ImX[k] = -\sum_{i=0}^{N-1} x[i]\sin(2\pi ki/N) \tag{4.5}$$

Inverzní DFT

Inverzní DFT [16] představuje opačný problém oproti DFT. Z frekvenční oblasti se data převádějí do oblasti časové. $x[i]$ ve vztahu 4.6 říká, že budeme počítat i-tý vzorek signálu v časové oblasti. i může být v rozsahu 0 až $N/2$.

$$x[i] = \sum_{k=0}^{N/2} Re\overline{X}[k]\cos(2\pi ki/N) + \sum_{k=0}^{N/2} Im\overline{X}[k]\sin(2\pi ki/N) \tag{4.6}$$

Vztahy 4.7, 4.8, 4.10, 4.11 jsou pro konverzi mezi sinusovými amplitudami a hodnotou ve frekvenční oblasti. $Re\overline{X}[k]$ a $Im\overline{X}[k]$ obsahuje amplitudy kosinusových a sinusových vln. k nabývá hodnot od 0 do $N/2$

$$Re\overline{X}[k] = \frac{ReX[k]}{N/2} \tag{4.7}$$

$$Im\overline{X}[k] = -\frac{ImX[k]}{N/2} \tag{4.8}$$

$$\tag{4.9}$$

Speciální případ:

$$Re\overline{X}[0] = \frac{ReX[0]}{N} \tag{4.10}$$

$$Re\overline{X}[N/2] = \frac{ReX[N/2]}{N} \tag{4.11}$$

Rychlá Fourierova transformace (FFT)

FFT (Fast Fourier Transform) [16] je chytrý algoritmus pro rychlý výpočet DFT. FFT rozloží DFT s N vzorky na N DFT s jedním vzorkem. FFT je standardně mnohonásobně rychlejší než ostatní metody.

Konvoluce ve frekvenční oblasti

Jedním ze způsobů, jak vypočítat konvoluci, je převést dva signály do frekvenční oblasti, vynásobit je a poté převést výsledek do časové oblasti. Konvoluce je nahrazena dvěma DFT, násobením a inverzní DFT.

Vztah pro násobení dvou signálů $X[f]$ a $H[f]$ ve frekvenční oblasti [16]:

$$ReY[f] = ReX[f]ReH[f] - ImX[f]ImH[f] \tag{4.12}$$

$$ImY[f] = ImX[f]ReH[f] + ReX[f]ImH[f] \tag{4.13}$$

FFT konvoluce obrazu

FFT konvoluce [16] poskytuje stejný výstupní obraz, jako klasická konvoluce(vztah4.3). Čas výpočtu FFT konvoluce nezáleží na velikosti jádra na rozdíl od klasické konvoluce, kde je čas výpočtu velice ovlivněn velikostí konvolučního jádra. Výpočetní čas FFT konvoluce je $N^2 \log_2(N)$, zatím co pro klasickou konvoluci je N^2M^2, pro obrázek o rozměru NxN bodů a konvolučním jádrem o rozměru MxM bodů. FFT konvoluci je výhodné použít pokud konvoluční jádro je větší jak 30x30 bodů. [16]

4.1.3 Sobelův operátor

Sobelův operátor [19] je diskrétní diferenční operátor počítající aproximaci gradientu funkce obrazu. Nejběžněji se používá při detekci hran v obraze nebo při jeho ostření. Sobelův operátor je založen na konvoluci obrazu s konvolučním jádrem. Konvoluční jádra mohou být pro všech osm směrů. Nejčastější konvoluční jádra jsou pro detekci hran v horizontálním a vertikálním směru:

$$h_x = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \quad h_y = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \tag{4.14}$$

4.2 Segmentace

4.2.1 Obecná definice segmentace

Segmentace obrazu $I(x, y)$ je jeho dělení na pod obrazy (též. regiony) $R_1, R_2, ..., R_N$ takové, že: [11]

$$\bigcup_{n=1}^{N} R_i = I(x, y) \tag{4.15}$$

Pro regiony platí, že se nepřekrývají ($\forall i \neq j, R_i \cup R_j = \emptyset$). Každý bod v regionu R splňuje nějaké tvrzení (podobná úroveň šedi/barvy ...). Segmentaci můžeme rozdělit na:

- Úplná segmentace - jednotlivé regiony korespondují s reálnými objekty scény
- Částečná segmentace - obraz je rozdělen na významné části

Při segmentaci se využívá znalost řešeného problému. Obecně je možné říct, že vybíráme takovou metodu segmentace, aby její výstup splňoval podmínky pro následující úpravu obrazu a aby bylo zachováno co nejvíce informací z původního obrazu. Segmentace je jeden z nejdůležitějších kroků analýzy obrazu, protože vyšší algoritmy pro zpracování obrazu často využívají informace získané segmentací. Proto segmentace velkou měrou ovlivňuje i výsledek hledání hloubky objektu v obraze při fotogrammetrii. Mezi nejběžnější metody segmentace patří: [21]

- Prahování založené na histogramu (Histogram-based thresholding)
- Metoda narůstání oblastí (Region growing)
- Metoda spojování regionů (Region merging)
- Metoda dělení regionů (Region splitting)
- Clustering/klasifikace (Clustering/Classification)
- Metody založené na teorii grafů (Graph theoretic approach)
- Znalostní metody (Knowledge-based)

4.2.2 Problémy segmentace

Metoda segmentace obrazu se musí vypořádat s obrazem, který je deformován šumem obsahujícím nehomogenní osvětlení objektů a překrýváním objektů. Z tohoto důvodu je úplná segmentace velice složitý proces.

4.2.3 Zvolená segmentace obrázků

Algoritmus použitý k segmentaci je založen na metodě semínkového vyplňování rastrových oblastí.

Semínkové vyplňování

Semínkové vyplňování (Flood-fill) [6] [14] je základní metoda v počítačové grafice k vyplňování oblastí. Oblast je definovaná hranicí. Jsou 2 způsoby definice hranice oblast:

1. Hraniční - Oblast je definována spojitou hranicí bodů dané barvy
2. Záplavová - Oblast je definována spojitou množinou vnitřních bodů dané barvy

Princip metody spočívá ve vložení semínka do oblasti, které obarví své okolí. Okolí semínka může být 4-okolí nebo 8-okolí. Nově obarvené body se stávají semínky a obarvují své okolí, dokud nenarazí na hranici oblasti.

4.2.4 Metoda narůstání oblastí

Metoda narůstání oblastí (Region Growing) [11] [4] [21] je založena na semínkovém vyplňování oblastí se záplavovou definicí hranice oblasti. Rozdíl spočívá v obarvování bodů ne jenom se stejnou barvou, ale obarvují se i podobné body (body oblasti splňují zadané kriterium homogenity). Prvotní semínko bývá náhodně umístěno do obrazu. Obdobně jako u semínkového vyplňování oblastí se mohou semínka rozrůstat s využitím 4-okolí nebo 8-okolí. Nevýhodou této metody je, že při různé poloze počátečního semínka mohou vznikat různé výsledky. Další nevýhodou je možný vznik velkého množství regionů (záleží na zvolení kritéria homogenity). Hlavní problémy metody narůstání oblastí:

- výběr počátečního bodu
- ne každé zvolené kritérium homogenity vede ke kvalitní segmentaci

Metoda narůstání oblastí může být rozšířena o post-processing, při kterém se využívá heuristika (např. spojení metody detekce hran[1] a metody narůstání oblastí). [21]

4.3 Rozpoznání objektů

4.3.1 Invarianty

Invariant [7] [23] je v matematice nějaká vlastnost, která se transformacemi nemění.

Invarianty pro obraz jsou definovány:

- $I(f) = I(D(f))$ pro každé přípustné D
- $I(f_1), I(f_2)$ jsou dostatečně rozdílné pro rozdílné f_1, f_2

D je operátor degradace, přičemž $g = D(f)$, kde $f(x, y)$ je ideální obraz a $g(x, y)$ je pozorovaný obraz(v případě jakém budeme invarianty využívat bude $f(x, y)$ objekt v obraze a $g(x, y)$ bude korespondující objekt v druhém obraze, přesněji to bude ukázáno v kapitole 5).

[1] edge-based

4.3.2 Momenty

Momenty (moments) [7] [23] jsou projekce obrazové funkce do polynomiální báze.

- $f(x, y)$ po částech spojitá funkce obrazu definovaná na ohraničení $\Omega \subset R \times R$
- $\{P_{pq}(x, y)\}$ množina polynomů definovaná na Ω

$$M_{pq} = \int \int P_{pq}(x, y) f(x, y) dx dy \tag{4.16}$$

Obecné momenty

Spojitá forma:

$$M_{pq} = \int \int x^p y^q f(x, y) dx dy \tag{4.17}$$

Diskrétní forma:

$$M_{pq} = \sum_X \sum_Y x^p y^q f(x, y) \tag{4.18}$$

M_{pq} je dvoudimenzionální moment funkce $f(x, y)$. Řád momentu je $(p + q)$, kde p a q jsou přirozená čísla.

Centrální momenty

$$\mu_{pq} = \sum_X \sum_Y (x - \overline{x})^p (y - \overline{y})^q \tag{4.19}$$

kde

$$\overline{x} = \frac{M_{10}}{M_{00}} \tag{4.20}$$

$$\overline{y} = \frac{M_{01}}{M_{00}} \tag{4.21}$$

Normalizované centrální momenty

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\frac{p+q}{2}+1}} \tag{4.22}$$

4.3.3 Invarianty momentů

Nelineární invariantní funkce, které jsou složeny z rozmanitých projekcí f do polynomiální prostoru jsou nazývány *invarianty momentů* (moment invariants) [7] [23]. Invarianty momentů jsou funkce momentů invariantních k určité třídě degradací.

Mezi tyto třídy patří:

- rotace, translace, změna měřítka
- konvoluce, rozostření
- afinní transformace
- složené invarianty

RTS (Rotate/Scale/Translate) invarianty momentů

$$\phi_1 = \eta_{20} + \eta_{02} \tag{4.23}$$
$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \tag{4.24}$$

Invarianty momentů jsou často používány jako charakteristika objektů při zpracování obrazu, dálkovém průzkumu země, rozpoznání objektů a klasifikaci. Momenty poskytují charakteristiky objektu, které jednoznačně reprezentují objekt.

4.3.4 Korelace

Jednou nejefektivnější metodou hledání známého vzoru v obraze je korelace implementovaná pomocí FFT konvoluce. [16]

Algoritmus hledání vzoru v obraze pomocí FFT konvoluce

Obrázek 4.4 názorně ukazuje, jak pomocí konvoluce je možné vypočítat korelaci.

Jádro v našem případě bude obsahovat hledaný obrázek. Budeme předpokládat, že jádro je čtverec a původní obrázek také, potom označíme šířku jádra w_k a šířku obrazu ve kterém hledáme w_i. Jádro i obrázek, ve kterém se bude hledat je nutné zvětšit tak, aby jeho šířka i výška byla rovna $w_k + w_i - 1$. Vzniklé místo se vyplní nulami. Před zvětšením jádra je potřeba jej otočit o 180°. Mnohem kvalitnějších výsledků je dosaženo, pokud jsou oba obrázky převedeny na hrany (například pomoci Sobelova hranového operátoru).

Poté oba obrázky převedeme do frekvenční oblasti pomocí DFT (viz kapitola 4.1.2), kde je vynásobíme (viz kapitola 4.1.2) a výsledek převedeme pomocí inverzní DFT do prostorové oblasti (viz kapitola 4.1.2). Extrém ve výsledném obrázku ukazuje pozici pravého dolního rohu hledaného obrázku.

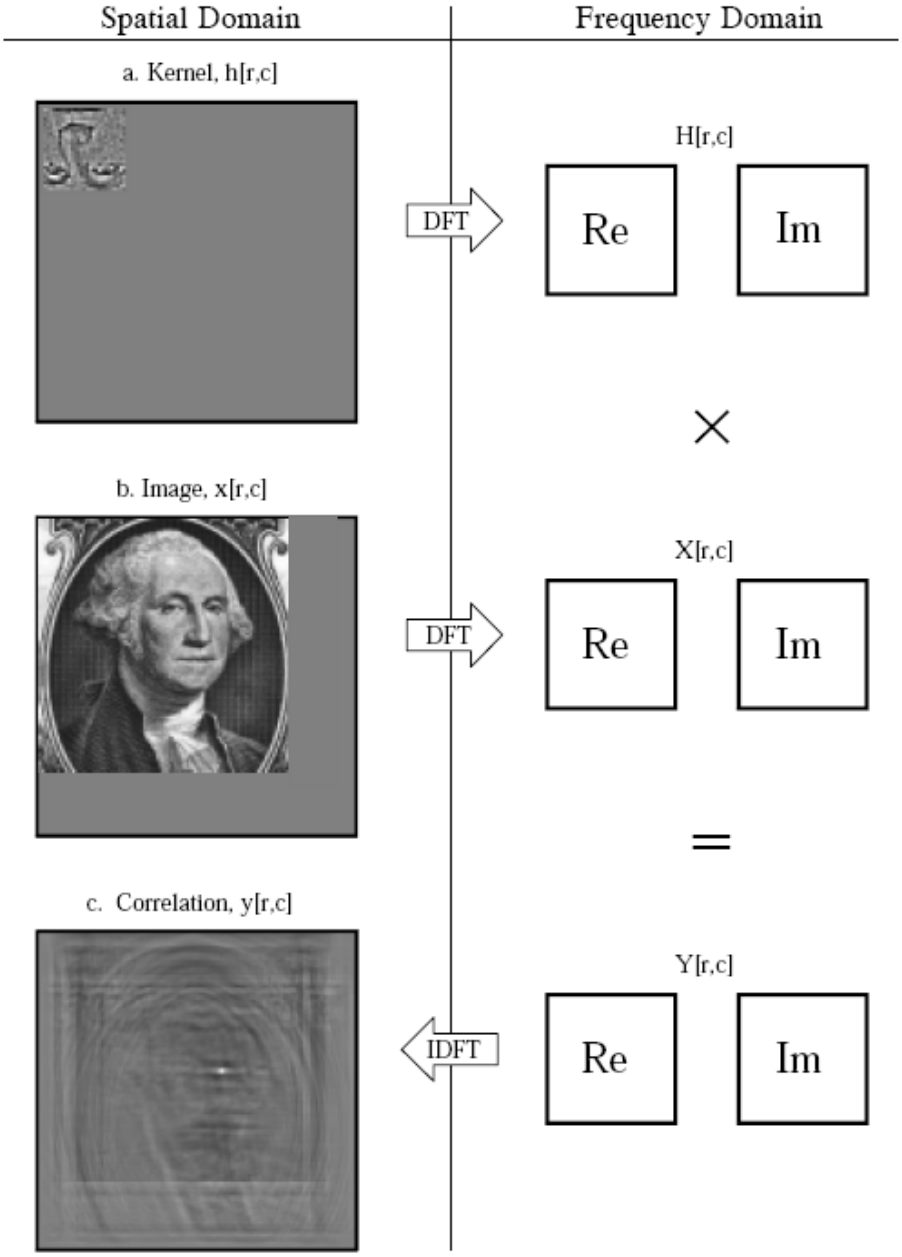
4.4 Algoritmus pro výpočet disparitní mapy v OpenCV

Algoritmus [18] je možné rozdělit na dvě části: *hlavní část výpočtu* a *postprocessing*.

4.4.1 Hlavní část

Tuto část výpočtu je možné rozdělit na další tři podčásti, a to:

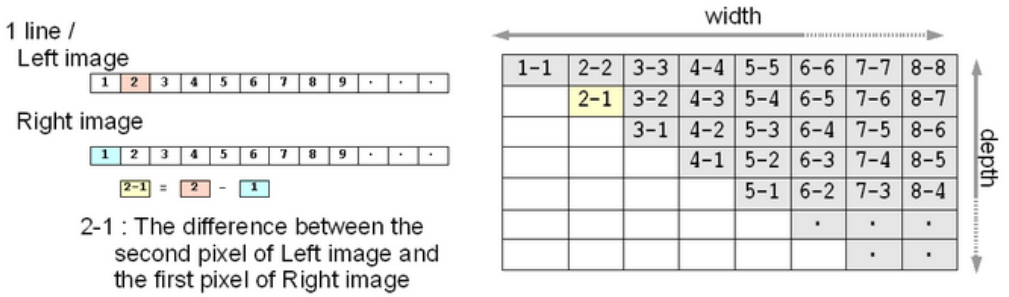
- prostor rozdílů
- DP tabulka
- mapa disparity



Obrázek 4.4: Obrázek znázorňující výpočet korelace pomocí FFT konvoluce [16]

Prostor rozdílů

V této části algoritmu se vypočítává rozdíl mezi pravým a levým obrázkem. Velikost prostoru je $hloubka * sirka$. Hloubka je zadána uživatelem. Na obrázku 4.5 je ukázán výpočet rozdílů prostoru pro jeden řádek levého a pravého obrázku. Pro první hloubku se nejprve položí řádky na sebe a vypočítají se rozdíly jednotlivých bodů. Následující řádek hloubky vypočítáme tak, že posuneme pravý řádek oproti levému doprava a opět spočítáme rozdíl. Takto pokračujeme dokud nedosáhneme hloubky zvolené uživatelem.



Obrázek 4.5: Obrázek popisuje výpočet rozdílů z levého a pravého obrázku [18]

DP tabulka

DP tabulka [18] je tabulka, kde každé buňce z tabulky rozdílů je přiřazena informace o ceně. DP tabulka je stejné velikosti jako tabulka rozdílů. Každá buňka tabulky ukazuje na buňku po své levici, nebo na buňku níže v tabulce. Cena každé buňky je vypočtena z jedné ceny v tabulce s nejnižší hodnotou, která na buňku ukazuje. Každá buňka v DP tabulce uchovává společně s cenou i pozici buňky, ze které počítala svoji hodnotu.

Mapa disparity

Disparitní mapa [18] obsahuje čísla řádků, na kterém se nachází buňka s nejmenší cenou. Disparitní mapu získáme optimálním průchodem DP tabulky.

4.4.2 Postprocessing

V této fázi se provádějí poslední úpravy mapy disparity, aby se odstranilo co nejvíce šumu a zřejmých chyb.

Kapitola 5

Knihovny a způsob získání obrazových dat

1. Java, Java Media framework
2. Python, Blender
3. C++, OpenCV

5.1 Knihovny

5.1.1 OpenCV

OpenCV[1] (Open Source Computer Vision) je volně dostupná grafická knihovna se zaměřením na počítačové vidění. Knihovna je vyvíjena společností Intel. Tato knihovna obsahuje velkou spoustu funkcí zaměřenou převážně na zpracování obrazu, uživatelské rozhraní a práci s grafickými formáty. Mezi algoritmy, které jsou implementovány v této knihovně patří například: detekce rohů, detekce hran (Sobelův hranový detektor, Cannyho hranový detektor, ...), práce s obrázky (histogram, podpora různých grafických formátu, filtry, konverze barev ...), převod obrázku na kontury, epipolární geometrie, kalibrace kamery, detekce objektů, nalezení korespondujících si bodů v obraze. Tento výčet není ani zdaleka úplný. Knihovna je dostupná pro programovací jazyky C/C++, Python a pro operační systémy Windows a UNIX.

5.1.2 FFTW

FFTW[2] (Fastest Fourier Transform in the West) je C knihovna pro výpočet diskrétní Fourierovy transformace (DFT) jedno nebo více dimenzionálního pole libovolné velikosti.

5.1.3 JMF

JMF[3] (Java Media Framework) je knihovna vyvíjená firmou Sun pro práci s multimédii. Knihovna obsahuje třídy pro přehrávání videa a zvuku, zaznamenávání videa a zvuku, získání videa/zvuku z kamery/mikrofonu, úpravu videa a zvuku v reálném čase, převod

[1] http://www.intel.com/technology/computing/opencv/, http://opencvlibrary.sourceforge.net/
[2] http://www.fftw.org/
[3] http://java.sun.com/products/java-media/jmf/

mezi formáty, přenos multimediálních dat po síti v reálném čase. Knihovna je napsána pro programovací jazyk Java a je dostupná pro operační systémy Linux, Solaris, Windows.

5.1.4 Blender

Blender[4] [13] je 3D studio pro modelování, animaci, tvorbu her, rendering a přehrávání. Toto 3D studio je založeno na grafické knihovně OpenGL a díky tomu je přenosné mezi většinou operačních systémů (Windows, Linux, Irix, Sun Solaris, FreeBSD, Mac OS X a dokonce existuje verze i pro pocket pc). Blender je k dispozici zcela zdarma včetně zdrojového kódu.

5.2 Získání obrazu

5.2.1 Získání obrazu z kamer

Hlavní součástí této aplikace je knihovna JMF pro práci s multimediálními daty, pomocí které je získáván obraz z kamer, upravován a v neposlední řadě i ukládán do souboru.

Největší důraz byl kladen na lehkou rozšiřitelnost kodeků, které byly zvoleny místo filtrů. Jelikož při snímání obrazu není jasné, které informace z kamer budou využity, neupravuje se snímaný obraz rovnou, ale místo toho je obraz zvětšen o velikost snímaného obrazu a do této nové části je nakopírován příchozí obraz, se kterým se následně pracuje a upravuje se. Tím je docíleno, že informace z předcházejících kodeků nebo z kamer nejsou následujícím kodekem zničeny, protože si následující kodek vytvoří novou oblast, se kterou teprve pracuje. Nové oblasti jsou přidávány nahoru. Výstup z kamer je nejspodnější malý obrázek ve videu, nad ním je obrázek který vnikl aplikováním kodeku na vstupní video a tak dále. Každý kodek si nejprve vybere, na který předcházející malý obrázek bude aplikován. Může si vybrat pouze výstup z kamer, nebo kodeky, které ho předcházely.

Úprava videa

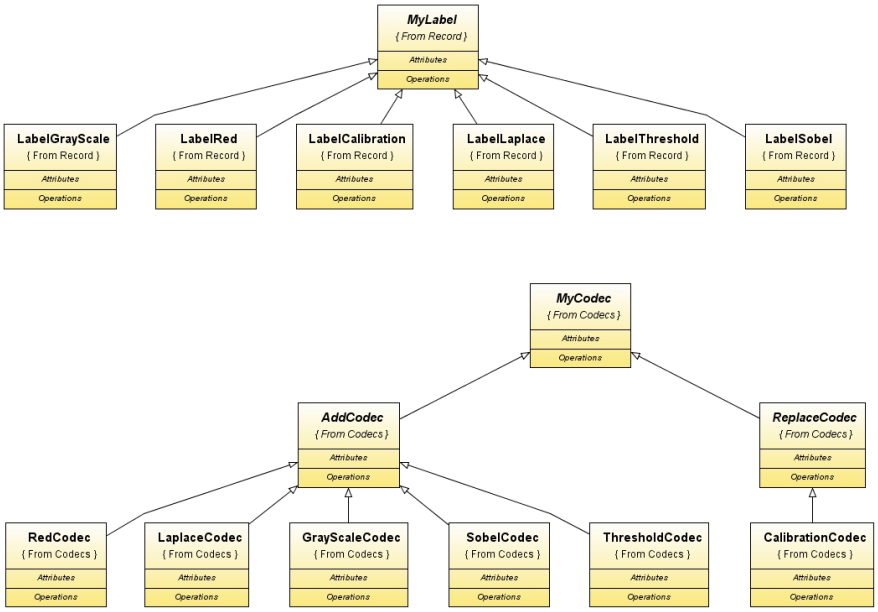
Po získání obrazu z levé a pravé kamery je obraz pomocí multiplexoru v procesoru spojen do jediného toku dat. V případě, že je požadováno zobrazit video, je vytvořen nový procesor, který je napojen na předcházející procesor. V něm jsou vytvořeny vybrané kodeky a výsledek je zobrazen do nového okna. Při ukládání je vytvořen další procesor s vybranými kodeky, ale tak aby je bylo možné pomocí *DataSink* ukládat do zvoleného souboru na disku. Při ukončení prohlížení a ukládání jsou oba tyto procesory zrušeny a první procesor je pozastaven, ale stále připraven k použití jako zdroj dat pro nové procesory. Teprve při ukončení aplikace je tento první procesor úplně zrušen a kamery odpojeny.

Přidání nového kodeku

Nejsnazším způsobem jak vytvořit nový kodek je odvodit jej z některé z abstraktních tříd. Na výběr jsou dvě možnosti *AddCodec* nebo *ReplaceCodec*. Třída *addCodec* přidává k získanému videu nový malý obrázek a ten teprve upravuje, zatímco třída *ReplaceCodec* upravuje vybraný malý obrázek z videa. Obě tyto třídy jsou zděděny z *MyCodec*, která implementuje rozhraní JMF Codec.

Při dědění z jedné z těchto abstraktních dvou tříd stačí implementovat metody:

[4] http://www.blender.org



Obrázek 5.1: Implementované kodeky a ukázka z jaké třídy byly odvozeny

```
public java.lang.String getName()

protected void process(Buffer input, Buffer output, byte[] inData,
                       byte[] outData, MyRGBFormat inputFormat,
                       MyRGBFormat outputFormat)
```

V metodě process je samotný kód kodeku. Není zapotřebí dále nic upravovat, protože vše je už nastaveno (formát, velikost výstupního buffer). Tímto způsobem vytvoříte standardní JMF Codec.

Další možný, ale pracnější způsob je zdědění z abstraktní třídy *MyCodec*.

Aby bylo možné propojit automaticky kodeky s GUI, je potřeba je obalit obalovací třídou. Touto třídou je abstraktní třída *MyLabel*. Následující ukázkový zdrojový kód ukazuje, co je zapotřebí implementovat a jakým způsobem.

```
class LabelGrayScale extends MyLabel {

    public LabelGrayScale() {
        super("GrayScale codec");                          // nastavuje jméno, které bude
    }                                                      // viditelné v˜GUI

    public MyCodec createMyCodec(MyRGBFormat inFormat) {   // zde je potřeba vytvořit instanci
        return new GrayScaleCodec(inFormat,getNumber());   // obalované třídy. getNumber()
    }                                                      // vrátí vybraný vstupní obrázek

    protected Object clone() {                             // zde je potřeba vytvořit kopii
        return new LabelGrayScale();                       // této třídy
    }
```

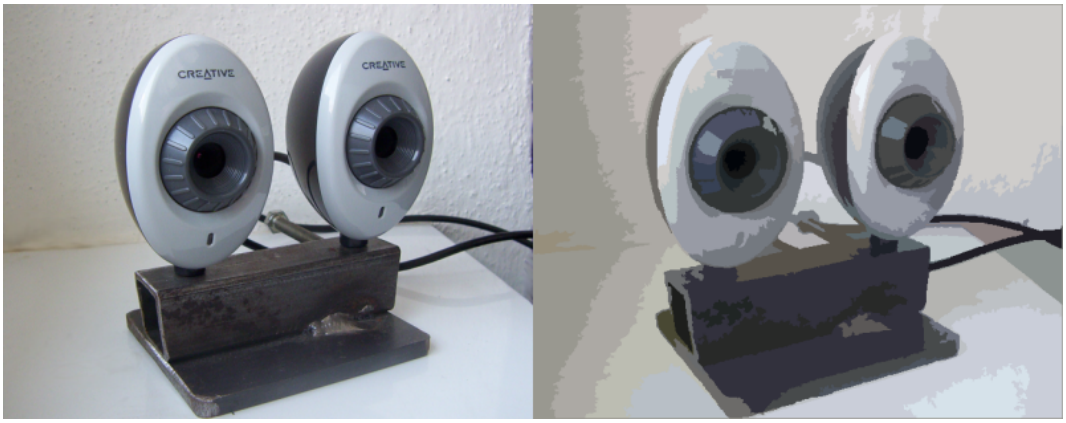
```
    public boolean isAddCodec() {                    // vrátit true pokud obalovaná třída
        return GrayScaleCodec.isAddCodec();          // nějakým způsobem zvětšuje obraz
    }                                                // tato informace je obsažena v
}                                                    // AddCodec i v ReplaceCodec
```

Poslední věcí, kterou je potřeba udělat, je připojit tuto obalovací třídu ke GUI. Provede se to přidáním jednoho řádku do konstruktoru v souboru RecordOptions.java

```
model.addElement(new LabelGrayScale());
```

Po přidání tohoto řádku je kodek správně připojen a je možné jej bez problému v aplikaci používat.

Problémy s kamerami



Obrázek 5.2: Dvě web kamery Creative Live! Cam Vista IM

Při experimentování, bylo zjištěno několik hlavních problémů, které nebylo možné odstranit.

Jedním z hlavních problémů je kalibrace kamer. Jak seřídit kamery, aby byly přesně paralelně vedle sebe? K experimentování byly použity dvě web kamery *Creative Live! Cam Vista IM* (obrázek 5.2). Přestože šlo o stejné kamery obraz z každé z nich se velice lišil. I při co nejvíce přesném seřízení, které bylo komplikované vzhledem k tvaru kamer, kamery nesměrovaly, kam by měly. Optika kamer je nepřesně uložena v plastovém krytu. Dalším podstatným problémem (hlavně s ohledem na následující projekt) je, že kamery nezobrazují stejně rychle. Při ručním výběru rozlišení se stává, že jedna kamera běží, zatímco obraz druhé kamery se mění po skocích. Ve výchozím nastavení tento problém nastává také, ale ne tak často a tak moc viditelně. Tyto okolnosti ztěžovaly první experimentování a navrhování algoritmu. Nejméně podstatným problémem je, že uložené video při přehrávání je zobrazováno rychleji, než je nahráváno.

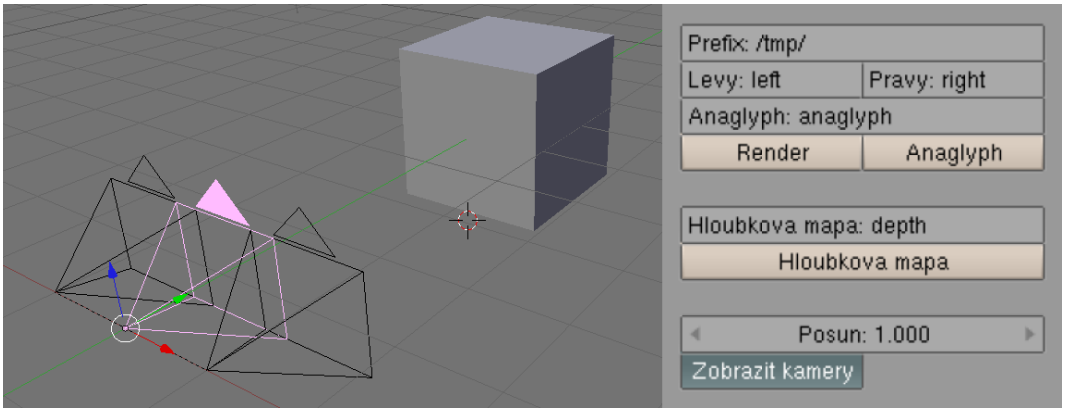
Kvůli chybě v JMF občas nastane restart počítače při ručním nastavení rozlišení kamer. Problém nastává při ručním výběru rozlišení a poté zapnutí ukládání. Tento problém

nastává většinou, pokud se vybere rozlišení 640x480, zatímco u menších rozlišení tato situace nenastává tak často.

5.2.2 Získání obrazu z 3D studia

Pro základní ověření algoritmu bylo využito 3D studio Blender.

Rendering scény



Obrázek 5.3: Obrázek ukazuje stereo kamery ve studiu Blender a skript na ovládání

Většinu úkonů v 3D studiu Blender je možné zautomatizovat skripty v jazyce Python. Pro zjednodušení renderingu scény byl vytvořen skript, který vloží do scény 3 kamery. Levou, prostřední a pravou. Levá a pravá kamera může být lehce schována a poté je možné pracovat pouze s prostřední kamerou. Při renderingu scény je nejprve vyrenderován obraz pro levou kameru, uložen do zvoleného adresáře a poté je vyrenderován obraz pro pravou kameru a následně uložen. Kam se budou ukládat vyrenderované obrázky a pod jakým názvem je možné nastavit ve skriptu ukázaném na obrázku 5.3.

Třídy pro práci se stereo kamerami:

- *MyRender* - Tato třída získává z vyrenderovaného obrázku Z-buffer, vytváří z obrázku/z-bufferu hloubkovou mapu a vloží ji do scény, ukládá z-buffer jako obrázek. Hlavní náplní této třídy je renderovat obraz ze zvolené kamery a ukládat jej do souboru.
- *StereoCameras* - Vytváří stereo kamery a vkládá je do scény. Nastavuje vzdálenost levé a pravé kamery od centrální kamery a pozici stereo kamery ve scéně.
- *St* - Tato třída vytváří grafické rozhraní pro předchozí třídy.

Bohužel doposud v poslední verzi (2.45) nebylo možné ve skriptech získat z-buffer scény. Proto bylo nutné upravit zdrojové kódy samotného Blenderu. Byla přidána nová funkce do API Blenderu, která získává z vyrenderovaného obrázku z-buffer a zpřístupňuje jej Pythonu. Funkce byla nazvána

```
getZBufferF
```

a vrací dvojrozměrné pole typu *float*.

5.2.3 Získání obrazu z webových stránek

Na internetových stránkách http://vision.middlebury.edu/stereo, zabývajících se metodami fotogrammetrie, jsou stereosnímky reálných scén vhodné k testování nových metod fotogrammetrie. Ke každému stereosnímku je připravena jeho hloubková mapa, která slouží k ověření správnosti nové metody.

Kapitola 6

Algoritmus a jeho implementace

V následujících kapitolách bude popsán výsledný algoritmus pro získání hloubkové mapy ze dvou obrázků.

6.1 Algoritmus

Algoritmus vychází z metod založených na korelaci, předpokládáme tedy, že levý a pravý obrázek jsou si do značné míry podobné. Hlavní snahou bylo vytvořit jednoduchý algoritmus, který se pokouší upřesnit disparitní mapu informacemi získanými segmentací. Vstupem algoritmu jsou dva barevné obrázky a výstupem algoritmu je disparitní mapa. Algoritmus je možné rozdělit na šest hlavních částí.

Detailní popis algoritmu:

1. *načtení levého a pravého obrázku ze souborů*
2. *předzpracování obrázků*
   (a) vytvoření kopií obrázků a převod těchto kopií obrázků na hrany pomocí Sobelova hranového detektoru (konvoluce)
   (b) vytvoření kopií obrázků a úprava těchto kopií pomocí konvoluce
3. *segmentace*
   (a) segmentace levého obrázku pomocí metody narůstání oblastí (k segmentaci se využije předzpracované kopie vstupního obrázku (viz krok 2b)
   výsledná barva segmentu je získána průměrem všech bodů, které jsou v tomto segmentu obsažené
   (b)
   - *způsob 1:*
     segmentace pravého obrázku stejným způsobem jako v kroku 3a
   - *způsob 2:*
     i. hledání segmentů z levého obrázku v pravém originálním obrázku
     – pokud je levý segment v pravém obrázku nalezen
     A. je vloženo nové semínko k segmentaci do předzpracovaného obrázku na nalezené místo
     B. pokud je již pravý obrázek v tomto místě segmentován, nic se neprovádí

- pokud je levý segment v pravém obrázku nenalezen, nic se neprovádí

ii. dokud existují segmenty z levého obrázku, pokračuje se krokem 3(b)i

iii. projití pravého obrázku, zda je zcela segmentovaný

A. pokud obsahuje nesegmentovanou část je tato část segmentována

4. *odstranění malých segmentů*

   (a) hledá se segment, který má menší obsah (počet bodů) než je kritérium zadané uživatelem

   (b) tento segment se spojí k segmentu, který z největší části obklopuje tento segment (kolika body se segment dotýká)

   (c) pokud existuje více segmentů se stejným počtem dotýkajících se bodů, je tento segment připojen k největšímu z těchto segmentů

   (d) dokud existují segmenty menší než kritérium, pokračuje se krokem 4a

5. *hledání korespondujících si segmentů*

   (a) porovnává se každý segment z levého obrázku s každým segmentem z pravého obrázku

   (b) segmenty, které se nejvíce shodují, jsou prohlášeny za korespondující si segmenty

   (c) *kritéria porovnávání segmentů*

   - podobnost momentových invariantů
   - segmenty jsou přibližně ve stejné výšce
   - segmenty jsou přibližně stejně velké (obsahují podobný počet bodů)

6. *vytvoření disparitní mapy*

   - *způsob 1:* pomocí OpenCV

     (a) z načteného pravého a levého obrázku se vypočítá pomocí OpenCV disparitní mapa (budeme nazývat hlavní disparitní mapa)

     (b) vybere se korespondující pár segmentů

     (c) tento pár slouží jako maska do původních načtených obrázků

     (d) všechny body, kromě těch, které se nalézají pod maskou, se odstraní

     (e) ze vzniklých dvou obrázků vypočítáme za pomocí OpenCV disparitní mapu

     (f) pokud existují ještě páry, které jsme nepoužili, skok na 6b

   - *způsob 2:* pomocí FFT konvoluce

     (a) z načteného pravého a levého obrázku se vypočítá pomocí DFT korelace disparitní mapa (bude nazývat hlavní disparitní mapa)

     (b) vybere se korespondující pár segmentů

     (c) tento pár slouží jako maska do původních načtených obrázku

     (d) všechny body, kromě těch které se nalézají pod maskou se odstraní

     (e) ze vzniklých dvou obrázků vypočítáme za pomocí FFT korelace disparitní mapu

     (f) pokud existují ještě páry, které jsme nepoužili, skok na 6b

   - *sjednocení disparitních map je společné pro způsob 1 i 2*

(a) postupně všechny disparitní mapy spojujeme s disparitní mapou vypočítanou pouze z pravého a levého původního obrázku
možné způsoby spojování:
- *způsob 1:* sjednocení
  i. odhadne se střed levého segmentu (který byl využit na výpočet disparitní mapy)
  ii. odhadne se střed pravého segmentu
  iii. pravý segment se posune na levý segment tak aby, se středy segmentu překrývaly
  iv. segmenty se sjednotí a tím vnikne maska, kterou použijeme na disparitní mapu, kterou jsme získali z těchto segmentů
  v. vymažeme body, které se nenalézají pod maskou, a sjednotíme s hlavní disparitní mapou
  vi. opakuje od kroku 6(a)i dokud nesjednotíme všechny disparitní mapy s hlavní disparitní mapou
- *způsob 2:* obdoba *sjednocení* s tím rozdílem, že se segmenty nesjednocují, použije se jenom segment z levého obrázku
- *způsob 3:* ořezání - segment, který převyšuje druhý segment, je zařízut tak, aby výška y byla u obou segmentů stejná

- *způsob 3:* získání disparitní mapy pouze pomocí informací získaných segmentací
  (a) při výpočtu segmentů každý segment obsahuje informaci o průměrné hodnotě x ze všech bodů obsažených v tomto segmentu
  (b) tuto hodnotu budeme považovat za střed segmentu v ose X
  (c) projdeme všechny páry segmentů a u každého páru spočítáme rozdíl mezi středem levého a pravého segmentu
  (d) do disparitní mapy vykreslíme levý segment z každého páru
  (e) každý bod segmentu bude obsahovat stejnou informaci, a to vypočtený rozdíl polohy středů v ose x vůči korespondujícímu pravému segmentu

6.2 Načtení vstupních obrázků

Tato implementace načítá pouze obrázky ze souboru, a to ve formátu *PNG*. Načítají se dva obrázky, levý a pravý. O načítání se stará knihovna OpenCV. Tento způsob byl zvolen proto, aby bylo možné kombinovat data z různých zdrojů (Blender, obrázky z WWW stránek, obrázky pořízené kamerami).

6.3 Předzpracování obrázků

6.3.1 Převod obrázků na hrany

Kvůli kvalitnějším výsledkům při FFT korelaci (viz kapitola 4.3.4) je vhodné obrázky převést na hrany. Byl využit Sobelův hranový detektor (viz kapitola 4.1.3), a to horizontální i vertikální. Při detekci hran se neupravují načtené obrázky, ale jejich kopie. Původní obrázky se nechávají v nezměněném stavu pro další zpracování.

6.3.2 Předzpracování obrázků

Předzpracování obrázků se provádí pomocí konvoluce. Vzhledem k tomu, že k tomuto úkolu stačí konvoluční jádro o velikosti $3x3$ nebo $5x5$ bodů, je v algoritmu použita klasická konvoluce (viz kapitola 4.1.1) místo FFT konvoluce.

6.4 Segmentace

Segmentace velkou měrou ovlivní hledání korespondujících si oblastí v pravém a levém obrázku. Pro segmentaci byla zvolena metoda narůstání oblastí. Jak již bylo zmíněno, záleží u této metody na počátečním bodu (semínku). Vzhledem k posunutí obrazu v pravém obrázku oproti levému může počáteční bod "zasáhnout" jiný objekt a výsledek segmentace může být rozdílný. Proto byla segmentace řešena dvěma způsoby. Společnou součástí obou těchto způsobů je segmentace levého obrázku. Ten je segmentován tak, že do levého horního rohu se umístí semínko a nechá se růst. Možnosti pro růst jsou dvě, osmi okolí nebo čtyř okolí. Jaký druh okolí se zvolí, určuje uživatel při spuštění programu. Informace o tom, jaké body byly postupně přidávány, se zaznamenává. Získáme tak seznam, kde první je semínko pro každý segment a dále následují body, které byly byly k segmentu přidány. Po dokončení růstu se zkontroluje, zda pravý bod od semínka je segmentován. Pokud ano, následuje test na další následující bod. Pokud je nalezen bod, který není ještě segmentován, je do něj vloženo semínko. Tím vytvoříme nový segment, který opět roste. Poté hledáme další bod, nepatřící do žádného segmentu. Toto opakujeme, dokud není celý obrázek rozdělen na jednotlivé segmenty. Velmi důležité při segmentaci je zvolení kritéria, které určuje, zda bod ještě bude patřit do segmentu, nebo ne. Kritérium segmentace je zadáno uživatelem při startu programu. Není využito žádného histogramu, případně jiných heuristik k výpočtu tohoto kritéria a to z důvodu, aby bylo možné lehce při experimentech nastavovat toto kritérium a sledovat, jaký má vliv na výslednou disparitní mapu.

Obrázek 6.1: a - originální obrázek, b - obrázek po segmentaci, c - obrázek po segmentaci s odstraněním malých oblastí

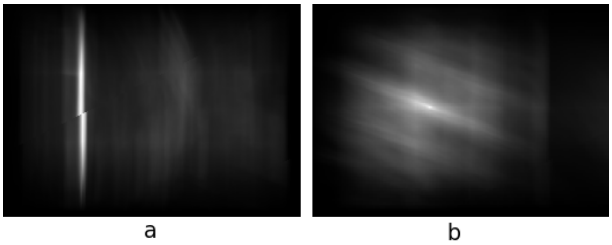
6.4.1 Způsob 1

Při tomto způsobu je semínko umístěno stejně jako při segmentaci levého obrázku, a to do levého horního rohu. Algoritmus na segmentaci pravého obrázku je totožný s výše popsaným algoritmem pro segmentaci levého obrázku.

6.4.2 Způsob 2

Tento algoritmus se pokouší odstranit posunutím pravého obrázku oproti levému a to tak, že se využije informací z již segmentovaného levého obrázku. Algoritmus nejprve vezme první segment z levého obrázku a pomocí FFT korelace se ho pokusí najít v pravém obrázku. Jak je ošetřeno, zda byl výsledný segment nalezen v pravém obrázku, bude popsáno později. Pokud byl segment nalezen (ne vždy korelace nalezne odpovídající segment v pravém obraze, protože úhel pohledu na objekty ve scéně je z každé kamery trochu jiný), vezme první bod segmentu a vloží jej jako semínko na nalezené místo. Pokud již zvolené místo je segmentováno, vezme další bod segmentu a pokusí se jej použít opět jako semínko pro nový segment v pravém obrázku. Toto se opakuje dokud se neprovede test se všemi body segmentu, nebo pokud některý bod nebyl úspěšně použit jako nové semínko. Následně se vezme další segment z levého obrázku a stejným způsobem se ho využije. Po vyčerpání všech segmentů se ještě pravý obrázek projde úplně stejně, jako když byl segmentován levý obrázek. Tím je zaručeno, že každý bod bude náležet do některého ze segmentů.

6.5 Určení existence korespondujícího obrázku při FFT korelaci



Obrázek 6.2: Výsledky po FFT korelaci. a - není možné určit místo extrému, b - výsledek obsahuje jeden dobře rozpoznatelný extrém

Pro určení pozice korespondujícího segmentu ve výsledku po FFT korelaci je potřeba nalézt globální extrém. V některých případech není možné tento globální extrém určit (obrázek 6.2a) ať už z důvodů, že extrému je víc, nebo naopak není žádný.

V implementaci se používají pouze výsledky, které se dají jednoznačně určit (obrázek 6.2b). V opačném případě není korespondující segment nalezen.

6.6 Odstranění malých segmentů

Při segmentaci vznikne velmi mnoho malých segmentů, které mohou být způsobeny například šumem v obraze, strukturou objektů. Pokud v obraze existuje velké množství těchto malých nevýrazných segmentů, není možné určit jejich korespondující segmenty. Z tohoto důvodu je vhodnější je sloučit s většími segmenty.

6.7 Hledání korespondujících si segmentů

V tomto kroce jsou již oba obrázky segmentovány a u každého segmentu známe jeho velikost (počet bodů), jeho pozici a momentové invarianty (ty jsou počítány během segmentace).

Pomocí těchto informací se pokusíme určit, které segmenty jsou vůči sobě korespondující a vytvořit tak korespondující páry segmentů.

První důležitá informace je pozice segmentů vůči sobě. Jelikož algoritmus je pro paralelní kamery, pozice segmentů bude ve stejné výšce. Pro každý levý segment jsou nalezeny všechny segmenty z pravého obrázku, které jsou přibližně ve stejné výšce. Následně jsou vyloučeny segmenty, které mají "moc" rozdílnou velikost. Ze zbylých segmentů, které by mohly být korespondující pro levý segment je vybrán ten segment, který má nejpodobnější momentové invarianty. Vyjde-li pro více levých segmentů stejný pravý segment, je tento segment přiřazen k levému segmentu, se kterým si je nejvíce podobný ve všech třech kritériích a pro zbylé segmenty na levé straně se hledají nové pravé segmenty. Levému segmentu nemusí být nalezen žádný pravý segment, pak zůstává nespárován.

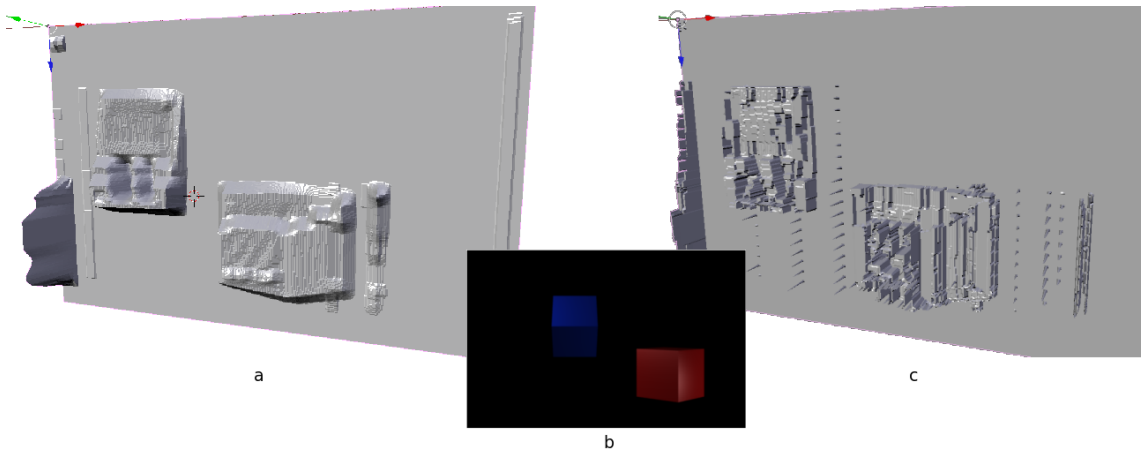
6.8 Vytvoření disparitní mapy

Pro výpočet disparitní mapy jsou využity dva způsoby. Při prvním se používá knihovna OpenCV a při druhém je využito FFT korelace.

Způsob 1 - OpenCV

Algoritmem popsaným v kapitole 4.4 se pomocí knihovny OpenCV vypočítá disparitní mapa, pro levý a pravý původní obrázek (obrázky nejsou nijak upraveny). Získáme tak hlavní disparitní mapu, kterou se pokusíme upřesnit pomocí informací získaných z korespondujících si segmentů, které byly vypočítány v předchozích krocích. Tento algoritmus vychází z předpokladu, že bychom měli dosáhnout přesnějších výsledků pokud se využije pro výpočet pouze oblasti v levém a pravém, která si navzájem korespondují. Sjednocením disparitních map od od všech korespondujících si páru segmentů bychom měli získat přesnější disparitní mapu. Vzhledem k tomu, že páry segmentů nemusí pokrývat celý prostor disparitní mapy je potřeba připojit disparitní mapu získanou z neupravených obrázků.

Způsob 2 - FFT korelace



Obrázek 6.3: Disparitní mapy zobrazené ve 3D studiu Blender
a - disparitní mapa získána průměrem; b - původní obrázek; c - disparitní mapa, kde každé korelaci odpovídá jeden bod

Způsob 2 vychází ze stejného předpokladu, ale místo OpenCV k výpočtu disparitní mapy využívá FFT korelace. Protože algoritmus je úplně stejný, jako při OpenCV bude zde popsáno pouze jak pomocí FFT korelace jsou počítány korespondující body.

1. levý a pravý obrázek je převeden na hrany pomocí Sobelova hranového detektoru
2. uživatel zvolí velikost jádra
3. jádro je čtverec z levého obrázku o zvolené velikosti
4. pomocí FFT korelace je jádro hledáno v pravém obrázku
5. po nalezení je do disparitní mapy vykreslen čtverec o velikosti jádra a to na pozici nalezené v pravém obrázku
6. tento čtverec, ale disparitní mapu nepřepisuje, s ostatními čtverci, které byly nalezeny tímto způsobem se provede aritmetický průměr jednotlivých bodů čtverců na stejné pozici a výsledek je teprve poté zapsán do disparitní mapy
7. jádro v levém obrázku se posune o jeden bod doprava (pokud je již celá šířka levého obrázku zpracována, nastaví se jádro na začátek nového řádku) a výpočet pokračuje od bodu 4 dokud se neprojde celý levý obrázek

Protože se jádra při výpočtu překrývají je využito aritmetického průměru k vyhlazení výsledků. Kdyby každé jádro korespondovalo pouze s jedním bode a tento bod by byl špatně vypočítán, působil by rušivě, zatímco tímto způsobem je hodnota bodu složena z několika výpočtů (srovnání viz obrázek 6.3).

Způsob 3 - získání disparitní mapy pouze pomocí informací získaných segmentací

Tímto způsobem získáme velmi hrubou disparitní mapu. Tento algoritmus předpokládá, že se nám podařilo získat větší množství dobře spárovaných segmentů. V nejlepším případě, že každý segment je objektem z reálné scény. Jelikož algoritmem jakým segmentujeme obrázky nezískáme úplnou segmentaci, bude segment obsahovat více objektů reálné scény.

Výpočet samotné disparitní mapy se provádí zjištěním posunutí levého segmentu vůči pravému korespondujícímu segmentu. Výpočtem rozdílů posunutí všech segmentů a normalizací těchto rozdílů získáme disparitní mapu.

Kapitola 7

Experimenty

7.1 Vliv nastavení programu na hledání korespondujících si segment

Veškeré experimenty byly prováděny s obrázky B.1, B.2, B.3, B.4, B.5, B.6, B.7, B.8.

Pro každý obrázek byl spuštěn výpočet s nastavením:

-t 40 -d 100 -4	-t 60 -d 100 -4	-t 80 -d 50 -4
-t 40 -d 100 -8	-t 60 -d 100 -8	-t 80 -d 50 -8
-t 40 -d 100 -4 -f	-t 60 -d 100 -4 -f	-t 80 -d 50 -4 -f
-t 40 -d 100 -8 -f	-t 60 -d 100 -8 -f	-t 80 -d 50 -8 -f

-t práh, udávající o kolik může být maximálně rozdílný sousední bod, aby byl ještě přidán do segmentu

-d pokud je segment menší než $(\frac{velikost\ nejvetsiho\ segmentu}{2})/d$, bude sloučen s větším segmentem

-4 "semínko" se bude zvětšovat pomocí čtyř okolí

-8 "semínko" se bude zvětšovat pomocí osmi okolí

-f pro segmentaci pravého obrázku se využije způsob, kdy se pomocí FFT hledá nejvhodnější místo na vložení "semínka"

Každý test byl puštěn 7x a to pokaždé s jiným konvolučním jádrem. Detailní popis konvolučních jader bude v následující podkapitole. Nastavení bylo vybráno tak, aby vygenerovaný obsah dat bylo možné ručně zpracovat.

Celkem bylo vygenerováno 3647 párů segmentů, u kterých bylo nutné ověřit, zda jsou opravdu k sobě korespondující.

7.1.1 Vliv konvoluce

K otestování vlivu předzpracování obrázků pomocí konvoluce bylo vybráno několik běžně používaných filtrů k ostření a rozostřování obraz.

$$h_1 = \begin{pmatrix} 2 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 2 \end{pmatrix} \quad h_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad h_3 = \begin{pmatrix} -2/8 & -2/8 & -2/8 \\ -2/8 & 1 & -2/8 \\ -2/8 & -2/8 & -2/8 \end{pmatrix} \tag{7.1}$$

Obrázek	h_1	h_2	h_3	h_4	h_5	h_6	h_7	součet
B.1	66/25	82/77	15/9	68/22	69/24	58/23	67/21	425/201
B.2	17/0	16/0	16/0	18/0	18/0	17/0	16/0	118/0
B.3	31/0	22/0	27/0	25/0	32/0	37/0	25/0	199/0
B.4	82/13	76/7	79/4	82/13	85/8	82/2	88/8	574/55
B.5	95/2	72/3	55/0	79/1	74/2	64/3	73/1	512/12
B.6	9/3	4/3	4/0	13/3	8/2	8/1	15/3	61/15
B.7	90/14	115/3	33/1	102/12	116/4	114/21	101/10	671/65
B.8	101/19	89/16	27/4	122/19	104/10	99/22	88/19	630/109
Součet	491/76	476/109	256/18	509/70	506/50	479/72	473/62	3190/457
%	(0.114)	(0.101)	(0.065)	(0.120)	(0.125)	(0.112)	(0.113)	(0.749)

Tabulka 7.1: Tabulka ukazující vliv konvolučního jádra na výsledné korespondující si segmenty. První číslo před lomítkem znamená kolik nalezených párů bylo dobře spárováno. Číslo za lomítkem říká kolik chybně. Řádek s % udává rozdíl procent dobře spočítaných párů segmentů - špatně spočítaných párů segmentů

$$h_4 = \begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix} \quad h_5 = \begin{pmatrix} 1/16 & 2/16 & 1/16 \\ 2/16 & 4/16 & 2/16 \\ 1/16 & 2/16 & 1/16 \end{pmatrix} \tag{7.2}$$

$$h_6 = \begin{pmatrix} 1/256 & 4/256 & 6/256 & 4/256 & 1/256 \\ 4/256 & 16/256 & 24/256 & 16/256 & 4/256 \\ 6/256 & 24/256 & 36/256 & 24/256 & 6/256 \\ 4/256 & 16/256 & 24/256 & 16/256 & 4/256 \\ 1/256 & 4/256 & 6/256 & 4/256 & 1/256 \end{pmatrix} \tag{7.3}$$

$$h_7 = \begin{pmatrix} 2/159 & 4/159 & 5/159 & 4/159 & 2/159 \\ 4/159 & 9/159 & 12/159 & 9/159 & 4/159 \\ 5/159 & 12/159 & 15/159 & 12/159 & 5/159 \\ 4/159 & 9/159 & 12/159 & 9/159 & 4/159 \\ 2/159 & 4/159 & 5/159 & 4/159 & 2/159 \end{pmatrix} \tag{7.4}$$

Jak je patrné z tabulky 7.1 nejlepších celkových výsledků dosáhlo konvoluční jádro h_5 a nejhorších h_3. Za referenční konvoluční jádro můžeme použít h_2, protože neovlivňuje vstupní obraz. Při porovnání výsledků dosažených h_5 a h_2 zjistíme, že použití předzpracování pomocí konvoluce zlepšuje výsledky při hledání korespondujících segmentů. Tabulka 7.3 říká, jaké konvoluční jádro je dobré zvolit pokud se levý a pravý obrázek segmentuje stejným způsobem (h_5 nebo h_6) a jaké je dobré zvolit, pokud se pravý obrázek segmentuje s využitím FFT korelace (h_5).

7.1.2 Vliv okolí

Tabulka 7.2 ukazuje vliv čtyř a osmi okolí "semínka" při segmentaci na hledání korespondujících segmentů. Z tabulky vyplývá, že ve většině případů (kromě při volbě konvolučního jádra h_2 a h_7) je výhodnější použít osmi okolí. Zlepšení výpočtu není oproti čtyř okolí velmi výrazné.

Obrázek		h_1	h_2	h_3	h_4	h_5	h_6	h_7	součet
B.1	4	21/16	46/37	6/5	38/10	33/17	25/12	29/17	198/114
B.1	8	45/9	36/40	9/4	30/12	36/7	33/11	38/4	227/87
B.2	4	9/0	8/0	8/0	10/0	9/0	9/0	10/0	63/0
B.2	8	8/0	8/0	8/0	8/0	9/0	8/0	6/0	55/0
B.3	4	15/0	11/0	14/0	13/0	16/0	17/0	11/0	97/0
B.3	8	16/0	11/0	13/0	12/0	16/0	20/0	14/0	102/0
B.4	4	39/7	38/4	42/1	36/8	41/3	35/1	48/3	279/27
B.4	8	43/6	38/3	37/3	46/5	44/5	47/1	40/5	295/28
B.5	4	44/2	33/2	26/0	38/1	36/0	27/1	36/0	240/6
B.5	8	51/0	39/1	29/0	41/0	38/2	37/2	37/1	272/6
B.6	4	4/3	3/1	2/0	4/1	6/0	3/0	8/2	30/7
B.6	8	5/0	1/2	2/0	9/2	2/2	5/1	7/1	31/8
B.7	4	50/8	56/2	17/1	54/8	61/2	62/11	54/6	354/38
B.7	8	40/6	59/1	16/0	48/4	55/2	52/10	47/4	317/27
B.8	4	37/14	53/12	15/4	65/12	49/7	47/12	50/8	316/69
B.8	8	64/5	36/4	12/0	57/7	55/3	52/10	38/11	314/40
Součet 4		219/50	248/58	130/11	258/40	251/29	225/37	246/36	1577/261
%		(0.046)	(0.052)	(0.033)	(0.060)	(0.061)	(0.052)	(0.058)	(0.361)
Součet 8		272/26	228/51	126/7	251/30	255/21	254/35	227/26	1613/196
%		(0.067)	(0.049)	(0.033)	(0.061)	(0.064)	(0.060)	(0.055)	(0.389)
Součet		491/76	476/109	256/18	509/70	506/50	479/72	473/62	3190/457
%		(0.114)	(0.101)	(0.065)	(0.120)	(0.125)	(0.112)	(0.113)	(0.749)

Tabulka 7.2: Tabulka ukazující vliv okolí při segmentaci na výsledné korespondující si segmenty. První číslo před lomítkem znamená kolik nalezených párů bylo dobře spárováno. Číslo za lomítkem říká kolik chybně. Řádek s % udává rozdíl procent dobře spočítaných párů segmentů - špatně spočítaných párů segmentů

Obrázek	h_1	h_2	h_3	h_4	h_5	h_6	h_7	součet
B.1	39/14	54/42	9/6	47/9	40/11	45/11	49/13	283/106
B.1 FFT	27/11	28/35	6/3	21/13	29/13	13/12	18/8	142/95
B.2	9/0	8/0	8/0	9/0	9/0	8/0	8/0	59/0
B.2 FFT	8/0	8/0	8/0	9/0	9/0	9/0	8/0	59/0
B.3	25/0	14/0	20/0	19/0	26/0	28/0	19/0	151/0
B.3 FFT	6/0	8/0	7/0	6/0	6/0	9/0	6/0	48/0
B.4	46/3	44/4	55/2	46/6	52/5	50/2	51/5	344/27
B.4 FFT	36/10	32/3	24/2	36/7	33/3	32/0	37/3	230/28
B.5	61/2	59/3	39/0	51/1	53/0	52/0	61/0	376/6
B.5 FFT	34/0	13/0	16/0	28/0	21/2	12/3	12/1	136/6
B.6	6/0	3/1	4/0	7/1	2/1	5/1	8/1	35/5
B.6 FFT	3/3	1/2	0/0	6/2	6/1	3/0	7/2	26/10
B.7	51/8	67/3	21/1	57/5	64/3	60/12	57/5	377/37
B.7 FFT	39/6	48/0	12/0	45/7	52/1	54/9	44/5	294/28
B.8	58/12	62/8	18/2	67/13	51/5	61/11	51/11	368/62
B.8 FFT	43/7	27/8	9/2	55/6	53/5	38/11	37/8	262/47
Součet	295/39	311/61	174/11	303/35	297/25	309/37	304/35	1993/243
%	(0.070)	(0.069)	(0.045)	(0.073)	(0.075)	(0.075)	(0.074)	(0.480)
Součet FFT	196/37	165/48	82/7	206/35	209/25	170/35	169/27	1197/214
%	(0.044)	(0.032)	(0.021)	(0.047)	(0.050)	(0.037)	(0.039)	(0.270)
Součet	491/76	476/109	256/18	509/70	506/50	479/72	473/62	3190/457
%	(0.114)	(0.101)	(0.065)	(0.120)	(0.125)	(0.112)	(0.113)	(0.749)

Tabulka 7.3: Tabulka ukazující vliv metody segmentace na výsledné korespondující si segmenty. FFT znamená, že pravý obrázek je segmentován za pomoci FFT korelace. První číslo před lomítkem znamená kolik nalezených párů bylo dobře spárováno. Číslo za lomítkem říká kolik chybně. Řádek s % udává rozdíl procent dobře spočítaných párů segmentů - špatně spočítaných párů segmentů

7.1.3 Vliv zvolené metody při segmentaci

Tabulka 7.3 srovnává metody segmentace. Jak je patrné z této tabulky segmentace pravého obrázku za pomoci FFT korelace nevykazuje zlepšení oproti stejné segmentaci levého a pravého obrázku, dokonce při použití FFT korelace jsou výsledky skoro 2x horší. Při výpočtu korespondujících segmentů z obrázku segmentovaného za pomoci FFT korelace algoritmus detekuje skoro jenom polovinu párů oproti stejné segmentaci levého a pravého obrázku. Tabulka 7.4 ukazuje celkový počet segmentů získaný se všech obrázků s korelačním jádrem h_5 a s osmi okolím. Segmentace za pomocí FFT korelace vygeneruje méně segmentů v pravých obrázcích, než metoda bez FFT korelace.

7.1.4 Vliv zvolených kritérií

Jak již bylo řečeno hodnoty pro práh a pro limit minimální velikosti oblasti byli určeny tak, aby bylo možné ověřit dosažené výsledky. Tyto parametry nejvíce ovlivňují segmentaci. Při velmi velkém prahu získáme jeden velký segment, zatímco při malém získáme velké množství malých segmentů. Tabulka 7.5 popisuje vliv tří vybraných parametrů na hledání korespondujících párů segmentů. Jak je vidět z tabulky pro každý obrázek vyhovuje jiné nastavení. Nastavení -t 40 -d 100 vygenerovalo nejvíce korespondujících segmentů, ale zároveň

h_5, osmi okolí			
		FFT	
počet levých seg.	počet pravých seg.	počet levých seg.	počet pravých seg.
1830	1781	1830	1728

Tabulka 7.4: Tabulka ukazující celkový počet vygenerovaných segmentů ze všech obrázků pro filtr h_5 s osmiokolím. První část tabulk se vztahuje k metodě segmentace, kdy je levý a pravý obrázek segmentován stejným způsobe. Druhá část tabulky ukazuje výsledek pro způsob segmentace ve které se využívá FFT korelace

Obrázek		h_1	h_2	h_3	h_4	h_5	h_6	h_7	součet	%
B.1	40 100	33/22	55/69	6/5	36/17	41/20	20/21	35/20	226/174	(0.014)
B.1	60 100	24/3	26/8	6/4	19/5	15/4	27/2	16/1	133/27	(0.029)
B.1	80 50	9/0	1/0	3/0	13/0	13/0	11/0	16/0	66/0	(0.018)
B.2	40 100	5/0	4/0	4/0	6/0	4/0	5/0	6/0	34/0	(0.009)
B.2	60 100	4/0	4/0	4/0	4/0	6/0	4/0	4/0	30/0	(0.008)
B.2	80 50	8/0	8/0	8/0	8/0	8/0	8/0	6/0	54/0	(0.015)
B.3	40 100	11/0	6/0	14/0	7/0	8/0	13/0	7/0	66/0	(0.018)
B.3	60 100	10/0	6/0	9/0	10/0	14/0	14/0	8/0	71/0	(0.019)
B.3	80 50	10/0	10/0	4/0	8/0	10/0	10/0	10/0	62/0	(0.017)
B.4	40 100	38/12	34/7	36/4	40/10	43/7	43/1	44/7	278/48	(0.063)
B.4	60 100	23/1	24/0	29/0	25/3	26/1	25/1	23/1	175/7	(0.046)
B.4	80 50	21/0	18/0	14/0	17/0	16/0	14/0	21/0	121/0	(0.033)
B.5	40 100	33/2	37/1	34/0	20/1	28/2	19/1	24/0	195/7	(0.052)
B.5	60 100	34/0	20/2	11/0	33/0	26/0	31/2	30/1	185/5	(0.049)
B.5	80 50	28/0	15/0	10/0	26/0	20/0	14/0	19/0	132/0	(0.036)
B.6	40 100	5/3	2/2	2/0	12/3	8/2	7/1	9/2	45/13	(0.009)
B.6	60 100	4/0	1/1	2/0	1/0	0/0	0/0	5/1	13/2	(0.003)
B.6	80 50	0/0	1/0	0/0	0/0	0/0	1/0	1/0	3/0	(0.001)
B.7	40 100	58/11	77/1	20/1	52/9	66/4	61/13	70/9	404/48	(0.098)
B.7	60 100	26/3	28/2	7/0	35/3	39/0	42/8	19/1	196/17	(0.049)
B.7	80 50	6/0	10/0	6/0	15/0	11/0	11/0	12/0	71/0	(0.019)
B.8	40 100	47/17	56/14	14/4	60/18	45/10	57/19	42/17	321/99	(0.061)
B.8	60 100	25/2	13/2	8/0	34/1	32/0	26/3	31/2	169/10	(0.044)
B.8	80 50	29/0	20/0	5/0	28/0	27/0	16/0	15/0	140/0	(0.038)
Součet 100 40		230/67	271/94	130/14	233/58	243/45	225/56	237/55	1569/389	
%		(0.045)	(0.049)	(0.032)	(0.048)	(0.054)	(0.046)	(0.050)		(0.324)
Součet 100 60		150/9	122/15	76/4	161/12	158/5	169/16	136/7	972/68	
%		(0.039)	(0.029)	(0.020)	(0.041)	(0.042)	(0.042)	(0.035)		(0.248)
Součet 50 80		111/0	83/0	50/0	115/0	105/0	85/0	100/0	649/0	
%		(0.030)	(0.023)	(0.014)	(0.032)	(0.029)	(0.023)	(0.027)		(0.178)
Součet		491/76	476/109	256/18	509/70	506/50	479/72	473/62	3190/457	
%		(0.114)	(0.101)	(0.065)	(0.120)	(0.125)	(0.112)	(0.113)	(0.749)	

Tabulka 7.5: Tabulka ukazující vliv zvolených parametrů na výsledné korespondující si segmenty. 40 100 - první číslo udává práh a druhé kritérium udávající jak musí být velká oblast, aby nebyla připojena k jinému segmentu. První číslo před lomítkem znamená kolik nalezených párů bylo dobře spárováno. Číslo za lomítkem říká kolik chybně. Řádek s % udává rozdíl procent dobře spočítaných párů segmentů - špatně spočítaných párů segmentů

velké množství těchto párů je chybně určených. Zatímco nastavení -t 80 -d 50 vygenerovalo nejméně páru, ale žádný z těchto párů nebyl chybně určen. Z pohledu následujícího zpracování bude výhodnější zvolit nastavení, které sice generuje méně korespondujících segmentů, ale zato určuje korespondující páry bez chyb. Algoritmus pro výpočet disparitní mapy počítá s tím, že segmenty jsou správně spárované a chybně určený pár znamená znepřesnění výsledku.

7.2 Výpočet disparitní mapy

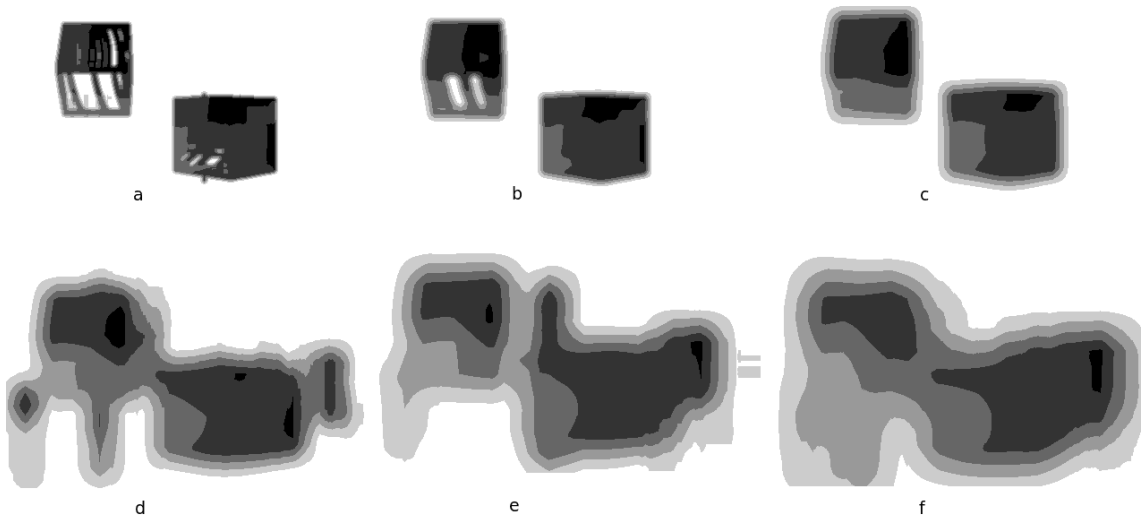
Při experimentování s nastavením pro výpočet disparitní mapy budeme vycházet z informací, které jsme zjistili v kapitole 7.1.

Nastavení:

- Konvoluční jádro - h_5
- Okolí - osmi okolí
- Způsobe segmentace - stejný způsob segmentace levého a pravého obrázku

Podle tabulky 7.6 bylo vybráno nastavení pro každý obrázek. Obrázek B.6 nebude v následujících experimentech využit, protože se zvoleným nastavením se nepodařilo najít žádný pár korespondujících segmentů.

7.2.1 Vliv velikosti konvolučního jádra



Obrázek 7.1: Srovnání výsledků pro šest různých konvolučních jader použitých pro hledání disparitní mapy na obrázku B.2.Čím je barva světlejší, tím je dál od pozorovatele. a - 5x5 bodů; b - 10x10 bodů; c - 20x20 bodů; d - 30x30 bodů; e - 40x40 bodů; f - 50x50 bodů

Velikost konvolučního jádra při korelaci velice ovlivňuje výslednou disparitní mapu. Nedá se stanovit univerzální velikost konvolučního, protože pro každý obrázek může být optimální velikost jiná. Při experimentování s obrázky byla zvolena velikost 10x10 bodů, protože vstupní obrázky jsou malého rozlišení.

Obrázek		h_5	%	FFT h_5	%
B.1	40 100	11/5	(0.037)	10/2	(0.070)
B.1	60 100	4/0	(0.025)	7/0	(0.061)
B.1	80 50	4/0	(0.025)	0/0	(0.000)
B.2	40 100	1/0	(0.006)	1/0	(0.009)
B.2	60 100	1/0	(0.006)	2/0	(0.017)
B.2	80 50	2/0	(0.012)	2/0	(0.017)
B.3	40 100	3/0	(0.019)	1/0	(0.009)
B.3	60 100	6/0	(0.037)	1/0	(0.009)
B.3	80 50	4/0	(0.025)	1/0	(0.009)
B.4	40 100	11/3	(0.050)	12/1	(0.096)
B.4	60 100	8/0	(0.050)	5/1	(0.035)
B.4	80 50	7/0	(0.043)	1/0	(0.009)
B.5	40 100	11/0	(0.068)	3/2	(0.009)
B.5	60 100	9/0	(0.056)	4/0	(0.035)
B.5	80 50	9/0	(0.056)	2/0	(0.017)
B.6	40 100	1/1	(0.000)	1/1	(0.000)
B.6	60 100	0/0	(0.000)	0/0	(0.000)
B.6	80 50	0/0	(0.000)	0/0	(0.000)
B.7	40 100	16/2	(0.087)	17/0	(0.148)
B.7	60 100	9/0	(0.056)	7/0	(0.061)
B.7	80 50	5/0	(0.031)	1/0	(0.009)
B.8	40 100	12/1	(0.068)	11/2	(0.078)
B.8	60 100	7/0	(0.043)	12/0	(0.104)
B.8	80 50	8/0	(0.050)	5/0	(0.043)
Součet 100 40		66/12	(0.335)	56/8	(0.417)
Součet 100 60		44/0	(0.273)	38/1	(0.322)
Součet 50 80		39/0	(0.242)	12/0	(0.104)
Součet		149/12	(0.851)	106/9	(0.843)

Tabulka 7.6: Tabulka ukazující vliv zvolených parametrů na výsledné korespondující si segmenty. Data v tabulce byla vygenerována s konvolučním jádrem h_5, osmi okolím. První část tabulky jsou výsledky ze segmentace, kdy se levý a pravý obrázek segmentoval stejným způsobem. Druhá část tabulky ukazuje výsledek po segmentaci za pomoci FFT korelace. První číslo před lomítkem znamená kolik nalezených párů bylo dobře spárováno. Číslo za lomítkem říká kolik chybně. Řádek s % udává rozdíl procent dobře spočítaných párů segmentů - špatně spočítaných párů segmentů

Obrázek	nastavení	oblasti	průměr	Y	bez úprav	nejlepší
B.5	40 100	4359863	4348064	4070501	4046164	bez úprav
B.7	40 100	4684334	4590594	3762209	3765809	Y
B.7	60 100	4492942	4407218	3771066	3766004	bez úprav
B.8	40 100	5550232	5436296	4462929	4457054	bez úprav
B.8	80 50	5401925	5442362	4474179	4455400	bez úprav

Tabulka 7.7: Tabulka porovnávající různé metody výpočtu disparitní mapy (OpenCV). Průměr - disparitní mapa je průměrem všech disparitních map ze segmentů, Y - disparitní mapa je získána se segmentů, které byly ořezány tak, aby byly stejně vysoké, Bez úprav - nevyužívá k výpočtu korespondující segmenty

Obrázek	nastavení	oblasti	průměr	Y	bez úprav	nejlepší
B.5	40 100	4359863	4579765	4588607	4819637	oblasti
B.7	40 100	4613972	4588506	5062449	5072874	průměr
B.7	60 100	4437076	3575323	4425194	5075558	průměr
B.7	80 50	4527030	3869166	4718147	5058714	průměr
B.8	40 100	5579213	5777274	6737111	6737111	oblasti
B.8	80 50	5487742	5938072	6570589	6863942	oblasti

Tabulka 7.8: Tabulka porovnávající různé metody výpočtu disparitní mapy (FFT korelace). Průměr - disparitní mapa je průměrem všech disparitních map ze segmentů, Y - disparitní mapa je získána se segmentů, které byly ořezány tak, aby byly stejně vysoké, Bez úprav - nevyužívá k výpočtu korespondující segmenty

7.2.2 Srovnání vypočítaných disparitních map s přesně změřenou disparitní mapou

Srovnání vypočtených a naměřených map je problém a to z důvodu, že způsob záznamu hloubky není jednotný. Každá hloubková mapa může být v jiném měřítku. Jednou z možností by byla normalizace hodnot, ale pouze za předpokladu, že maximální a minimální hloubka je správná. V okamžiku, kdy disparitní mapa obsahuje několik hodnot s velmi velkým rozdílem oproti maximu/minimu, není normalizaci možné použít. Dalším problémem je neexistence změřené disparitní mapy.

Problém se špatným měřítkem byl vyřešen tak, že optimální měřítko bylo hledáno hrubou silou. Měřítko s nejmenší sumou rozdílů bodů se změřenou disparitní mapou bylo zvoleno za optimální.

7.2.3 Výpočet disparitní mapy za pomoci OpenCV

Z tabulky 7.7 je patrné, že žádná z metod nevedla ke zpřesnění disparitní mapy. Tyto metody dokonce tento výsledek zhoršují. Nejméně přesná je metoda založená pouze na korespondujících segmentech, přesto výsledky z této metody nejsou o moc horší než, když je počítán průměr disparitních map pro jednotlivé páry segmentů.

7.2.4 Výpočet disparitní mapy za pomoci FFT korelace

Z tabulky 7.8 je vidět že při metodě s FFT korelací velice záleží na obrázku. Na rozdíl od metody založené na OpenCV, upřesňování disparitní mapy funguje. Nejlepších výsledků dosahuje metoda založená na průměrech.

Z důvodu velmi dlouho trvajících (několik hodin jeden obrázek) výpočtů disparitní mapy pomocí metody založené na FFT korelaci nebylo možné vytvořit velké množství

testovacích dat. Ale už první experimenty ukázaly, že mnohem lepších výsledků dosahuje metoda založená na OpenCV bez úprav.

Kapitola 8

Závěr

8.1 Implementace

Při návrhu metody pro stereovidění bylo snahou vyzkoušet více možností, poté vybrat vhodné kandidáty a s těmi experimentovat. Proto implementace není optimalizována na rychlost, ale na jednoduchou úpravu. Snahou byl co nejobjektovejsi návrh. Byly vytvořeny základní objekty pro práci s obrázky, které byly postupně doplňovány o nové metody. Snaha co nejvíce zapouzdřit data měla i své nedostatky, a to v rychlosti výpočtu algoritmu. Aby nemohlo dojít k nechtěné úpravě po změně algoritmu, byla většina dat kopírována. Při metodách segmentace, ukládání, konvoluci obrázků tento přístup nevadil, ale při hledání korespondujících bodů pomocí FFT korelace, se výpočet velice prodloužil. U velkých obrázků, při kterých bylo nalezeno větší množství korespondujících segmentů, se výpočet protáhl na několik hodin, protože korespondující body se hledají v tolika obrázcích, kolik je nalezených párů segmentů. Při využití OpenCV výpočet trval několik vteřin, jelikož knihovna OpenCV využívá ve velké míře SIMD a je značně optimalizovaná.

8.2 Algoritmus

Algoritmus je kombinací algoritmu založeného na příznacich a algoritmu založeného na korelaci. Z obou těchto algoritmů si nese svoje výhody i nevýhody.

Nejprve začneme s částí segmentace a hledáním korespondujících segmentů. Tato část patří mezi metody založené na příznacích, z čehož plyne její použití. Této části algoritmu nejvíce vyhovují větší plochy s ne moc složitou texturou. Nejlépe byly segmentovány a určeny jednoduché objekty vyrenderované ve 3D studiu. Naopak s čím si algoritmus neporadil, byl obrázek dítěte sedícího před mapou (obrázek B.6). Zde vzniklo velké množství malých segmentů u kterých se nedařilo dobře určit korespondující segmenty nebo naopak vzniklo velmi málo velkých segmentů, což se v podstatě rovná vyřazení metody hledající korespondující segmenty ze hry a využití pouze metody založené na korelaci. K nejlepším testovacím obrázkům z reálného světa patřil obrázek B.7, který se dá velice dobře segmentovat. Mezi různými částmi obrázku jsou kontrastní přechody a obrázek obsahuje množství přiměřeně velkých objektů ve scéně. Vhodné použití této části aplikace by mohlo být například uvnitř budov, kde je možné dobře od sebe rozlišit objekty.

Při testování bylo dosaženo nejlepšího výsledku úspěšnosti rozpoznání korespondujících párů 85.1%, přičemž se potvrdilo, že velký vliv na výsledek má počáteční nastavení. Bohužel pouze 9% segmentů ze všech segmentů v obraze bylo spárováno.

Zbylá část algoritmu patří mezi metody založené na korelaci. Tato metoda není vhodná pro obrázky pořízené z velmi vzdálených kamer. Další její nevýhodou je velká citlivost na osvětlení. Ke správnému výpočtu je zapotřebí, aby objekty ve scéně byly dobře otexturovány.

Snahou bylo rozšířit klasickou metodu založenou na korelaci. Předpokladem bylo předložit metodě dva segmenty, které jsou si podobné, a tím zúžit místa pro hledání. Viditelného zlepšení výsledků bylo dosaženo pouze s metodou počítající korelaci pomocí FFT korelace. Naproti tomu při použití s OpenCV metoda selhává. Výsledek u metody používající FFT korelaci ovlivňuje i velikost zvoleného jádra.

8.3 Možná vylepšení

Na prvním místě optimalizace rychlosti. Dále pokusit se zvýšit počet nalezených párů segmentů, například využitím FFT korelace přímo k rozpoznávání, zda segmenty jsou korespondující a ne jak je tomu nyní k hledání ideální polohy semínka. K lepším výsledkům by mohla přispět některá z heuristik k nalezení optimálního prahu.

K možným vylepšením druhé části algoritmu by mohlo patřit využít některý propracovanější způsob hledání korespondujícího bodu, oproti tomu nejzákladnějšímu, co byl využit. Pro vylepšení stávajícího algoritmu by mohlo pomoct při provádění průměrů vyčlenit jednu barvu za pozadí, která by neovlivňovala výsledek. V současném algoritmu se nejprve vypočítají korespondující body pro všechny páry segmentů. Vzhledem k tomu, že segment je menší než celý obrázek, spočítáním průměru s ostatními obrázky černá místa ztmavují ostatní segmenty. Nejvíce to ovšem ovlivní hlavní disparitní mapu a pak mohou vznikat velké rozdíly. Vhodné by bylo zdokonalit algoritmus na porovnávání výsledných disparitních map a v neposlední řadě propojit aplikaci na získávání obrázků z kamer přímo s algoritmem.

Literatura

[1] A. Bauer. *Lexikon výtvarného umění*. FIN, 1996. ISBN 80-71820-23-7.

[2] StereoGraphic Corporation. *The StereoGraphics Developer's Handbook*. StereoGraphics Corporation, 1997.

[3] Torsten N. Wiesel David H. Hubel. *Brain and Visual Perception*. Oxford University Press, 2004. ISBN 01-95176-18-9.

[4] E. R. Davies. *Machine vision*. 2005. ISBN 0-12-206093-8.

[5] I. Parberry F. Dunn. *3D math primer for graphics and Game Development*. Wordware Publishing, 2002. ISBN 15-56229-11-9.

[6] Andries van Dam James D. Foley. *Computer Graphics: Principles and Practice*. Addison Wesley, 1997. ISBN 0-201-84840-6.

[7] Tomas Suk Jan Flusser, Barbora Zitova. *Moments and moment invariants in image analysis*. 2007.

[8] Daniel Laubr. *Stereoskopická projekce*. 2006.

[9] J. Oplt. *Perspektiva*. Polygrafie Brno, 1925.

[10] F. Orság. *Robotika ROB - Studijní opora*. 2006.

[11] Michal Španěl Pavel Zemčík. *Počítačové vidění*. 2007.

[12] P. Pelant. *Optická metoda měření 3D posuvu pneumatické pružiny pomocí stereokorelace*. Technická univerzita v Liberci, 2006.

[13] Pavel Pokorný. *Blender naučte se 3D grafiku*. BEN, 2006. ISBN 80-7300-203-5.

[14] Ing. Michal Španěl Přemysl Kršek. *Základy počítačové grafiky*. 2007.

[15] M. Shah. *Fundamentals of computer vision*. Computer Science Department University of Central Florida, 1997.

[16] Steven W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1999. ISBN 0-9660176-6-8.

[17] WWW stránky. Encyklopedie wikipedia. `http://en.wikipedia.org`.

[18] WWW stránky. Opencv on the cell. `http://cell.fixstars.com/opencv/index.php/StereoMatching`.

[19] WWW stránky. Sobel operator. `http://en.wikipedia.org/wiki/Sobel_operator`.

[20] WWW stránky. Stereo vision research page. `www.middlebury.edu/stereo`.

[21] Ajay K. Ray Tinku Acharya. *Image processing - Principles and Applications*. Wiley, 2005. ISBN 978-0-471-71998-4.

[22] Trucco Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998. ISBN 01-32611-08-2.

[23] František Zbořil. *Základy umělé inteligence*. 2007.

Dodatek A

Ovládání programů

A.1 Popis obsluhy programu pro získání disparitní mapy

```
./main -l left_image -r right_image -t limit -d divide [-c correlation_size] [-8|-4] [-f]

 -l  levý obrázek
 -r  pravý obrázek
 -t  práh
 -d  kolikrát může být nejmenší segment maximálně menší než polovina největšího segmentu
 -c  k˜výpočtu disparitní mapy se využije FFT korelace místo OpenCV
 -8  osmiokolí
 -4  čtyřokolí
 -f  při segmentaci pravého obrázku se použije FFT korelace
```

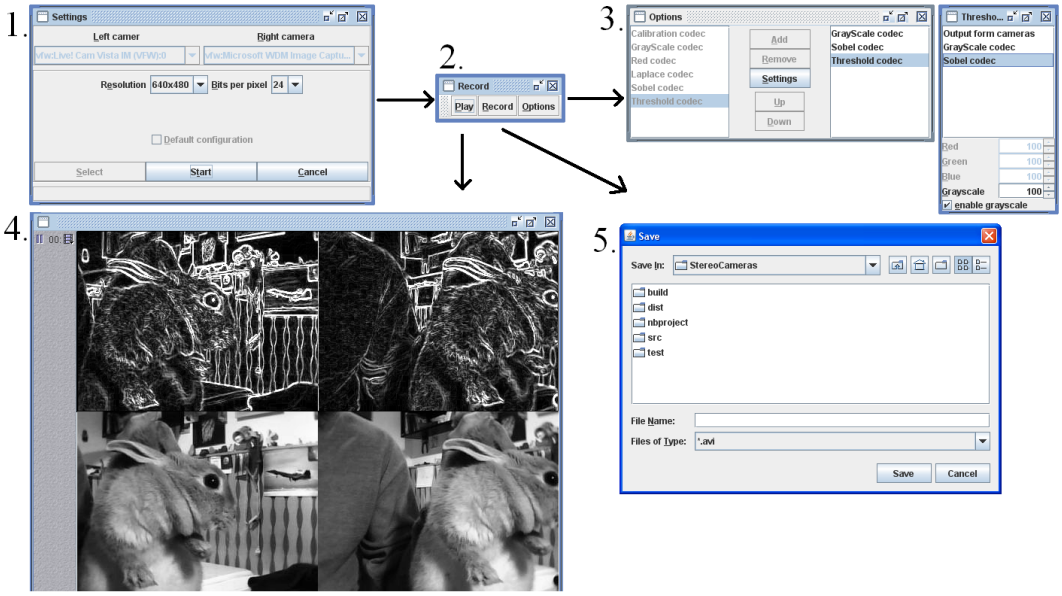
A.1.1 Výstup programu

Po spuštění programu, bude vytvořen adresář do kterého budou postupně ukládány veškeré výsledky.

```
[0-9][0-9][0-9]_[left|right].png       levý/pravý segment z páru vložený na modré pozadí
[0-9][0-9][0-9]_[left|right]_all.png   levý/pravý segment z páru zvýrazněný v˜původním obrázku
[0-9]+depth_[y|prumer]_[0-9]+.png      disparitní mapa získaná ze levého a˜pravého segmentu
z_conv_[left|right]_image.png          levý/pravý vstupní obraz po konvoluci s˜filtrem
z_out_[left|right]_image.png           levý/pravý vstupní obraz po segmentaci
depth_[Y|oblasti_l|oblasti_r|original|prumer].png   výsledná disparitní mapa pro konkrétní
                                                    algoritmus
```

A.2 Popis obsluhy programu pro získání obrazku z kamer

- *Settings* - Po spuštění aplikace je zapotřebí vybrat 2 rozdílné kamery, které podporují RGB formát.
  - V případě ponechání vybrané možnost: *Default configuration*, zvolí se standardní nastavení kamer (rozlišení a hloubka barev)
  - V druhém případě je možné si rozlišení a hloubku barev zvolit ručně



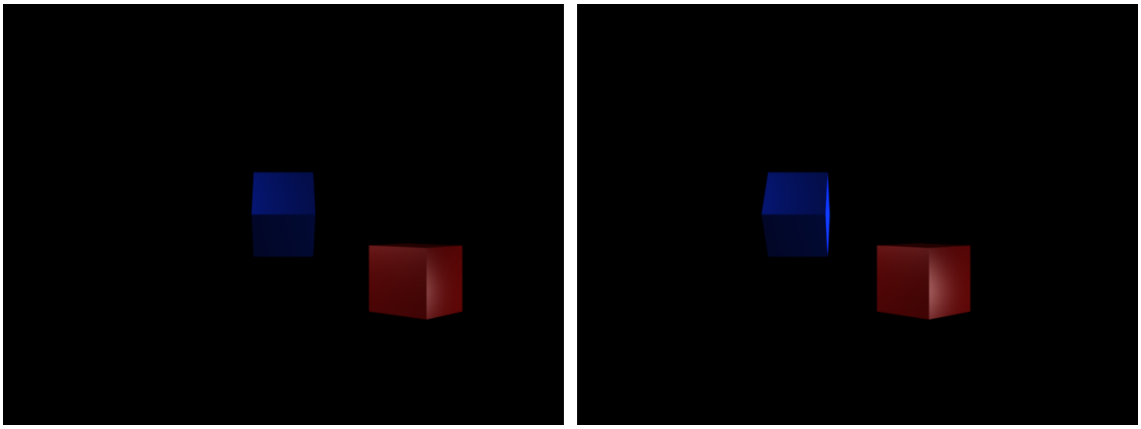
Obrázek A.1: Obrázek ukazující hlavní části programu pro zachytávání videa

- *Record* - Hlavní nabídka pro práci s videem. Nejprve je vhodné zvolit *Options*, kde se vybírají a nastavují kodeky. Tlačítkem *Play* je možné zobrazit obraz kamer a tlačítkem *Save* tento obraz ukládat do souboru.
- *Options*
  - *Add / Remove* - Vybírá a přidává/odebírá kodeky z levé strany (seznam všech dostupných kodeků). Přidávání a odebírání je možné pouze pokud není video zobrazeno a ani se nenahrává. Důležité je i zvolení pořadí kodeků. První kodek může pracovat pouze s výstupem z kamer, zatímco následující kodeky mohou pracovat s jedním výstupem některého z předcházejících kodeků nebo s výstupem kamer. Kodeky mohou vstupní video: rozšíří o další malý obraz (výsledné video bude obsahovat několik menších obrazů v jenom obraze) upravovat stávající malý obraz
  - *Settings* - Nastavuje který kodek, případně výstup z kamer, bude pro zvolené kodek vstupním. Kodeky, které nerozšiřují video, nebude možné vybrat jako vstupní kodeky. Nastavování je možné i při spuštěném přehrávání i záznamu. Je-li zvoleno nastavení při nahrávání/záznamu je možné u některých kodeků nastavovat více parametrů.
  - *Up / Down* - Pro úpravu pořadí kodeků.
- *Play* - Toto okno zobrazuje video po aplikování všech kodeků.
- *Save* - Požaduje výběr souboru, do kterého bude video uloženo. Podporovaný formát výstupu je AVI. Jakmile potvrdíte soubor, začne se video nahrávat.

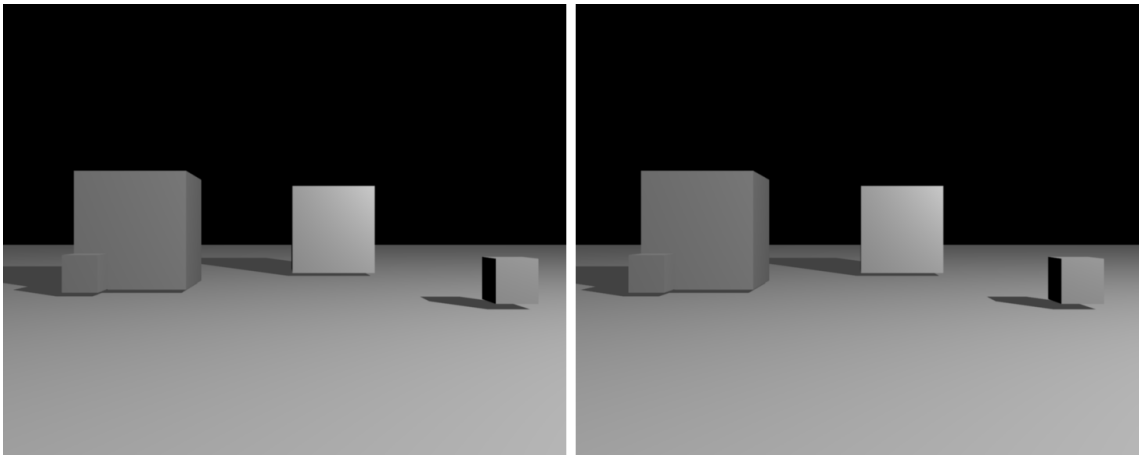
Dodatek B

Testovací obrázky

Obrázek B.1: Zdroj: http://www.tridakt.cz/galerie/zaslane/vase-3d.htm



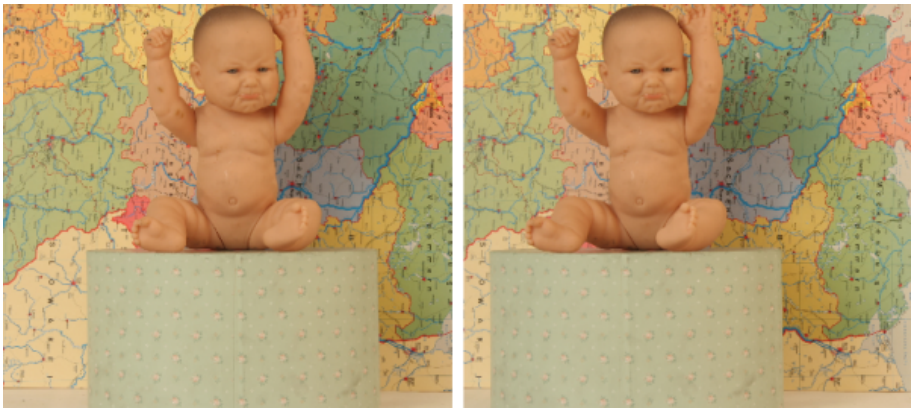
Obrázek B.2: Zdroj: Blender



Obrázek B.3: Zdroj: Blender

Obrázek B.4: Zdroj: kamery

Obrázek B.5: Zdroj: [20]



Obrázek B.6: Zdroj: [20]

Obrázek B.7: Zdroj: [20]

Obrázek B.8: Zdroj: [20]