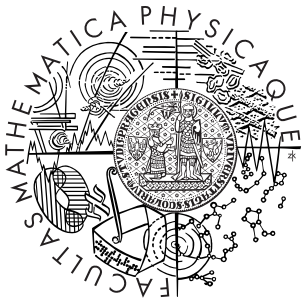
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Jindřich Pilmann

3D scannování laserem

Katedra softwarového inženýrství

Vedoucí bakalářské práce: Mgr. Viliam Holub, Ph.D.
Studijní program: Informatika, obecná informatika

2008

Na tomto místě bych rád poděkoval vedoucímu této práce, Mgr. Viliamu Holubovi Ph.D., za pomoc při návrhu a implementaci programu stejně jako při psaní bakalářské práce. Také bych mu rád poděkoval za konstrukci a poskytnutí použitého laseru. Chtěl bych poděkovat také fakultě za zapůjčení webkamery pro tento projekt. V neposlední řadě pak děkuji celé své rodině za veškerou podporu.

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 28. 7. 2008 Jindřich Pilmann

Obsah

Název práce: 3D scannování laserem
Autor: Jindřich Pilmann
Katedra (ústav): Katedra softwarového inženýrství
Vedoucí bakalářské práce: Mgr. Viliam Holub, Ph.D.
e-mail vedoucího: Viliam.Holub@mff.cuni.cz

Abstrakt: Obsahem této práce je popis a implementace jednoduchého 3D scanneru. Scanner je tvořen levnou webovou kamerou, jednoduchým zařízením vytvořeným z laseru ze staré CD mechaniky a programovým vybavením. Je založený na triangulační metodě scannování, pro převod obrazových souřadnic na světové využívá DLT a výstup poskytuje ve formátu VRML. Nemá být konkurentem profesionálních nástrojů, ale jen popisem možného jednoduchého řešení.
Klíčová slova: 3D snímání, DLT, vrml, webová kamera

Title: 3D laser scanner
Author: Jindřich Pilmann
Department: Department of Software Engineering
Supervisor: Mgr. Viliam Holub, Ph.D.
Supervisor's e-mail address: Viliam.Holub@mff.cuni.cz

Abstract: This work contains a description and an implementation of simple 3D scanner. The scanner is formed by cheap web cam, simple device created from laser from old CD-ROM drive and software. It is based on triangulation method of scanning. It uses the DLT method for the transformation of image coordinations into the world space. The output is in VRML format. It is not supposed to compete with professional tools, it should be only a description of a possible simple solution.
Keywords: 3D scanning, DLT, vrml, web cam

Kapitola 1

Úvod

Cílem této práce je popis a implementace jednoho z možných způsobů převádění trojrozměrných objektů reálného světa do digitálního modelu. Při použití této metody je objekt snímám kamerou a při tom je osvětlován laserem. Zatímco kamera a snímaný předmět žádný pohyb nekonají, laser se v průběhu scannování otáčí a to jak horizontálně, tak v omezené míře i vertikálně.

Pro převádění analogových veličin do digitální podoby se užívá termín vzorkování, častěji ale z anglického výrazu "scan" scannování. Cílem projektu je tedy sestrojit jednoduchý 3D scanner.

Projekt nemá být konkurentem nějakého z profesionálních či vědeckých řešení, protože jejich cena se mnohdy pohybuje i ve stovkách tisíců korun, je spíše návodem či inspirací jak takovýto problém vyřešit sice méně kvalitně ale nesrovnatelně levněji.

1.1 Motivace

To, co umí zachytit dobrý malíř či portrétista, můžeme stejně dobře zachytit pomocí fotoaparátu. A ačkoli má fotografie již velmi dlouhou historii, až do nedávné doby neexistovalo nic, co by umožnilo obyčejnému člověku zachytit něco i prostorově a ne jen jako obrázek. Příchod 3D scannerů tedy alespoň částečně umožňuje lidem vyrovnat se v něčem sochařům.

Toto bezpochyby velmi atraktivní odvětví prodělává v poslední době rychlý vývoj a má široký rozsah použití. Hlavní motivací tedy bylo zkusit si něco, co je v dnešní době používáno v mnoha oborech, od lékařství až po stavebnictví. Bylo jasné, že nebude možno realizovat nějaký dokonalý

3D scanner, ale ani konstrukce jednoduchého řešení není triviální a vypadá velmi zajímavě.

Získávání trojrozměrných informací z obyčejných rastrových snímků mě zaujalo a chtěl jsem vyzkoušet, zda to jde jednoduše implementovat. Ačkoli je princip velmi podobný prostorovému vidění člověka to, že nemáme k dispozici dva snímky ale jen jeden, situaci výrazně ulehčuje.

1.2 Obsah práce

Kapitola 3D scannování se zabývá tvorbou trojrozměrných modelů reálných věcí obecně. Snaží se představit jednotlivé techniky, které se ke scannování používají a u některých upozorňuje na jejich přednosti či slabiny. Dále zmiňuje jednotlivé obory, ve kterých se scannery (i když se jim tak mnohdy neříká) v dnešní době používají.

Kapitola Analýza nejdříve do větších podrobností popisuje techniku zvolenou k implementaci. Následně se zabývá tím, jak zjistit některé parametry kamery, které jsou pro scannování potřeba. V další části je pak jedna z metod, která se k tomu používá, popsána dopodrobna. Podrobněji je zde také popsáno zařízení laseru, jak se s ním bude zacházet a jak lze na snímku přítomnost laseru zjišťovat. Poslední je v této kapitole rozpracováno, jaký by měl mít program výstup, aby bylo možno s nascannovanými informacemi dále pracovat.

Kapitola Implementace popisuje jak, se jednotlivé myšlenky z předcházející kapitoly podařilo naplnit a jakým způsobem byly realizovány. Zmiňuje také problémy, které se objevily až při tvorbě programu, a jejich řešení.

Kapitola Použití programu radí uživateli, jak program používat. Upozorňuje, co je třeba mít k řádnému použití. Podává návod k instalaci a také k tomu, jak programu zadat potřebné informace tak, aby mohl provést úspěšné scannování.

Kapitola Závěr rekapituluje použité řešení a hodnotí úspěšnost implementace. V další části navrhuje, co by šlo v budoucnu zlepšit nebo co udělat nějak jinak.

Kapitola 2

3D scannování

3D scannování je jeden ze dvou způsobů, jak získat digitální třírozměrný model nějakého objektu z reálného světa. Prvním způsobem je vyrobit model ručně pomocí nějakého systému pro 3D modelování (například 3dMax nebo AutoCAD [8]). Tento způsob je však poměrně pracný a náchylný k chybám. Zvláště pokud se jedná o předmět ze světa přírody, či umělecké dílo a nejde o nějakou technickou součástku vyznačující se pravidelnými geometrickými tvary. Naproti tomu scannování může být velmi rychlé, téměř bez lidské asistence a mnohdy i mnohem přesnější. Pokud by však vznikla nějaká chyba, lze vzniklý výstup použít jako koncept a pomocí modelování může být dále upravován, dokud nebude požadované přesnosti dosaženo.

Aplikací 3D scanování je v dnešním světě opravdu mnoho. Používá se v průmyslu, ve stavebnictví, v kinematografii, ve zdravotnictví nebo může být použit třeba v reklamě. Navíc se používá i ke zdokumentování historických či uměleckých děl [12]. V průmyslu může být využit například k návrhům nových prototypů, kdy je nejprve ručně vymodelován v plastelíně, nebo jinak sestrojen, malý model, ten je pak nascannován do počítače, upraven a teprve následně se vyrábí první prototyp. Pokud se rozbije nějaká součástka a nejsou k dispozici žádné technické nákresy, může být scanner použit k přesnému opětovnému navržení nové součástky. Ve stavebnictví se po nascannováni stavby nebo modelu mohou hledat slabá místa, či kritické oblasti stavby důležité z hlediska statiky [3]. Kinematografie používá počítačová kouzla už více jak třicet let a 3D scannování je opět nejjednodušší způsob jak získat model skutečného předmětu, pak ho trikoví specialisté rozpohybují nebo upraví tak, že už jako skutečný nevypadá. Ve zdravotnictví jsou například data z magnetické rezonance používána ke konstrukci

třírozměrných modelů, kde můžou být některé věci lépe patrné. 3D modely nových výrobků mohou být použity v prezentacích na Internetu a tak mohou zákazníci vidět jak věc, co si chtějí koupit, opravdu vypadá. To je určitě lepší reklama než pouhá 2D fotografie.

2.1 Druhy 3D scannerů

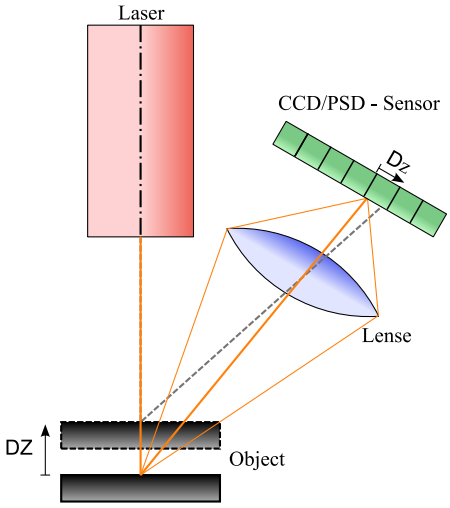
Scannery lze klasifikovat více způsoby, nejčastější je rozdělení podle techniky scannování. Primárně se scannery dělí na kontaktní a bezkontaktní. Bezkontaktní scannery se pak dále dělí na aktivní a pasivní.

Kontaktní scanner zkoumá objekt pomocí přímého dotyku. Většinou tak činí pomocí nějakého ramene nebo výsuvné tyče a díky znalosti natočení, ohnutí či vysunutí ramene zjišťuje polohu bodu na povrchu objektu. Velkou nevýhodou tohoto způsobu scannování je, že objekt může být v průběhu scannování poškozen nebo dokonce zničen. Další nevýhodou je nízká rychlost scannování. Kontaktní scannery většinou nepřesáhnou frekvenci několika hertzů, což může být v porovnání s jinými metodami i více než tisíckrát pomalejší.

Bezkontaktní aktivní scannery vysílají ke zkoumanému objektu nějaké záření (nejčastěji světlo, ale není to podmínkou) a na základě odrazu tohoto záření určují polohu respektive tvar objektu. I bezkontaktní scannery je možno rozdělit na několik dalších druhů.

Prvním druhem jsou tzv. "time-of-flight" scannery [2]. Jedná se o scannery, které vysílají ke snímané scéně laserový paprsek (nejčastěji) a následně měří čas mezi vysláním paprsku a přijmutím jeho odrazu. Vzdálenost místa odrazu může být snadno dopočítána díky tomu, že rychlost světla je konstantní. Nevýhodou je, že velmi vysoká rychlost světla klade velmi vysoké nároky na přesnost měření časového intervalu. Například světlu trvá jen 3,3 pikosekundy než překoná vzdálenost jednoho milimetru, pokud nám tedy technické prostředky nedovolí měřit čas dostatečně přesně, nelze měřit přesně ani vzdálenosti. Výhodou je naopak vysoká rychlost scannování pomocí této metody. Pokud je laser směrován soustavou zrcadel, může dosahovat frekvence až 100 000 bodů za sekundu.

Jiný druh scannerů využívá metodu triangulace [4]. Základem této metody je, že je objekt ozařován laserem z jiného úhlu, než z kterého je snímán kamerou. Poloha ozářeného bodu na pořízeném snímku je pak závislá na vzdálenosti bodu od kamery (viz obrázek 2.1). Technika je nazývána triangulační neboť kamera, zdroj záření a ozařovaný bod se nacházejí ve vr-



Obrázek 2.1: Princip triangulační metody, zdroj:[15]

cholech pomyslného trojúhelníku. Díky znalosti vzdálenosti zdroje záření od kamery (jedna strana trojúhelníku), směru záření a směru pohledu kamery (dva vnitřní úhly), lze dopočítat vzdálenost bodu od kamery (druhá strana trojúhelníku). Cílem této práce bylo sestrojit scanner právě tohoto druhu, nakonec je však silně využívána i myšlenka Structure light scannerů. Výhodou triangulačních scannerů oproti "time-of-flight" scannerům, je jejich přesnost při použití na krátké vzdálenosti. Tato přesnost může být vyšší než jen v řádu milimetrů, což je většinou limit konkurenční technologie. Naproti tomu nejsou triangulační scannery vhodné pro scannování velkých objektů, jako jsou například budovy, což by naopak "time-of-flight" scannerům nemělo činit potíže.

Další typ scannerů (tzv. Structured light 3D scanner [5]) promítá na snímaný objekt nějaký vzorek. Vzorek může být jednorozměrný (např. pruhy) nebo dvoudimensionální (např. mřížka). Podle změny tvaru vzorku jsou pak dopočítávány polohy jednotlivých bodů ozářených vzorkem. Hlavně u jednorozměrných vzorků je způsob dopočítávání hodně podobný triangulační metodě. Výhodou tohoto typu scannerů je nepochybně jejich rychlost. A to především díky tomu, že v jednom kroku nezískávají polohu jen jednoho bodu, ale polohu všech bodů v jednom záběru.

Bezkontaktní pasivní scannery, narozdíl od aktivních, žádný druh záření směrem ke snímanému objektu nevysílají. Většinou pracují s přirozeným

osvětlením scény nebo s nějakým jiným druhem záření (např. s infračerveným zářením vycházející ze snímaných objektů). Zařízení využívající pouze přirozeného osvětlení pak bývají zpravidla levnější, protože nepracují s žádným speciálním hardwarem.

Příkladem pasivních scannerů jsou stereoskopické systémy. Pracují většinou se dvěma kamerami, které snímají stejnou scénu, avšak jsou od sebe trochu vzdálené. Systém pak analyzuje mírné rozdíly mezi těmito dvěma záběry a pomocí toho určuje vzdálenost jednotlivých bodů ve scéně. Celá myšlenka je inspirována prostorovým viděním člověka.

Jiný druh scannerů pracuje pouze se siluetou zkoumaných objektů. Předmět je při této metodě snímkován z různých úhlů, ale vždy proti kontrastnímu pozadí. Z jednotlivých snímků se pak získává jejich průsečík a výsledkem těchto průsečíků je přibližný povrch předmětu.

Kapitola 3

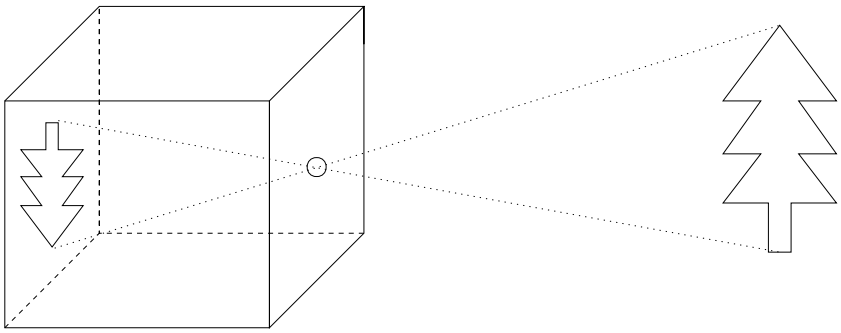
Analýza

3.1 Triangulační metoda

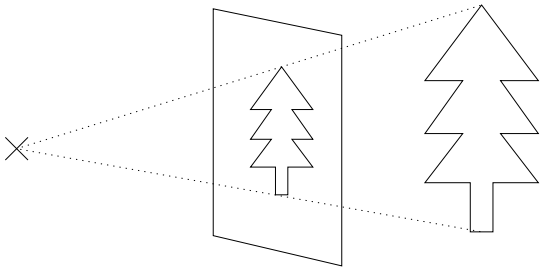
Pro tento projekt byla vybrána triangulační metoda 3D scannování a to hlavně z důvodů finanční nenáročnosti (bude využíváno jen levné webové kamery a laseru ze staré CD mechaniky) a poměrně dobré přesnosti na krátké vzdálenosti, tzn. při scannování malých objektů.

Princip této metody je poměrně jednoduchý. Každý pixel jednoho snímku kamery reprezentuje směrový vektor. Z tohoto směru daný pixel přijímá světelné záření, které interpretuje jako svou barvu. Pokud známe polohu kamery a směrové vektory pro jednotlivé pixely, můžeme pro každý pixel určit přímku, na které leží body reprezentované tímto pixelem. Dále potřebujeme znát polohu laseru a směr, kterým svítí. Poté můžeme na snímku najít světelný bod laseru. Jelikož poloha laseru a jeho směr zadávají jednu přímku a pixel, který reprezentuje obraz světelného bodu zadává druhou, můžeme najít jejich průsečík (respektive bod, v kterém jsou k sobě přímky nejblíže, pokud nebude měření zcela přesné) a tím získat jeho souřadnice. Pokud se bude postupně měnit směr laseru, získáme takto více bodů, tzv. oblak bodů (cloud of points). Z těch pak sestrojíme trojrozměrný model snímané scény.

Přesnost této metody tedy úzce souvisí s tím, jak přesně budeme znát polohu kamery a laseru a směrové vektory pro jednotlivé pixely a laser. Zjišťovat tyto informace pomocí měření a analytických výpočtů však prakticky není možné. Bylo by třeba znát např. přesnou polohu světločivného čipu nebo přesnou polohu čočky a její ohniskovou vzdálenost. Pro zjištění polohy čipu a čočky by však kamera pravděpodobně musela být rozebrána, což by představovalo velké riziko poškození. Kvůli výše zmíněným problémům



Obrázek 3.1: Dírková kamera



Obrázek 3.2: Středové promítání

se ve 3D scannování používá jiná metoda pro zjištění parametrů kamery.

3.2 Kalibrace kamery

Většinou se používá zjednodušený model kamery a to tzv. dírkové kamery, obr. 3.1. Jedná se o nejstarší model kamery známý pod latinským názvem camera obscura. Kamera je v tomto modelu představována např. dutou krabicí, v jejíž jedné stěně je malý otvor. Tím pronikají světelné paprsky dovnitř a na protější stěně vytvářejí obraz scény. Pokud je protější stěna rovná, je dírková kamera ekvivalentní středovému promítání (obr. 3.2) známému z 3D počítačové grafiky a deskriptivní geometrie.

Středové promítání je v počítačové grafice nejpoužívanějším druhem promítání a to z toho důvodu, že nejlépe odpovídá lidskému vidění, viz [7]. Stejně tak model dírkové kamery dobře odpovídá skutečným kamerám. Pokřivení obrazu skutečných kamer vůči modelu dírkové kamery vzniklé optickými chybami čoček lze pak buď odstranit transformací obrázku ještě před použitím nebo je lze uvažovat jako další parametr kamery. Pokud jsou

tyto deformace malé, lze je zanedbat. Mezi nejznámější, zde nežádoucí, jindy však chtěné deformace obrazu patří například tzv. efekt rybího oka.

Spojitost mezi středovým promítáním a camerou obscurou je následující. Otvor uvnitř stěny je v terminologii středového promítání ekvivalentní středu promítání. Protější stěna je ekvivalentní průmětně (rovinně promítání) s tím rozdílem, že průmětna je od středu promítání směrem ke scéně, zatímco promítací stěna je směrem opačným. Z toho důvodu vzniká na stěně dírkové kamery převrácený obraz. Dalším důležitým parametrem obou těchto modelů je vzdálenost středu promítání od průmětny respektive otvoru od promítací stěny, to ovlivňuje, jak velký bude obraz scény na průmětně.

Všechny parametry můžeme rozdělit na vnější parametry

Poloha kamery světové souřadnice středu promítání

Orientace kamery vektor kolmý k rovině promítání

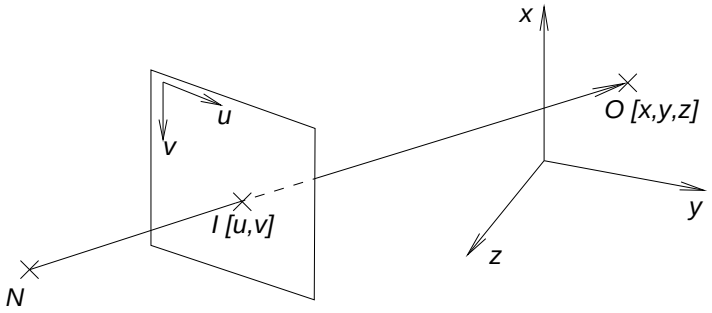
a vnitřní parametry

Ohnisková vzdálenost vzdálenost středu promítání od průmětny

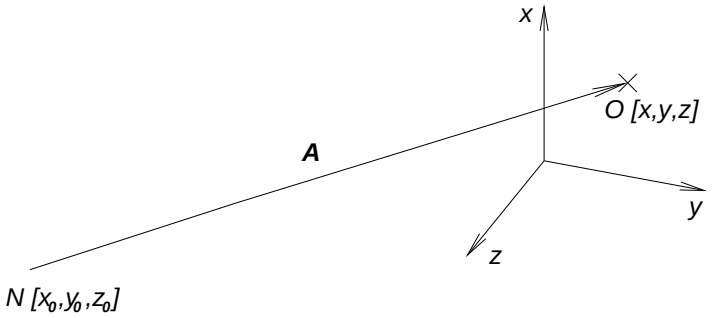
Hlavní bod obrazové souřadnice průsečíku optické osy (přímka procházející středem promítání kolmá k průmětně) s průmětnou

Změna měřítka poměr obrazových a světových jednotek pro udávání souřadnic

Při znalosti všech těchto parametrů je přesně zadaná korespondence mezi body obrazu a promítacími paprsky. Pokud je navíc uvažován dírkový model kamery, je tento vztah lineární. Nejčastěji se tento vztah zadává pomocí matice. Takovou matici nazýváme kalibrační maticí kamery a proces zjišťování těchto parametrů kalibrací kamery. Kalibrační techniky můžeme rozdělit na fotogrammetrické a automatické. Při fotogrammetrické kalibraci je před kamery předkládán speciální kalibrační objekt a ze znalosti jeho přesné geometrie jsou určeny parametry kamery. Pro automatickou kalibraci není potřeba žádný kalibrační objekt. Místo toho je ale třeba znalost vzájemně si odpovídajících bodů v obrazu a jejich polohy v prostoru. Pomocí těchto dvojic jsou následně parametry odhadnuty. Takovýchto bodů je potřeba znát dostatek, např. pro jedenáct neznámých parametrů kamery, je třeba znát alespoň šest sobě odpovídajících dvojic bodů z obrazu a ze scény. Pokud je známo více dvojic než je nezbytně nutné, jsou redundantní informace použity ke zpřesnění odhadů.



Obrázek 3.3: Kolinearita bodů scény a obrazu



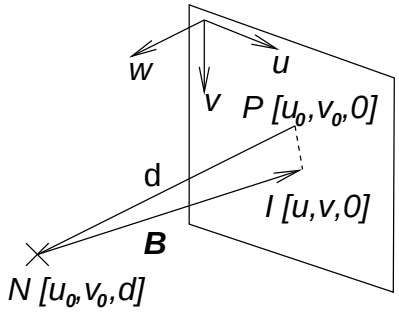
Obrázek 3.4: Vektor ze středu promítání do prostoru scény

3.3 Direct Linear Tranformation

Jednou z nejstarších metod pro automatickou kalibraci kamery je metoda Direct Linear Transformation [1]. Metoda hledá závislost souřadnic bodu v prostoru a jeho obrazu na promítací rovině. Na promítací rovině je totiž použit jiný 2D systém obrazových souřadnic u, v než je systém 3D systém světových souřadnic x, y, z, pomocí něhož popisujeme polohu objektů ve scéně.

Kamera zobrazuje bod scény O se souřadnicemi $[x, y, z]$ na bod obrazu I (se souřadnicemi $[u, v]$) takový, že je průsečíkem průmětny s promítacím paprskem, tedy přímkou obsahující střed promítání N a bod O. Body I a O tedy leží na jedné přímce (jsou kolineární) a tato kolinearita je základem metody DLT, viz obrázek 3.3.

Nechť střed promítání N má světové souřadnice $[x_0, y_0, z_0]$. Potom pro vektor A ze středu promítání do bodu scény O platí $A = (x-x_0, y-y_0, z-z_0)$, viz obrázek 3.4.



Obrázek 3.5: Vektor ze středu promítání na průmětnu

Pokud obrazové souřadnice rozšíříme o osu W, kolmou na osy U a V, můžeme zapsat polohu středu promítání i v souřadnicovém systému obrazu jako $[u_0, v_0, d]$, kde $P = [u_0, v_0, 0]$ je tzv. hlavní bod, nejbližší bod promítací roviny ke středu promítání a d je tzv. ohnisková vzdálenost, vzdálenost středu a roviny promítání, viz obrázek 3.5.

Nechť $B = (u - u_0, v - v_0, -d)$ je vektor ze středu promítání do bodu obrazu I, potom díky kolinearitě platí

$$B = cA, \tag{3.1}$$

kde c je skalár. Jelikož ale byly vektory A a B popsány každý v jiném souřadnicovém systému, musíme je pospat v jednom společném systému. Jednou z možností jak toho docílit je převod vektoru A do obrazových souřadnic. To lze provést následovně

$$A^{(I)} = T_{(O\to I)}A^{(O)} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} A^{(O)}, \tag{3.2}$$

kde vektor $A^{(I)}$ značí vektor A zapsaný pomocí obrazových souřadnic, zatímco $A^{(O)}$ značí stejný vektor jen zapsaný v souřadnicích světových. $T_{(O\to I)}$ je transformační matice pro převod z prostoru světových souřadnic do prostoru obrazu. Z 3.1 a 3.2 plyne

$$B = c\ T_{(O\to I)}\ A \tag{3.3}$$

to lze přepsat jako

$$\begin{pmatrix} u - u_0 \\ v - v_0 \\ -d \end{pmatrix} = c \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{pmatrix}$$

nebo

$$u - u_0 = c\left[r_{11}\left(x - x_0\right) + r_{12}\left(y - y_0\right) + r_{13}\left(z - z_0\right)\right], \tag{3.4}$$
$$v - v_0 = c\left[r_{21}\left(x - x_0\right) + r_{22}\left(y - y_0\right) + r_{23}\left(z - z_0\right)\right], \tag{3.5}$$
$$-d = c\left[r_{31}\left(x - x_0\right) + r_{32}\left(y - y_0\right) + r_{33}\left(z - z_0\right)\right]. \tag{3.6}$$

Z 3.6 můžeme získat konstantu c

$$c = \frac{-d}{r_{31}\left(x - x_0\right) + r_{32}\left(y - y_0\right) + r_{33}\left(z - z_0\right)}. \tag{3.7}$$

a po dosazení do 3.4 a 3.5 získáme

$$u - u_0 = -d\,\frac{r_{11}\left(x - x_0\right) + r_{12}\left(y - y_0\right) + r_{13}\left(z - z_0\right)}{r_{31}\left(x - x_0\right) + r_{32}\left(y - y_0\right) + r_{33}\left(z - z_0\right)}, \tag{3.8}$$
$$v - v_0 = -d\,\frac{r_{21}\left(x - x_0\right) + r_{22}\left(y - y_0\right) + r_{23}\left(z - z_0\right)}{r_{31}\left(x - x_0\right) + r_{32}\left(y - y_0\right) + r_{33}\left(z - z_0\right)}. \tag{3.9}$$

Je třeba si všimnout, že souřadnice u, v, u_0 a v_0 jsou ve všech rovnicích udávány ve stejných jednotkách jako světové souřadnice, tedy pravděpodobně v nějakých běžně používaných jako třeba v cm. Pro nás je ale daleko přirozenější udávat tyto jednotky v pixelech. Pro převod na pixely tedy zavedeme dvě konstanty λ_u a λ_v, které budou zajišťovat změnu měřítka.

$$u - u_0 \Rightarrow \lambda_u\left(u - u_0\right)$$

$$v - v_0 \Rightarrow \lambda_v\left(v - v_0\right)$$

$$u - u_0 = -\frac{d}{\lambda_u}\,\frac{r_{11}\left(x - x_0\right) + r_{12}\left(y - y_0\right) + r_{13}\left(z - z_0\right)}{r_{31}\left(x - x_0\right) + r_{32}\left(y - y_0\right) + r_{33}\left(z - z_0\right)} \tag{3.10}$$
$$v - v_0 = -\frac{d}{\lambda_v}\,\frac{r_{21}\left(x - x_0\right) + r_{22}\left(y - y_0\right) + r_{23}\left(z - z_0\right)}{r_{31}\left(x - x_0\right) + r_{32}\left(y - y_0\right) + r_{33}\left(z - z_0\right)} \tag{3.11}$$

Nyní můžeme rovnice naposledy přepsat do přehlednějšího tvaru:

$$u = \frac{L_1 x + L_2 y + L_3 z + L_4}{L_9 x + L_{10} y + L_{11} z + 1} \tag{3.12}$$
$$v = \frac{L_5 x + L_6 y + L_7 z + L_8}{L_9 x + L_{10} y + L_{11} z + 1}, \tag{3.13}$$

kde

$$d_u = -\frac{d}{\lambda_u} \qquad d_v = -\frac{d}{\lambda_v}$$

$$D = -\left(r_{31}x_0 + r_{32}y_0 + r_{33}z_0\right)$$

$$L_1 = \frac{u_0 r_{31} - d_u r_{11}}{D} \qquad L_2 = \frac{u_0 r_{32} - d_u r_{12}}{D} \qquad L_3 = \frac{u_0 r_{33} - d_u r_{13}}{D}$$

$$L_4 = \frac{\left(d_u r_{11} + u_0 r_{31}\right) x_0 + \left(d_u r_{12} + u_0 r_{32}\right) y_0 + \left(d_u r_{13} + u_0 r_{33}\right) z_0}{D}$$

$$L_5 = \frac{v_0 r_{31} - d_v r_{21}}{D} \qquad L_6 = \frac{v_0 r_{32} - d_v r_{22}}{D} \qquad L_7 = \frac{v_0 r_{33} - d_v r_{23}}{D}$$

$$L_8 = \frac{\left(d_v r_{21} + v_0 r_{31}\right) x_0 + \left(d_v r_{22} + v_0 r_{32}\right) y_0 + \left(d_v r_{23} + v_0 r_{33}\right) z_0}{D}$$

$$L_9 = \frac{r_{31}}{D} \qquad L_{10} = \frac{r_{32}}{D} \qquad L_{11} = \frac{r_{33}}{D}.$$

Koeficienty L_1 až L_{11} nazýváme DLT parametry. Vyjadřují vztah obrazových a světových souřadnic. Pokud známe bod v prostoru můžeme k němu dopočítat souřadnice jeho průmětu. Na druhou stranu pro bod na obrazové rovině máme jednoznačně určený promítací paprsek (přímku v prostoru). Jelikož obrazové informace jsou pouze dvoudimenzionální, musíme pro jednoznačné určení polohy bodu v třídimenzionálním prostoru získat ještě nějakou další informaci (tu získáme z laseru).

Při automatické kalibraci obrazu se koeficienty L_1 až L_{11} získávají ze znalosti několika dvojic korespondujících si bodů v prostoru a v obrazu. Rovnice 3.12 a 3.13 můžeme přepsat následovně:

$$L_1 x + L_2 y + L_3 z + L_4 - u L_9 x - u L_{10} y - u L_{11} z = u$$

$$L_5 x + L_6 y + L_7 z + L_8 - v L_9 x - v L_{10} y - v L_{11} z = v$$

To je ekvivalentní vektorovému zápisu:

$$\begin{pmatrix} x & y & z & 1 & 0 & 0 & 0 & 0 & -ux & -uv & -uz \\ 0 & 0 & 0 & 0 & x & y & z & 1 & -vx & -vy & -vz \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \\ L_6 \\ L_7 \\ L_8 \\ L_9 \\ L_{10} \\ L_{11} \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix}$$

Pro jednu dvojici, bod obrazu - bod scény, získáme dvě rovnice, kde máme celkem jedenáct neznámých (DLT parametry L_1 - L_{11}). Pro jednoznačné určení všech parametrů je tedy potřeba alespoň šesti dvojic bodů, tyto body navíc musejí být nezávislé. Pokud budeme znát šest dvojic nebo ještě více, stačí nám to na odhadnutí parametrů. Jelikož je ale neznámých jen jedenáct a my ze šesti bodů získáme rovnic dvanáct, je možné, že nebudou vzájemně "kompatibilní" (použití různých jedenácti rovnic bude dávat různé výsledky).

Proto větší počet rovnic využijeme pro zpřesnění odhadu parametrů. Metoda, která se na to používá se nazývá Metoda nejmenších čtverců pro soustavu lineárních rovnic.

Mějme matici A o rozměrech n krát m a sloupcový vektor b délky m. Úkolem je najít takový vektor x (délky n), aby platilo $Ax = b$. Pokud je $m < n$, je takových řešení nekonečně mnoho a pro jednoznačné určení všech neznámých to nestačí. Pokud je $m = n$ a matice A je regulární, existuje právě jeden vektor x, tak aby byla splněna požadovaná rovnice. V našem případě je však $m > n$, pak nazýváme soustavu rovnic $Ax = b$ přeučenou. U přeučených soustav se může stát, že nebude mít žádné řešení.

Cílem metody nejmenších čtverců je najít pro přeučenou soustavu rovnic takový vektor x, aby rozdíl $Ax - b$, byl co nejmenší. Tento rozdíl je nazýván chybovým vektorem. Chybový vektor bude nejmenší, právě když bude kolmý na vektor Ax a dva vektory jsou kolmé, právě když je jejich skalární součin roven nule. Hledáme tedy vektor x, pro nějž by platilo

$$(Ax - b)^T (Ax) = 0,$$

$$(x^T A^T - b^T)Ax = 0,$$
$$(x^T A^T A - b^T A)x = 0.$$

Poslední rovnice platí pokud, $x = 0$ (toto řešení však obvykle není to hledané) nebo pokud

$$x^T A^T A = b^T A,$$
$$A^T Ax = A^T b,$$
$$x = (A^T A)^{-1} A^T b.$$

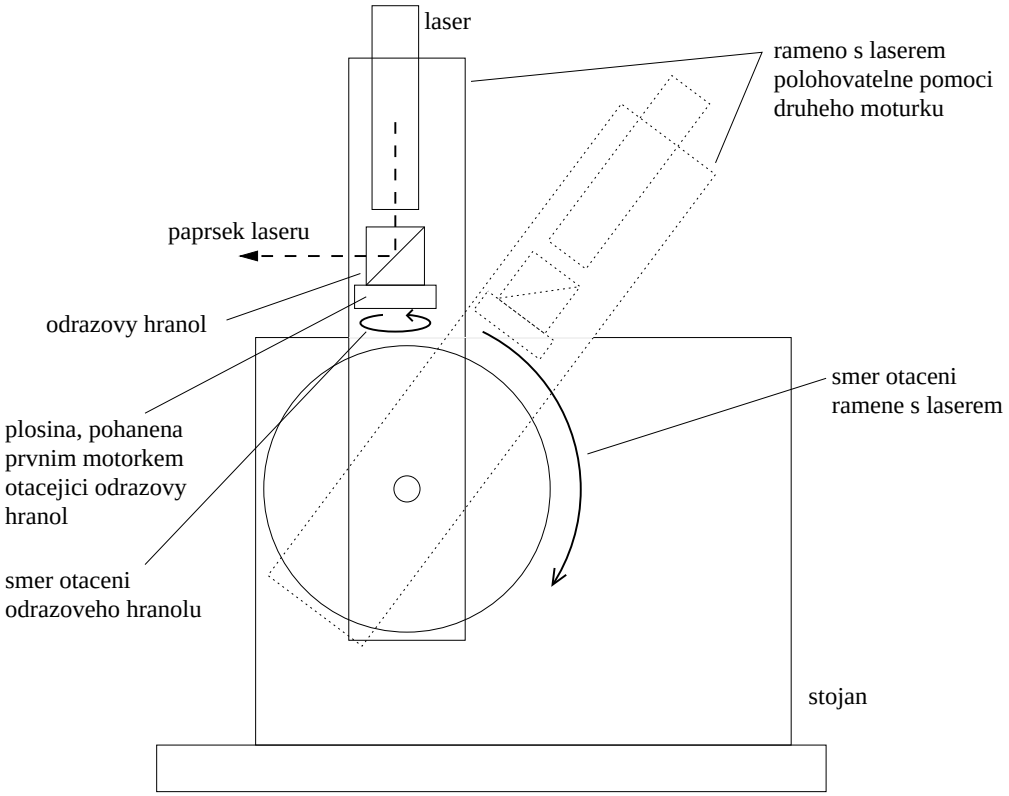
Pomocí metody nejmenších čtverců tedy můžeme najít takový vektor DLT parametrů, aby pro všechny vstupní dvojice bodů ze scény a bodů ze snímku byla chyba (respektive vektor chyb) minimální.

3.4 Laser

Jak již bylo uvedeno, laser se v triangulační metodě používá pro získání informace o třetím rozměru. V dvourozměrné fotografii totiž můžeme zjistit, jaké přímky v prostoru jsou reprezentovány jednotlivými pixely obrázku, ale ačkoli můžeme na snímku detekovat hrany nebo s určitou úspěšností i nalézat spojité stěny objektů, nebude získaná reprezentace zcela přesná. Laser nám v podstatě poskytuje pohled na předmět z jiného úhlu, než z kterého je snímán kamerou a to nám poskytne možnost získat informaci o hloubce, stejně jako nám fakt, že máme dvě oči, které sledují okolí každé z mírně odlišné pozice, umožňuje vidět prostorově.

Původní plán na realizaci tohoto projektu byl následující. Kamera se v průběhu scannování nebude pohybovat a bude snímat předmět z jednoho místa. Stejně tak ani laser nebude měnit svoji polohu a bude svítit na předmět stále jedním směrem. Naproti tomu snímaný předmět bude na otočné ploše. Tím pádem se bude v průběhu scannování otáčet a navíc se bude otočná plocha spolu s předmětem pomalu zvedat. Výhodou tohoto způsobu by byla možnost nascannovat předmět najednou ze všech stran. Nevýhodou naopak to, že by bylo třeba mít dvě specializované součástky, laser a otočnou plochu. Navíc by se buď musela detekce laserového bodu řešit jinak, než jak je tomu zde, tzn. pomocí rozdílových snímků, nebo by musel laser blikat, což by zkomplikovalo jeho konstrukci.

Nakonec byl zvolen ten model scanneru, ve kterém je předmět stejně jako kamera statický a nemění svoji polohu ani natočení, zatímco laser mění



Obrázek 3.6: Zjednodušený náčrt laseru

směr svého záření. Tento způsob má hlavní nevýhodu v tom, že neposkytuje náhled na scannovaný předmět ze všech úhlů, ale jen z toho z kterého je snímán kamerou. Výsledek je kvůli tomu spíše plastickým obrazem než plnohodnotným 3D modelem. Je v něm tedy zachycena 3D informace, neposkytuje však například možnost podívat se na předmět ze všech stran. Na druhou stranu nám tato metoda poskytne možnost pohledu na celou scénu a ne jen na scannovaný předmět, zda je to výhodou záleží na situaci.

Laser byl vyroben a poskytnut k užívání vedoucím práce, za což bych mu ještě jednou rád poděkoval. Laser je napájen z USB, ovládán je ale pomocí sériového portu. Jeho schéma je vidět na obrázku 3.6. Po zapojení do USB začne laser svítit. Sám však nemíří přímo na scénu, ale na odrazový hranol, který mění směr jeho záření o 90°. Hranol se při zapojení laseru do USB začne otáčet, čímž se směr záření neustále mění, stále jsou však ozařovány jen body v jedné rovině, v rovině pro níž je původní směr záření vycházející

přímo z laseru normálovým vektorem. Další možností změny směru paprsku je druhý motor, který otáčí ramenem, na němž je umístěn jak samotný laser, tak i otočný odrazový hranol. Toto rameno se otáčí kolem osy kolmé na původní směr laseru (osa je rovnoběžná s rovinou prozatím ozařovaných bodů) a jeho pohybem se dá "ozařovaná" rovina naklánět. První motor tak mění horizontální úhel, zatímco druhý mění vertikální.

Laser umožňuje přes sériový port následující možnosti. V průběhu otáčení odrazového hranolu je vždy v určitém úhlu natočení sepnut mikrospínač a tím poslán signál na sériový port. Tuto informaci lze použít pro zjištění natočení odrazového hranolu (směru paprsku po odražení) ve chvíli pořízení snímku kamerou. Nakonec ale tato informace využita není a tento signál zůstává neindikován. Je to z toho důvodu, že použitá kamera nedokáže snímek pořídit dostatečně rychle. Při této rychlosti otáčení (rychlost už nelze snížit) se laserový bod na snímku rozmaže do poměrně dlouhé čáry a informace o natočení hranolu, tak ztrácí svou přesnost a s ní i význam. Proto nebudeme detekovat bod, ale "čáru", která na snímku zářením laseru vznikne.

Druhou možností, kterou sériové rozhraní laseru umožňuje je sepnout obvod s druhým motorem. Po zapojení tohoto obvodu se motor rozeběhne a tím přes soustavu ozubených koleček začne otáčet ramenem s laserem a odrazovým hranolem. Stejně jako v prvním případě je i tento pohyb jednosměrný, rozdíl je však v tom, že tento motor nemůže otočit ramenem o 360° a tudíž jakmile je ramenem jednou pohnuto, nelze ho programově vrátit zpět. Pohyb na výchozí pozici se musí realizovat manuálně. Teoreticky umožňuje motor natáčet rameno o libovolný úhel, ve skutečnosti je ale lepší otáčet ramenem jen krokově. Čím je totiž pohyb ramene delší, tím je rychlost ramene větší a nelze přesně určit výsledné otočení.

3.5 Kalibrace laseru

Stejně jako pro kameru je i pro laser třeba zjistit informace o jeho poloze a natočení, aby se je následně dalo použít při rekonstrukci 3D informací. V zásadě jsou dvě možnosti jak to dělat.

Tyto informace se mohou zjistit na začátku scannování, podobně jako se to dělá v případě DLT pro kameru. Rozdíl je však v tom, že laser se pohybuje a je proto třeba tyto informace upravovat. Každý pohyb ramene se zdrojem laseru způsobí mírné otočení roviny záření a zároveň s tím i její malý posun. Pokud by šlo tyto změny zadat přesně tak, jak se následně vykonají, šlo

by tento způsob bez problémů použít. Laser však na stejné příkazy nemění polohu vždy zcela identicky (změny jsou mírně odlišné).

Proto bylo navrženo použít druhý způsob, při kterém nejsou žádné informace o laseru zjišťovány předem a vše je vyhodnocováno až v průběhu scannování. Myšlenka tohoto způsobu je následující. Předmět je umístěn v rohu a informace o tomto rohu jsou známy (informacemi jsou míněny rovnice všech tří rovin tvořící tento roh). Jak je předmět osvětlován laserem, jsou zároveň osvětlovány i stěny rohu. Podmínkou v tomto případě je, že scannovaný předmět nezabírá celý záběr kamery (po stranách musí být trochu místa). Předpokládejme, že jsou osvíceny alespoň dvě stěny. Pokud detekujeme ozářené body na snímku, můžeme pomocí DLT a informace, na které stěně body leží, zjistit jejich 3D souřadnice. Všechny tyto body však leží v jedné rovině (rovině záření laseru), ze znalosti jejich 3D souřadnic tedy můžeme odhadnout polohu a orientaci, respektive rovnici, této roviny. Tu pak můžeme použít pro zjištění souřadnic zbylých ozářených bodů (bodů scannovaného předmětu). Předpoklad, že jsou osvíceny alespoň dvě stěny, je zde z toho důvodu, že osvícení jedné stěny nám nestačí. Rovina záření se totiž s nějakou stěnou (také rovinou) protne v jediné přímce. Ozářené body na jedné stěně tedy leží v jedné přímce, to ale nestačí pro odhad roviny záření. Přímka totiž prochází nekonečně mnoha rovinami. Rovina je zadaná teprve třemi body, které v přímce neleží, proto je třeba, aby byly osvíceny alespoň dvě stěny.

3.6 Detekce laseru na snímku

Ačkoli už bylo několikrát zmíněno vyhledání laseru na snímku kamery, nebyla prozatím tato technika vysvětlena. Stejně jako u zbylých problémů i zde existuje více způsobů řešení. Je možné na snímku vždy vyhledávat určité rysy, jako jsou například vysoké hodnoty jednotlivých barevných složek, konkrétní barva, prudká změna barev na okrajích laserového bodu a podobně. Každá z těchto metod má kromě svých silných stránek i slabé. Vysoké hodnoty se například kromě laseru mohou vyskytnout i na nějakých odrazových hranách, stejně tak mohou být na snímku i rychlé změny barev. Zaměření se na detekci určité barvy by zase mohlo přinést omezení na to, jaké barvy mohou mít scannované předměty. Celkově bychom mohly říct, že tyto techniky jsou zaměřené na hledání určitých rysů či vzorů a nejlepší by pravděpodobně bylo hledat kombinaci více jednotlivých znaků.

V našem případě se však dá využít faktu, že kamera ani scéna se při scan-

nování nehýbou. Jediné, co se tedy na snímku mění, je místo, na které dopadá laserové záření. Pro detekci laseru jde tohoto faktu využít následovně. Před počátkem scannování bude pořízen tzv. referenční snímek, na kterém bude zachycena scéna ještě bez laserového záření. Následně vždy, když bude požadavek na to nalézt na nějakém novém snímku laser, nejdříve od tohoto snímku odečteme ten referenční a teprve ve výsledku budeme laser vyhledávat. Výhodné je to proto, že ve scéně se kromě laserového bodu nic nehýbe a tudíž pokud odečteme referenční snímek, zbude tam právě jen tento bod. Navíc tam samozřejmě může zbýt i nějaký šum, ale takto by mělo být daleko snazší se s ním vypořádat.

Předpokládejme tedy, že na snímku máme jen ty pixely, na nichž je zachycen laser. Zbývá nám ještě nějak určit, který konkrétní je právě osvícen. To znamená vybrat všem ozářeným pixelům jediného reprezentanta. Intuitivně lze říct, že by měl být někde ve středu. Možností je například spočítat průměrné souřadnice všech pixelů laserového bodu. Jako lepší se ale pravděpodobně ukáže počítat průměr vážený. Váhou zde bude hodnota barevných složek pixelů. Lepší asi bude použít hodnoty pixelů z původního snímku s laserem ještě před odečtením referenčního snímku. Hodnoty pixelů laserového bodu jsou totiž dosti podobné, ať svítí na jakkoli barevný povrch (záleží spíše na materiálu a jeho odrazových vlastnostech). Pokud by se tedy použil snímek rozdílový a laser by svítil zrovna na rozhraní světlé (vysoké hodnoty barevných složek) a tmavé plochy (nižší hodnoty barev), byla by vyšší váha přiřazena pixelům z tmavé části. Od přibližně stejných hodnot laserového bodu by se totiž v tmavé části odečetly nižší hodnoty než v části světlé. To by následně způsobilo posun hledaného středu do tmavých míst.

3.7 Výstup

Pomocí již popsaných technik by mělo jít získat souřadnice bodů scannovaného předmětu a měli bychom mít k dispozici tzv. oblak bodů (cloud of points). Z referenčního snímku můžeme zase vyčíst jejich barvy. Zbývá tedy jen popsat, jak tyto informace předat dál tak, aby mohly být zobrazeny nebo s nimi šlo jinak dále pracovat. Před začátkem projektu bylo navrhnuto napsat jednoduchý program pro zobrazování 3D dat a zobrazovat získané informace pomocí něho. Nakonec bylo ale rozhodnuto, že lepší bude vydat se cestou uznávaných standardů. Výstup by měl být v nějakém uznávaném standardním formátu, aby šel následně zobrazovat více různými programy a pomocí různých nástrojů mohl být převáděn do dalších formátů ke speci-

fickému použití.

Za formát pro výstup byl tedy zvolen jazyk VRML [14]. Jedná se o celosvětově uznávaný formát a ačkoli je už trochu starší je stále podporován. VRML soubory jsou obyčejné textové soubory, které popisují vždy objekt nebo scénu. Mají hierarchickou strukturu a jednotlivé objekty se v nich mohou skládat z mnoha menších. Základní objekty mohou být např. jednoduchá geometrická tělesa jako je koule, válec nebo kvádr. To pro naše účely ale není relevantní, my chceme zobrazit data bez toho abychom je aproximovali základními tělesy. Pro výstup projektu budou spíše vhodné objekty IndexedFaceSet, IndexedLineSet nebo PointSet.

Uzel PointSet definuje množinu bodů, které mohou mít různou barvu, to je tedy přesně to, co prozatím získáme při použití popsaných technik. Nevýhodou ovšem je, že objekt nebude vypadat jako jedno těleso, ale opravdu jako množina bodů. Uzel IndexedLineSet definuje objekt složený z lomených čar. Ideálně se proto hodí na reprezentaci drátěného modelu. Pro zobrazení dat by bylo možné použít tohoto uzlu, stále by však nebylo docíleno toho, aby scéna alespoň přibližně vypadala tak jako ve skutečnosti (aby předměty měly stěny). To by měl umožňovat uzel IndexedFaceSet. Jedná se o indexovanou množinu mnohoúhelníků tzn. objekt složený z množiny polygonů.

Nyní však potřebujeme rozhodnout, z kterých získaných bodů udělat jeden polygon. V počítačové grafice se většinou nepoužívají obecné mnohoúhelníky, ale zpravidla vždy se pracuje pouze s trojúhelníky. My zvolíme stejný přístup. Jelikož získáváme jednotlivé body postupně tak, že najednou dostaneme body v jedné rovině a následně získáváme body z roviny o něco výš, stačí vždy do jednoho trojúhelníku vzít dva body z jedné roviny a přidat k nim jeden bod z roviny nad nimi, který se nachází blízko nich. Ten najdeme tak, že bude mít na snímku přibližně stejnou vertikální souřadnici. Takto vytvořené trojúhelníky sdružíme do jednoho tělesa uzlem IndexedFaceSet. Jeden uzel nám tedy bude vytvářet jeden "řádek" trojúhelníků. Všechny získané řádky pak spojíme do jednoho nadřazeného tělesa a to bude požadovaným výstupem programu.

Kapitola 4

Implementace

Implementace by měla sledovat postupy navrhnuté v analytické části. Objevili se ale některé problémy, s kterými se v při analýze nepočítalo. Implementační sekce se tedy zabývá tím, jak uskutečnit analytické vize a jak se vyrovnat s neočekávanými problémy.

Jako první se zaměříme na jednotlivé moduly, které následně používá hlavní část programu k tomu, aby uskutečnila scannování a vydala požadovaný výstup. Jedním z požadavků na projekt bylo, že program bude napsán v jazyce C pro operační systém Linux, to však bylo určující hlavně pro části programu určené pro komunikaci s hardwarem (kamera a sériový port), popřípadě pro využívání souborového systému a vytváření uživatelského prostředí. Části, které se nezabývají těmito problémy, nejsou na Linuxu nijak závislé.

4.1 Komunikace s kamerou

Pro účely tohoto projektu byla fakultou poskytnuta k užívání webová kamera Logitech QuickCam chat [9]. Pro tuto kameru jsou pro Linux běžně dostupné ovladače Gspca [10], s jejichž pomocí už je komunikace s kamerou poměrně snadná a to díky tomu, že Linux nabízí pro spojení s video zařízeními, jednotné rozhraní prostřednictvím jaderného modulu V4L (Video For Linux) [16]. O komunikaci s kamerou prostřednictvím V4L se starají funkce z modulu `dev_video.c`, alespoň zběžně tu ty nejdůležitější popíšeme.

Funkce `open_device()` inicializuje video zařízení (kameru) tak, aby se s ním dalo komunikovat a mohly z něj být získávány jednotlivé snímky. První parametr je řetězec určující adresu zařízení v souborovém systému (nejčastěji

Obrázek 4.1: Chybný snímek

soubor `/dev/video0`). Další dva parametry určují jas a kontrast s jakými se má kamera inicializovat. Návratová hodnota je odkaz, pomocí kterého se dále se zařízením komunikuje. Každé otevřené zařízení má takovýto odkaz. Uvnitř funkce se pomocí systémové funkce `ioctl()` získávají a nastavují další parametry, jako je například jméno kamery, informace o ovladačích či rozměry snímku.

Druhou důležitou funkcí v modulu je funkce `take_picture()` Umožňuje pořízení několika po sobě následujících snímků. První parametr je ukazatel na předem alokované pole pro výsledek. Jelikož je každý pixel snímku uložen ve třech bytech, je požadovaná velikost tohoto pole rovna šířce snímku krát výška snímku krát tři krát požadovaný počet snímků. Druhým parametrem je právě počet za sebou jdoucích snímků, které chceme zaznamenat a posledním argumentem je již zmíněný odkaz na zařízení, co nám má snímky dodat.

Komunikace s kamerou tedy není nijak složitá, ale právě při ní se vyskytly nečekané problémy. První z nich byl ten, že jednou za čas kamera vydala chybný snímek, viz obrázek 4.1. Nicméně s upgradem Linuxového jádra a ovladačů kamery se problém zhoršil tak, že při původním rozlišení se už vyskytovaly jen takovéto snímky. Po snížení požadavků na rozlišení snímků, tento problém zmizel, je to však jeden z důvodů, proč není uživateli dovoleno rozlišení měnit.

Druhý, závaznější problém s kamerou je, že se občas po zadání požadavku kameře program zastaví a to tak, že uvízne ve volání některé ze systémových funkcí. Ačkoli byl zkoušen ještě jeden přístup ke čtení dat z kamery (místo funkce `read()` pomocí funkcí `ioctl()` a `mmap()`, viz [16]), zlepšení nebylo

dosaženo. Otázkou zůstává, zda to je problém kamery nebo problém ovladačů. Projekt byl totiž testován se dvěma kamerami, které mají sice rozdílný chipset, ale jsou stejného typu. S úplně jinou kamerou by tedy tyto problémy možná nevznikaly. Řešení tohoto problému je popsáno dále v části 4.5.

4.2 Komunikace s laserem

Dle návrhu z analytické části je jediná věc kvůli níž s laserem komunikujeme jeho vertikální natočení. Laser pravidelně vysílá signál, když při horizontální rotaci prochází určitým bodem, ten je však ignorován. Veškeré předávání informací probíhá přes sériový port, protože ačkoli je třeba laser zapojit do USB, je z něho pouze napájen. O sériový port se starají funkce z modulu `dev_serial.c`.

Stejně jako video zařízení je i sériový port třeba otevřít a získat na něj odkaz. To dělá funkce `open_serial_device()`, jejíž jediným parametrem je cesta k portu v souborovém systému (nejčastěji `/dev/ttyS0`).

Jsou dvě možnosti, jak způsobit požadovaný vertikální pohyb laseru. První možností je nastavit na konkrétní komunikační lince požadovaný výstupní bit, počkat určitou dobu a následně nastavit opět bit znamenající žádný pohyb. Druhým způsobem, aplikovaným v tomto projektu, je nastavit sériovému portu nějakou komunikační rychlost, to zajišťuje funkce `set_serial_speed()` a zapisovat znaky na sériový port. Komunikační rychlost určuje, jak dlouho je každý výstupní bit dlouho na komunikační lince, zatímco znaky posílané na výstup určují, jaké bity tam budou. Jeden horizontální krok je pak uskutečňován pomocí funkce `serial_step()`, jejíž argumentem je pak právě znak, který posíláme na výstup.

Více informací o sériovém portu pod Linuxem je např. v [13].

4.3 DLT

Metoda DLT byla implementována dle návrhu a je obsažena v modulu `dlt.c`. Díky funkci `spocti_koeficienty()`, lze po dodání šesti a více dvojic bodů z obrazu a ze scény, určit dlt koeficienty. S takto zjištěnými parametry a pomocí funkce `najdi_bod()`, můžeme pro nějakou rovinu (například rovinu záření laseru) a nějaký bod obrazu, spočítat přesnou polohu bodu ve scéně. To stejné, ale s více body respektive se s jejich spojovým seznamem umožňuje funkce transformuj souřadnice.

Modul `dlt.c` využívá mnoho funkcí pro práci s maticemi. Ty jsou obsaženy v modulu `matice.c` a umožňují násobení, transpozici, inverzi matic. A dále s nimi jde vyřešit soustavu rovnic a nebo najít řešení s nejmenší chybou (pro přeučené soustavy).

4.4 Odhad roviny laserového záření

Proto abychom určili rovinu, kterou se šíří laserové záření, musíme nejdříve zjistit, které body obrazu reprezentují dopad laseru. K tomu slouží funkce z modulu `image_operation.c`. Tento modul nám umožňuje provádět relativně jednoduché operace s jednotlivými snímky. Můžeme je odčítat a pokud odečteme snímek, na kterém není laser zachycen, od nějakého, kde laser je, získáme právě body laserem ozářené. Pokud toto zopakujeme se všemi snímky, můžeme je opět pomocí jedné z funkcí sečíst do jednoho.

Poslední důležitou součástí tohoto modulu je funkce `najdi_laser()`, jako vstup bere nějaký rozdílový snímek na němž je zachycen laser. Postupně pak prochází celý snímek po sloupcích a pokud najde nějaké sousední pixely reprezentující laser, spočítá vážený průměr jejich souřadnic a tím určí jejich nejvhodnějšího reprezentanta (hledá vlastně střed laserového paprsku).

Pokud tedy známe souřadnice bodů reprezentujících laser, můžeme se snažit určit nebo alespoň odhadnout rovinu, v které se laserové záření šíří. K tomu slouží modul `laser_kalib.c` a uvnitř tohoto modulu především funkce `odhad_roviny_laseru()`. Má poměrně hodně parametrů, prvním parametrem jsou jednotlivé body laseru nalezené na snímku, druhým jsou dlt koeficienty. Dalším je informace o tom, ve kterých krajních sloupcích snímku již není scannovaný předmět, ale stěny umístěné v pozadí. Posledním vstupním parametrem jsou pak rovnice rovin popisujících právě tyto stěny. Jelikož je to jedna z nejdůležitějších funkcí, alespoň zběžně jí zde popíšeme.

Funkce postupně prochází všechny body laseru a u těch, které nejsou ze scannovaného předmětu ale z krajních sloupců (respektive z pozadí tvořeného třemi stěnami v jejichž rohu je předmět umístěn), se snaží určit jejich polohu ve scéně. To dělá tak, že vybere stěnu, do které bod patří a pomocí metody nejmenších čtverců (v modulu `matice.h`) odhadne pomocí rovnice této roviny a znalosti dlt koeficientů polohu bodu ve scéně. Pokud jsou body alespoň ze dvou stěn, měli bychom v ideálním případě určeny dvě různoběžné přímky určující jednu rovinu, rovinu laseru. Ve skutečnosti máme jen množinu bodů laseru, pomocí nichž rovnici roviny odhadneme opět díky metodě nejmenších čtverců.

4.5 Ukládání informací

Jak již bylo popsáno dříve v části 4.1, program se občas "zasekne" při volání nějaké ze systémových funkcí pracujících s kamerou. Není to problém špatné práce s kamerou, neboť tyto problémy mají i ostatní programy, které s kamerou komunikují (např. známý program Ekiga určený pro videotelefonii). Přesto byla snaha nějak tento problém obejít, aby mohl být projekt dokončen. Řešení problému nakonec poskytlo ukládání veškerých potřebných informací na disk, kde zůstávají, dokud je program sám nepřepíše nebo do konce scannování. Diskové operace jsou z principu výrazně pomalejší než práce z pamětí, programu to však vůbec nevadí neboť z časového hlediska je pro něj kritická práce s kamerou a laserem a diskové operace ho nijak znatelně nezpomalují.

Dočasné informace se ukládají do několika souborů, které jsou umístěny v adresáři `/tmp/SimpleScanner."login uživatele"`. Pokud by se login nepodařilo zjistit je tato část vynechána. Adresář je tedy vždy unikátní pro každého uživatele, ale pro daného uživatele je to jediný "odkládací" prostor. Tím však vzniká hrozba současného vícenásobného běhu a vzájemnému přepisování uložených informací. Při logickém používání programu tato situace sice nemůže nastat, protože laser je unikátní a nelze tedy předpokládat, že by program mohl být spuštěn vícekrát zároveň, protože laser stěží může scannovat dvě různé věci najednou. Navíc lze video zařízení otevřít vždy naráz jen jedním programem, to znamená, že k přepisu nedojde, jelikož je vždy před ukládáním informací kamera "otevírána", nemůžou oba spuštěné programy zapisovat do souborů. A i pokud by to nastalo, informace jsou stejně vždy načítány jen na začátku, takže daný běh programu by tím ovlivněn nebyl.

4.6 Výstup

I při implementaci výstupu nebyly opuštěny původní myšlenky navrhnuté při analýze. Tvorbu výstupního souboru má na starosti modul `vrml.c` a jeho funkce. Jedna z funkcí má za úkol zapsat do VRML souboru nezbytnou hlavičku, informace o virtuálním světě, zapsat pozici kamery při scannování a v neposlední řadě zapsat úvod pro definici jediného objektu, který bude reprezentovat celou scénu.

Tento objekt se vždy skládá z uzlů typu IndexedFaceSet, zápis těchto uzlů provádí funkce `put_vrml_triangle_row()` a jejich syntax je následující (po

jméně poduzlu následuje jeho implicitní hodnota):

```
IndexedFaceSet {
    coord NULL
    coordIndex [ ]
    color NULL
    colorIndex [ ]
    colorPerVertex TRUE
    normal NULL
    normalIndex [ ]
    normalPerVertex TRUE
    texCoord NULL
    texCoordIndex [ ]
    ccw TRUE
    convex TRUE
    solid TRUE
    creaseAngle 0.0
}
```

Definice, kterou lze najít ve specifikaci jazyka VRML, je však ještě složitější. Zmíněná funkce, ale využívá jen některé z polí uzlu. Uzel coord je určen pro zápis bodů určujících mnohoúhelníky pomocí jejich souřadnic. Uzel coordIndex pak říká, které body jsou spolu spojeny do jediného polygonu. Pomocí uzlů color, colorIndex a colorPerVertex jsou vyjádřeny barvy. Color obsahuje pole barev, colorIndex přiřazuje buď bodům a nebo celým ploškám jednotlivé barvy a uzel colorPerVertex rozhoduje, která z těchto dvou variant to bude. Posledním prvkem, který je využit, je uzel solid. Ten říká zda mají být mnohoúhelníky viditelné z obou stran a nebo zda mají být ze zadní strany průhledné.

Pro určení, které body budou spojeny do jedné plošky, je využívána funkce `triangulizuj()`, která dostává jako své parametry spojové seznamy bodů ze dvou po sobě jdoucích vzorkování. Jako výstup dává spojový seznam trojic indexů do těchto seznamů tak, aby mohly být vyplněny do uzlu coordIndex. Funkce vybírá vždy dva body z jednoho vzorkování takové, že jsou na snímku v sousedních sloupcích a zároveň od sebe nejsou moc daleko ve scéně. K nim pak hledá třetí bod z druhého seznamu takový, že na snímku má stejnou vertikální souřadnici jako jeden z první dvojice a zároveň je u něj dostatečně blízko. Takto nalezené trojice už zbývá jen zapsat na výstup.

4.7 Průběh scannování

Právě jsme popsali řešení jednotlivých podproblémů, které jsou použity v hlavní části programu. Teď se budeme zabývat tím, jak pracuje program jako celek. Na začátku musíme zjistit vstupní parametry, těmi jsou adresy v souborovém systému pro kameru (video zařízení), laser (sériový port) a výstupní soubor. Pro kameru také potřebujeme znát hodnoty jasu a kontrastu, s nimiž má kamera pracovat. A nakonec potřebujeme nějaké informace o scéně, pomocí kterých budeme rekonstruovat polohu jednotlivých nasnímaných bodů. Těmi budou

- souřadnice šesti dvojic korespondujících bodů ze scény a ze snímku,
- čísla krajních sloupců snímku, na kterých se již vyskytuj jen pozadí a
- rovnice tří rovin tvořících roh v němž se nachází snímaný předmět.

Poslední, co zjistíme před počátkem skutečného scannování, budou souřadnice bodu, po jehož překročení rovinou laseru bude program ukončen. Všechny tyto informace získáme na začátku od uživatele. Po získání všech potřebných informací, spočítáme z korespondujících dvojic dlt koeficienty a ty spolu se vším ostatním uložíme.

Nyní už začíná práce na skutečném scannování. Jako první zapíšeme "hlavičku" výstupního VRML souboru. Poté otevřeme video zařízení s daným jasem a kontrastem a následně i sériový port, kterému nastavíme komunikační rychlost. Teď již poprvé, začneme brát snímky z kamery. Jako první vezmeme několik snímků a z nich spočítáme jeden průměrný. Ten nazveme referenčním snímkem a z něj budeme brát barvy, které budeme ukládat do výstupního souboru. Navíc si tento snímek uložíme.

Pro další pokračování již potřebujeme zapnutý laser (respektive laser napájený a svítící), ale nevadí pokud byl zapnutý již předtím, nicméně teď s ním začneme pracovat. Teď začneme brát další snímky. Vezmeme jich o trochu více, než kolik jich stačí kamera vzít za dobu jedné otočky laseru kolem dokola (to je přibližně 22 snímků). Z těchto snímků spočítáme opět jeden "průměrný" (průměr je však brán z těch hodnot, které leží mezi prvním a třetím kvartilem). Průměrný snímek od ostatních odečteme a rozdílové snímky sečteme. V tomto součtu budou nenulové jen pixely, které zachycují laserem ozářená místa. V součtu z těchto pixelů spočítáme vážený průměr jejich souřadnic a tím získáme spojový seznam obrazových souřadnic bodů laseru. Z těchto bodů s pomocí informací o rovinách, čísel krajních sloupců a

dlt koeficientů odhadneme rovinu, v které laser září. Tento odhad využijeme pro převod všech bodů laseru z obrazových souřadnic na souřadnice scény. Zároveň s tím vyřadíme všechny "nesmyslné" body, které na snímku vznikly například z odraženého laserového záření, ale díky tomu, že nejsou v rovině záření, vyjdou jejich obrazové souřadnice mimo snímanou oblast a my je můžeme smazat. Body, které takto nevyřadíme, uložíme.

Teď můžeme vertikálně pohnout laserem a celý postup opakujeme. Tím jsme získali dva spojové seznamy bodů a z těch sestrojíme trojúhelníky, které ihned zapisujeme na výstup. Nyní dříve získaný seznam uvolníme znovu pohneme laserem a pokračujeme stále dokola. Pro ukončení cyklu slouží uživatelem zadané souřadnice bodu, jehož překročení má scannování ukončit. Dokud je tento bod nad rovinou laseru, cyklus pokračuje stále stejně, jakmile ho rovina překročí, je cyklus ukončen, do výstupního souboru je zapsána "patička", je smazán úložný adresář a scannování je ukončeno.

Pokud se kamera "zasekla", můžeme po načtení uložených informací, pokračovat tam, kde byl program přerušen. Po načtení všech uložených souborů, pak vždy začínáme snímkováním se zapnutým laserem.

Kapitola 5

Použití programu

Program je napsán pro operační systém Linux (testován byl na distribuci openSUSE). K jeho použití je tedy třeba počítač s instalací tohoto OS. Dále musí být počítač vybaven alespoň dvěma USB porty, pro připojení kamery a napájení pro laser a sériovým portem pro komunikaci s laserem. Operační systém potom musí podporovat komunikaci s těmito zařízeními. To znamená, že součástí jeho jádra musí být moduly, které to umožní, např. modul V4L pro spolupráci s kamerou. Navíc je třeba podpora knihovny GTK+, protože uživatelské prostředí programu tuto knihovnu využívá. V neposlední řadě musíme mít kameru, která bude schopna s programem spolupracovat (testováno bylo více typů kamery Logitech QuickCam chat) a zařízení laseru.

Uživatel, který bude chtít program používat, musí pak mít práva s těmito zařízeními pracovat. Například pro kameru, toho lze většinou docílit tak, že uživatel je přidán do skupiny `video`. Pro ovládání laseru musí mít zase práva k zařízení sériového portu (nejčastěji `/dev/ttyS0`).

5.1 Instalace

Pokud jsou splněny všechny předpoklady, musíme program před použitím nejdřív nainstalovat. Jako první je potřeba vytvořit adresář v němž bude moci uživatel, který provádí instalaci, vytvářet a modifikovat soubory. Do tohoto adresáře, se pak překopíruje obsah adresáře Scanner. Tím končí přípravné akce a následuje už samotná instalace. Ta se při bezproblémovém průběhu odbude spuštěním následujících příkazů. Příkazy předpokládají, že se nacházíme ve zmíněném nově vytvořeném adresáři.

```
sh autogen.sh
make
cd src/mysrc
make all
cd ../..
```

Význam jednotlivých příkazů je následující. Při návrhu uživatelského prostředí bylo využito nástroje Glade [11] a první příkaz automaticky generuje soubory, které převádí návrh uživatelského prostředí do zdrojových kódů. Druhý příkaz pak spouští skutečný překlad. Třetím se přepínáme do adresáře se zdrojovými soubory. Čtvrtým se spouští překlad pomocného programu, jehož význam je vysvětlen v následující sekci. Poslední příkaz nás vrátí do adresáře, z kterého jsme instalaci začínali.

Při úspěšném průběhu vzniknou dva spustitelné soubory, `src/scanner` a `src/mysrc/pokracovani`, které provádí vlastní scannování.

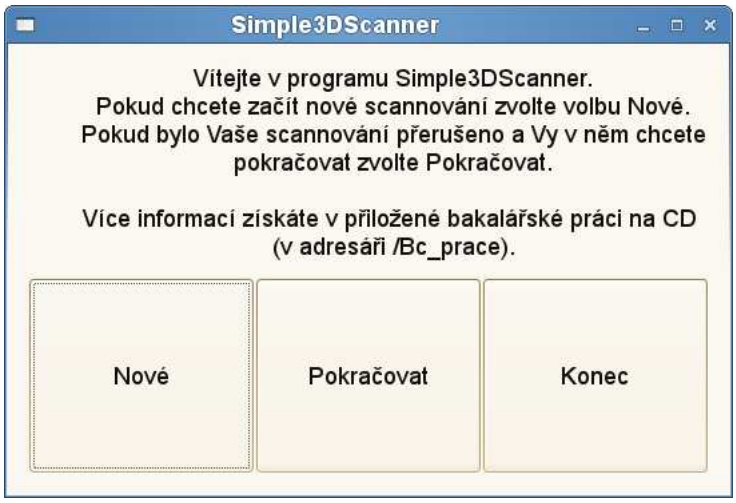
5.2 Ovládání

Nezmínili jsme ještě poslední předpoklad, který program používá, protože k instalaci nebyl zapotřebí. Tím je, že pokud chceme scannovat nějaký předmět je třeba umístit ho do rohu. Tento roh musí být tvořen třemi rovinami (nejlépe na sebe kolmými) a musíme znát obecné rovnice těchto rovin, protože program je využívá.

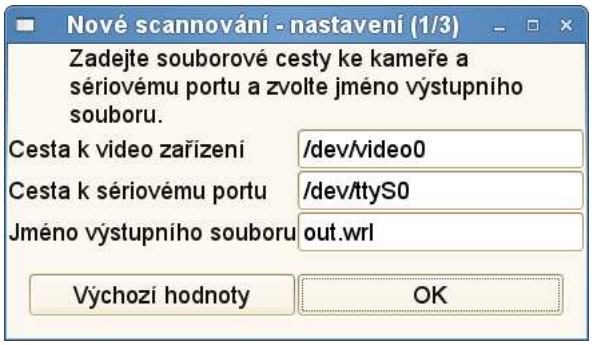
Pokud se tedy instalace zdařila, můžeme začít se scannováním. Předpokládejme, že jsme stále ve stejném adresáři jako po dokončení instalace, pak můžeme hlavní program spustit příkazem `src/scanner`. Jako první po spuštění uvidíme okno (obrázek 5.1), které dává možnost začít zcela nové scannování, pokračovat v nějakém rozpracovaném avšak předčasně přerušeném scannování, nebo program ukončit.

Volba možnosti nového scannování okamžitě začne s přípravou na scannování a začne se uživatele dotazovat na další informace potřebné pro scannování, viz níže. Zároveň vymaže informace o minulém scannování, což však nevadí, neboť se k němu stejně nepůjde vrátit, protože např. laser již nepůjde vrátit do původní polohy.

Pokud je vybrána volba Pokračovat, program okamžitě začne načtením uložených prozatímních informací a poté pokračuje tam, kde bylo snímání přerušeno. Pokud by se načtení informací nezdařilo a to buď proto, že poslední scannování přerušeno ve skutečnosti nebylo a není tedy v čem po-



Obrázek 5.1: Úvodní okno aplikace



Obrázek 5.2: Okno pro zadání souborových adres

kračovat nebo proto, že byly dočasné soubory poškozeny, program se ukončí a vypíše důvod nezdaru.

Pokud jsme zvolili možnost nového scannování, musíme nejdříve zadat cesty k jednotlivým zařízením a to nám umožní okno, které se zjeví jako druhé (obrázek 5.2). Zařízení mají nejčastěji takovou adresu, jaká je implicitně předvyplněna, mohou se však lišit. Zároveň s cestami zadáváme i jméno výstupního souboru (můžeme zadat i plnou cestu). Po zadání požadovaných vstupů můžeme s nastavováním pokračovat pomocí tlačítka OK. Pokud jsme však zvolili cesty, které k zařízení nevedou nebo nemáme práva na jejich používání program se ukončí a vypíše důvod.

Pokud jsme zadali správné údaje, zobrazí se další okno (obrázek 5.3),

Obrázek 5.3: Výběr vhodných hodnot pro jas a kontrast

které nám nabídne upravit jas a kontrast, s kterými kamera snímá scénu. Pomocí posuvníků můžeme zvolit pro oba parametry nějakou hodnotu mezi nulou a dvěstěpadesátipěti. Pokud hodnoty změníme, můžeme si prohlédnout efekt nového nastavení pomocí tlačítka Zkus. Pokud jsme již s volbou spokojeni, pokračujeme pomocí volby OK.

Poslední a nejsložitější okno určené k nastavování parametrů můžeme vidět na obrázku 5.4. Nyní je poslední možnost, kdy můžeme měnit polohu kamery, scannovaného předmětu nebo jeho pozadí, pokud tak učiníme, můžeme obraz nové scény získat pomocí tlačítka Nový snímek.

Jako první musíme zadat šest bodů na snímku a k nim jejich pozici ve scéně. K tomu slouží levá horní část okna. Body snímku se zadávají tak, že nejdříve zvolím, který bod chci zadávat (stiskem korespondujícího tlačítka Zadat na snímku) a poté kliknutím na snímek určím pozici bodu (souřadnice se sami zapíší do tabulky a na místě kliknutí se objeví červená tečka). Souřadnice bodu ve scéně pak vyplníme ručně do příslušných políček. Je třeba upozornit, že v programu se stejně jako ve VRML používá pravotočivý souřadný systém, kde kladná poloosa y ukazuje směrem nahoru. Tedy pokud kladná poloosa z ukazuje směrem ke kameře, směřuje kladná poloosa x na snímku doprava.

Body by měly být vybírány tak, aby žádné tři z nich neležely v přímce.

Velmi dobré je zvolit body ležící přibližně ve vrcholech pravidelného šestiúhelníku a které budou hodně blízko okrajům snímku. Špatná volba bodů může velmi negativně ovlivnit kvality výstupu.

Druhou věcí, kterou je třeba programu sdělit, jsou čísla krajních sloupců snímku, na nichž již není vidět scannovaný předmět, ale jen pozadí (jako pozadí může být za snímkem vidět jen již výše zmíněný roh, respektive tři roviny, které ho tvoří). Vlevo od levého krajního sloupce a vpravo od pravého by tedy mělo být možno zachytit jen to laserové záření, které dopadlo přímo na jednu ze tří rovin pozadí. Jejich pozice se zadává stejně jako body na snímku tedy jedním kliknutím na tlačítko a druhým na snímek.

Předposledními informacemi, které program potřebuje získat, jsou rovnice tří rovin tvořící pozadí scannovaného předmětu. Tyto rovnice jsou uvažovány ve tvaru $ax + by + cz = d$ a pro zadání koeficientů a, b, c a d slouží dvanáct kolonek v levé dolní části okna.

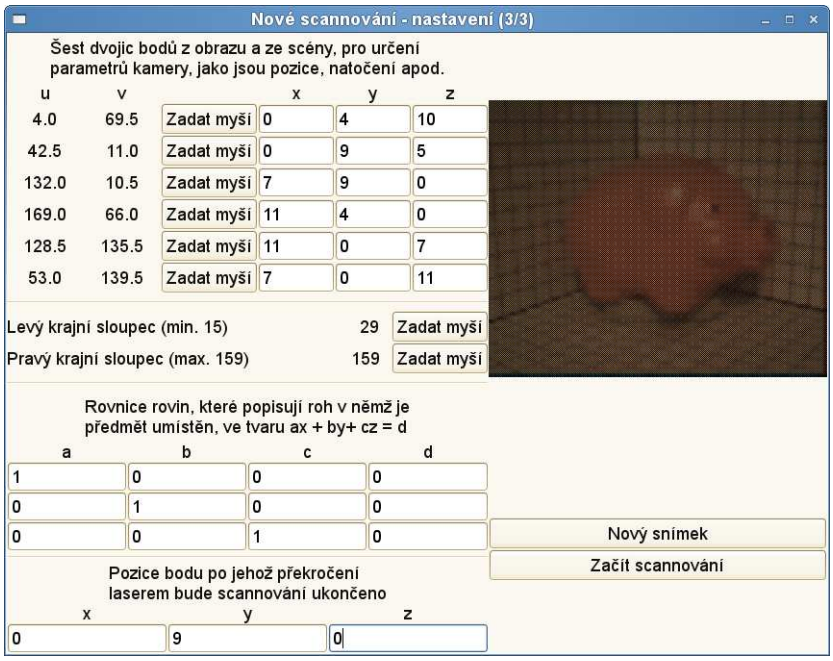
Poslední, co programu zadáme, jsou světové souřadnice bodu, který slouží k ukončení scannování. Pokud bude tohoto bodu dosaženo laserem v průběhu scannování, tzn. pokud by rovina laserového záření byla již nad tímto bodem, bude scannování ukončeno a výstup připraven k zobrazení.

Po stisku tlačítka Začít scannování je zahájeno scannování.

5.3 Pomocný program

Jelikož kamera nepracuje zcela správně, může se stát, že se při snaze pořídit snímek program "zasekne" a nebude na nic reagovat (problémy jsou pravděpodobně způsobovány ovladači). V tom případě je jedinou možností jak dále pokračovat ve scannování předčasné ukončení programu a to posláním nějakého signálu k ukončení. Při novém spuštění programu lze pak v započatém snímání pokračovat volbou Pokračovat v okně na obrázku 5.1. Při použití grafického uživatelského rozhraní se však tento problém vyskytuje tak často, že je úspěšné scannování prakticky znemožněno. Proto byla implementována druhá možnost, jak ve scannování pokračovat. Tou je program `pokracovani` umístěný v adresáři `src/mysrc`, jehož průběh je zcela stejný jen s tím rozdílem, že nevyužívá grafické uživatelské prostředí ale jen standardní výstup.

Program předpokládá, že všechny potřebné parametry již byly nastaveny výše popsaným způsobem, pokud tomu tak není, program se ukončí a vypíše, kde nastaly potíže. Důležité však je, že pokud jsme zadali relativní adresu



Obrázek 5.4: Okno pro zadání parametrů potřebných pro scannování

k výstupnímu souboru, musíme tento program pouštět ze stejného adresáře jako program `scanner`.

Kapitola 6

Závěr

6.1 Shrnutí

Cílem této práce bylo navrhnout a implementovat jednoduchý 3D scanner, který bude schopný vytvářet modely skutečných věcí jen pomocí levné webové kamery a zařízení vyrobeného z laseru staré CD mechaniky. Výstup měl být uskutečněn do souboru ve formátu VRML. Zadaní se splnit podařilo, zrekapitulujme tedy naše řešení.

Kamera v něm snímá nehybnou scénu. Jediné, co se na snímku pohybuje, je bod dopadu laserového paprsku. Laser mění neustále směr svého záření tak, že otáčí odrazovým hranolem pořád dokola. Bod dopadu laseru tedy horizontálně obíhá kolem. Naproti tomu vertikální směr záření mění zařízení jen po impulsu od programu.

Horizontální rotace laseru je však tak rychlá, že bod dopadu je na snímku zachycen jako "čára". Pomocí zjištění rozdílu mezi snímky jsme tuto "čáru" schopni najít. Metoda DLT nám zajišťuje převod jednotlivých souřadnic pixelů "čáry" na paprsky ve scéně. Pomocí informací o pozadí za scannovaným předmětem převádíme informace o paprscích zachycujících laserové body na souřadnice scény.

Když známe pozici dostatečného počtu bodů dopadu ve scéně, odhadneme rovinu, ve které se laserové záření šíří. Pomocí této roviny a znalosti paprsků reprezentujících jednotlivé pixely laseru už můžeme určit pozici všech bodů, které laser ozářil. Takto získáme pozici bodů v jedné řádce. Po tom, co provede laser vertikální pohyb, můžeme obdobným způsobem zachytit další řádky bodů.

Z jednotlivých po sobě následujících řádek vybíráme trojice bodů, které

jsou si dostatečně blízké, a z těch sestrojujeme polygony, které zapisujeme do VRML souboru. Ten je pak výstupem našeho programu.

6.2 Návrhy na další vývoj

Lze říci, že zadání se splnit povedlo, řešení však nelze považovat za konkurence schopné nějakým ekvivalentním profesionálním nástrojům. Ačkoli nebylo záměrem vytvořit takto výkonný scanner, pokusme se nalézt faktory, které by nás při snaze o zlepšení nejvíce limitovali.

Nejvíce omezujícím je v tomto směru nepochybně kamera. Rozlišení, které bylo nakonec použito, bylo pouhých 176 na 144 pixelů, při čemž poslední dva řádky i sloupce byly nepoužitelné. Je jasné, že pro zpřesňování scanneru je třeba zvyšovat rozlišení. Vyšší rozlišení zvýší počet různých paprsků, které jsme schopni od sebe odlišovat a tím umožní něco jako "vyšší rozlišení" i pro výsledný model. Zároveň však přímo zvýší například i možnost lepšího odhadu roviny laseru.

Pokud by kamera poskytovala lepší barvy, mohla by být detekce laseru na snímcích založena více i na barvě laseru a ne jen na intenzitě, jak tomu bylo v naší implementaci. Není jisté, že by to znamenalo zlepšení, ale byla by šance na zpřesnění odhadu jednotlivých pozic laserových bodů.

Zasekávání kamery velmi znepříjemňovalo vývoj aplikace. Navíc čím delší je doba scannování, tím se zvyšuje pravděpodobnost nějaké změny v záběru, která je pro použitý způsob detekce laseru ve snímku kritická. Změna může být způsobena nejen pohybem kamery či předmětů ve scéně, ale třeba i změnou intenzity slunečního záření.

Další částí, která by se dala vylepšit je zařízení laseru. První možné zlepšení je pevné spojení laseru s kamerou, tím by byla zajištěna neměnná vzdálenost laseru od kamery. Konstantní polohou laseru vůči kameře by se usnadnilo hledání pozice laseru, protože tuto pozici by bylo možné odvodit přímo z pozice kamery. To by nepochybně pomohlo přesnější práci s laserem.

Druhou šanci na zlepšení vidím v detekci laserového bodu místo detekce "čáry". Toho by šlo dosáhnout buď zpomalením horizontální rotace odrazového hranolu nebo zrychlením braní jednotlivých snímků. Detekce bodu by nám vlastně řekla, jakým paprskem laser září, zatímco takto máme informaci pouze o rovině, v které září. Pokud víme rovnici této roviny, pro zjištění pozice bodu ve scéně hledáme její průsečík s paprskem reprezentující pixel na snímku. I malá chyba v odhadu naklonění roviny pak může znamenat velký posun detekovaného bodu ve scéně. Naproti tomu, pokud

bychom znali pro laserové záření jediný paprsek, hledali bychom nejbližší bod dvou přímek. Při stejné změně naklonění by se pak detekovaný bod ve scéně pohnul nejvýše o tolik, o kolik v předchozím případě, někdy však méně.

Kalibrace kamery by se dala více automatizovat. Pokud bychom měli nějaký předmět předem daných geometrických vlastností, šlo by některé body předmětu na snímku detekovat automaticky a následně provést kalibraci bez další asistence uživatele. Bylo by to spíše zpříjemnění ovládání programu a výrazné zpřesnění by to pravděpodobně nepřineslo.

Předposlední možností, jak by šlo program zlepšit, je nepoužívání rozdílových snímků. Ty jsou totiž silně závislé na osvětlení scény, které se však, zvláště pokud je přirozené, může samovolně poměrně rychle měnit. Nejsnáze by se toho dalo docílit pomocí použití speciálního zařízení pro detekci laseru, to by však naprosto změnilo celý projekt. Naproti tomu přímá detekce laseru bez použití rozdílových snímků by nejspíše vyžadovala již zmíněné zkvalitnění pořízených snímků.

Poslední návrh je spíše doporučení ke směru dalšího vývoje. Jelikož formát VRML je již poměrně starý, bylo by pravděpodobně správné zvolit pro další verze programu jiný výstupní formát. Jako ideální se jeví formát X3D, který byl i standardizačním konsorciem Web3D označen za nástupce VRML.

Literatura

[1] Abdel-Aziz Y.I., Karara H.M.: *Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry in Proceedings of the Symposium on Close-Range Photogrammetry (pp. 1-18)*, VA: American Society of Photogrammetry, Falls Church, 1971

[2] Jarvis R.A.: *A Laser Time-of-Flight Range Scanner for Robotic Vision*

[3] Kašpar M., Křemen T.: *3D laserové skenovací systémy*, `http://si.vega.cz/clanky/d-laserove-skenovaci-systemy/`

[4] Kovács T.: *Construction of an Active Triangulation 3D Scanner for Testing a Line Following Strategy*, `www.bmf.hu/conferences/sami2006/Kovacs.pdf`

[5] Rocchini C., Cignoni P., Montani C., Pingi P., Scopigno R.: *A low cost 3D scanner based on structured light*, `http://vcg.isti.cnr.it/publications/papers/vcgscanner.pdf`

[6] Voráčová Š.: *Aplikace epipolární geometrie*, `http://www.fd.cvut.cz/personal/voracova/Epipolar.pdf`

[7] Žára J., Beneš B., Sochor J. Felkel P.: *Moderní počítačová grafika*, str. 312, Computer Press, Brno, 2004.

[8] Internetové stránky Autodesk (výrobce 3D modelovacích nástrojů), `http://www.autodesk.com`

[9] Internetové stránky Logitech (kamera QuickCam chat), `http://www.logitech.com/index.cfm/38/281&cl=us,en`

[10] Linuxové ovladače pro webové kamery,
http://mxhaard.free.fr

[11] Nástroj pro návrh uživatelského prostředí pro knihovnu GTK+,
http://glade.gnome.org/

[12] Projekt dokumentace Stonhenge,
http://www.stonehengelaserscan.org/

[13] Seriový port v Linuxu,
http://www.linuxdocs.org/HOWTOs/Serial-HOWTO.html

[14] Specifikace jazyka VRML,
http://www.web3d.org/x3d/specifications/vrml/

[15] Wikipedia - 3D scannery,
http://en.wikipedia.org/wiki/3d_Scanner

[16] Wikipedia o modulu Video For Linux,
http://www.linuxtv.org/v4lwiki

Dodatek A

Obsah přiloženého CD

- `/Bc_prace` adresář obsahující bakalářskou práci ve formátu PDF.
- `/Scanner` adresář obsahující soubory, pomocí nichž lze provést instalaci aplikace, viz část 5.1.
- `/Priklady` adresář obsahující příklady nascannovaných objektů. Pro porovnání jsou připojeny obyčejné snímky pořízené kamerou (ve formátu PNG).