Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

DIPLOMOVÁ PRÁCE

Virtuální studio

Plzeň, 2010 Václav Bystřický

Poděkování

Rád bych poděkoval především Ing. Petru Lobazovi, za vedení mé práce, při kterém projevil neskonalou trpělivost a pozitivní přístup při řešení všech problémů, na které jsme narazili. Poděkování patří také mé přítelkyni, rodině a přátelům za podporu a neskonalé porozumění.

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 17. 5. 2010, Václav Bystřický,……

Abstract

Virtual studios are today widespread tools in professional TV production that allows putting real actors or objects into a virtual scene in a seamless way. The Biggest problem connected with their practical usage is high initial investment needed for realization of such virtual studio. As with ongoing digital revolution in TV broadcasting demand for such system still rises, we have decided to analyze and design low–cost virtual studio. Our goal isn´t to compete with professional s solutions, but to analyze processes running in virtual studio and propose their low cost implementation. Results of first stage of our projects are described in this paper.

Obsah

1 ÚVOD 5

2 TEORETICKÝ ZÁKLAD 6

2.1 Model projektivní kamery Equation Chapter 2 Section 1 6
2.2 Epipolární geometrieEquation Chapter (Next) Section 1 8
2.2.1 Výpočet esenciální matice z korespondencí 10
2.2.2 Zjednodušení esenciální matice 10
2.2.3 Výpočet zjednodušené esenciální matice 11
2.2.4 Rozklad esenciální matice 11
2.3 Kalibrace kamery 12

3 VIRTUÁLNÍ STUDIO 13

3.1 Maskování (Matting) 14
3.2 Sledování pozice kamery (Tracking) 16
3.2.1 Elektromechanické systémy 16
3.2.2 Systémy pro infračervené trasování 18
3.2.3 Optické systémy 20
3.3 Renderování (Rendering) 21

4 EXISTUJÍCÍ SYSTÉMY 23

5 NÍZKOROZPOČTOVÉ STUDIO 24

6 HARDWARE 26

7 NÁVRH VIRTUÁLNÍ SCÉNY 29

7.1 Nástroje pro návrh virtuální scény 29
7.2 Formát virtuální scény 31

8 SOFTWARE 31

8.1 Struktura softwaru 32
8.2 Kalibrace 33
8.3 Trasování 34
8.4 Proudy 35
8.4.1 DirectShow 35
8.4.2 Directshow filtry 36
8.4.3 Popis proudů 38
8.5 Procesor 40

9 KLÍČOVÁNÍ STEREOSKOPICKÉHO OBRAZU 45

10 MOŽNÁ ROZŠÍŘENÍ 47

11 ZÁVĚR 48

LITERATURA 49

PŘÍLOHA A: UŽIVATELSKÁ PŘÍRUČKA 51

A.1 Kalibrace kamery 51
A.2 Kalibrace trackeru 55
A.3 Virtuální studio 58

PŘÍLOHA B: ZÁSADY PRO VYTVOŘENÍ VIRTUÁLNÍ SCÉNY 64

1 Úvod

Klíčování na modré (či zelené) pozadí je televizní a filmová technika, s jejímiž výstupy se setkáváme již řadu desetiletí a například hlasatelku, v jejímž pozadí se nám zobrazují ve virtuální scéně informace o počasí na příští den, je většina z nás zvyklá vídat téměř denně. Přestože lze pomocí klíčování dosáhnout velmi zajímavých vizuálních výsledků, má tato technika jedno zásadní omezení. Při klasickém klíčovaní téměř jakýkoliv pohyb kamery zruší iluzi virtuálního prostoru, kterou se snažíme vytvořit. Proto byl tento koncept počátkem devadesátých let minulého století rozšířen a vznikla první experimentální virtuální studia. Virtuální studio je systém, který kromě klíčování dokáže rovněž sledovat pozici kamery, která snímá scénu. Tyto informace pak dokáže přenést do procesu vytváření (vykreslování) virtuálního pozadí tak, že dojem, že snímané postavy či objekty se nacházejí ve virtuálním prostředí, zůstává zachován. Umožňuje tak svým uživatelům téměř absolutní volnost při zasazování postav do různých virtuálních světů. Právě tato skutečnost byla hnacím motorem pro větší rozšíření těchto systémů v televizní produkci, které je v poslední dekádě zcela jasně patrné. Opravdu masovému rozšíření virtuálních studii však brání jejich vysoká pořizovací cena, která může jít v extrémním případě až do stovek miliónů korun. Především pro malé produkce, jejichž počet v poslední době spolu s postupující digitalizací televizního vysílání roste, jsou takové ceny neakceptovatelné. Z toho důvodu jsme se rozhodli prozkoumat systémy virtuálních studií, analyzovat jejich jednotlivé části a navrhnout nízkorozpočtové virtuální studio.

Naším cílem není konkurovat vysoko rozpočtovým komerčním systémům nebo dovést jednotlivé části virtuálního studia k naprosté dokonalosti. Budeme se snažit spíše popsat jednotlivé procesy, které ve virtuálních studiích probíhají a navrhnout možnosti jejich realizace za pomoci nám dostupných prostředků, tak aby takto vzniklý systém určený v první fázi především ke vzdělávacím účelům bylo možné dále vyvíjet a dovést jej do produkční kvality.

V následujících kapitolách postupně popíšeme jednotlivé části virtuálního studia (kapitola 3) uvedeme příklady existujících systémů (kapitola 4) a postupně představíme náš návrh nízkorozpočtového studia jak z pohledu použitého hardwaru (kapitola 6) tak softwaru (kapitola 8). Nejprve si však představíme teoretický základ, ze kterého budeme v dalších částech vycházet.

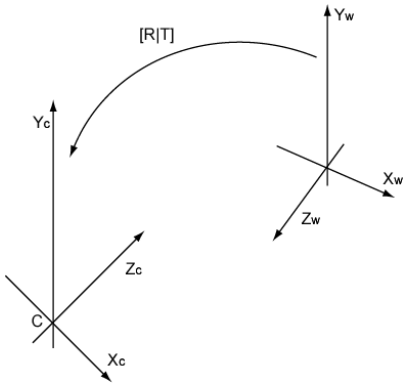
2 Teoretický základ

Před tím, než se pustíme do popisu samotného virtuálního studia, si je nutné představit teoretický (matematický) základ, na kterém jsou postaveny klíčové části systému.

Modelování reálných kamer pomocí matematických modelů a následná aplikace těchto modelů pro získání přidané informace z jejich obrazu, je problematikou zasahující do mnoha odvětví nejen počítačové grafiky ale i počítačového vidění, robotiky a dalších. V posledních několika dekádách se tomuto tématu věnovala řada odborníků a výstupem jejich práce je rozsáhlá teorie vztahující se k různým aspektů dané problematiky. Následující tři podkapitoly jsou věnovány popisu základního matematického aparátu, který z této teorie vychází, a který je použit v našem projektu.

2.1 Model projektivní kamery

Nejprve zavedeme tři základní souřadné systémy typicky použité ve všech modelech kamer.



Obr. 2.1 Zobrazení vzájemné polohy souřadného systému světa (X_w, Y_w, Z_w) a souřadného systému kamery (X_c, Y_c, Z_c). Přechod mezi těmito systémy lze realizovat pomocí transformace dané rotační maticí R a translační vektorem T.

Souřadný systém kamery

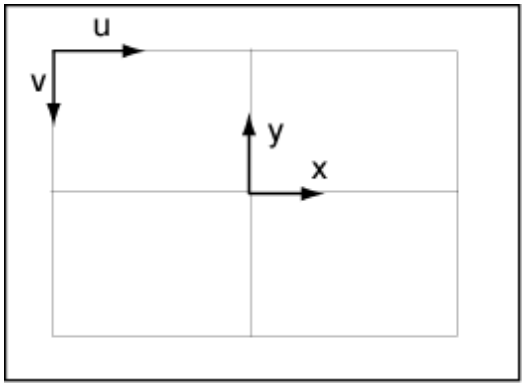
Třídimenzionální souřadný systém, jehož počátek je typicky umístěn v centru promítání kamery a kladná osa Z_c je orientována shodně se směrem pohledu kamery. Značení souřadných os tohoto systému je X_c, Y_c, Z_c, viz Obr. 2.1.

Souřadný systém světa

Rovněž třídimenzionální systém. Jedná se o systém, v jehož souřadnicích jsou udávány pozice objektů, které jsou následně snímány kamerou. Značení souřadných os tohoto systému je X_w, Y_w, Z_w, viz Obr. 2.1.

Obrazový souřadný systém

Jedná se o dvojdimenzionální systém. Někdy se nazývá též pixelový a za jeho počátek je typicky udáván levý horní roh obrazu, který je získán z kamery. Souřadnice bodů v tomto systému jsou výsledkem projektivní transformace. Osy tohoto systému se označují u, v, viz Obr. 2.2.



Obr. 2.2 Dvě možné representace obrazového souřadného systému (u,v) a (x,y). Souřadný systém (u, v) se někdy rovněž nazývá pixelový.

Model kamery popisuje, jakým způsobem jsou objekty ze souřadného systému světa kamerou promítnuty do obrazového souřadného systému. Takových modelů je celá řada. My se zaměříme pouze na jeden z nich, a to model perspektivní kamery, který velmi dobře aproximuje reálnou kameru. Popišme si tedy tento model kamery a jeho vlastnosti.

Položíme-li optický střed kamery *C* do počátku soustavy souřadnic a označíme-li rovinu kolmou na osu *z* ležící ve vzdálenosti *f* od *C* jako rovinu promítaní (průmětnu) viz Obr. 2.3 (jinými slovy pokud sjednotíme souřadný systém světa a souřadný systém kamery viz výše), pak perspektivní projekcí bodu $x = (X, Y, Z)$ je bod $x = \left(\frac{fX}{Z}, \frac{fY}{Z}, f\right)$. Pokud ignorujeme poslední souřadnici, získáme hledanou transformaci mezi trojrozměrným a dvojrozměrným prostorem doby ve formě:

$$(X, Y, Z) \rightarrow \left(\frac{fX}{Z}, \frac{fY}{Z}\right) \tag{2.1}$$

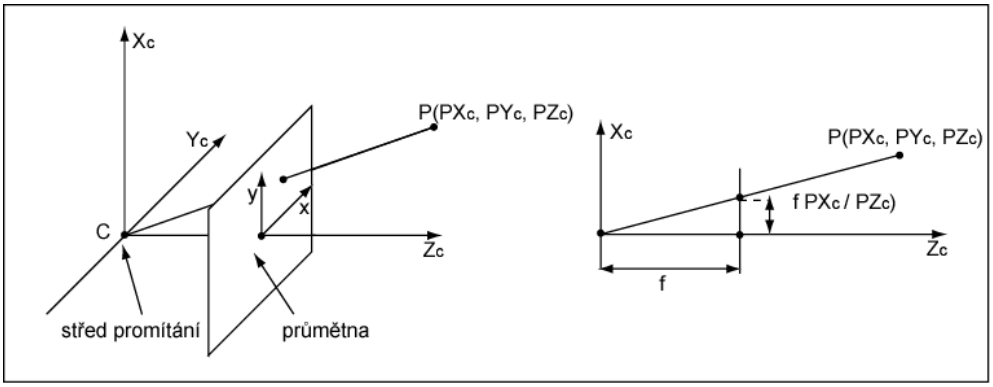
Použijeme-li homogenní souřadnice, můžeme tento vztah maticově vyjádřit následujícím způsobem:

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.2}$$

Osa *Z* se v tomto případě nazývá hlavní osa kamery. Průsečík hlavní osy kamery a roviny promítání se nazývá střed promítání nebo také principální bod (u_0, v_0).

Ve výrazu (2.2) předpokládáme počátek obrazových souřadnic (x, y) umístěný ve středu promítání. Ve skutečnosti tomu tak nemusí být, v obecném případě je používán tzv. obrazový souřadnicový systém (u, v). Matice vyjadřující převod mezi těmito dvěma souřadnými systémy se nazývá matice vnitřních parametrů kamery a označujeme ji *K*. Matice vnitřních parametrů kamery:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} \alpha f & 0 & u_0 \\ 0 & \beta f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.3}$$

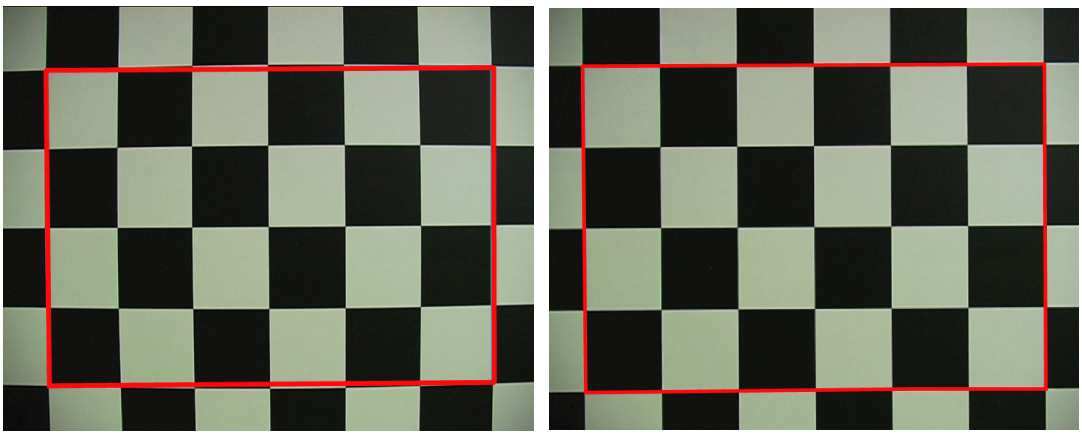


Obr. 2.3 Souřadný systém kamery se středem v bodě C a s průmětnou umístěnou ve směru kladné osy Z ve vzdálenosti f (ohnisková vzdálenost).

Model kamery se kromě vnitřních parametrů skládá z tzv. vnějších parametrů. Vnější parametry kamery definují pozici a orientaci kamery v prostoru a jsou reprezentovány ortogonální maticí R a vektorem T určujícím pozici ohniska kamery. Rozšíříme-li rovnici (2.2) o vnitřní a vnější parametry, je možno zapsat promítnutí bodu maticí ve tvaru:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} \beta f & 0 & 0 \\ 0 & \alpha f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_{11} & y_{12} & y_{13} & t_x \\ y_{21} & y_{22} & y_{23} & t_y \\ y_{31} & y_{32} & y_{33} & t_z \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.4}$$

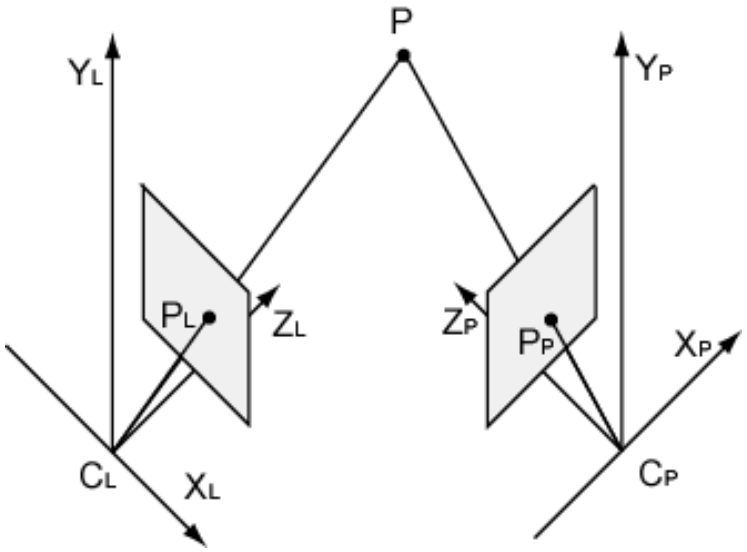
Je nutné poznamenat, že tento model kamery nezohledňuje tzv. radiální zkreslení způsobené průchodem světla přes čočku kamery. Toto zkreslení může být u některých typů kamer značné (viz Obr. 2.4) a v takových případech je nutné model kamery rozšířit. Jeden z nejpoužívanějších modelů, které zahrnují radiální zkreslení, lze nalézt v práci [1].



Obr. 2.4 vlevo výstup z kamery ovlivněný radiálním zkreslením, vpravo obraz po korekci radiálního zkreslení

2.2 Epipolární geometrie

Epipolární geometrie popisuje základní geometrický vztah mezi dvěma perspektivními kamerami nezávisle na pozorované scéně. V následující části popíšeme základní vztahy epipolární geometrie bez odvození (kompletní popis epipolární geometrie včetně všech odvození lze nalézt například v [2].



Obr. 2.5 Geometrická representace epipolární geometrie

Geometrická reprezentace epipolární geometrie je zobrazena na Obr. 2.5, kde vidíme dvě kamery určené středy promítání C_L a C_P s příslušnými projektivními rovinami a dále bod v prostoru P a jeho projekce na levé a pravé průmětně P_L a P_P.

Algebraická reprezentace

Epipolární geometrie je algebraicky reprezentovaná esenciální maticí E a z ní vycházející fundamentální maticí F. Esenciální matice je definována vztahem:

$$P_P^T E\, P_L = 0 \tag{2.5}$$

Kde P_P, P_L jsou zápisy bodu P v souřadných systémech pravé a levé kamery. Pokud známe transformaci mezi souřadnými systémy kamer danou rotační maticí R a vektorem posunutí T viz (2.6):

$$P_P = R(P_L - T) \tag{2.6}$$

můžeme matici E vyjádřit vztahem:

$$E = SR \tag{2.7}$$

kde R, je známá matice rotace a

$$S = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \tag{2.8}$$

kde T_x, T_y, T_z jsou prvky vektoru T. Pro úplnost definujeme fundamentální matici F:

$$p_P^T F p_L = 0 \tag{2.9}$$

Kde p_P p_L jsou projekce bodu P v na průmětny pravé a levé kamery v pixelových souřadnicích.

2.2.1 Výpočet esenciální matice z korespondencí

Esenciální matice má pět stupňů volnosti. Tři stupně volnosti reprezentují rotaci mezi kamerami, tři translaci mezi kamerami a jeden stupeň je možné odebrat, protože každá esenciální matice musí mít hodnost 2 (dokonce lze dokázat, že každá esenciální matice musí mít dvě singulární čísla shodná a třetí rovno nule). Z toho vyplývá, že pro spočtení esenciální matice je třeba znát minimálně pět bodových korespondencí. Skutečně existují algoritmy, které dokážou vypočíst esenciální matici s použitím pouhých pěti známých korespondencí. Jejich zásadní nevýhodou je, že produkují až deset řešení, a přestože lze některá řešení většinou vyloučit, protože hodnoty rotace a translace z nich získané nabývají komplexních hodnot, jsou tyto metody pro výpočet esenciální matice obecně nevhodné (viz [3]). Vhodnější se ukazují algoritmy používající větší počet korespondencí, nejznámější a nejčastěji používaný je takzvaný osmibodový algoritmus.

Osmibodový algoritmus

Esenciální matice E je definována rovnicí (2.5) $P_P^T E\, P_L = 0$, kde P_P a P_L jsou projekce stejného bodu na průmětně levé, respektive pravé kamery (tzv. korespondenční body). Označíme-li jednotlivé členy esenciální matice následujícím způsobem:

$$E = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix}$$

a korespondenční body jsou ve tvaru, $P_P = (x_p, y_p, 1)$, $P_L = (x_L, y_L, 1)$, což můžeme předpokládat bez újmy na obecnosti, můžeme pro každou dvojici korespondenčních bodů sestavit rovnici:

$$x_p x_L e_{11} + x_p y_L e_{12} + x_p e_{13} + y_p x_L e_{21} + y_p y_L e_{22} + y_p e_{23} + x_L e_{31} + y_L e_{32} + e_{33} = 0 \quad \textbf{(2.10)}$$

Pokud máme alespoň osm korespondencí, můžeme sestavit soustavu rovnic:

$$Ae = 0 \quad \textbf{(2.11)}$$

kde e je vektor ve tvaru $e = (e_{11}, e_{12}, e_{13}, e_{21}, e_{22}, e_{23}, e_{31}, e_{32}, e_{33})$ a řádky matice A mají tvar $(x_p x_L,\ x_p y_L,\ x_p,\ y_p x_L,\ y_p y_L, y_p, x_L, y_L, 1)$. Řešením této soustavy a uspořádáním prvků vektoru e do matice získáme požadovanou esenciální matici. Je nutné zmínit, že rovnice (2.11) nemá vlivem šumu ve vstupních datech obecně jiné než triviální nulové řešení. V takovém případě můžeme nalézt alespoň přibližné řešení pomocí metody nejmenších čtverců například za použití SVD (Singular Value Decomposition).

2.2.2 Zjednodušení esenciální matice

Esenciální matice reprezentuje obecnou třídimensionální transformaci mezi kamerami. Pokud tuto transformaci vhodným způsobem omezíme, může esenciální matice získat podstatně jednoduší tvar. V následující části se podíváme, jak se esenciální matice změní, pokud vzájemnou transformaci kamer omezíme na rovinnou. V takovém případě připouštíme vzájemnou rotaci kamer pouze kolem

jediné osy (např. Y) a posun pouze v rovině kolmé na tuto osu. V tomto případě přechází rovnice (2.8) do tvaru

$$S = \begin{bmatrix} 0 & -T_z & 0 \\ T_z & 0 & -T_x \\ 0 & T_x & 0 \end{bmatrix} \quad \textbf{(2.12)}$$

matice rotace kolem osy Y (pravotočivá) má tvar

$$R(\alpha) = \begin{bmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix},$$

výslednou esenciální matici lze pak podle vztahu (2.7) vyjádřit jako

$$E = \begin{bmatrix} 0 & -T_z & 0 \\ T_z\cos(\alpha) + T_x\sin(\alpha) & 0 & T_z\sin(\alpha) - T_x\cos(\alpha) \\ 0 & T_x & 0 \end{bmatrix} \quad \textbf{(2.13)}$$

2.2.3 Výpočet zjednodušené esenciální matice

Čtyřbodový algoritmus

Zjednodušená esenciální matice pro rovinný pohyb má tvar

$$E = \begin{bmatrix} 0 & e_1 & 0 \\ e_2 & 0 & e_3 \\ 0 & e_4 & 0 \end{bmatrix},$$

obdobně jako u osmibodového algoritmu můžeme sestavit soustavu rovnic pro korespondenční body $P_P = (x_p, y_p, 1)$, $P_L = (x_L, y_L, 1)$:

$$x_p y_L e_1 + x_L y_p e_2 + y_p e_3 + y_{Lp} e_4 = 0 \quad \textbf{(2.14)}$$

Pro řešení soustavy je opět vhodné použít metodu nejmenších čtverců, viz část 2.2.1.

2.2.4 Rozklad esenciální matice

Existuje velké množství postupů vyvinutých za účelem získání rotační matice a vektoru posunutí z esenciální matice (např.[2],[4],[5]). My uvedeme (bez odvození) pouze nejpoužívanější z nich založenou na SVD rozkladu esenciální matice.
Provedeme-li tento rozklad, získáme vyjádření esenciální matice v následujícím tvaru.

$$E = \mathrm{U}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathrm{V}^{\mathrm{T}}$$

Lze dokázat, že rotační matice příslušná této esenciální matici má jeden z následujících dvou tvarů.

$$R = \mathrm{UWV}^{\mathrm{T}} \tag{2.15}$$

$$R = \mathrm{UW}^{\mathrm{T}}\mathrm{V}^{\mathrm{T}} \tag{2.16}$$

kde

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Hledaný vektor posunutí nabývá jeden z následujících dvou tvarů

$$T = -\mathrm{u}_3 \tag{2.17}$$

$$T = +\mathrm{u}_3 \tag{2.18}$$

kde u_3 značí třetí sloupec matice U. Pokud zkombinujeme dvě možná vyjádření rotační matice s dvěma vektory posunutí, získáme čtyři možná řešení pro vzájemnou polohu kamer. Lze dokázat, že pokud určíme souřadnice některého z korespondenčních bodů pro všechny tyto kombinace, ve třech případech nám vždy vyjde fyzikálně nemožné řešení (bod se bude nacházet za jednou nebo oběma kamerami). Hledanou vzájemnou pozici kamer tedy vyjadřuje poslední zbývající kombinace rotační matice a vektoru posunutí.
Pro zjednodušenou verzi esenciální matice není třeba provádět žádnou speciální operaci. Pro získání rotace z rovnice (2.13) stačí použití standardních goniometrických funkcí a hodnoty posunutí jsou v matici přímo obsaženy.

2.3 Kalibrace kamery

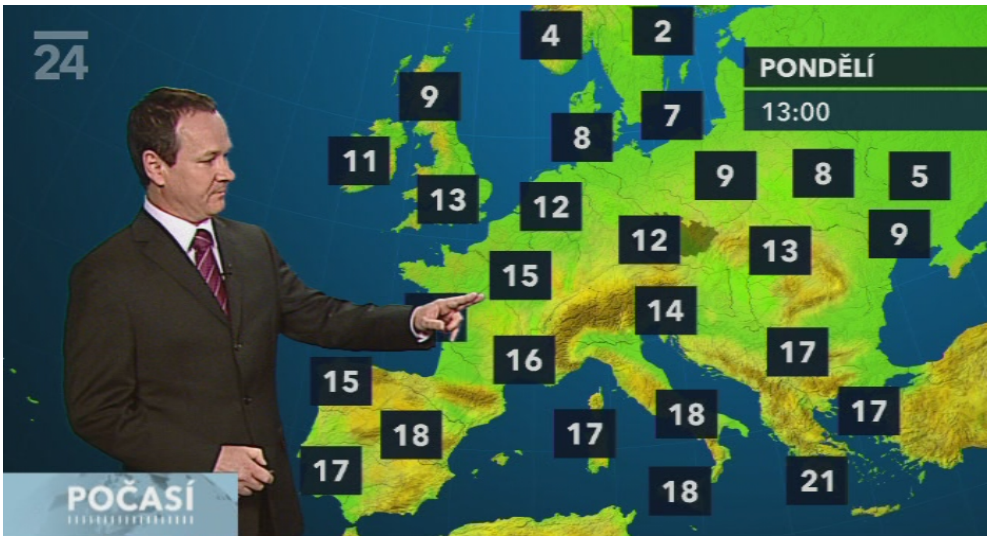
Kalibrace je klíčový úkol při realizaci virtuálního studia. Jejím cílem je určení vnitřních a v některých případech i vnějších parametrů kamery (pro popis těchto parametrů viz část 2.1). Některé modely se navíc dokáží vypořádat i s radiálním zkreslením kamery.
Existuje mnoho metod, které tento problém řeší ať již částečně nebo úplně. Tyto metody lze rozdělit do dvou hlavních skupin (Pozn. metod pro kalibraci kamery existuje několik desítek a jejich rozdělení by mohlo být mnohem podrobnější, my si však pro naše potřeby vystačíme s následujícím základním dělením). Komplexnější rozdělení kalibračních metod lze nalézt například v [6].

Metody používající kalibrační vzory

Tyto metody kalibrace využívají znalosti geometrie snímaného obrazu. Často se používají takzvané kalibrační vzory a to jak 2D vzory, jejichž klasickým zástupcem je šachovnicový vzor, tak 3D vzory. Tyto metody se často využívají v oblasti počítačového vidění, což je velmi rozsáhlý obor zabývající se porozuměním obecné 3D scéně. Mezi tyto metody patří například [1] a [7].

Auto-kalibrační metody

Tyto metody nevyžadují žádnou apriorní znalost snímané scény. Ke kalibraci využívají informace získané ze znalosti vzájemných korespondencí bodů detekovaných ve scéně v několika po sobě jdoucích snímcích. Tyto metody se často využívají ve fotogrammetrii, což je obor zabývající se rekonstrukcí 3D objektů či povrchů z 2D snímků. Typickým případem kdy je vhodné použít auto-kalibrační metody je letecké snímkování (při kterém jsou jiné způsoby kalibrace jen obtížně realizovatelné).
My používáme pro kalibraci kamery metodu [1] implementovanou v knihovně OpenCV. Tato metoda využívá mnoha různých pohledů na 2D šachovnicový vzor o známých rozměrech (viz Obr. 2.1). Algoritmus nejprve spočte projektivní transformaci pro všechny vstupní obrazy. Poté spočte počáteční odhad vnitřních a vnějších parametrů pomocí metody nejmenších čtverců a jako poslední krok provede nelineární optimalizaci reprojekční chyby pomocí Levenberg-Marquardtovy metody (viz [8]).

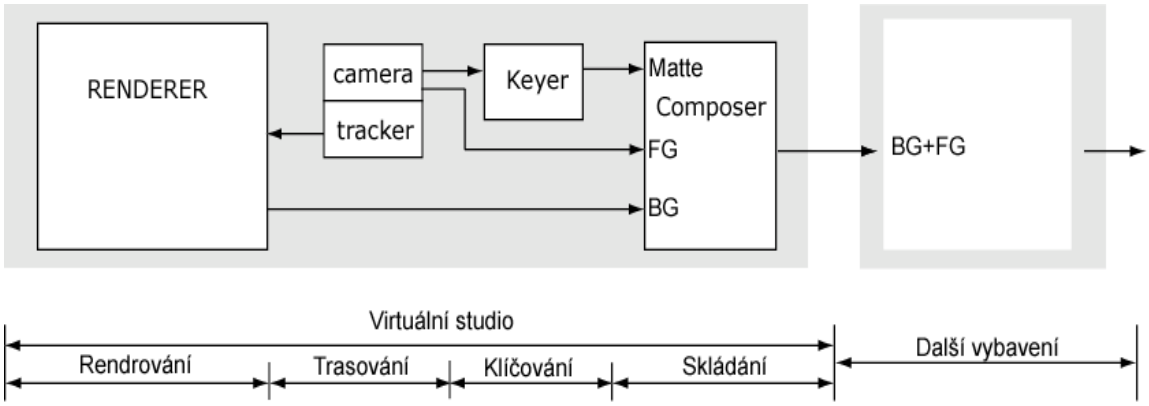


Obr. 2.6 Typické použití tzv. animovaného pozadí (v tomto případě pro předpověď počasí).

3 Virtuální Studio

Virtuální studia jsou dnes již běžně používaným nástrojem v televizní praxi. Vývoj virtuálních studií přímo navazuje na techniku klíčování na modré pozadí (dnes se běžně používají i jiné barvy pozadí, především pak zelená v některých speciálních případech i červená a další barvy, více viz část 3.1), kterou vyvinul na počátku šedesátých let minulého století Petro Vlahos, tvůrce speciálních efektů a zakladatel společnosti Ultimatte. Tento tradiční proces klíčování na modré pozadí se v současné televizní praxi stále velmi často používá, a to zejména při tvorbě tzv. animovaných pozadí jaké můžeme vidět například v televizních předpovědích počasí, viz Obr. 2.6. Tato technika má však jednu značnou nevýhodu, která omezuje její použití v moderních televizních produkcích. Při procesu snímání obrazu, který má být následně klíčován, musí kamera zůstat statická. Jakákoliv změna pozice kamery či prostá změna ohniskové vzdálenosti (zoom) nebo ostření většinou zásadním způsobem naruší výsledný vjem z virtuálního prostoru, který se snažíme divákovi zprostředkovat. Technologie virtuálního studia má tedy za cíl umožnit tvůrcům kromě prostého klíčování i sledovat pozici kamery snímající reálnou scénu a pomocí této informace měnit parametry virtuální kamery, pomocí níž je vytvářen obraz

virtuální scény tak, aby divák získal pocit, že se reální herci skutečně nachází v tomto virtuálním prostředí. Přestože neexistuje žádná oficiální definice pro virtuální studio, z předchozí části lze vyčíst několik zásadních vlastností, které jsou pro všechna virtuální studia společné (viz [9]).
Virtuální studio musí umožňovat klíčování na modré (či jiné vhodné) pozadí. Dále musí umožňovat sledovaní pozice reálné kamery či kamer a v neposlední řadě musí umět vykreslovat virtuální scénu. Obecné schéma virtuálního studia je zobrazeno na Obr. 3.1. V následujících částech si tyto jednotlivé části virtuálního studia rozebereme blíže a pokusíme se nastínit výhody a nedostatky jejich různých implementací. Pozn. pro lepší orientaci v literatuře uvádíme za českým názvem daného odvětví i název anglický.



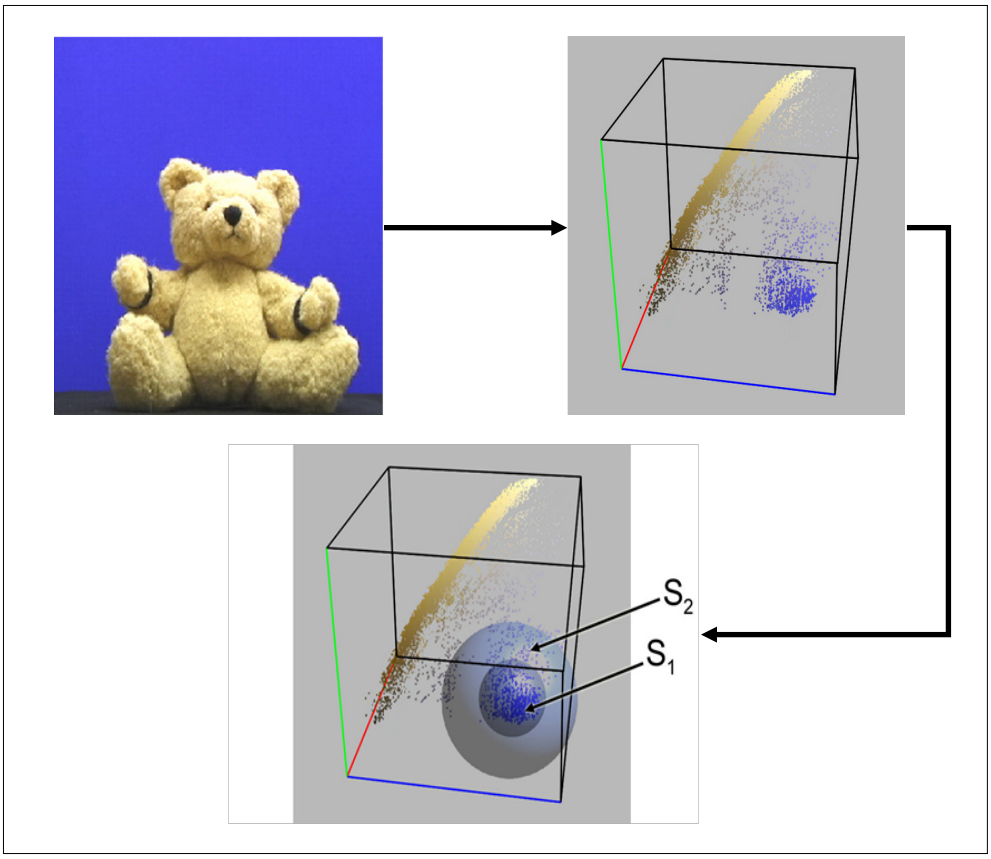
Obr. 3.1 Obecné schéma virtuálního studia, které se skládá ze 4 základních částí: rendrování, trasování, klíčování a skládání (převzato z [9], upraveno)

3.1 Maskování (Matting)

Maskování je jedna ze základních operací při zpracování obrazu, a to nejen v televizní či filmové praxi, ale i v mnoha dalších oborech. K problematice maskování obrazu existuje velmi rozsáhlá teorie vztahující se k různým jejím aspektům. My se zaměříme pouze na jednu konkrétní oblast maskování, a to na maskování částečně průhledného popředí (jímž může být například obraz moderátora ve studiu) na neprůhledném pozadí (například modré pozadí). Tento proces lze popsat rovnicí:

$$C = \alpha_f F + (1 - \alpha_f) B \qquad \textbf{(3.1)}$$

Kde C je známý obraz (v teorii skládání obrazu nazýván jako kompozit) a my se snažíme nalézt barvu popředí F, barvu pozadí B a masku α_f, která popředí od pozadí odděluje. Je zjevné, že tato rovnice nemá obecně řešení (máme jedinou rovnici a tři neznámé). Bylo vyvinuto mnoho metod, které se snaží tento problém řešit (typicky přidáním nějaké další informace do rovnice). Jednou z nich je již dříve zmíněná metoda maskování na modré pozadí. Tato metoda přináší do předchozí rovnice informaci o barvě pozadí B. Je zřejmé, že i takto upravená rovnice stále nemá obecné řešení, protože nám zůstávají dvě neznámé F a α_f. Byly však vyvinuty postupy založené na teorii barev a barevných systémů, které se snaží odhadnout parametr α_f především na základě uživatelem či heuristicky definovaných konstant. Tři z těchto postupů jsou popsány v následující části.



Obr. 3.2 Vlevo obraz, pro který má být vytvořena maska. Vpravo zobrazení jednotlivých bodů obrazu v barevném prostoru RGB. Uprostřed vyznačení oblastí S1 pro jejíž body má být maska zcela průhledná a S2 pro jejíž body má být maska částečně průhledná.

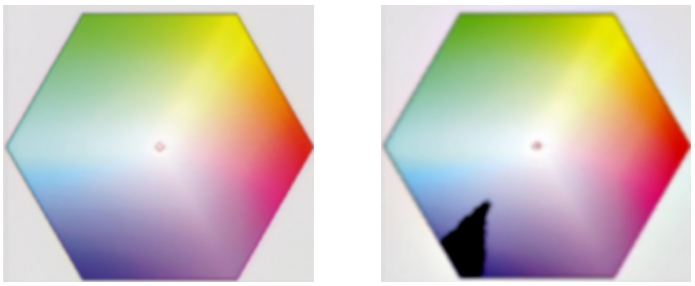
3D maskování

Jedná se o skupinu maskovacích metod založených na reprezentaci barvy v trojrozměrném barevném prostoru (většinou RGB). Pokud si zobrazíme maskovaný obraz v daném prostoru, typicky nám vzniknou minimálně dva shluky bodů, z nichž jeden reprezentuje body náležící k pozadí. Při 3D maskování se pak snažíme definovat vhodné obalové těleso, pro které by platilo, že všechny body pozadí jsou uvnitř tohoto tělesa a všechny body popředí jsou vně. Na základě prostého rozhodovacího pravidla (typu je bod uvnitř tělesa) je pak vytvořena hledaná maska. Tuto metodu lze dále rozšířit tak, aby produkovala i částečně průhledné masky, a to buď přidáním druhého obalového tělesa, viz Obr. 3.2 či podmínkou určující transparentnost masky například na základě vzdálenosti bodu od těžiště obalového tělesa.

Chroma maskování

Tato metoda maskování vychází z vlastností barevného prostoru HSV, ve kterém je barva reprezentována pomocí trojice hodnot: odstín barvy H, saturace S, jas V. Při samotném maskování je pak definována oblast barevných odstínů (typicky se jedná o souvislou oblast kolem konkrétního například modrého odstínu), které jsou označeny za pozadí. Použitím jednoduchého rozhodovacího pravidla je opět vytvořena hledaná maska. Stejně jako u předchozí metody lze pro vytvoření částečně průhledných

masek definovat druhý region či doplnit jiné jednoduché pravidlo. Pro ukázku, jak může vypadat typická maskovací oblast pro modré pozadí, viz Obr. 3.3.



Obr. 3.3 Vlevo diagram zobrazující různé barevné odstíny. Vpravo tentýž diagram s černě vyznačenou oblastí barevných odstínů, které mají být vymaskovány.

Difference Matting

Tato metoda je založena na předpokladu, že pro všechny body pozadí reprezentované v prostoru RGB platí

$$B > max(G, R).$$ ***(3.2)***

Pozadí by mělo mít co „nejčistší“ modrou barvu, tedy barvu, jejíž obraz v RBG má co největší rozdíl mezi modrou složkou a ostatními dvěma složkami. Pro barvu objektů popředí by měla samozřejmě platit opačná nerovnost. Maska je pak vytvořena podle vztahu:

$$\alpha = \frac{\mathrm{B} - \max(\mathrm{G}, \mathrm{R})}{\mathrm{B}_r - \max(\mathrm{G}_r, \mathrm{R}_r)}, \quad \textbf{(3.3)}$$

Kde (R_r G_r B_r) je referenční barva pozadí zvolená uživatelem.

3.2 Sledování pozice kamery (Tracking)

Sledování pozice kamery je zcela zásadní součástí systému virtuálního studia. Bez přesného a spolehlivého sytému, který dokáže určit základní parametry kamery, kterými jsou její pozice, natočení, ohnisková vzdálenost a v některých případech clona (pokud chceme při rendrování použít efekt hloubky ostrosti, viz část 4) nelze dosáhnout skutečně dobré iluze virtuálního prostoru. Systémy pro sledování kamery se dělí do tří základních skupin podle principu, na kterém jsou založeny. Jsou to elektromechanické systémy, optické systémy a systémy pracující na principu trasování (infračerveného) bodu v prostoru (pro jednoduchost budeme dále tyto systémy označovat jako systémy pro infračervené trasování).

3.2.1 Elektromechanické systémy

Tyto systémy jsou založeny na soustavě čidel, která jsou umístěna v těle platformy, na které je kamera umístěna a často i na kameře samotné. Obecně lze tyto systémy rozdělit na aktivní, které mají místo čidel servomotory a nelze je ovládat přímo, ale ovládá je na dálku operátor u příslušného řídícího pultu, a pasivní, které jsou ovládány přímo kameramanem. Jak bylo zmíněno dříve, soustava čidel či servomotorů je součástí platformy, na které je kamera umístěna a není tedy možné,

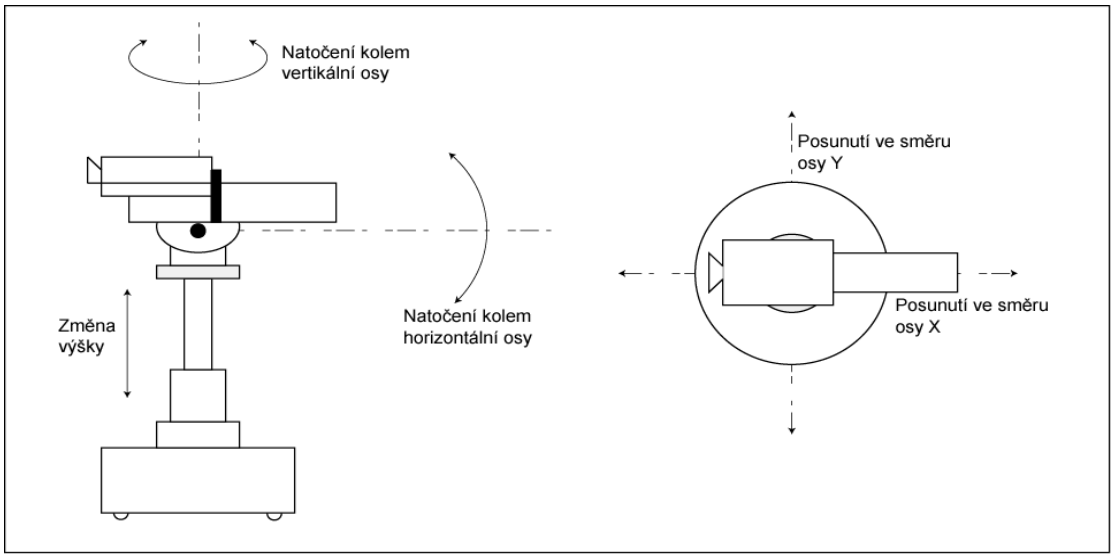
aby byla kamera umístěna mimo tuto platformu, například aby ji držel kameraman v ruce. Tyto platformy můžeme rozdělit na:

- Stativy s výkyvnou hlavou (dva stupně volnosti pro náklon hlavy stativu ve dvou osách).

- Statické podstavce s výkyvnou hlavou a měnitelnou výškou hlavou (tři stupně volnosti, dva pro náklon hlavy a jeden pro měnitelnou výšku podstavce v ose viz Obr. 3.4).

- Pohyblivé podstavce s výkyvnou hlavou, měnitelnou výškou a možností pohybovat celým podstavcem buď v jedné či dvou osách (až pět stupňů volnosti dva pro náklon hlavy a tři pro pohyb podstavce v osách X, Y, Z) viz Obr. 3.4.

- Kamerové jeřáby (většinou se statickou základnou) umožňující natočení kamery kolem čtyř os.



Obr. 3.4 Schéma kamery umístěné na pohyblivém podstavci a zobrazení 5 stupňů volnosti, které kamera má.

Krom výše zmíněných parametrů musí být každý takový systém schopen získávat další parametry přímo z kamery a to především ohniskovou vzdálenost (zoom) a clonu. Dále popíšeme výhody a nevýhody těchto systémů.

Výhody:

- Vysoká přesnost, která u většiny komerčních systémů dosahuje tisícin stupně pro natočení a milimetrů pro určení pozice.

- Možnost použití v „exteriéru“. Použití tohoto systému není přímo vázáno na konkrétní řízené prostřed, lze jej proto využít pro „přenosná“ virtuální studia, která lze dle potřeby postavit na různých místech.

Nevýhody:

- Vysoká cena především při použití komplexních systémů s více kamerami, které jsou umístěny na platformách s maximálním stupněm volnosti.

- Náročná obsluha. Tyto systémy jsou poměrně komplikované a typicky je třeba poměrně značné úsilí pro jejich kalibraci před každým použitím, především v případech, kdy se parametry systému mezi jeho jednotlivými spuštěními mění. Tuto nevýhodu lze částečně eliminovat použitím vhodné formy optického trasování pro kalibraci systému viz, část 3.2.3.

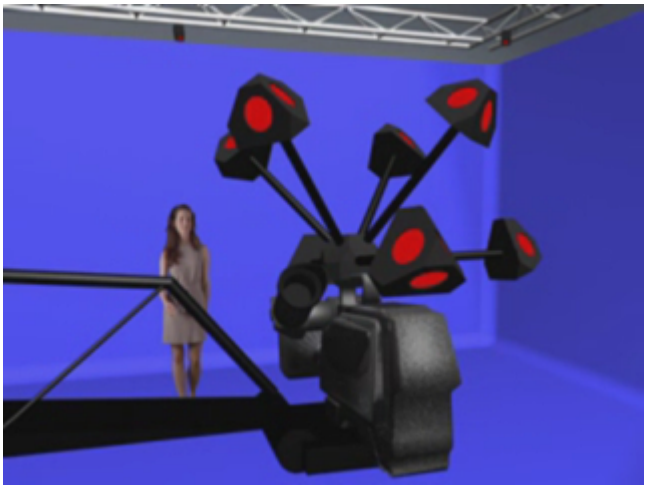
- Omezená volnost pohybu kamery. Kamera musí být pevně spojena s příslušnou platformou, což je limitujícím faktorem pro volnost jejího pohybu. V typické televizní produkci, která zahrnuje virtuální studio, si však s výše uvedeným počtem stupňů volnosti vystačíme.

3.2.2 Systémy pro infračervené trasování

Tyto systémy jsou založeny na trasování objektů (obvykle bodů) vydávající infračervené záření. Pro snadnější trasování jsou tyto body obvykle uspořádány do různých speciálních vzorů, viz Obr. 3.5. Tyto vzory jsou trasovány pomocí systému statických kamer umístěných na zdech a stropě studia. Pro profesionální trasování se obvykle požívá 12 a více takových kamer s vysokým rozlišením. Typické rozmístění kamer pro systém OptiTrack (profesionální trasovací systém, viz [10]) je zobrazeno na Obr. 3.6.

Samotný proces trasování se skládá z několika částí. Nejprve je třeba celý systém kalibrovat, tedy určit vnitřní i vnější parametry kamer (viz část 2.3). Vnější parametry všech kamer musí být samozřejmě určeny vzhledem k jedinému společnému souřadnému systému, který by měl být vhodně zvolen tak, aby bylo jednoduše možné jej ztotožnit se souřadným systémem virtuálního světa. Pokud se poloha kalibrovaných kamer nezmění, není nutné proces kalibrace opakovat mezi jednotlivými použitími systému. Všechny další části procesu trasování je samozřejmě nutné opakovat pro každý snímek zvlášť.

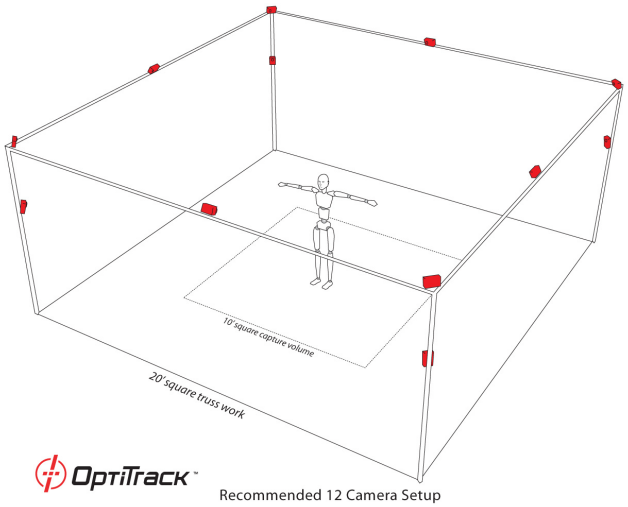
Další fází trasování je detekce trasovaných bodů v obrazech kamer. Tento proces značně zjednodušuje použití infračervených trasovacích objektů, které nám dovolují za předpokladu, že se ve scéně nenachází další zdroj infračerveného záření, použít jednoduché metody pro zpracování obrazu, jako je prahování, s velmi dobrými výsledky. Po provedení detekce je třeba nalézt vzájemné korespondence mezi detekovanými body (tedy určit, které z bodů detekovaných v obrazech kamer, si vzájemně odpovídají). Existuje cela řada strategii, jak tento problém řešit, některé z nich lze nalézt například v [11]. Ve chvíli kdy jsme určili všechny tyto korespondence, můžeme použít jednoduchou triangulaci pro určení pozic jednotlivých trasovaných objektů v souřadném systému definovaném během kalibrace kamer. Ze znalosti pozic jednotlivých objektů v prostoru a způsobu jejich upevnění na kameru můžeme již poměrně přímočaře určit pozici kamery a její natočení. Informace o ohniskové vzdálenosti a fokusu je třeba získat přímo z kamery a to buď přímo (pokud to kamera umožňuje), nebo pomocí přídavného čidla umístěného na kameře. Následuje souhrn kladů a záporů této metody určení pozice kamery.



Obr. 3.5 Infračervené objekty tvořící speciální trasovací vzor (převzato z [12])

Výhody:

- Poměrně velká přesnost. Nelze ovšem předpokládat stejnou přesnost jako u elektromechanických systémů.
- Náklady na přidání druhé a každé další kamery do systému jsou minimální. Je nutné dokoupit pouze další trasovací objekty a připevnit je na kamery.
- Flexibilita. Systém není omezen ani z hlediska pozice kamery ani z hlediska velikosti jejího natočení. Tento systém není přímo vázán na žádnou konkrétní platformu, ke které by musela být kamera připojena, lze jej tedy použít i na tzv. „natáčení z ruky“.
- Neklade žádné další požadavky na fyzické uspořádání studia (do kterého patří například rozložení a velikost klíčovacích ploch, či jejich osvětlení). Jedinou podmínkou je, že trasovaná kamera musí být v zorném poli dostatečného množství statických infračervených kamer.



Obr. 3.6 Jedno z možných rozmístění trasovacích kamer (převzato z [10])

Nevýhody:

- Poměrně vysoká pořizovací cena základního vybavení.
- Nemožnost použití v exteriérech.

3.2.3 Optické systémy

Optické systému pracují na poněkud jiném principu než dva předchozí. Zatímco elektromechanické a „infračervené“ systémy využívají pro určení pozice kamery přídavných zařízení, která jsou připevněna ke kameře a/nebo umístěna ve studiu, optické systémy se snaží získat informaci o pozici kamery přímo z obrazu, který kamera snímá.
Nejčastěji používaná technika v optických systémech se nazývá rozpoznávání vzorů (viz např. [13]). Tato technika vychází z předpokladu, že ve snímané scéně se nachází objekt či objekty s předem známou vhodně definovanou geometrií a tyto objekty se pak snaží detekovat. Pokud se jí to podaří, snaží se ze známé výchozí pozice trasovacího vzoru a jeho nově detekované pozice vypočítat hledanou pozici kamery. Pomocí této metody lze získat všechny informace potřebné pro trasování kamery, kterými jsou pozice a natočení kamery a její ohnisková vzdálenost. Při online zpracování obrazu se žádná z těchto metod příliš neosvědčila. Důvody jsou dva a jsou zcela zásadní. Prvním z nich je přesnost, která se nemůže rovnat ani jedné z dříve popsaných metod a druhým zásadním důvodem je nízká stabilita s ohledem na šum ve vstupních datech, která zapříčiňuje nekonzistenci mezi obrazem virtuální a skutečné scény (dochází k nenadálým záchvěvům). S jejich nasazením se častěji setkáme v postprodukci, kde lze tyto artefakty minimalizovat.
Dalším problém spojeným s uvedením této metody je fyzická reprezentace vzoru, který má být v obraze detekován. Je zřejmé, že takový vzor musí splňovat několik požadavků:

- Snadná detekovatelnost
- Vzor se nesmí objevit ve výsledném obraze
- Geometrie vzoru musí být známa velmi přesně

Jedinou rozšířenou implementací této metody, která splňuje všechny předchozí požadavky včetně požadavku na přesnost a stabilitu je systém Xync firmy Orad (viz část 4), který používá speciální trasovací vzor ve tvaru dvourozměrné mřížky (viz Obr. 3.7). Tento systém je však schopen detekovat pouze natočení kamery a ohniskovou vzdálenost, pro určení pozice kamery používá zjednodušenou verzi systému pro trasování infračervených objektů. Jedná se tedy o systém kompozitní a nelze jej pokládat za čistě optický. Následuje shrnutí výhod a nevýhod systému:
Výhody:

- Nízká cena
- Možnost přidávat libovolné množství kamer

Obr. 3.7 Modré klíčovací pozadí se speciálním vzorem pro optické trasování kamery

Nevýhody:

- Problematická přesnost a stabilita systému
- Nutnost začlenit trasovací vzor do klíčovacího pozadí, související problémy při klíčování, problematická tvorba vysoce kvalitních kompozitních obrazů obsahujících poloprůhledné objekty a stíny
- Omezení při osvětlení scény tak, aby splněn požadavek na snadnou detekovatelnost trasovacího vzoru

3.3 Renderování (Rendering)

Rendering je dynamicky se rozvíjející oblastí počítačové grafiky, která se zabývá vykreslováním obrazu virtuální scény (nejčastěji definované jako 3D model) a jako taková má nezastupitelné místo v systému virtuálního studia. Úkolem rendrovací části virtuálního studia je vytvoření obrazu virtuální scény ve správné synchronizaci s trasovacím systémem studia, který udává polohu reálné kamery.
Na rendrovací systém jsou kladeny různé požadavky v závislosti na charakteru virtuální scény a míře realističnosti, které chceme při jejím zobrazení dosáhnout (scéna může obsahovat například dynamické stíny, nerovinná zrcadla, různé druhy průhledných povrchů a speciálních materiálů mnoho dalších efektů). Tyto požadavky se pak samozřejmě odrážejí v nárocích na software i hardware, který je musí být schopen zpracovat v reálném čase (čas pro vykreslení jednoho snímku může být buď 40ms nebo 20ms v závislosti na tom, zda jej vykreslujeme pro každý snímek nebo půlsnímek z reálné kamery (předpokládáme-li výstup z kamery ve formátu PAL)). V posledních letech se stále více prosazují televizní formáty s vysokým rozlišením (největší z nich má rozměr 1920x1080 obrazových bodů), jejichž nároky na výpočetní výkon rendrovací jednotky jsou samozřejmě násobně větší než u systémů se standardním rozlišením, které je pro normu PAL 720x576 obrazových bodů. Existují dva různé přístupy, kterými je vytvářen obraz virtuální scény. Prvním z nich je tzv. 2D rendering a druhým 3D rendering.

2D rendering

Tato technika využívá tzv. předrendrovaného pozadí. Její použití předpokládá, že máme 2D obraz pozadí, který byl vytvořen se známou polohou kamery. Reálnou kameru pak musíme umístit do polohy co možná nejbližší té virtuální a poté její obraz skládáme přímo s předrendrovaným pozadím. Pohyb reálné kamery je tedy zásadně omezen (v ideálním případě by se kamera neměla pohybovat vůbec, pak bychom již ovšem stěží mohli mluvit o virtuálním studiu). Existují techniky, pomocí nichž lze simulovat některé pohyby kamery, a to především rotaci kolem svislé a vodorovné osy a zoom, to samozřejmě jedině za předpokladu, že nám to rozsah předrendrovaného 2D pozadí dovolí. Kvalita iluze pohybu ve virtuálním světě je však u všech těchto metod minimálně sporná. Vyvstává tedy otázka, zda se vůbec najdou případy, ve kterých je použití tohoto systému vhodné. Takové případy existují minimálně dva. Prvním z nich je jednoduše případ, kdy není možné ať už s technických či jiných důvodu rendrovat scénu v reálném čase. Tento scénář není úplně nepravděpodobný, pokud si uvědomíme, že vykreslení některých scén ve fotorealistické kvalitě (a právě fotorealismus je typicky meta, které se chceme přiblížit) může trvat i na výkonných strojích mnoho hodin. Druhým případem je situace, kdy od virtuální scény požadujeme atypický vzhled, jehož nelze pomocí standardně používaného softwaru jednoduchým způsobem dosáhnout. Dobrým příkladem takové scény je virtuální prostředí na Obr. 3.8.

2D rendering se tedy může obecně uplatnit v případech, kdy máme na virtuální scénu velmi specifické nároky. Navíc je třeba si uvědomit, že pokud využíváme pro rendrování pouze této techniky nejsme schopni využít informace udávající případnou změnu polohy kamery a při fyzickém návrhu virtuálního studia je vhodné s tím počítat.

Obr. 3.8 Typické použití 2D renderingu pro vytvoření nefotorealistické virtuální scény

3D rendering

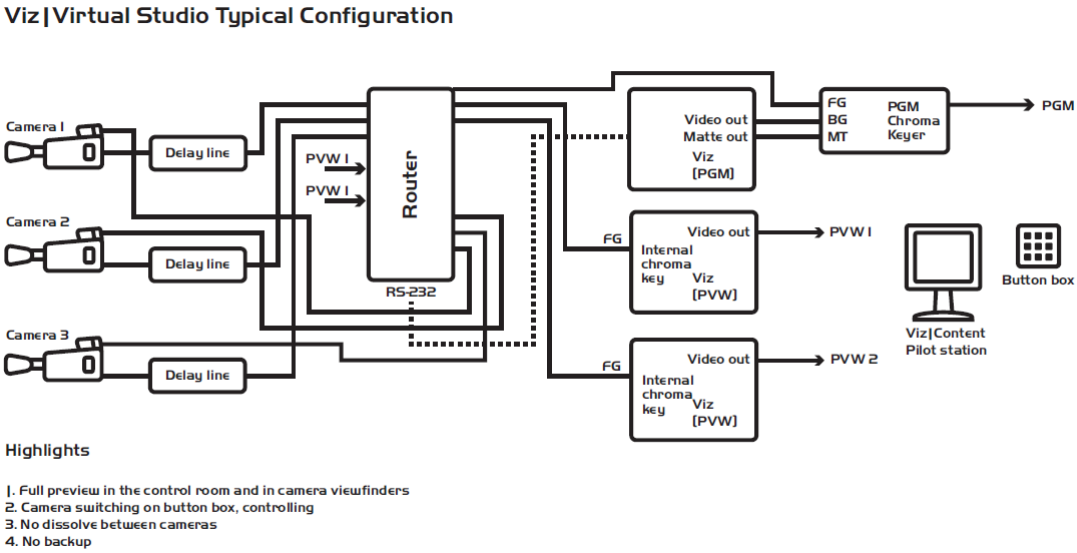
Při použití této metody lze plně rozvinout všechny možnosti, které nám koncept virtuálního studio nabízí. Virtuální scéna musí být v tomto případě definována jako 3D model (pro vytvoření takového modelu lze použít některý z mnoha CAD/CAM systémů, viz část 7.1). Tato scéna je pak snímána virtuální kamerou, která má stejné parametry (pozici, orientaci a ohniskovou vzdálenost) jako kamera reálná. Na pozici reálné kamery nejsou v tomto případě z hlediska renderingu kladeny žádné speciální

požadavky. 3D rendering také umožňuje použití celé řady efektů, jako jsou například dynamické stíny 3D animace nebo simulace ostření. Obecně lze říct, že jedinými faktory, které nás při použití této metody ve snaze po dosažení co nejlepšího výsledku omezují, jsou míra detailu použitého 3D modelu a dostatečný výpočetní výkon.

4 Existující Systémy

První systémy začaly vznikat již kolem roku 1990. Od té doby prošla studia obrovským vývojem a přerodily se z jednoduchých aplikací (ve smyslu rozsáhlosti) nabízejících pouze omezenou funkcionalitu a bojujících o úsporu bezmála každé operace a každého trojúhelníku do komplexních systémů s obrovskou škálou možností a téměř neomezeným výpočetním výkonem. Tomuto odpovídá i jejich cena. Přestože je i v této oblasti jasně viditelný trend snižování cen a virtuální studia se stávají dostupnější i pro menší televizní produkce, cena špičkových systémů (které jsou vždy stavěny na míru podle požadavků zákazníka a to se týká nejen hardwaru ale i softwaru) může dosáhnout stovek milionů korun.

Popsat detailně vlastnosti jednotlivých existujících komerčních systémů je velmi obtížné bez možnosti vidět je v praxi a případně srovnat jednotlivé systémy mezi sebou. Lze však s velkou mírou jistoty tvrdit, že současné špičkové systémy se v základních vlastnostech popsaných v kapitole 3 liší jen minimálně a větší rozdíly můžeme nalézt především v komplexnosti a rozsahu částí, které se dodávají spolu se systémem nebo které lze k systému připojit (opět se to týká jak hardwaru, tak softwaru). Pro úplnost uvedeme dva nejrozšířenější komerční systémy.



Obr. 4.1 typická konfigurace systému Viz Virtual Studio (převzato z [14])

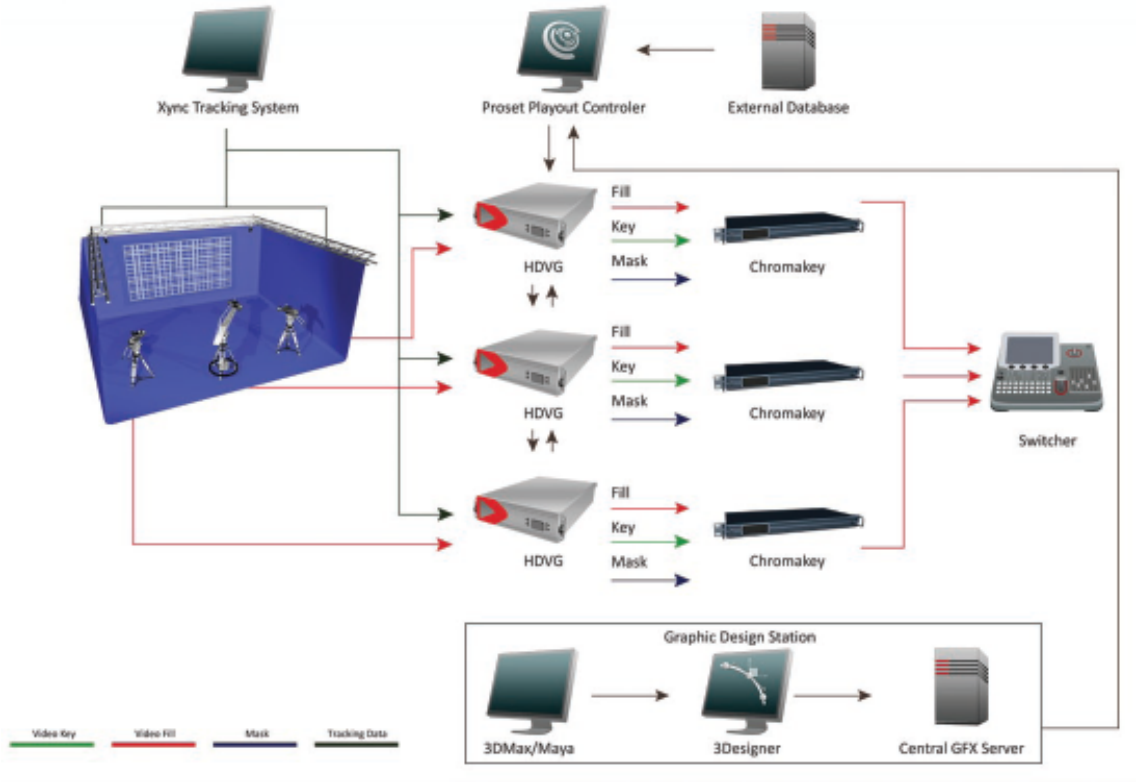
Viz Virtual Studio

Viz Virtual Studio je produktem firmy Vizrt, která je světovým lídrem na poli produkce virtuálních studií (její studia používají přední televizní stanice jako americké CNN a ABC či německá ZDF). Viz Virtual Studio splňuje všechny požadavky, které jsou kladeny na virtuální studia nejvyšší kvality, jako je vysoká přesnost a spolehlivost trasování kamer (podporuje všechny popsané trasovací

metody), kvalitní rendering a maskování ve vysokém rozlišení jsou samozřejmostí. Kromě toho umožňuje studio integraci s celou řadou další produktů firmy Vizrt a je tak skutečně komplexním systémem (pro více informací viz [14]). Typická konfiguraci tohoto virtuálního studia je zobrazena na Obr. 4.1

Orad

Virtuální studio firmy Orad je dalším široce rozšířeným systémem. Toto studio má velmi podobné vlastnosti, jako výše zmiňované Viz Studio. Podporuje všechny běžné metody trasování kamer, rendrování i maskování ve vysokém rozlišení. Další informace o tomto systému je možné nalézt v [12]. Typická konfigurace tohoto studia je na Obr. 4.2.



Obr. 4.2 Typické konfigurace systému Orad (převzato z [12])

5 Nízkorozpočtové Studio

Ve dvou předchozích částech jsme popsali, co je virtuální studio, z jakých částí se skládá a k jakým účelům se používá v praxi. Dále jsme uvedli konkrétní příklady dvou špičkových virtuálních studií, jejichž cena se může šplhat až do stovek miliónů korun, a popsali jsme jejich vlastnosti. Jak již bylo zmíněno v úvodu, naším cílem není těmto špičkovým produktům konkurovat, ale navrhnout nízkonákladové virtuální studio s pokud možno co nejlepšími vlastnostmi, použitelné v případech kdy není realizace některého komerčního systému z finančních důvodů možná.

V části 3 jsme definovali čtyři základní úkony, které musí být schopné realizovat každé virtuální studio. Jsou jimi sledování pozice kamery, klíčování, rendrování virtuální scény a skládání výsledného obrazu. V dalším textu se zaměříme na popis nejvhodnějších způsobů realizace těchto procesů v nízkorozpočtovém studiu a definujeme ještě některé další vlastnosti, které by toto studio mělo splňovat.

První proces, na který se podíváme, je klíčování. Při výběru nejvhodnější metody pro nás byli podstatná dvě kritéria a to kvalita vygenerované masky a rychlost jejího vygenerování. Podívejme se, jakým způsobem tato kritéria splňují metody popsané v části 3.1 (Chroma Mattinig, 3D Matting, Difference Matting), pokud jsou jejich vstupem data ze zařízení standardu DV, jejichž použití v nízkorozpočtovém studiu přepokládáme.
Technika Chroma Matting se nedokáže vypořádat s artefakty, které jsou spojeny s DV daty (jako například poměrně vysoký stupeň barevného šumu u čipů běžných DV zařízení) a které mají za následek, že hodnoty H (odstín barvy) jsou především v tmavých místech špatně definované. Důsledkem toho jsou masky produkované touto technikou značně nekvalitní. Rychlost není u této metody zásadním problémem.
Technika 3D Matting si i vzhledem k tomu, že je poměrně sofistikovaná, s DV daty dokáže poradit velmi dobře a produkuje kvalitní masky. Jejím problémem je rychlost, která vzrůstá spolu se složitostí použitých obalových těles a pro kvalitní maskování je nutné použít skutečně komplexní obalová tělesa.
Difference Matting je osvědčená metoda, která je známa již několik desítek let. Je to nejpoužívanější způsob klíčování v televizní praxi, a to ze dvou důvodů. Jednak produkuje velmi kvalitní masky, jednak je dokáže díky své jednoduchosti, díky které se také jako jediná dočkala i hardwarové implementace, produkovat velmi rychle.
Z předchozího popisu jasně vyplývá, že nejvhodnější metodou pro realizaci klíčování v nízkorozpočtovém studiu je Difference Matting.
Další funkcí, kterou je nutné implementovat, je rendrování virtuální scény. V praxi se používají především dvě technologie pro rendrování 3D objektů a to OpenGL a DirectX. Obecně nelze určit, která z těchto technologií je lepší. S velkou mírou pravděpodobnosti však lze tvrdit, že pro účely virtuálního studia se dají použít obě bez zásadních rozdílů pro kvalitu nebo rychlost požadovaného výstupu. Kromě technologie rendrování je nutné definovat nároky, které klademe na virtuální scénu samotnou (tedy co vše by měla obsahovat a jak by měla po vykreslení vypadat). Ideálem je samozřejmě dosažení fotorealističnosti, což je však meta pro nízkonákladový systém zcela nedosažitelná. Pokusili jsme se tedy alespoň definovat minimální soubor vlastností, které by měl každý systém splňovat.

- Rendrování objektů složených ze základních grafických primitiv (nejčastěji trojúhelníků)
- Rendrování textur
- Rendrování materiálů
- Implementace některého z osvětlovacích modelů (s minimálně jedním polohovatelným světlem)
- Možnost manipulace s objekty scény (změna velikosti, posunutí, rotace)

Jedná se skutečně o naprosté minimum, které postačuje jen pro vytvoření základní iluze virtuálního prostoru. Výčet funkcí, které by bylo možné do systému pro zlepšení výsledného vněmu doplnit, je téměř neomezený. Pro příklad uveďme implementace dynamických stínů, umožnění animace objektů ve scéně, rendrování průhledných objektů atd.

Částí úzce provázanou s rendrováním scény a klíčováním je skládání výsledného obrazu. Tento proces je v našem případě, kdy skládáme pouze dva obrazy a máme vytvořenou potřebnou masku, velmi přímočarý a spočívá v prosté aplikaci rovnice (3.1). V případě, že bychom chtěli skládat více obrazů (například přidat do obrazu řádek s titulky), museli bychom vytvořit další masku (pro každý obraz) a celý proces by se pak opakoval.
Posledním ze základních úkolů virtuálního studia je sledování pozice kamery. V této oblasti jsme na rozdíl od ostatních, kde jsme požadavek na minimální ceny virtuálního studia vůbec nezmiňovali, tímto požadavkem výrazně omezeni. Cena všech komerčních systémů uvedených v části 3.2 je značná. Pokud se máme rozhodnout, který z nich použít, musíme rozlišit dvě situace. Pokud chceme ve virtuálním studiu používat pouze jednu kameru a neplánujeme v budoucnu jeho rozšíření, je pro nás nejlepší volbou použití speciálního stativu, v jehož hlavě jsou senzory snímající natočení kamery, spolu se senzorem pro snímání ohniskové vzdálenosti kamery (pokud nám kamera neumožňuje tento údaj získávat jiným způsobem). Toto řešení spadá do kategorie tzv. elektromechanických systémů (viz část 3.2.1). Náklady na takový systém jsou srovnatelné či menší než u konkurenčních systémů (IR, nebo optický systém) a přesnost jaké jsme schopni v tomto případě dosáhnout je zdaleka nejlepší ze všech. Problém nastává ve chvíli, kdy bychom chtěli použít více než jednu kameru. V tomto případě se náklady na jakýkoliv elektromechanický systém zvětšují úměrně k počtu použitých kamer. Z finančního hlediska je pak takové řešení pro nízkorozpočtové studio nevhodné. Musíme tedy volit jiný systém. Jako nejvýhodnější vzhledem k poměru cena výkon se pro více kamer jeví některý z optických systémů. Nevýhoda, se kterou se v takovém případě musíme smířit, je omezená přesnost.
V našem projektu jsme se z finančních důvodů rozhodli implementovat zjednodušenou verzi optického systému, který umožňuje sledování natočení kamery pouze kolem jedné osy. Implementace plnohodnotného optického systému či elektromechanického systému je jedním z námětů na další práci.

6 Hardware

V této části si nastíníme, jaké minimální hardwarové požadavky by mělo virtuální studio splňovat, a také uvedeme, jaký hardware jsme použili při tvorbě a testování našeho studia. Je nutné poznamenat, že autor tohoto textu se necítí být odborníkem na hardware a některé části následujícího textu je třeba brát spíše orientačně.
Všechna virtuální studia mají několik společných hardwarových částí, kterými jsou jedna či více kamer, klíčovací pozadí, osvětlení, systém pro sledování pozice kamery, řídící a výpočetní jednotku pro zpracování dat.

Kamera
Použitá kamera samozřejmě významně ovlivňuje výstup celého virtuálního studia. Kritérií pro výběr kamery je několik. V první řadě je vhodné zohlednit formát záznamu a velikost čipu kamery. Formátů pro záznam, které kamery podporují, je celá řada, v poslední době se však stále více prosazují kamery formátem záznamu HDV nebo AVCHD. To se ukazuje jako problém, protože data z těchto kamer procházejí tzv. chroma podvzorkováním (4:2:0), jehož následkem je omezení množství informace v barvonosných kanálech (jeden vzorek barvy na čtyři vzorky jasu). Tento fakt značně komplikuje klíčování obrazu z těchto kamer, při kterém vzniká v masce nepříjemný kostičkovaný vzor, viz Obr. 8.14. Je tedy lepší použít

kameru s nižším stupněm podvzorkování (4:2:2) nebo v ideálním případě bez podvzorkování. Pokud jde o velikost čipu kamery, obecně platí pravidlo čím větší čip tím lepší obraz.
Při testování našeho systému jsme použili kameru Canon XM2, jež zaznamenává data ve starším DV formátu, který trpí obdobnými neduhy jako dva výše zmíněné. Naše výsledky ukázali, že použití kamery s DV formátem záznamu, a tedy potažmo i s formáty HDV a AVCHD, je možné avšak nikoliv ideální.

Obr. 6.1 Klíčovací pozadí z PVC folie použité v našem systému.

Klíčovací pozadí

Volba klíčovacího pozadí se často podceňuje. Přitom pozadí se správnými vlastnostmi přispívá podstatným dílem ke zvýšení kvality výsledných kompozitních obrazů. Při výběru vhodného pozadí je třeba dbát především na dvě věci. Zaprvé je třeba věnovat pozornost správnému barevnému odstínu pozadí. Ten je pro klíčovací metodu Difference Matting, která je implementována v našem systému, zcela zásadní. Tato metoda vytváří masky pomocí rozdílu mezi jednotlivými barevnými kanály ve snímku. Z čehož vyplývá, že ideální pozadí by mělo odrážet pouze světlo vlnové délky, která je příslušná barvě, již chceme klíčovat (modrá nebo zelená). To je samozřejmě v praxi nemožné, je ovšem vhodné vybrat materiál, jenž se této podmínce alespoň blíží (literatura se v tomto ohledu často odvolává na pojem „nejzelenější zelená“). Druhým faktorem, který ovlivňuje kvalitu výsledného obrazu, je způsob, jakým se barva pozadí odráží na předmětech či osobách, které stojí před ním. V případě, že je pozadí příliš odrazivé, vzniká ve výsledném obraze kolem všech objektů jakási zelená aura (v literatuře se tento artefakt nazývá green spill), jenž se jen velmi těžko odstraňuje (pozadí by tedy mělo být matné). Na vzniku tohoto artefaktu se kromě vlastností klíčovacího matriálu často rovněž podílí nevhodné nasvícení scény nebo umístění objektů do přílišné blízkosti ke klíčovacímu pozadí. Je nutné poznamenat, že výběr nevhodného klíčovacího pozadí sníží kvalitu výsledného kompozitního obrazu a v podstatě neexistuje způsob, jakým by bylo

možné tento nepříjemný fakt během dalšího zpracování (především v „online“ systémech) zcela kompenzovat.
Při testování našeho systému jsme z finančních důvodů použili zelené klíčovací pozadí vyrobené ze samolepící PVC folie, jejíž cena je 100 Kč/m^2 (viz Obr. 6.1). Vlastnosti tohoto materiálu rozhodně nejsou ideální a na kvalitě výstupů se to projevilo.

Osvětlení
Osvětlení televizních (virtuálních) studií je poměr komplexní disciplína (největší studia mají až několik stovek světelných zdrojů různých druhů) a proto si popíšeme pouze několik zásad, které je vhodné při osvětlení virtuálního studia dodržet.
Nejdůležitějším požadavkem je, aby klíčovací pozadí bylo nasvíceno rovnoměrně a s dostatečnou intenzitou. Je však třeba pamatovat na to, že pokud pozadí přesvítíme, většinou se zvětší nepříjemné barevné odlesky na předmětech v popředí. Tyto odlesky působí ve většině případů ve výsledném obraze velmi rušivě. V případě, že ve výsledné scéně nechceme mít stíny vrhané předměty v popředí, je vhodné nasvítit zvlášť klíčovací pozadí a zvlášť zbytek scény. V opačném případě musíme samozřejmě nasvítit celou scénu najednou. Je však třeba poznamenat, že realistické přenesení stínů do virtuálního prostředí je netriviální úkol.
V našem studiu jsme se museli vypořádat s naprosto nevhodným nasvícením, jehož důsledkem byla bohužel ztráta některých jemných detailů z obrazu kamery.

Obr. 6.2 Infračervená kamera TrackIR 5 použitá v našem optickém trasovacím systému

Systém pro trasování kamery
Různé systémy pro sledování pozice kamery a jejich hardwarové součásti jsme v kostce popsali v části 3.2. Nyní se tedy již budeme věnovat pouze popisu námi realizovaného systému. Pro naše studio jsme realizovali optický sledovací systém pro měření natočení kamery kolem jedné osy pracující na principu epipolární geometrie (jeho softwarová část včetně všech postupů je popsána v části 8.2). Hardwarová část systému se skládá ze zařízení Trackir5 firmy NaturalPoint (cena cca 3000Kč), což je kamera určená ke snímání obrazu v infračerveném spektru, viz Obr. 6.2. Tuto kameru jsme připevnili ke stativu kamery tak, aby se při natočení hlavy stativu kolem vertikální osy pohybovala spolu s ní, viz Obr. 6.3. Druhou částí systému jsou různé trasovací vzory tvořené infračervenými LED diodami.

Výpočetní jednotka
Mozkem celého systému je kontrolní a výpočetní jednotka. Podrobnosti o konfiguraci a vlastnostech těchto částí u profesionálních studií nám bohužel nejsou

Obr. 6.3 Kamera Canon XM2 s připevněným trasovacím zařízením (žlutá krabička)

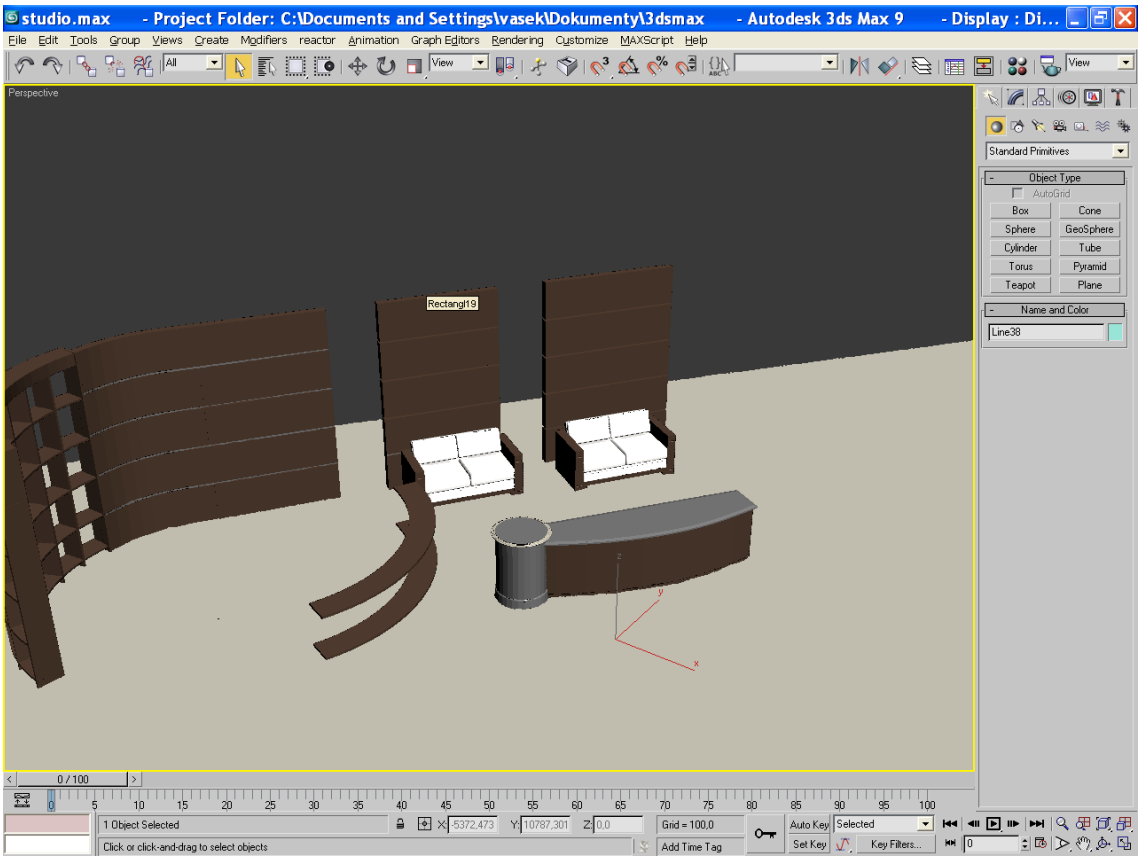
známi. V našem vlastním studiu je pro tuto funkci určen běžný stolní počítač. Při výběru vhodného počítače pro (naše) virtuální studio je vhodné sledovat několik parametrů. Za prvé v každém případě by se mělo jednat o stroj s více jádrovým procesorem (naše aplikace velmi aktivně využívá vlákna). Velikost operační paměti není zcela zásadní, naše aplikace si bez problémů vystačí s 2GB, důležitější je rychlost operační paměti (vzhledem k neustálému přesunu velkého množství dat v aplikaci). Počítač by měl mít jeden, ale v ideálním případě (především pokud chceme data z aplikace ukládat) několik vysokorychlostních velkokapacitních pevných disků. Zcela nevhodné se kvůli omezené rychlosti přenosu dat ukazuje použití většiny externích disků. Poslední podstatnou komponentou je grafická karta. U grafických karet použitých při vývoji a testování naší aplikace se kromě jejich výkonu ukázal ještě další důležitý faktor, který je třeba vzít v potaz. Tímto faktorem je propustnost kanálu mezi operační pamětí počítače a pamětí grafické karty, která především při přesunu dat z grafické karty značně limitovala výkon celého systému.

7 Návrh virtuální scény

Rendrování virtuální scény je nedílnou součástí každého virtuálního studia. Než můžeme jakoukoliv virtuální scénu rendrovat je třeba ji samozřejmě nejprve vytvořit a uložit ve vhodném formátu a nakonec ji načíst do našeho systému. Vytvoření nějakého speciálního nástroje pro návrh virtuální scény není správnou volbou vzhledem k tomu, že existuje mnoho komerčních i volně dostupných nástrojů pro návrh 3D modelů, které lze k tomuto účelu použít. Jediným problémem, který je v tomto případě nutné vyřešit, je výstupní formát takto vytvořené scény. Tento formát musí být schopen uložit všechna data, která virtuální scéna obsahuje (geometrii, materiály, textury, animace atd.), měly by do něj být schopny ukládat svá data aplikace, které chceme k definici virtuální scény použít (viz následující část), a v poslední řadě bychom ho měli být schopni načíst do naší aplikace (ideálně s použitím co nejmenšího úsilí). Všechny tyto požadavky splňuje X File formát vytvořený společností Microsoft primárně pro použití v platformě DirectX (pro více informací o tomto formátu viz část 7.2).

7.1 Nástroje pro návrh virtuální scény

Jak bylo řečeno, existuje mnoho aplikací pro návrh 3D modelů, a to jak komerčních tak nekomerčních. My si v krátkosti představíme jednoho typického zástupce z obou těchto kategorií.



Obr. 7.1 3D Studio Max

3D Studio Max

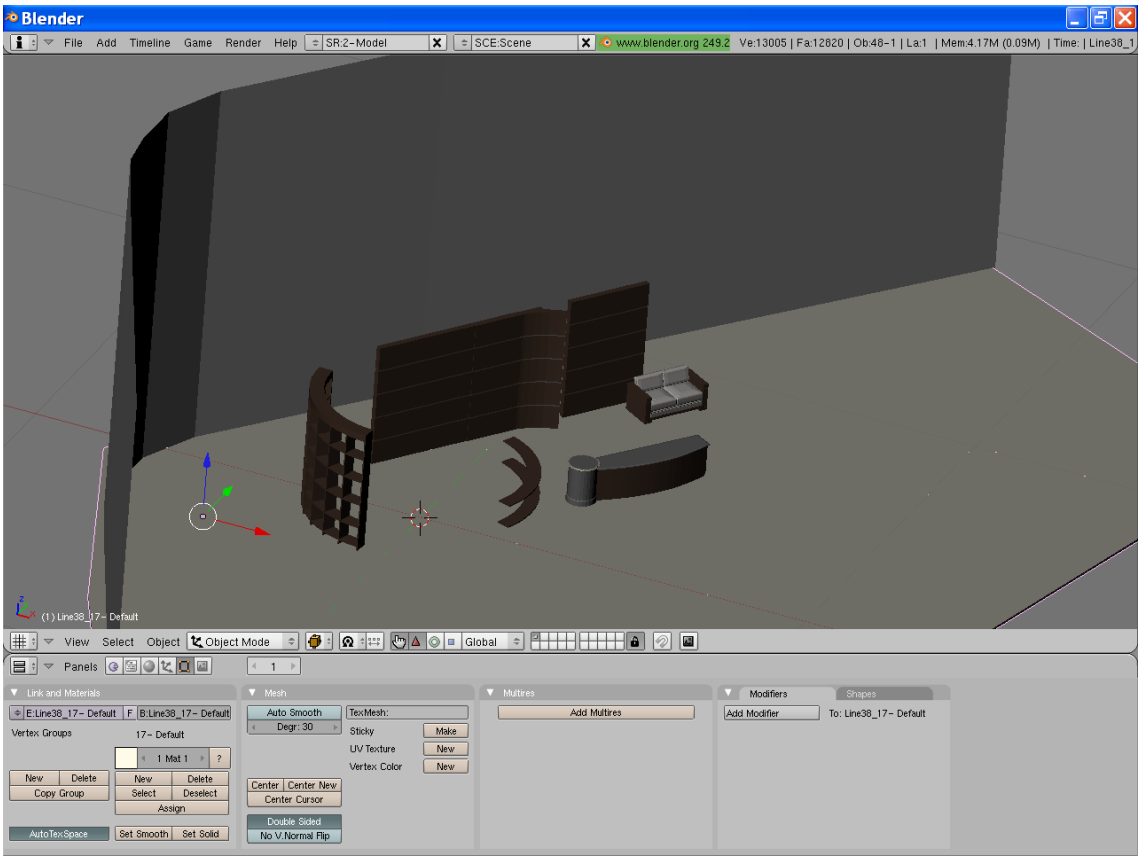
Jedná se profesionální komerční program pro vytváření 3D grafiky od firmy Autodesk (viz [15]). Je běžně využíván pro 3D modelování, návrh počítačových her, v televizní a filmové postprodukci a mnoha dalších oblastech. Svým uživatelům nabízí mimo jiné širokou škálu nástrojů pro modelování a animaci objektů, různé druhy renderingu (včetně globálních osvětlovacích modelů), vlastní skriptovací jazyk (MaxSkript), a mnoho dalších možností. Společnost Autodesk nabízí k tomu produktu také API, pomocí nějž lze vytvářet různé plug-iny. Modely vytvořené v tomto programu podporují i komerční virtuální studia jako Viz nebo Orad (viz část 4).

Blender

Blender je příkladem nekomerčního open source nástroje pro 3D modelování a animace (viz [16]). Díky tomu, že je zdarma, je poměrně rozšířen v počítačové komunitě. Svým uživatelů nabízí širokou škálu možností pro modelování objektu (včetně použití parametrických ploch a křivek) a animaci včetně populární inverzní kinematiky, podporuje rovněž skriptovací jazyk Python a je možné jej rozšířit pomocí plug-inů.

Do srovnávání popsaných modelovacích nástrojů se pouštět nebudeme, oba mají své příznivce i odpůrce a asi nelze jednoznačně určit, který z nich je lepší (byť autoři této práce se kloní na stranu 3D Studia Max). Důležité je, že oba tyto nástroje umožňují definovat plnohodnotnou virtuální scénu pro naše virtuální studio a tuto scénu (po

naistalování příslušných volně dostupných plug-inů) uložit v námi požadovaném formátu.



Obr. 7.2 Blender

7.2 Formát virtuální scény

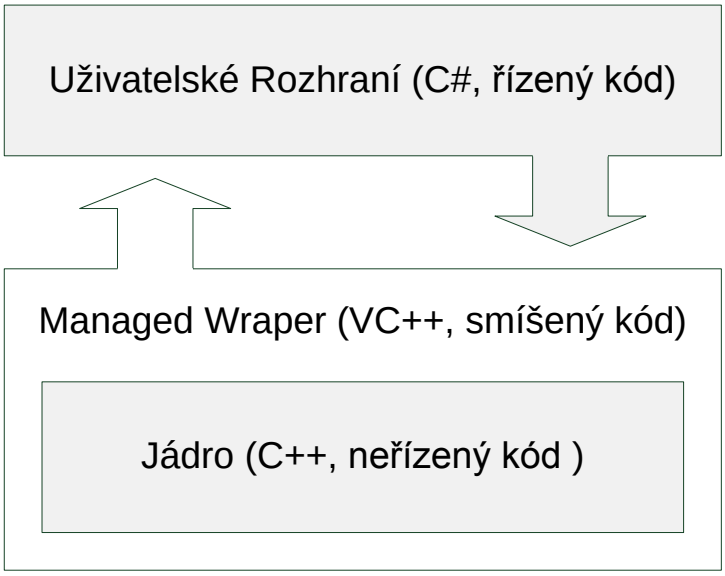
Jako vstupní formát virtuální scény pro náš systém jsme zvolili X File formát (důvody pro toto rozhodnutí jsou popsány v předchozí části). Nyní si tedy tento formát stručně popíšeme.

X File Format je formát založený na šablonách a je určen k ukládání obecných dat (byť je používán především pro ukládání grafických dat pro technologii DirectX). Struktura X souboru je následující. Na začátku každého souboru je Header, který obsahuje informace o použité verzi formátu, způsobu uložení dat (textový soubor, binární soubor) a některé další informace. Po něm následuje definice jednotlivých šablon, které určují, jaký způsobem jsou data v souboru uložena (formát sám již má definováno 27 standardních šablon pro uložení grafických dat). Pak již přicházejí na řadu samotná data strukturovaná dle jednotlivých šablon.

8 Software

Jak již bylo zmíněno v předchozích částech, každé virtuální studio je alespoň z části originál, a to i pokud jde o softwarovou část. Ve většině případů se při vytváření nového virtuálního studia nezačíná na zelené louce, ale systém se buduje pomocí souboru existujících částí (modulů), jejichž funkcionalita je společná pro všechna virtuální studia. Mezi takové moduly může patřit například modul pro trasování pozice kamery nebo modul pro práci s multimediálním obsahem.

Jako součást této diplomové práce bylo realizováno softwarové vybavení implementující základní funkcionalitu pro vytvoření virtuálního studia (v dalším textu budeme pro jednoduchost označovat tuto část softwaru jako „jádro“), s jehož pomocí lze, po patřičných úpravách, vytvořit systémy odpovídající specifickým požadavkům kladeným na konkrétní virtuální studio. Byla také vytvořena aplikace demonstrující správnou funkčnost a možnosti tohoto jádra.
V této části se budeme dále zabývat především popisem softwarového jádra systému, jeho struktuře, principům zásadních procesů, které v něm probíhají, a možnostem, které svým uživatelům nabízí.



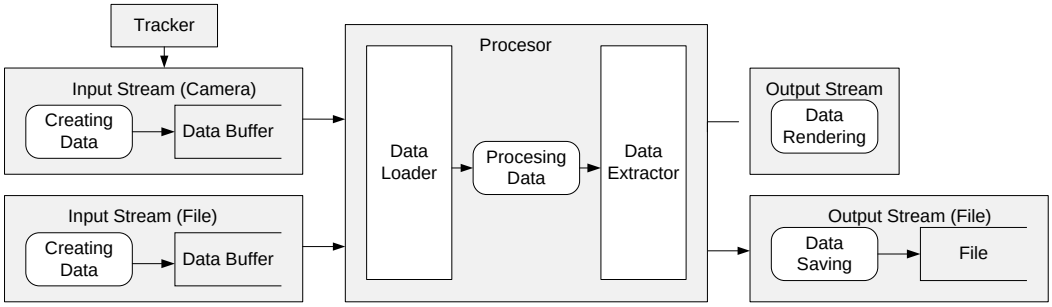
Obr. 8.1 Třívrstvá struktura našeho systém z pohledu použitých platforem

8.1 Struktura softwaru

V tomto oddíle se podrobněji podíváme na strukturu jádra systému a to z několika různých pohledů. Nejprve se podíváme na jednotlivé funkční celky, které tvoří samotné jádro.
V předcházejícím textu jsme vyjmenovali jednotlivé úkony, které musí zvládnout každé virtuální studio. Mezi tyto úkony patří trasování kamery, vykreslování virtuální scény a práce s multimediálním obsahem, do které spadá mimo jiné klíčování a skládání obrazu, načítání a ukládání multimediálních dat v různých formátech a další. Při návrhu jádra systému jsme se snažili se této struktury pokud možno co nejvíce držet. Jádro se tedy skládá ze čtyř funkčních částí (modulů), kterými jsou: modul pro trasování kamery (Tracker), modul pro klíčování obrazu, vykreslování virtuální scény a jejich skládání (Procesor), modul pro načítání, ukládání a zobrazování multimediálních dat (Streams) a konečně modul pro kalibraci kamery (CameraCalibrator). Na způsob, jakým mezi sebou jednotlivé části spolupracují, se zaměříme trochu detailněji, protože jeho pochopení je velmi důležité pro pochopení systému jako celku.
První část, na jejíž výstupy se zaměříme, je CameraCalibrator, protože kalibrace kamery či kamer je první krok, který je nutný po spuštění každého virtuálního studia učinit. CameraCalibrator poskytuje modulu Tracker informace o vnitřních parametrech kamery a umístění kamery na stativu (viz část 8.2). Druhá část jádra, do které spadají moduly Tracker, Streams a Procesor, je aktivní při celém běhu aplikace,

který následuje po kalibraci. Při tomto běhu je nutné průběžně v reálném čase zpracovávat vstupní data z kamer, určovat pozici kamer a s použitím těchto informací vykreslovat virtuální scénu a skládat výsledný obraz. Ten je pak nutné distribuovat na příslušná výstupní zařízení. Načítání a ukládání multimediálních dat zajišťuje modul Streams pomocí různých typů multimediálních proudů, které jsou v něm implementovány. Při načítání dat z kamery příslušný proud také zajišťuje, aby k nim byly připojeny odpovídající informace o parametrech kamery (vnitřních i vnějších viz 2.3), které poskytuje modul Tracker. Modul Procesor pak vybírá data z jednotlivých vstupních proudů, tyto data zpracovává a vytváří s jejich pomocí výsledný obraz. Výstupní data následně předává výstupním proudům, které jsou k němu připojeny. Celý tento proces je znázorněn na Obr. 8.2 ve formě data flow diagramu.

Na strukturu systému se podíváme ještě z pohledu použitých platforem. Celé jádro je, krom modulu pro kalibraci kamery, implementováno pomocí jazyka C++ a tedy v takzvaném neřízeném kódu. Neřízený kód je kód, který je překladačem přeložen přímo do strojového jazyka. Takto přeložený kód lze již přímo spustit na procesoru, pro který byl přeložen, a z čehož plyne jeho největší výhoda, kterou je jeho rychlost a zároveň jeho největší omezení, kterým je nepřenositelnost. Pro použití neřízeného kódu jsme se rozhodli ze dvou důvodů. Kromě již zmíněné rychlosti pro nás byla při rozhodování zásadní i skutečnost, že frameworky, které jsme pro realizaci jádra použili (zejména DirectShow viz [17]), jsou vytvořeny právě v tomto neřízeném kódu a jejich volání z neřízeného kódu je přímé a přirozené. Nicméně trendem posledních let je používání tzv. řízeného kódu. To je kód, který je překládán do tzv. mezijazyka, kterým může být například dobře známý bytecode pro javu nebo IL pro platformu .NET. Při spuštění programu je pak tento kód zpracován a řízen (proto řízený kód) pomocí tzv. virtuálního stroje, který jej následně přeloží do strojového kódu. Z předchozího vyplývá nevýhoda řízeného kódu, kterou je jeho relativně menší rychlost způsobená jeho překladem až v době běhu aplikace. Tento neduh je však kompenzován celou řadou výhod, které nám použití řízeného kódu nabízí. Mezi největší z nich patří automatická správa paměti, snadnější ladění a v neposlední řadě i rozsáhlá podpora ze strany největších softwarových výrobců jako jsou Sun či Microsoft. Především z těchto důvodů jsme pro náš systém vytvořili rozhraní, které umožňuje jeho použití v řízeném kódu. Struktura systému z pohledu použitých platforem je zobrazena na obrázku Obr. 8.1.



Obr. 8.2 Data flow diagram znázorňující typický běh programu.

8.2 Kalibrace

První část našeho virtuálního studia, kterou si popíšeme, je modul pro kalibraci. Úkolem tohoto modulu je poskytnout ostatním částem systému informace o vnitřních a vnějších parametrech kamery či kamer, které jsou použity pro snímání scény. Kalibraci kamer provádí třída Calibration, která k tomuto používá funkcí knihovny

OpenCv (viz [18]). V této knihovně jsou implementovány funkce pro kalibraci kamery metodou [1], která na svém vstupu potřebuje několik snímků planárního kalibračního vzoru (v našem případě se jedná o šachovnici, viz Obr. 2.4). Třída Calibration umožňuje uživateli uložit jednotlivé snímky z kamery, v nichž dokáže detekovat šachovnicový vzor. Samotná kalibrace probíhá pomocí všech uložených snímků a jejím výstupem je matice vnitřních parametrů kamery a koeficienty popisující její radiální zkreslení. Pokud jsou tyto parametry známy, umožňuje třída spočítat vnější parametry kamery. Rotace a translace, které jsou tímto způsobem získány, jsou vztaženy k souřadnému systému, definovanému polohou kalibračního vzoru. Pro získání korektních údajů je tedy třeba tento vzor vhodně umístit do snímané scény. Návod na správný postup při kalibraci kamery lze nalézt v příloze A této práce. Posledním parametrem, který je třeba při kalibraci kamery získat, je vzájemná poloha středu promítání kamery a vertikální osy kolem které se může otáčet hlava stativu na kterém je kamera umístěna. I tento údaj lze získat pomocí metod třídy Calibration. Ta k tomu používá epipolární geometrie (viz část 2.2) je tedy nutné uložit dva pohledy na statický kalibrační vzor. Mezi pořízením prvního a druhého snímku se musí hlava stativu (samozřejmě spolu s kamerou) otočit kolem příslušné osy. Žádný jiný pohyb kamery není v té chvíli přípustný.

8.3 Trasování

Trasování pozice kamery je funkce, která virtuálnímu studiu umožňuje vytvoření iluze virtuálního prostoru. V našem virtuálním studiu je implementována zjednodušená metoda optického trasování umožňující sledovat natočení kamery kolem jedné osy. Jak jsme již uvedli v předchozích částech, náš systém je realizován pomocí infračervené kamery připevněné ke stativu, na kterém je umístěna kamera snímající scénu. Tato infračervená kamera snímá trasovací vzory vytvořené z infračervených LED diod a pomocí epipolární geometrie určuje natočení stativu mezi jednotlivými snímky.

Trasovací systém je před jeho použitím nejprve třeba kalibrovat. Při tomto procesu je pomocí epipolární geometrie ze dvou snímků, které obsahují trasovací vzory s dostatečným počtem bodů (minimum je osm bodů korespondenčních bodů v obou snímcích), získána osa rotace trasovacího zařízení. Tato osa musí být pro všechny polohy trasovacího zařízení stejná, což lze zajistit jeho umístěním na vhodnou část stativu kamery. Ve chvíli kdy známe tuto osu, můžeme výstup z infračervené kamery upravit tak, aby odpovídal situaci, kdy se tato kamera otáčí kolem osy Y (viz popis souřadného systému kamery v části 2.1). V této chvíli se nám značně zjednodušuje model epipolární geometrie pro náš trasovací systém (viz část 2.2.2) a jsme schopni spočítat úhel natočení kamery mezi dvěma snímky přímo z esenciální matice vytvořené pomocí znalosti pouhých čtyř korespondenčních bodů. Nyní si popíšeme nejdůležitější třídy, které patří do toho modulu.

CTracker

Tato třída je mozkem celého procesu trasování. Jejím úkolem je vyhledání a připojení příslušné infračervené kamery k čemuž používá Optitrack API. Dále z kamery získává jednotlivé snímky (tyto snímky jsou v kameře předzpracovány a uživatel již dostává přímo souřadnice jednotlivých bodů detekovaných v obraze). Tato data posílá třídě CPointTracker, která provádí trasování jednotlivých bodů a vrací seznam korespondencí mezi jednotlivými snímky. Z těchto korespondencí je pak pomocí metod třídy CMatrixCalculations vytvořena esenciální matice a spočten úhel natočení mezi snímky. CTracker počítá tento úhel buď pro každé dva po sobě

jdoucí snímky, nebo dává uživateli možnost zvolit referenční snímek a úhel je pak vypočítáván mezi aktuálním a tímto referenčním snímkem.

CPointTracker
Je třída určená k trasování bodů nalezených v jednotlivých snímcích z infračervené kamery. Třída určuje korespondence vždy mezi poslední dvojicí snímků. Metod pro trasování bodů je celá řada, my v naší aplikaci používáme nejjednodušší z nich, která hledá korespondence na základě minimální vzdálenosti bodů mezi snímky. Použití této metody si můžeme dovolit ze tří důvodů, jednak díky dostatečné rychlosti použité kamery (její frekvence je sto dvacet snímků za vteřinu) a také díky znalosti charakteru pohybu, kterým se body detekované ve snímcích pohybují, a také rigiditě použitých trasovacích vzorů.

CMatrixCalculations
Poslední třídou o které se zmíníme je CMatrixCalculations. Tato třída obsahuje metody pro výpočet a manipulaci s esenciální maticí a to jak v jejím úplném tak zjednodušeném tvaru. Teoretický rozbor jednotlivých metod lze nalézt v části 2.2.

8.4 Proudy

Tento modul obsahuje několik tříd, které uživateli umožňují načítat, ukládat a zobrazovat multimediální data v různých formátech a z různých zdrojů. Všechny tyto třídy jsou postaveny na technologii DirectShow, což je framework pro proudové zpracování multimediálních dat od firmy Microsoft (pro více informací o této teologii viz část 8.4.1). V následujících částech nejprve popíšeme technologii DirectShow, po té se zaměříme na strukturu jednotlivých proudů a to jakým způsobem pracují s daty a jak spolupracují s ostatními moduly jádra.

8.4.1 DirectShow

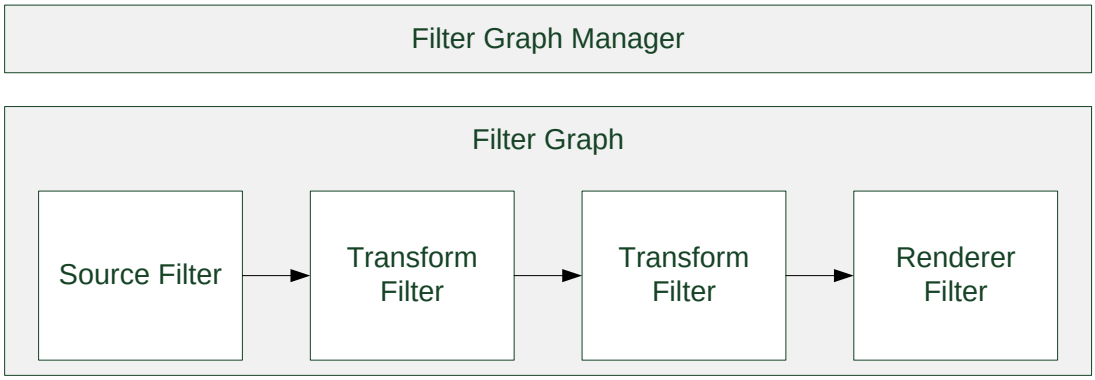
DirectShow je technologie pro proudové zpracování multimediálních dat, která umožňuje načítání zpracování a zobrazování či ukládání dat v různých formátech (např. MPEG, AVI, DV a v mnoha dalších). Tento framework je založen na COM, což principielně umožňuje jeho použití v mnoha programovacích jazycích, které COM rovněž podporují (pozn. společnost Microsoft oficiálně podporuje pouze rozhraní pro programovací jazyk C++).
DirectShow je navržen tak, aby každý dílčí úkol spojený se zpracování dat byl prováděn jedním COM objektem, který se v terminologii DirectShow nazývá filtr. Tato modulární struktura má několik výhod. Především umožňuje komplikované problémy spojené se zpracování multimediálních dat rozdělit do jednotlivých částí a každou z nich řešit samostatně v rámci realizace konkrétního filtru. DirectShow obsahuje sadu standardních filtrů, které poskytují základní funkčnost pro přehrávání a manipulaci s různými typy vstupních dat. Krom toho je možné díky jasně definovanému rozhraní vytvářet vlastní filtry, které tuto funkčnost rozšiřují. Díky této možnosti vzniklo množství nových filtrů, a to jak komerčních, tak nekomerčních. Díky nim technologie DirectShow pokrývá celou škálu problémů spojených se zpracováním multimediálních dat a dokáže velmi pružně reagovat na nové trendy, které se v této oblasti objevují. Nyní několik slov o tom, jak DirectShow funguje.
Jak jsme se zmínili v předcházejícím odstavci, Directshow je založen na systému filtrů. Tyto filtry lze podle jejich účelu rozdělit do několika kategorií:

- |Zdrojové filtry, jejichž úkolem je načítání dat. Tato data mohou pocházet z různých zdrojů, například ze souboru uloženého na disku počítače, z kamery, ze síťového multimediálního serveru a dalších.

- Transformační filtry, které nějakým způsobem mění formát či povahu vstupních dat a ty poskytují dalším filtrům.

- Prezentační filtry, jejichž úkolem je prezentace dat, která dostávají na vstupu.

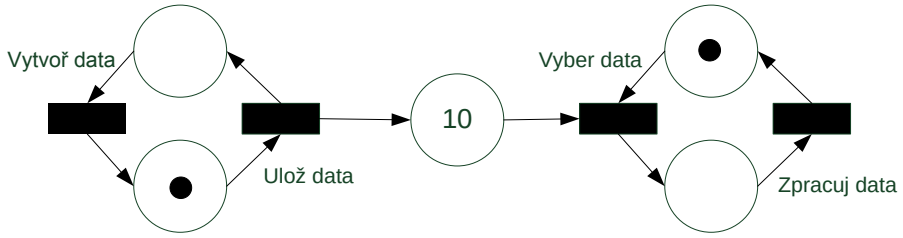
Aby spolu mohly jednotlivé filtry komunikovat, DirectShow je skládá do takzvaných grafů. Graf je COM objekt, který obsahuje několik vzájemně spojených filtrů (typicky v pořadí zdrojový filtr, transformační filtr či filtry a prezentační filtr). Každý graf je spravován takzvaným Filter Graph Managerem, který uživateli umožňuje mimo jiné kontrolovat běh jeho filtrů. Typická konfigurace grafu a Filter Graph Manageru je na Obr. 8.3.



Obr. 8.3 Typická konfigurace Filter Graph Manageru a grafu, který je pomocí něj spravován.

8.4.2 Directshow filtry

Pro potřeby naší aplikace jsme vytvořili dva speciální DirectShow filtry, jejichž hlavním úkolem je umožnit napojení jednotlivých proudů na ostatní moduly aplikace, jmenovitě na modul Procesor. V následujícím textu jsou popsány funkce těchto filtrů a některé jejich specifické vlastnosti.

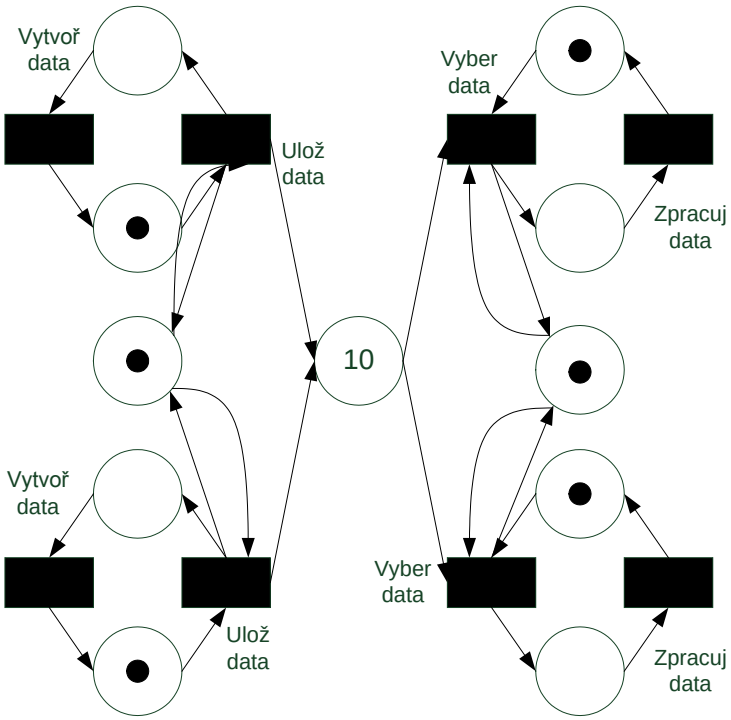


Obr. 8.4 Synchronizační model Producent-Konzument znázorněný pomocí Petriho sítě.

CSinkFilter

Tato třída implemetuje filtr, který umožňuje vstupním proudům (odvozeným od třídy CInputStream) napojení na modul Procesor. Filtr je v grafu vstupního proudu umístěn jako poslední (mohli bychom jej tedy zařadit do kategorie prezentačních filtrů). V rámci svého grafu se filtr chová jako buffer vstupních snímků, které jsou ve formátu IMediaSample, což je standardní formát všech dat, která si filtry mezi sebou

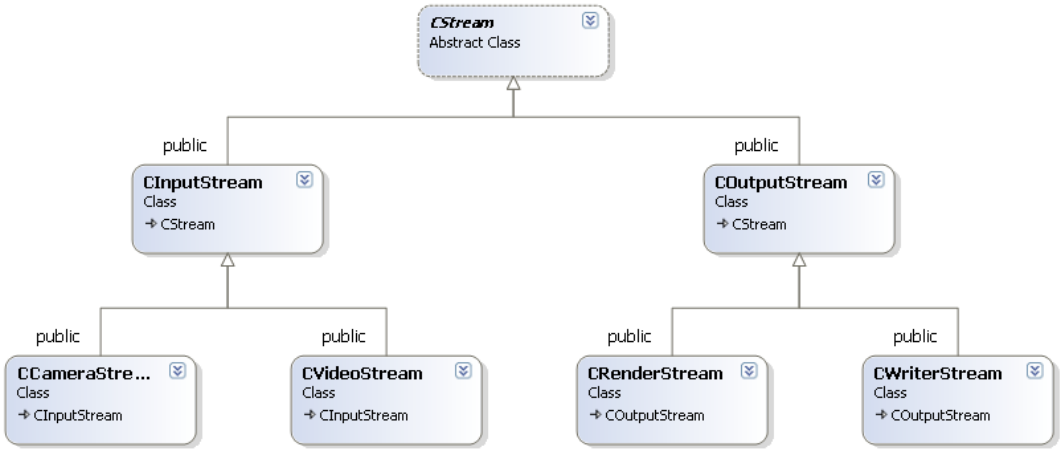
předávají. Velikost tohoto bufferu je omezená a nastavuje se jedinkrát při vytváření filtru.
Filtr má dva módy pro přístup k datům, která má uložená ve svém bufferu. První mód je synchronní, ten je reprezentován metodami PushSampleSync pro zápis snímku a PopSampleSync pro vyjmutí snímku z bufferu. Druhý mód je asynchronní a je reprezentován metodami PushSampleASync a PopSampleSync.
Při synchronním přístupu do bufferu se volající vlákno zablokuje ve chvíli, kdy není možné požadovanou operaci provést (tedy pokud chceme číst z prázdného bufferu či zapisovat do plného) a zůstává zablokované do doby, než se tento stav změní. Jedná se zde o klasický synchronizační problém typu producent-konzument, kde vlákno, které se snaží zapsat data do bufferu, vystupuje jako producent, a vlákno, které se snaží z bufferu číst, jako konzument. Tento model znázorněn pomocí Petriho sítě na Obr. 8.4. Buffer je synchronizován i pro případný přístup několika producentů a konzumentů zároveň (použijeme-li předchozí terminologii). V takovém případě lze model chování bufferu popsat Petriho sítí na Obr. 8.5.
Oproti tomu při asynchronním přístupu do bufferu se volající vlákno nezablokuje nikdy, a pokud požadovanou operaci nelze provést, vrátí se volání příslušné funkce s chybovým kódem.



Obr. 8.5 Petriho síť znázorňující ukládání a vybírání snímků z bufferu CSinkFilteru v situaci, kdy se toto ukládání a výběr pokouší více vláken současně.

CSourceFilter

Tato třída implementuje filtr, který umožňuje napojení výstupních proudů k ostatním částem systému. Oproti předchozímu filtru tento neprovádí žádné bufferování snímků a jeho struktura je velmi jednoduchá. Jeho jedinou funkcí je předávání snímků, které jsou do něj vkládány pomocí metody ConsumeSample, dalšímu filtru, na který je napojen. Jediným rozšířením je možnost zvolit, zda má filtr data předávat přímo nebo zda má předávat jejich kopie (což má své opodstatnění ve chvíli, kdy následující filtr data modifikuje, a my nechceme, aby se tato změna projevila v originálních vstupních datech).



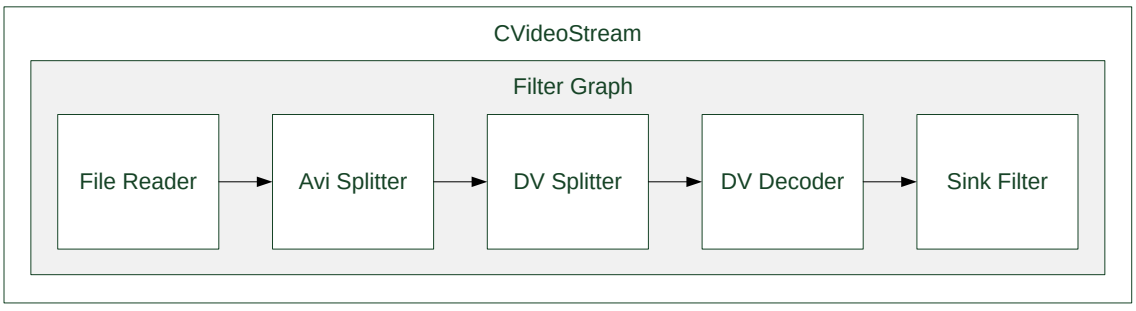
Obr. 8.6 UML diagram multimediálních proudů

8.4.3 Popis proudů

Všechny proudy, jak souhrnně nazýváme třídy našeho projektu, které se zabývají načítáním či ukládáním multimediálních dat, jsou postaveny na technologii DirectShow. Každý proud je vlastně obalovou třídou pro jeden graf filtrů, který vykonává činnost, kterou od proudu požadujeme (například načítání dat ze souboru). Stejná základní struktura všech proudů je zajištěna jejich společným předkem, jímž je třída CStream (viz Obr. 8.6). Nyní si podrobněji popíšeme jednotlivé třídy.

CStream

Třída CSream je abstraktní bázová třída pro všechny ostatní proudy. Mezi její potomky patří třídy CVideoStream, CCmeraStream, CrenderStream i CWriterStream (viz Obr. 8.6). Tato třída definuje strukturu obecného proudu jako obalové třídy pro graf filtrů. Od ní odvozené třídy pak specifikují jednotlivé filtry tohoto grafu v závislosti na funkci, kterou mají vykonávat. Dále jsou v ní definovány metody společné pro všechny ostatní proudy. Mezi tyto metody patří především metody pro řízení filtrů (Start, Stop, Pause) a pro změnu stavu proudu (Activate, Passivate). Pokud je proud v pasivním stavu, dává tím najevo ostatním částem systému, že nemá zájem s nimi jakýmkoliv způsobem komunikovat (tedy že jim nebude poskytovat, případně od nich přijímat, žádné snímky). Pokud je proud v aktivním stavu, signalizuje naopak svou připravenost ke spolupráci.
Způsob, jakým reagují grafy jednotlivých proudů na metody Start, Stop a Pause je rozepsán v následujících částech.



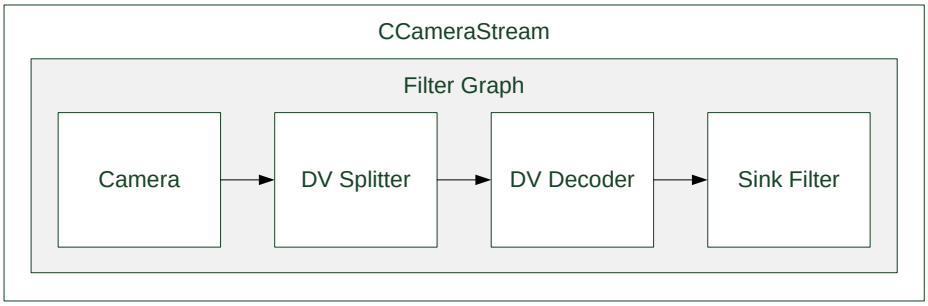
Obr. 8.7 Konfigurace filtrů v grafu třídy CVideoStream

CVideoStream
Tato třída implementuje proud pro načítání dat z libovolného vstupního souboru. Tento soubor může být v jakémkoliv formátu, který je podporován některým z DirectShow dekodérů, které jsou nainstalovány na počítači, kde je systém spuštěn. Cesta ke vstupnímu souboru musí být proudu předána při jeho vytváření jako parametr konstruktoru. Po té proud vytvoří příslušný graf filtrů. Ukázka, jak může takto vytvořený graf vypadat, pokud je na vstupu soubor ve formátu DV a je zabalen v kontejneru typu AVI je na Obr. 8.7. Na tomto obrázku vidíme, že graf se skládá ze standardních DirectShow filtrů s výjimkou posledního z nich, kterým je SinkFilter. Jak bylo zmíněno v předchozí části, tento filtr umožňuje napojení proudu na ostatní části systému pomocí svého rozhraní ISampleSource.
Kromě načítání dat ze vstupního souboru umožňuje tato třída uživateli nastavit několik atributů vstupního proudu. Dva nejdůležitější z nich jsou:

- Atribut, který určuje, zda je proud synchronní či asynchronní (ve třídě je reprezentován proměnou m_isSynchrounous)

- Atribut, který určuje, zda se jedná o takzvaný hlavní proud. Hlavní proud je proud, který obsahuje data z kamery, která v aktuální chvíli snímá scénu.

Poslední věcí, o které se zmíníme v souvislosti s touto třídou, je, jak se mění její chování v závislosti na stavu, ve kterém se nachází její graf filtrů. Volání metod Run a Stop vyvolá očekávanou reakci, tedy graf se buď spustí, nebo zastaví (a uvolní při tom všechny prostředky, které během svého běhu alokoval, včetně všech snímků) a proud tedy buď produkuje, nebo neprodukuje nové snímky. Zajímavější je stav, do kterého graf filtrů přejde volání metody Pause. V takovém případě SinkFilter při volání metod RequestSampleSync a RequestSampleASync neuvolňuje vrácený snímek ze svého bufferu. Obraz, který vychází ze streamu, se tedy zastaví, ale nedochází k žádné ztrátě snímků a po případném volání metody Run může načítání vstupního souboru bez problémů pokračovat.



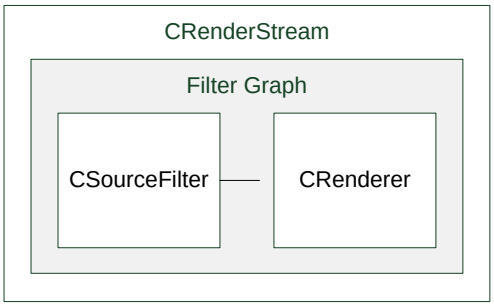
Obr. 8.8 Konfigurace filtrů v grafu třídy CCameraStream

CCameraStream
Tato třída implementuje proud pro načítání dat z DV kamery. V mnoha ohledech je její chování i struktura totožná s předcházející třídou CVideoStream, což je patrné i z Obr. 8.8, na kterém je zobrazen graf filtrů toho proudu, který se od grafu filtrů třídy CVideoStream liší jen velmi málo. Největším rozdílem v jejich grafech je první tzv. zdrojový filtr, který u CCameraStreamu načítá data z kamery zatím co u CVideoStreamu ze souboru. Dalším rozdílem mezi těmito třídami, který se ovšem již netýká jejich grafů, je fakt, že třída CCameraStream spolupracuje s modulem pro trasování kamery (Trackerem). Z tohoto modulu získává data o aktuální pozici

kamery, ke které je proud připojen, a ty přiřazuje do struktury jednotlivých snímků. Tato data jsou tedy následně přístupná i ostatním částem systému, které se snímkem manipulují.

CRenderStream

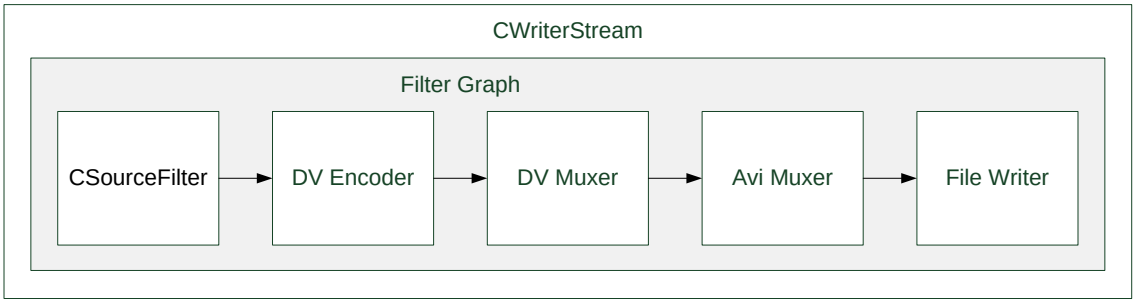
Tato třída implementuje výstupní proud, který vykresluje data, která jsou mu předkládána na vstupu. Graf filtrů tohoto proudu je velmi jednoduchý. Skládá se totiž pouze ze dvou spojených filtrů (viz Obr. 8.9). Funkci CSourceFilteru jsme si již popsali v jedné z předchozích částí. Druhým filtrem je CRender. Jedinou funkcí tohoto filtru je vykreslit obsah snímků, které dostává od CSourceFilteru. Toto vykreslování se provádí pomocí technologie DirectX 9.0.



Obr. 8.9 Konfigurace filtrů v grafu třídy CRenderStream

CWriterFilter

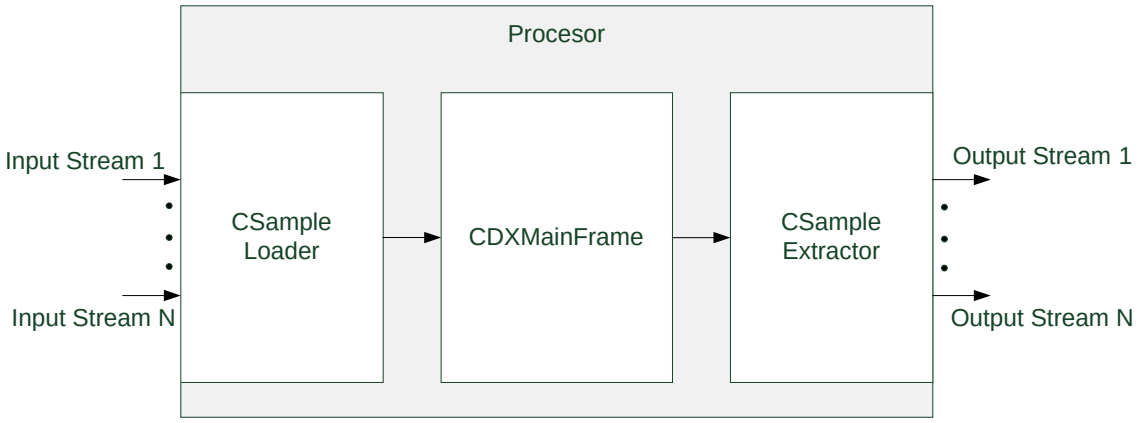
Poslední třídou, kterou si představíme, je třída CWriterFilter, která implementuje výstupní proud pro konverzi dat do DV formátu a jejich následné ukládání do souboru. Graf filtrů tohoto proudu má obdobnou strukturu jako předchozí proud a kromě již známého CSourceFilteru obsahuje jen standardní DirectShow filtry (viz Obr. 8.10).



Obr. 8.10 Konfigurace filtrů v grafu třídy CWriterStream

8.5 Procesor

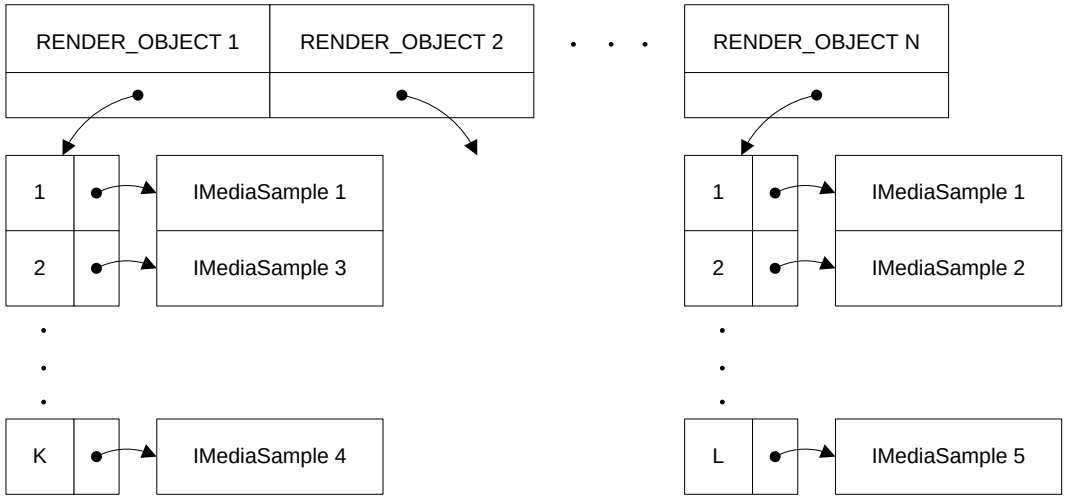
Modul Procesor je mozkem celého systému. Jeho úkolem je přijímat data ze všech vstupních proudů, ty zpracovat a výsledný obraz poslat do všech výstupních proudů, které jsou k němu připojeny. Na Obr. 8.11 jsou vidět tři základní části tohoto modulu. Vstupní bod Procesoru tvoří třída CSampleLoader, která je zodpovědná za připojování vstupních proudů k Procesoru a za získávání dat z těchto proudů. Výstupním bodem Procesoru je třída CSampleExtractor, která zodpovídá za připojování výstupních proudů k procesoru a rozesílá jim výstupní data, která produkuje třída CDXMainFrame. Všechny tyto třídy si nyní popíšeme podrobněji.



Obr. 8.11 Struktura modulu Procesor. (jako vstupní bod slouží třída CSampleLoader, jako výstupné bod třída CSampleExtractor, zpracování dat má za úkol třída CDXMainFrame)

CSampleLoader

Tato třída zajišťuje bezproblémové připojování a odpojování vstupních proudů k Procesoru. Její hlavní činností je však vybírání dat z připojených vstupních proudů. K tomuto účelu si třída při svém instalování vytváří pracovní vlákno, které cyklicky probíhá seznamem připojených proudů a vybírá z nich data. Při tom samozřejmě respektuje atributy těchto proudů udávající, zda je proud aktivní a lze z něj data vybírat a jakým způsobem (synchronně či asynchronně). Načtená data ukládá do speciální struktury, v níž jsou jednotlivé příchozí snímky roztříděny podle toho, na který objekt virtuální scény se mají navázat (kde ve scéně se mají vykreslit). Každému vstupnímu proudu tento objekt přiřadí uživatel. Tato struktura umožňuje, aby se na jeden objekt scény navázalo zároveň několik snímků (tyto jsou při další zpracování prolnuty pomocí třídy CDXComposer) a také aby se jeden snímek navázal na několik objektů scény (takový snímek se pak ve virtuální scéně zobrazí několikrát), viz Obr. 8.12. Po načtení snímků ze všech vstupních proudů jsou tyto poslány na zpracování do třídy CDXMainFrame.

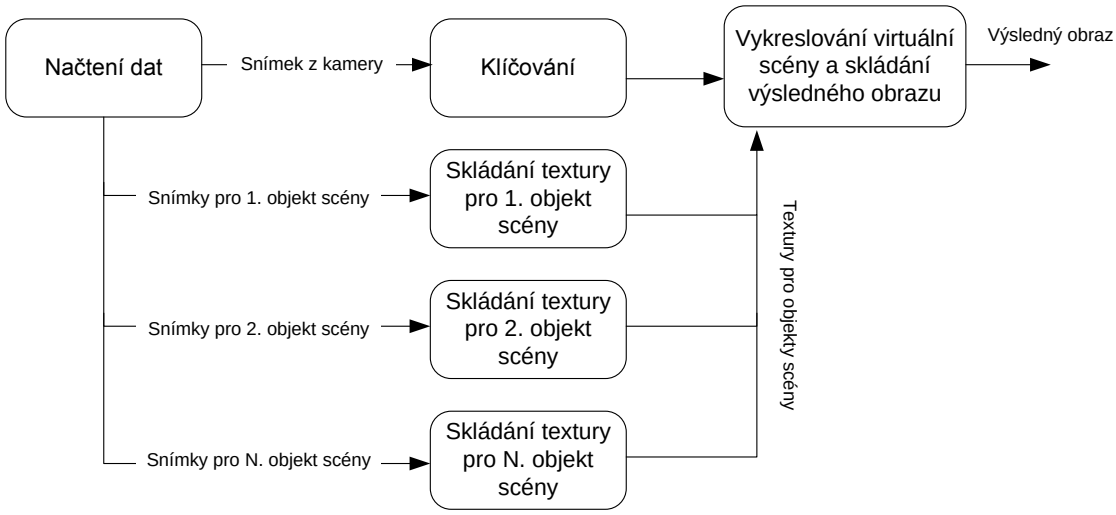


Obr. 8.12 Datová struktura pro předávání dat mezi třídami CSampleLoader a CDXMainFrame

CSampeExtractor
Tato třída spravuje výstupní proudy připojené k Procesoru. Zajišťuje jejich správné připojování a odpojování a ve chvíli kdy jsou k dispozici výstupní data z CDXMainFramu je všem těmto proudům postupně pošle.

CDXMainFramu
Tato třída a její podtřídy společně zajišťují veškerou funkcionalitu, která je potřebná pro hned tři zásadní součásti virtuálního studia a to pro klíčování, korektní vykreslování virtuální scény a skládání výsledného obrazu. Mezi její podtřídy patří CDXKeyer, určený pro klíčování vstupního obrazu, CDXComposer, určený pro skládání obrazů, a CDXProcessor, který se stará o načítání a vykreslování virtuální scény. Všechny tyto třídy jsou popsány v následujícím textu.
Samotná třída CDXMainFrame vystupuje pouze jako řídící prvek a stará se o správné předávání dat mezi svými podtřídami. Na jejím vstupu je struktura popsaná na Obr. 8.12, která obsahuje všechny vstupní snímky rozdělené podle objektu scény, na který jsou navázány a také snímek z tzv. hlavního proudu, který obsahuje data z připojené kamery. CDXMainFrame následně pošle tyto snímky CDXComposerům (pro každý objekt scény má vytvořen jeden CDXComposer), které z nich složí obraz, který se má ve scéně na příslušném objektu vykreslit. Snímek z hlavního proudu je poslán do třídy CDXKeyer, která provede klíčování. Výstup z CDXKeyer a všech CDXComposerů je následně poslán pro finální zpracování třídě CDXProcessor, která vykreslí virtuální scénu a její obraz složí se vstupem z kamery. Takto vzniklý obraz je následně distribuován dále do systému. Celý tento proces je znázorněn na data flow diagramu Obr. 8.13

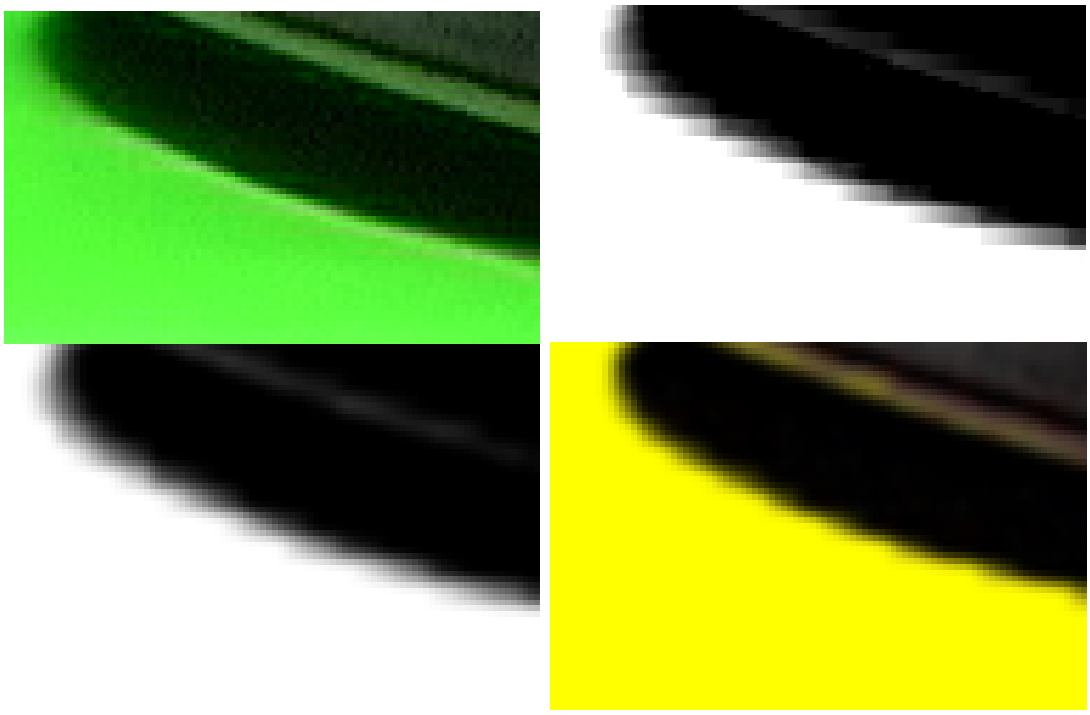
Data flow diagram zobrazující činnost třídy CDXMainFrame



Obr. 8.13 Data flow diagram zobrazující činnost třídy CDXMainFrame

CDXKeyer
Tato třída provádí klíčování obrazu pomocí technologie DirectX. Ke klíčování je použito rendrování do textury a technologie shaderů. Postup klíčování má dvě fáze. V první fázi je vstupní textura obsahující snímek z kamery zpracována pomocí shaderu a je vytvořena příslušná maska metodou Difference Matting (viz část 3.1), která je uložena v alfa kanálu výstupní textury. Zároveň je provedeno odstranění

radiálního zkreslení ze vstupního obrazu (radiální zkreslení je popsáno v části 2.3.) a výsledek je opět uložen do výstupní textury.
Pokud byl vstupní snímek před jeho dekódováním ve formátu DV (což je v našem případě velmi pravděpodobné), vzniklá maska je poměrně rozkostičkovaná (viz Obr. 8.14). Tento jev je způsoben chroma podvzrokováním, které v sobě DV formát zahrnuje (typicky narazíme na 4:1:1 nebo 4:2:0 podvzorkování). Proto následuje druhá fáze, ve které se snažíme tento nepříjemný artefakt zmírnit rozmazáním původní masky. Výsledná textura obsahující masku a upravený vstupní obraz je následně poslána k dalšímu zpracování.



Obr. 8.14 Jednotlivé fáze vytváření masky obrazu. Ze vstupního obrazu (vlevo nahoře) je vytvořena první maska (vpravo nahoře), která je značně rozkostičkovaná, na tuto masku je aplikováno rozmazání pro zmírnění tohoto artefaktu (vlevo dole). Výsledný kompozitní obraz vzniklý pomocí této masky je zobrazen vpravo dole

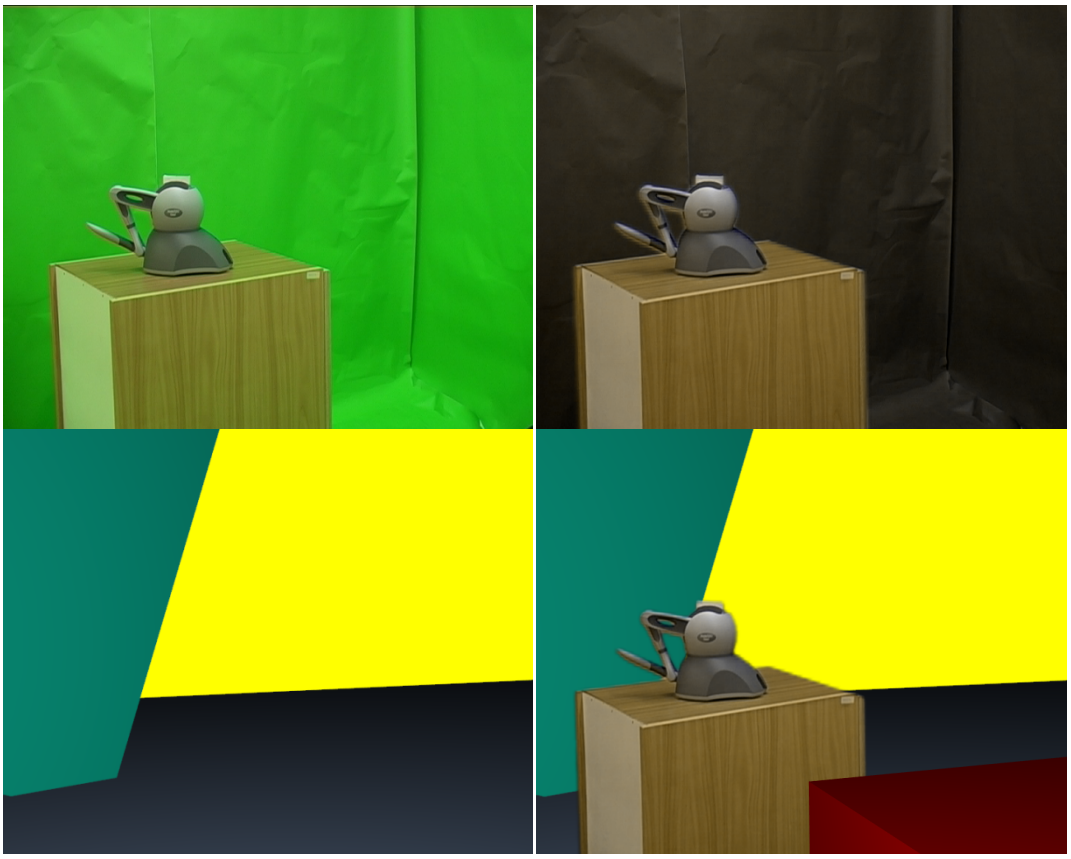
CDXComposer

Tato třída slouží ke skládání vstupních snímků do jediného výsledného obrazu. Každá instance této třídy má přiřazen objekt virtuální scény, na kterém se její výstup bude zobrazovat. Tento objekt mimo jiné určuje, jaký má být poměr stran textur, které se na něm mají zobrazovat. CDXComposer tedy adekvátně upravuje poměr stran všech vstupních snímků metodou LetterBox (k vstupnímu obrazu jsou přidány černé pruhy).

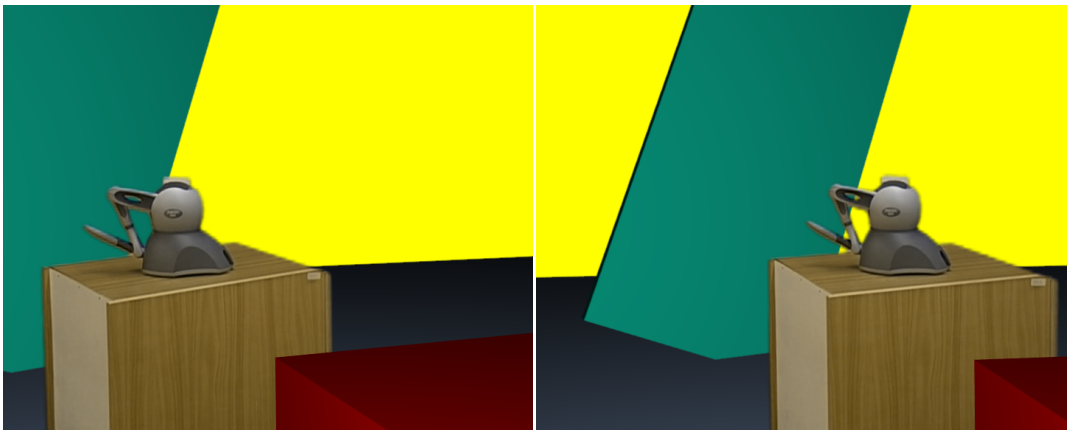
CDXProcesor

Tato třída a její podtřídy zajišťují všechny úkony spojené se správným vykreslováním virtuální scény v závislosti na pozici skutečné kamery. Samotné vykreslení má tři fáze.
V první fázi je třeba virtuální scénu načíst z uživatelem definovaného souboru (scéna musí být v X File Formátu viz část 7.2). Každá scéna se samozřejmě načítá pouze jednou. Tuto činnost zajišťuje třída CXDModel, která načtenou scénu uchovává v jedné ze svých struktur a umožňuje ostatním třídám přistupovat k jednotlivým

objektům scény, modifikovat jejich pozici a vykreslovat je. Při načítání scény je zárovně vytvořen seznam takzvaných RenderObjektů, což jsou objekty scény, na které lze vykreslovat data ze vstupních proudů. Každý z RenderObjektů ve scéně musí mít vlastní texturu a název této textury musí začínat předponou „render_“, podle toho jsou tyto objekty také při načítání virtuální scény identifikovány.
Druhou fází je správné nastavení virtuální kamery. Parametry skutečné kamery včetně její pozice a natočení si v sobě nese snímek z každého vstupního proudu typu CCameraStream. Tyto informace nezpracovává třída CDXProcesor přímo, ale předává je své podtřídě CDXCamera, která sama vytvoří matice view a projection, jež jsou potřebné pro vykreslování v prostředí DirectX. V této fázi jsou také aktualizovány transformační matice, které náleží jednotlivým objektům virtuální scény, na základě uživatelem definovaných změn pozic těchto objektů.
Poslední fází je samotné vykreslení virtuální scény. Protože všechny potřebné parametry jsou v této chvíli již nastaveny, je vykreslení velmi přímočarý proces. Od standardního vykreslení scény načtené z X souboru se liší pouze v podpoře pro vykreslování objektů do popředí (takové objekty se ve výsledném kompozitním obraze jeví vždy před obrazem z reálné kamery) a do pozadí (takové objekty se ve výsledném kompozitním obraze jeví vždy za obrazem z reálné kamery). Tento proces je znázorněn na Obr. 8.15 a Obr. 8.16. Samotný proces vykreslení vypadá následujícím způsobem. Nejprve jsou do textury vykresleny objekty v pozadí. Tato textura je následně s pomocí příslušné masky složena s texturou obsahující snímek z kamery a do takto vzniklého kompozitu jsou přikresleny objekty v popředí.



Obr. 8.15 Proces skládání obrazu. Vlevo nahoře je obraz z kamery, vpravo nahoře obraz po odstranění zelené aury z okrajů objektů (green spillu), vlevo dole obrázek virtuálního pozadí, vpravo dole je pak výsledný kompozit složený z předchozích dvou snímků, do kterého je navíc přidám jeden objekt v popředí (pravý dolní roh snímku).



Obr. 8.16 Výsledný kompozitní obraz pro dvě různé pozice kamery.

9 Klíčování stereoskopického obrazu

Stereoskopie je soubor technik umožňující navození iluze trojrozměrného vjemu pomocí dvourozměrných obrazů. Existuje mnoho postupů jak získat stereoskopický obraz, patří mezi ně i použití tzv. stereo nástavce (viz Obr. 9.1). My jsme se rozhodli prozkoumat problematiku klíčování stereoskopického obrazu, získaného právě pomocí takového stereo nástavce. Výsledky, kterých jsme dosáhli, jsou popsány v následující části.

Obr. 9.1 Kamera s připojeným stereo nástavcem

Snímek z kamery, opatřené stereo nástavcem, obsahuje obraz jak pro levé, tak i pravé oko (jeden v sudých a jeden v lichých řádcích snímku), viz Obr. 9.2. Tento formát uložení stereo snímků s sebou přináší několik problémů. Prvním z nich je

ztráta kvality způsobená polovičním vertikálním rozlišením snímků, která se samozřejmě musí projevit i v kvalitě masky, kterou se snažíme z obrazu získat. Na další ještě závažnější problém narazíme, pokud obraz z kamery rozseparujeme a zobrazíme si snímky pro pravé a levé oko zvlášť, viz Obr. 9.3. Na těchto snímcích jsou jasně patrné duchy způsobené částečným smíšením obou snímků. Tyto artefakty, které pravděpodobně vznikají při procesu chroma podvzorkování , činí klíčování tohoto stereo obrazu velmi obtížným. Limitujícím faktorem se ukázal i samotný stereo nástavec, jehož aplikací došlo ke ztrátě velkého množství světla dopadajícího na čip kamery a výsledný obraz byl velmi tmavý. Masky, které jsme byli schopni vyprodukovat pomocí našeho systému (viz Obr. 9.4), byly natolik nekvalitní, že nemělo smysl dále posuzovat kvalitu stereo vněmu, který bychom z kompozitních obrazů získali.

Z předchozího je zjevné, že klíčování obrazu získaného pomocí stereo nástavce je velmi obtížné a jeho výsledky nejsou ani zdaleka uspokojivé. Jeho kvalitu by bylo možné zlepšit dvěma způsoby. Jednoznačně by pomohlo použití kamery s menším stupněm chroma podvzorkování (například 4:2:2), která by do klíčovacího procesu přinesla větší množství informace a je pravděpodobné, že by se omezil vznik výše popsaných duchů v obraze. Zvýšená pozornost by měla být rovněž věnována správnému nasvícení scény, protože masky vzniklé z nedostatečně nasvícených obrazů jsou méně kvalitní. Prozkoumání klíčování stereo obrazu získaného pomocí dvou kamer ponecháváme jako jeden z námětů pro další práci.

Obr. 9.2 Obraz z kamery se stereo nástavcem (obraz pro jedno oko je obsažen v sudých a pro druhé v lichých řádcích).

Obr. 9.3 Obraz z kamery rozdělený pro pravé a levé oko

Obr. 9.4 Kompozitní obraz získaný pro pravé a levé oko.

10 Možná rozšíření

Lze nalézt velké množství námětů na rozšíření námi vytvořeného systému. Budoucí práce se může zaměřit například na některou z těchto oblastí.

- Implementace úplného optického trasovacího sytému včetně navržení vhodného hardwaru
- Prozkoumání možností připojení některého z exitující elektromechanických systémů pro trasování kamery do námi vytvořeného systému
- Rozšíření možností vytvořeného modulu pro rendrování virtuální scény. (dynamické stíny, průhlednost, …)
- Prokoumání možností pro zkvalitnění obrazu z amatérských a poloprofesionálních DV kamer a z nich generovaných masek

11 Závěr

V této práci jsme prozkoumali systém virtuálního studia. Popsali jsme, ze kterých částí se skládá a jak jsou tyto části realizovány v komerčních systémech. Uvedli jsme rovněž příklady požití virtuálních studií v praxi.

Dále jsme navrhli systém nízkorozpočtového virtuálního studia, pro který jsme vytvořili potřebný software. Tento software umožňuje kalibraci kamery a její následné trasování, zpracování multimediálních datových proudů, klíčování videa a rendrování virtuální scény v závislosti na pozici reálné kamery. Prozkoumali jsme rovněž nástroje vhodné pro definici virtuální scény a navrhli jsme formát pro její uložení, který je možné jednoduchým způsobem načíst do našeho systému. V neposlední řadě jsme definovali hardwarové nároky, které má nízkorozpočtové studio včetně úskalí, která jsou s výběrem vhodného hardwaru spojena.

Naše pozornost se zaměřila i na problematiku klíčování stereoskopického obrazu. Prozkoumali jsme úskalí spojená s touto technikou a navrhli pro ně možná řešení.

Hlavním přínosem této práce je, že poskytuje velmi dobrý teoretický i praktický základ pro tvorbu nízkorozpočtových virtuálních studií a pro další bádání v oblasti jejich návrhu. Jednotlivé části našeho systému jistě nejsou dotaženy k dokonalosti, ale to ani nebylo naším cílem, jsou však funkční a byli navrženy tak aby se dali snadno upravit či rozšířit.

Pro mě osobně byla práce na tomto projektu velmi přínosná. Rozšířil jsem si vědomosti v oblastech zpracování obrazu, realizace multimediálních systémů i v oblasti používané televizní techniky.

Literatura

[1] Zhang, Z. Flexible Camera Calibration By Viewing a Plane From Unknown Orientations. Redmond : Microsoft Research, 1998.

[2] Hartley, R. and Zimmerman, A. Multiple View Geometry in Computer Vision. Cambridge : Cambridge University Press, 2003. 0521540518.

[3] Hongdong, L. a Hartley, R. Five-Point Motion Estimation Made Easy. Washington : IEEE Computer Society, 2006. 1051465.

[4] Horn, B.K.P. Recovering Baseline and Orientation from Essential Matrix. 1990.

[5] Trucco, E. a Verri, A. Introductory Techniques for 3D Computer Vision. Upper Saddle River : Prentice Hall PTR, 1998. 0132611082.

[6] Remondino, F. a Fraser, C. Gigital Camera Calibration Methods. International Archives of Photogrammetry, Remote Sensisng vol. 36. 2006.

[7] Horn, B.K.P. Tsai's camera calibration method revisited. 2000.

[8] Marquardt, D. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. SIAM Journal on Applied Mathematics . 1963, Sv. 11.

[9] Gibbs, S., Arapis, C. a Breiteneder, C. Virtual Studios:An Overview. IEEE Multimedia. 1998, stránky 18 - 35 .

[10] Optitrack Motion Capture Solutions. NaturalPoint. [Online] NaturalPoint. Citace: 2. květen 2010.] http://www.naturalpoint.com/optitrack/.

[11] Feature Point Tracking Algorithms. [Online] 1998. [Citace: 2. květen 2010.] http://visual.ipan.sztaki.hu/psmweb/.

[12] Virtual Studio - Orad. Orad. [Online] Orad, 2007. [Citace: 2. kvěren 2010.] http://www.orad.co.il/en/page.asp?id=94.

[13] Teisler, M. a Bernas, M. A Determination Accuracy of Camera Position in a Virtual Set. Conference Proceedings of Radioelektronika 2004. 2004, 1., stránky 159-161.

[14] Viz Virtual Studio. Vizrt. [Online] Vizrt, 26. duben 2010. [Citace: 2. květen 2010.] http://www.vizrt.com/products/article128.ece.

[15] 3ds Max. [Online] Autodesk. [Citace: 2. květen 2010.] http://usa.autodesk.com/adsk/servlet/pc/index?siteID=123112&id=13567410.

[16] Blender.org. [Online] Blender Foundation. [Citace: 2. květen 2010.] www.blender.org/.

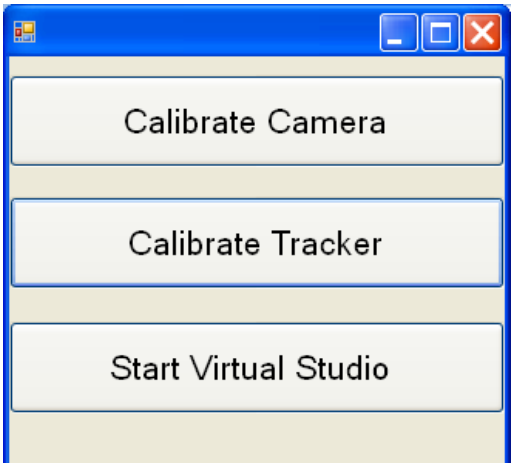
[17] DirectShow. [Online] Microsoft. [Citace: 2. květen 2010.] http://msdn.microsoft.com/en-us/library/dd375454(VS.85).aspx.

[18] OpenCV. [Online] [Citace: 2. květen 2010.] http://opencv.willowgarage.com/wiki/.

Příloha A: Uživatelská Příručka

Naše aplikace se skládá ze tří funkčních částí s vlastním uživatelským rozhraním. Jsou jimi část pro kalibraci kamery, část pro kalibraci trasovacího zařízení a samotné virtuální studio. V následujícím textu si popíšeme jejich ovládání a uvedeme doporučení, kterými by se měl uživatel při používání naší aplikace řídit.

Po spuštění programu se nám zobrazí rozcestník, pomocí něhož můžeme spouštět jednotlivé části aplikace, viz Obr A.1. Před spuštěním virtuálního studia je většinou nutné provést kalibraci kamery a trasovacího zařízení (pokud jsme potřebná kalibrační data neuložili při některém z předchozích spuštění aplikace). Po stisknutí tlačítka Calibrate Camera nebo Calibrate Tracker se objeví příslušné kalibrační rozhraní (viz následující části). Po dokončení procesu kalibrace stačí tato rozhraní zavřít, program předá získaná data k dalšímu zpracování ve virtuálním studiu automaticky.



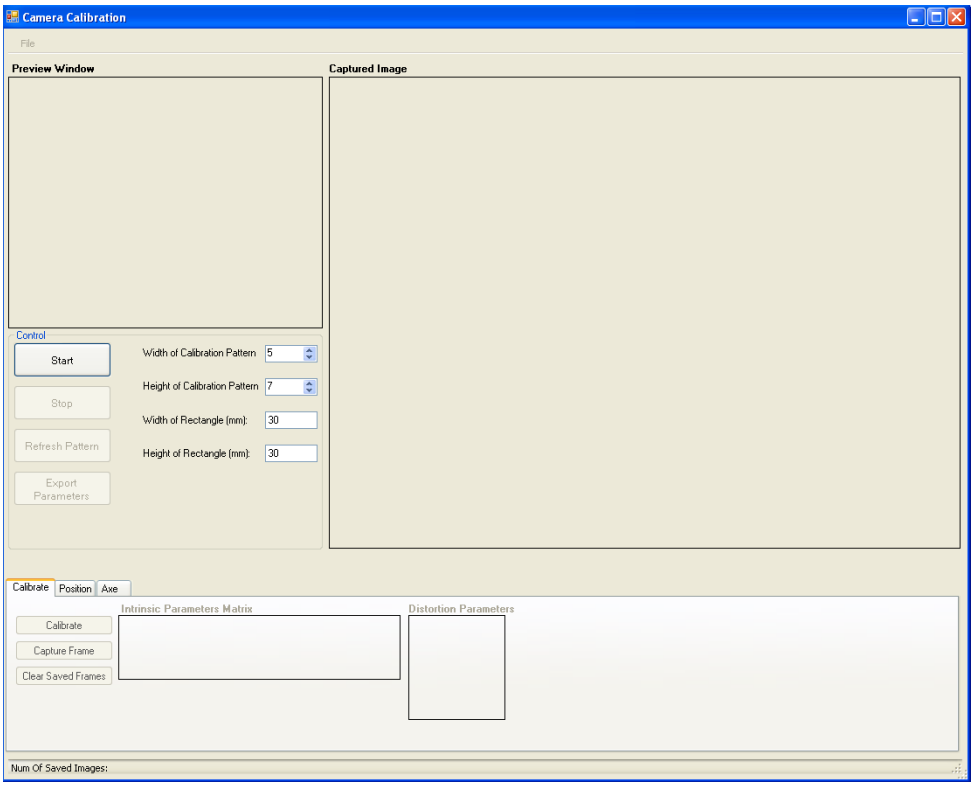
Obr A.1 Rozcestník aplikace

A.1 Kalibrace kamery

Cílem kalibrace kamery je získání parametrů kamery připojené k našemu systému. Tyto parametry jsme v naší aplikaci rozdělili do tří skupin: na vnitřní parametry (ohnisková vzdálenost, parametry radiálního zkreslení kamery), vnější parametry (pozice kamery v prostoru) a parametry určující umístění kamery na stativu (osa rotace kamery a vzdálenost ohniska kamery od této osy). Postup, kterým lze tyto parametry získat, je popsán v následujících částech.

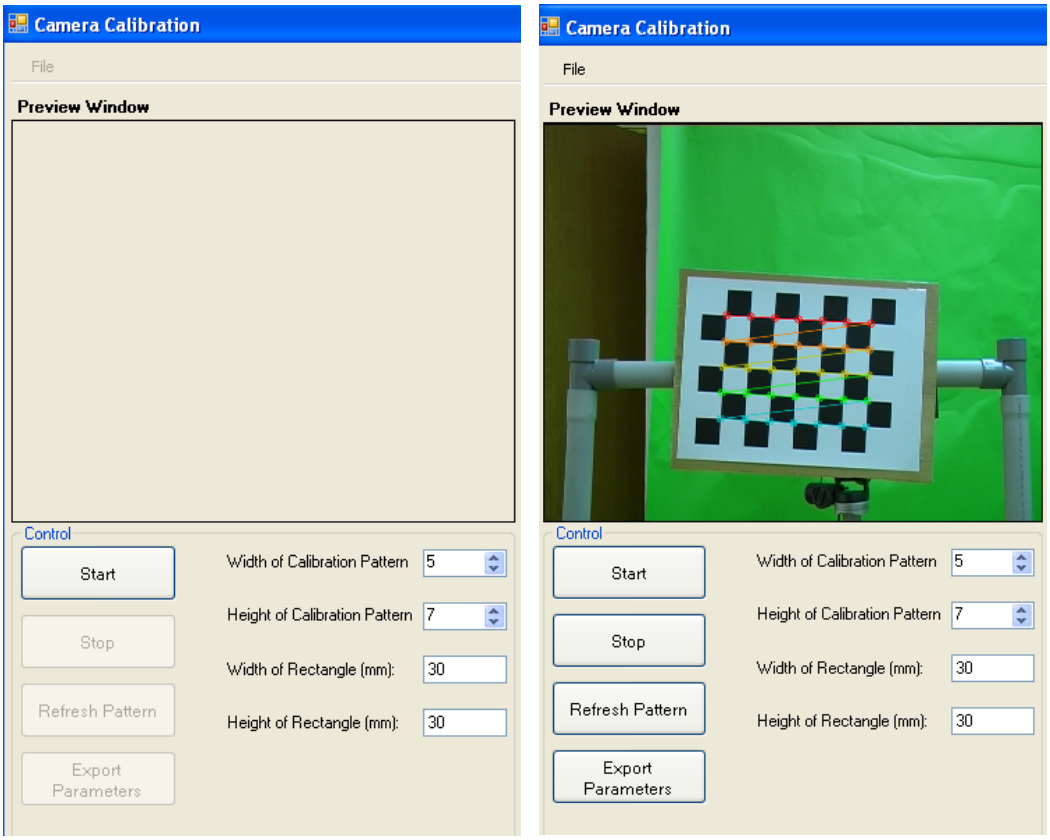
Před samotnou kalibrací musíme nastavit parametry dvourozměrného kalibračního vzoru (ten musí mít formu černobílé šachovnice, viz Obr. 2.4). To provedeme pomocí následujících ovládacích prvků (viz Obr. A.3):

Width of Calibration Pattern:	Počet polí kalibrační šachovnice ve směru x snížený o jedna
Height of Calibration Pattern:	Počet polí kalibrační šachovnice ve směru y snížený o jedna
Width Of rectangle:	Šířka jednoho pole šachovnice v milimetrech
Height of rectangle:	Výška jednoho pole šachovnice v milimetrech



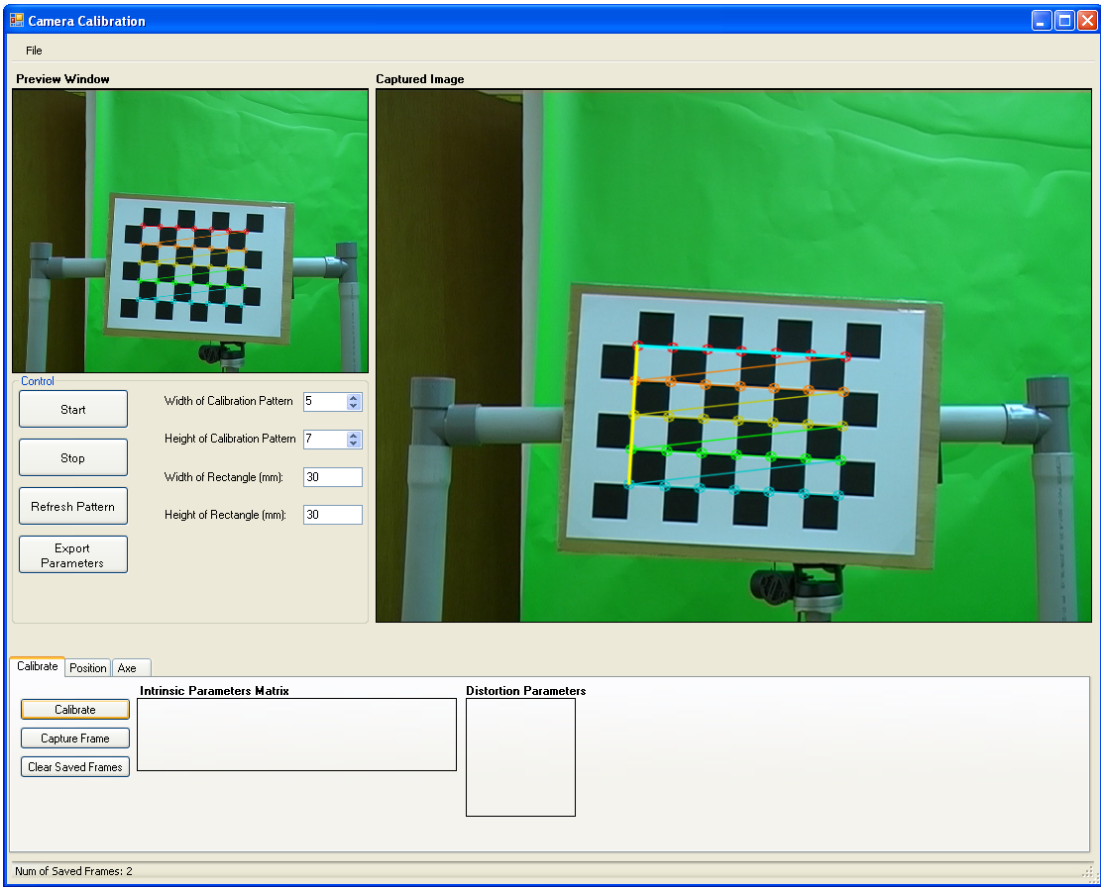
Obr A.2 Okno kalibrace kamery

Pokud bychom chtěli tyto parametry během kalibrace změnit lze to provést pomocí tlačítka Refresh Pattern.



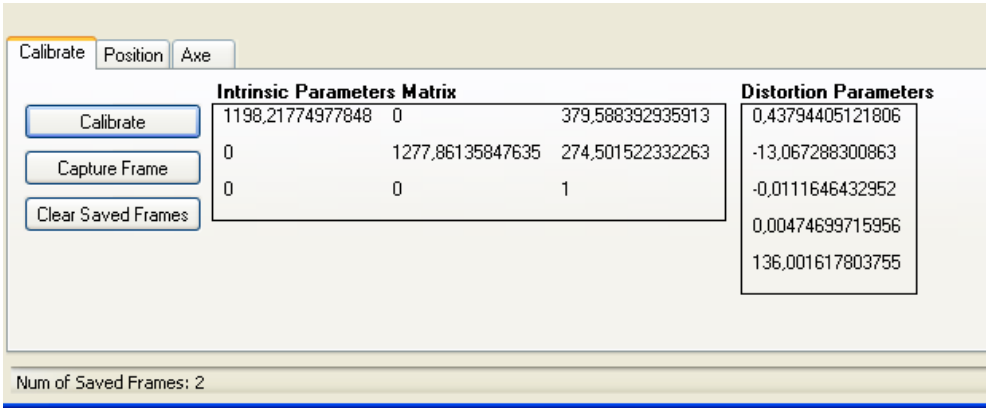
Obr A.3 Hlavní ovládací prvky kamery před (vlevo) a po (vpravo) připojení a spuštění kamery tlačítkem Start

Samotnou kalibraci spustíme tlačítkem „Start“. Následně se nám v okně „Preview Window“ začne zobrazovat obraz z připojené kamery, viz Obr. A.3. Pokud chceme zobrazování dat z kamery zastavit, můžeme použít tlačítko „Stop“. Nyní již můžeme začít se samotnou kalibrací kamery. Nejprve je nutné určit vnitřní parametry kamery. Pro správné určení vnitřních parametrů kamery potřebujeme uložit několik snímků s různě natočenou kalibrační šachovnicí (rotace šachovnice mezi jednotlivými snímky je nezbytná, pokud bychom šachovnici pouze posunuli nebo s ní nehýbali vůbec, nepřinesly by takové snímky do procesu kalibrace žádnou další informaci). Obecně se doporučuje použít pro kalibraci 30 a více snímků s různě natočenou šachovnicí v různých vzdálenostech od kamery. Šachovnice musí být samozřejmě na všech snímcích vidět celá a v poloze ve které je možné detekovat jednotlivé kalibrační body. Pokud je šachovnice detekována správně, objeví se na ní barevný vzor Obr. A.4.



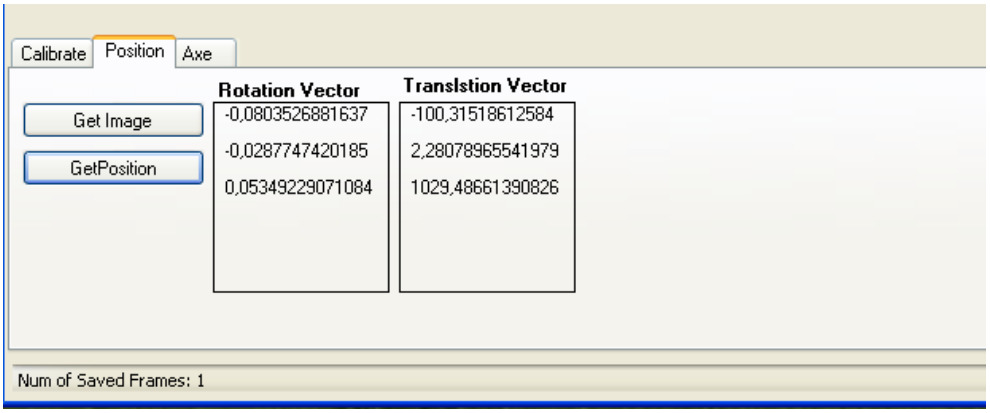
Obr A.4 uložený snímek s vydetekovaným kalibračním vrozrem se objeví v okně Captured Image.

K výpočtu vnitřních parametrů kamery slouží ovládací prvky v záložce „Calibrate“. Pomocí tlačítka „Capture Frame“ lze ukládat jednotlivé snímky z kamery. Ty se zobrazují v okně „Captured Image“ (viz Obr A.4) a jejich počet je zobrazen v liště ve spodní části okna. Pokud chceme uložené snímky odstranit lze použít tlačítko „Clear Saved Images“. Samotné získání vnitřních parametrů provedeme tlačítkem „Calibrate“. Získané parametry se zobrazí v polích „Intrinsic Parameters Matrix“ a „Distortion Parameters“, viz Obr A.5.



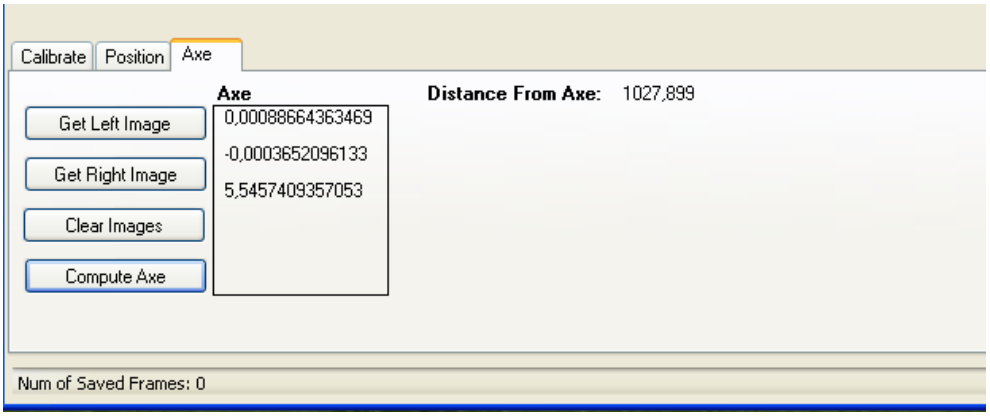
Obr A.5 Záložka pro kalibraci vnitřních parametrů kamery

Po získání vnitřních parametrů lze přistoupit k výpočtu vnějších parametrů. Vnější parametry udávají pozici kamery v prostoru. V našem případě je tento prostor definován pozicí kalibračního vzoru. Kladné poloosy X a Y jsou v obraze zobrazeny modrou respektive žlutou čárou, viz Obr A.4. Počátek souřadného systému je pak v průsečíku těchto čar. K výpočtu vnějších parametrů slouží záložka „Position“. Pokud máme kalibrační vzor v požadované pozici, stačí uložit jeho obraz tlačítkem „Get Image“ a spustit výpočet tlačítkem „Get Position“. Výsledky se zobrazí v poli „Rotation Vector“, které udává osu a rotace (velikost rotace je rovna velikosti tohoto vektoru), a „Translation Vector“, které udává pozici kamery.



Obr A.6 Záložka pro kalibraci vnějších parametrů kamery.

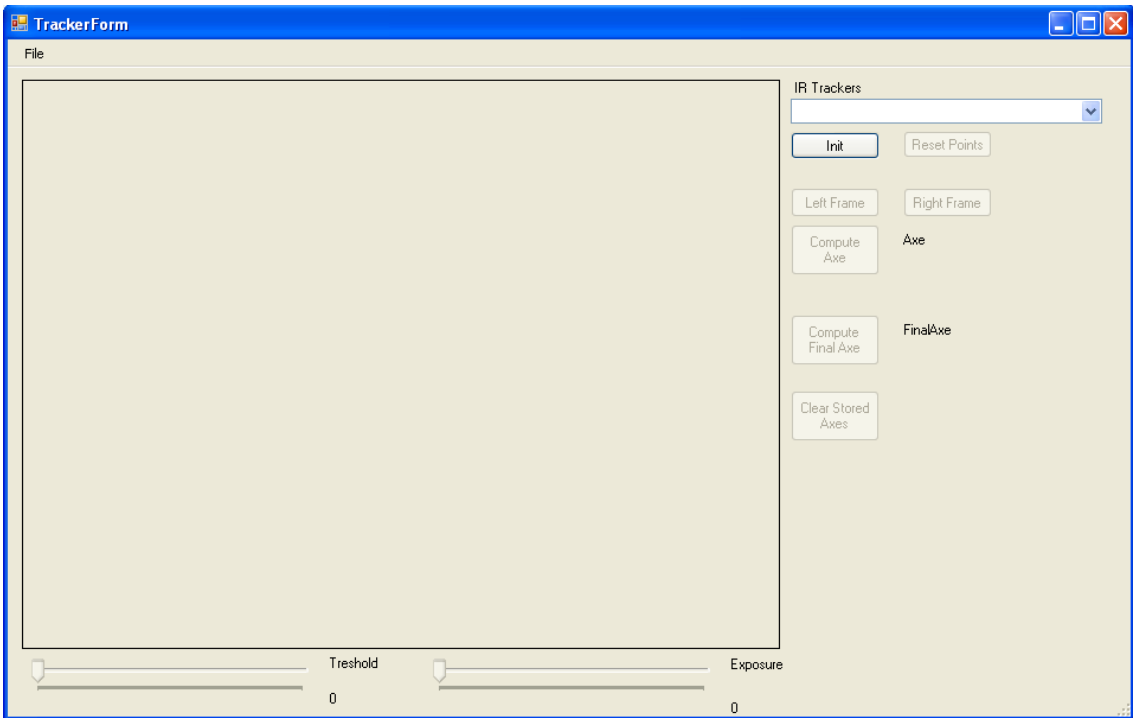
Poslední částí kalibrace kamery je určení polohy kamery na stativu. Protože náš systém umožňuje pohyb kamery pouze kolem vertikální osy, stačí nám určit vzdálenost ohniska kamery od této osy. K tomu slouží záložka „Axe“. Abychom mohli spočítat osu rotace, musíme uložit dva snímky kalibračního vzoru (pomocí tlačítek „Get Left Image“ a „Get Right Image“), mezi jejichž pořízením musíme kameru pootočit právě kolem vertikální osy. Po uložení snímků lze hledanou osu a její vzdálenost od ohniska kamery spočítat pomocí tlačítka „Compute Axe“.
Po dokončení kalibrace kamery můžeme získané parametry uložit do xml souboru a to buď pomocí menu File nebo tlačítka „Export Parameters“. Pokud parametry neuložíme, zůstanou nám přístupné pouze po dobu běhu programu.



Obr A.7 Záložka pro zjištění pozice kamery na stativu

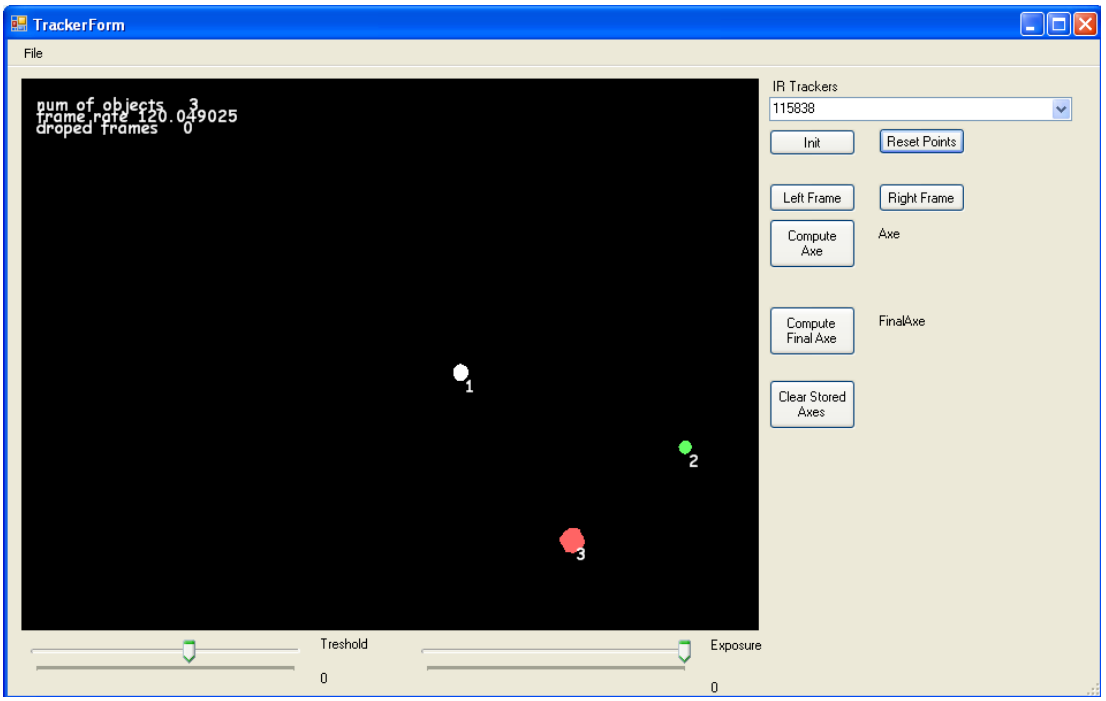
A.2 Kalibrace trackeru

Kalibrace trasovacího zařízení je druhým úkonem, který je třeba provést před spuštěním virtuálního studia. Trasovací zařízení je v našem systému realizováno pomocí infračervené kamery, připevněné ke stativu kamery, která snímá referenční infračervený vzor (realizovaný pomocí IR LED diod). Tato konfigurace trasovacího zařízení nám umožňuje určovat natočení kamery kolem jedné osy. Cílem kalibrace je pak určit vyosení infračervené kamery vzhledem k této ose. Po spuštění kalibrace se nám objeví následující okno.



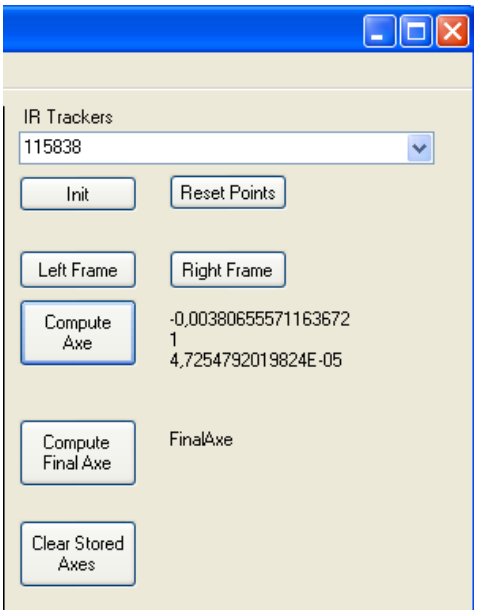
Obr A.8 Okno kalibrace trasovacího zařízení

Nejprve musíme vybrat kameru, kterou chceme kalibrovat. To provedeme pomocí kombo boxu „IR Trackers“, který je umístěn v pravém horním rohu kalibračního okna. Samotnou kalibraci spustíme tlačítkem „Init“. Po jeho stisknutí se nám začne v okně zobrazovat obraz z kamery.



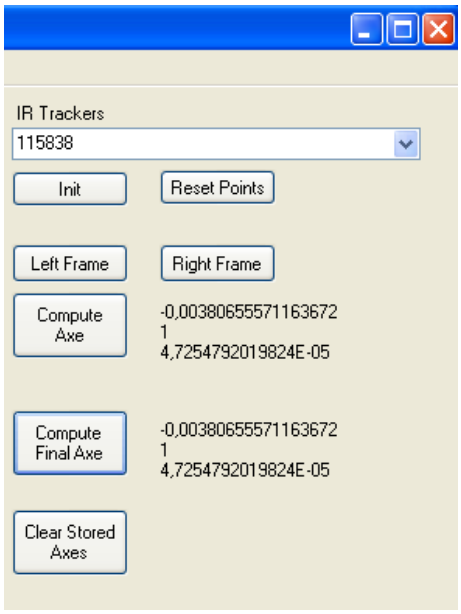
Obr A.9 Okno kalibrace po připojení infračervené kamery

Parametry tohoto obrazu můžeme měnit pomocí posuvníků „Threshold“ (pro nastavení velikosti prahu, který je použit při detekci objektů v obraze) a „Exposure“ (pro nastavení délky expozice kamery). Při kalibraci je třeba uložit dva snímky referenčního infra vzoru (kalibrační vzor musí mít minimálně 8 bodů), mezi jejichž pořízením je nutné pootočit stativ, na kterém je připevněna infra kamera, kolem odpovídající osy. Uložení snímků se provádí pomocí tlačítek „Left Frame“ a „Right Frame“. Výpočet osy rotace pak provedeme pomocí tlačítka „Compute Axe“. Výsledek výpočtu se nám zobrazí vedle tohoto tlačítka ve formě sloupcového vektoru (X,Y,Z).



Obr A.10 rozhraní pro kalibraci trasovacího zařízení s vypočtenou hodnotou osy rotace

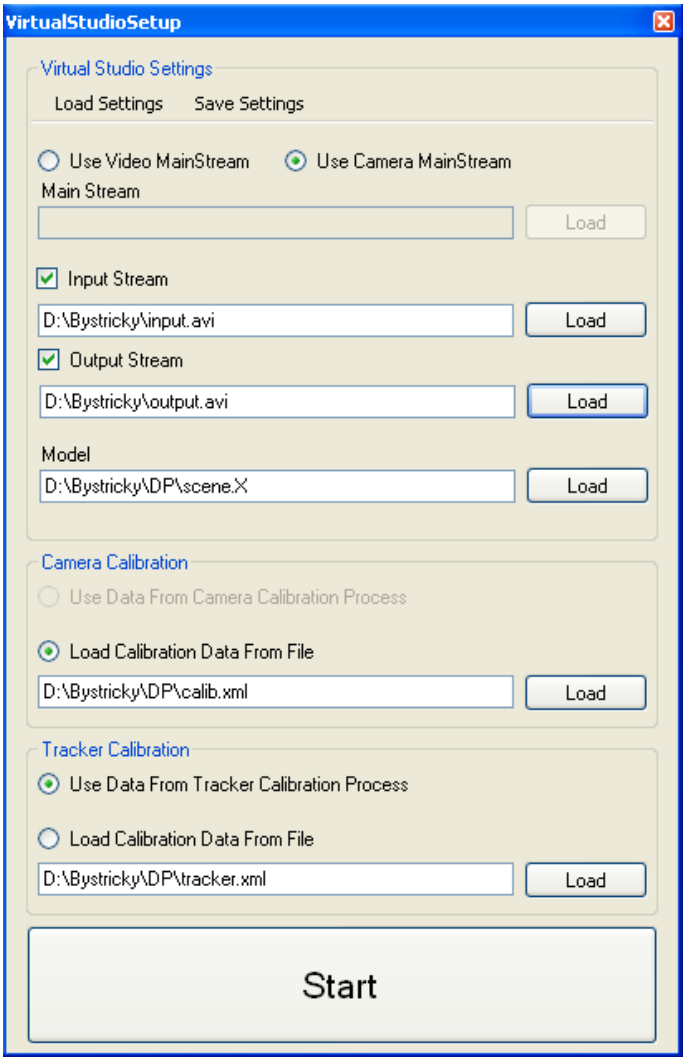
Protože obraz z kamery je zatížen určitou mírou šumu je vhodné tento postup několikrát opakovat (všechny vypočtené osy se ukládají) a výslednou osu určit jako průměr všech vypočtených hodnot. K výpočtu tohoto průměru slouží tlačítko „Compute Final Axe“. Pomocí tlačítka „Clear Stored Axe“ lze uložené osy vymazat.



Obr A.11 rozhraní pro kalibraci trasovacího zařízení s vypočtenou finální hodnotou osy rotace

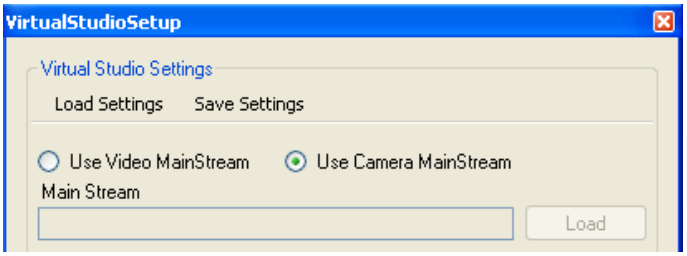
Získaná kalibrační data můžeme uložit pomocí menu File.

A.3 Virtuální studio



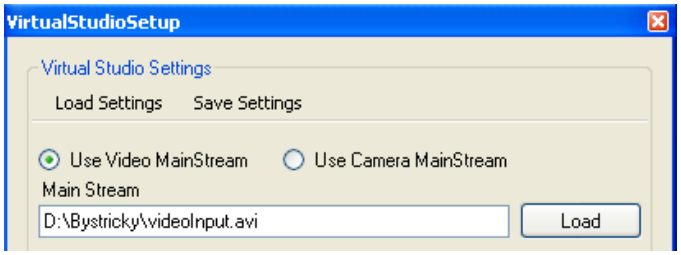
Obr A.12 Konfigurační dialog pro virtuální studio

Při spuštění virtuálního studia se nám nejprve zobrazí konfigurační dialog. Pomocí něj můžeme nastavit několik základních parametrů studia. V první řadě musíme určit, zda se má vstupní obraz získávat z kamery nebo ze souboru. To se provádí pomocí ovládacích prvků označených „Use Video Mainstream“ a „Use Camera Mainstream“.



Obr A.13 Prvky pro výběr zdroje obrazových dat (soubor, nebo kamera)

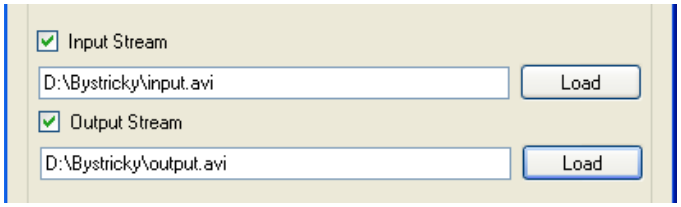
Pokud vybereme vstup ze souboru, musíme určit soubor, ze kterého se má obraz získávat.



Obr A.14 Gui s vybraným zdrojem dat ze souboru

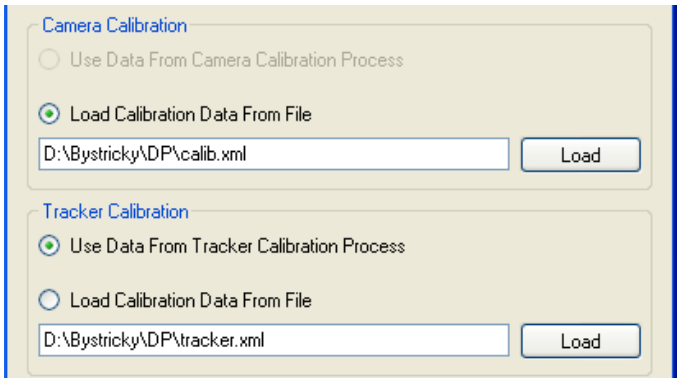
Ovládací prvky „Input Stream a „Output Stream“ slouží k výběru video souboru, jehož obsah se má zobrazovat na vybraných objektech scény (aplikace podporuje pouze video formáty, pro nějž jsou nainstalovány potřebné dekodéry plus skripty systému AviSynth), a souboru do kterého se má ukládat výstup z aplikace (ten je vždy ve formátu DV, který je zabalen v Avi kontejneru).

Obr A.15 Gui pro výběr zdroje dat pro objekty scény a výstupního souboru do kterého se má ukládat výstup z virtuálního studia (oba parametry jsou nepovinné)

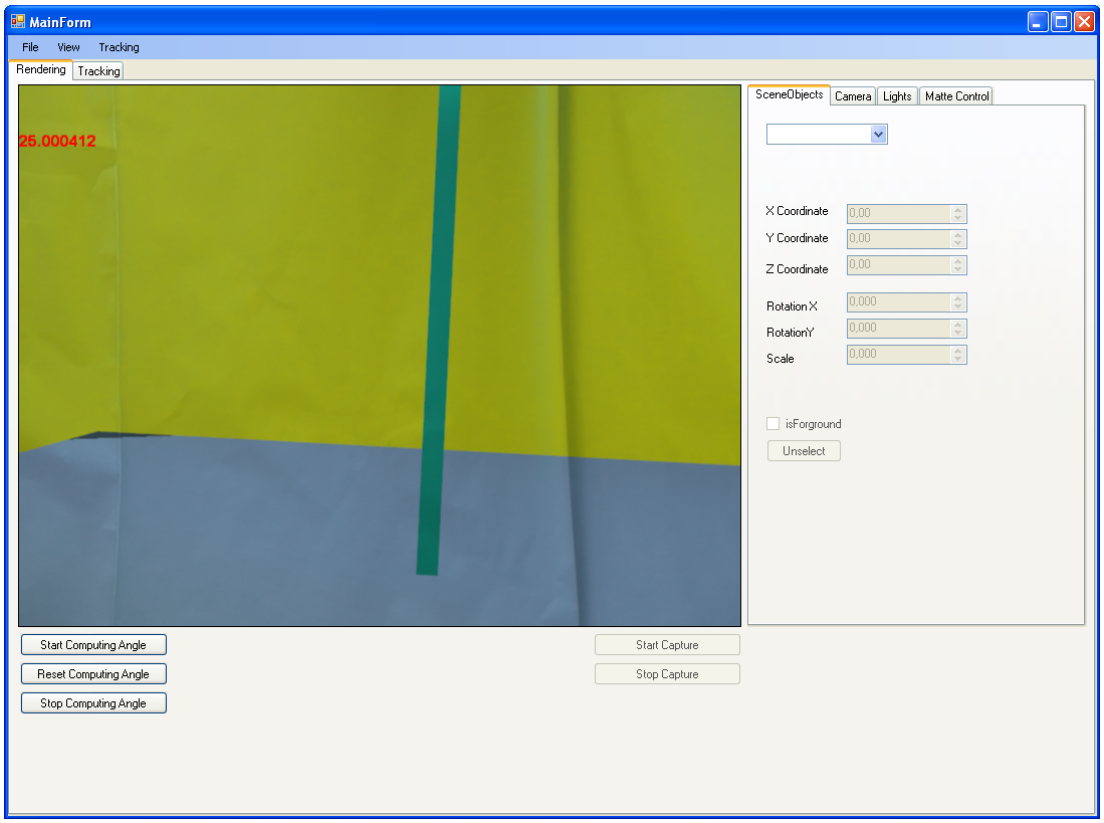


Obr A.16 Gui pro výběr zdroje dat pro objekty scény a výstupního souboru do kterého se má ukládat výstup z virtuálního studia s vyplněnými hodnotami (jejich vyplnění není povinné).

V poslední části inicializačního okna je nutné nastavit zdroj kalibračních dat pro kameru a trasovací systém. Pokud jsme provedli kalibraci zařízení před spuštěním virtuálního studia, systém nám nabídne možnost použít tato data („Use Data From Camera Calibration Process“, „Use Data From Tracker Calibration Process“). V opačném případě je naší jedinou možností načíst příslušná data ze souborů. Po nastavení všech parametrů můžeme konečně spustit virtuální studio pomocí tlačítka „Start“.

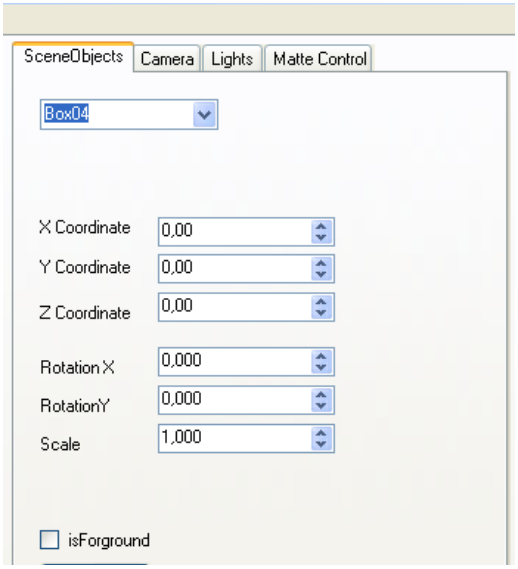


Obr A.17 Gui pro výběr zdroje kalibračních dat (povinné parametry)



Obr A.18 Okno virtuálního studia

V okně virtuálního studia se nám ihned po spuštění zobrazí výsledný obraz složený z obrazu z kamery (či souboru) a z obrazu virtuální scény. V této fázi ještě neprobíhá trasování kamery ani ukládání obrazu do výstupního souboru, je tedy vhodné nastavit požadované parametry scény. Toto nastavení lze provést pomocí jednotlivých záložek v pravé části okna.

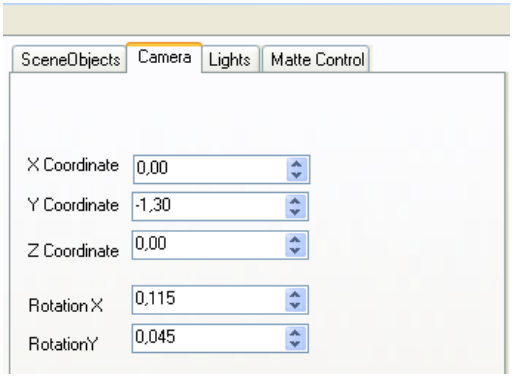


Obr A.19 Záložka pro manipulaci s objekty scény

V záložce „Scene Objects“ můžeme změnit polohu a velikost jednotlivých objektů virtuální scény. Nejprve musíme pomocí kombo boxu vybrat objekt, jehož vlastnosti chceme změnit (vybraný objekt se obarví načerveno). Změnu parametrů vybraného objektu pak provádíme pomocí těchto ovládacích prvků:

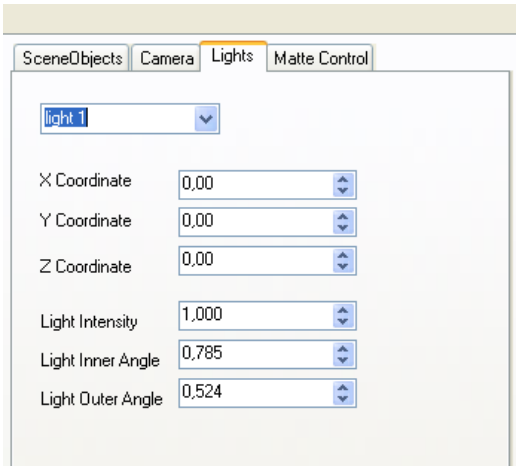
X Coordinate	Posun objektu ve směry osy X
Y Coordinate	Posun objektu ve směry osy Y
Z Coordinate	Posun objektu ve směry osy Z
Rotation X	Rotace objektu kolem osy X
Rotation Y	Rotace objektu kolem osy Y
Scale	Změna velikosti objektu
isForground	Určení zda má být objekt ve výsledném kompozitním obraze vždy v popředí

Změny všech parametrů se okamžitě projeví ve výsledném obraze.



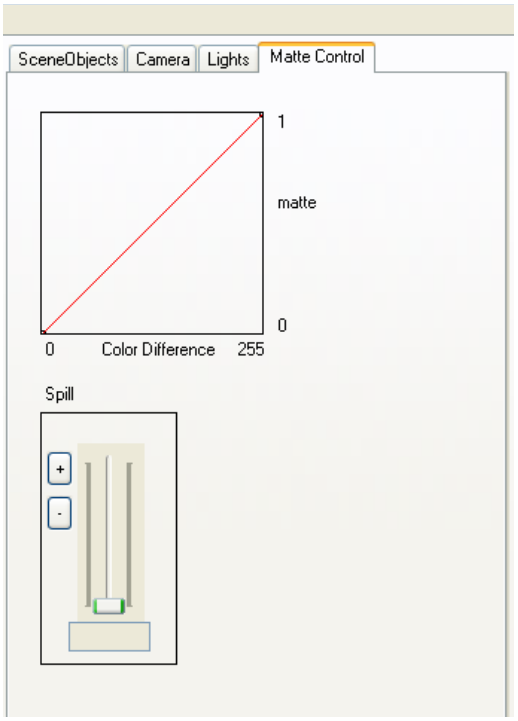
Obr A.20 Záložka pro manipulaci s virtuální kamerou

V další záložce označené „Camera“ můžeme nastavit počáteční pozici virtuální kamery. Funkce ovládacích prvků je shodná jako u předcházející záložky (posuv a rotace kamery).



Obr A.21 Záložka pro manipulaci se světli ve virtuální scéně

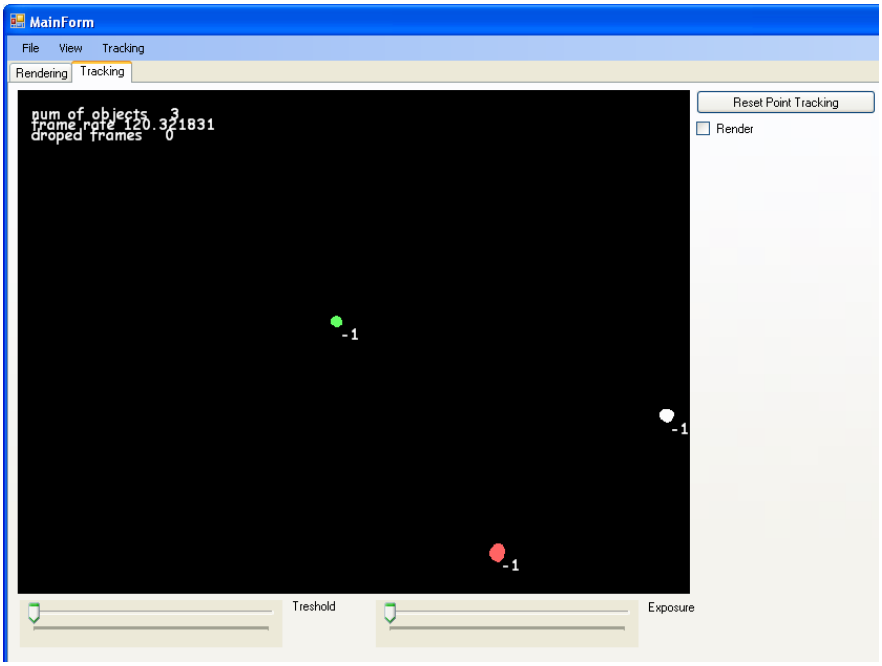
Ve třetí záložce máme možnost měnit parametry světel, která jsou ve virtuální scéně (pozn. ve scéně se mohou nacházet pouze tzv. reflektorová světla). Světlo, jehož parametry chceme změnit, opět vybereme pomocí příslušného kombo boxu. Následně máme možnost změnit pozici světla (X Coordinate, Y Coordinate, Z Coordinate) a také intensitu světla „Light Intensity“, a průměr vnitřního a vnějšího kužele reflektorového světla („Light Inner Angle“, „Light Outer Angle“). Velikosti těchto kuželů se zadávají v radiánech.



Obr A.22 Záložka pro nastavení parametrů klíčovacího proces

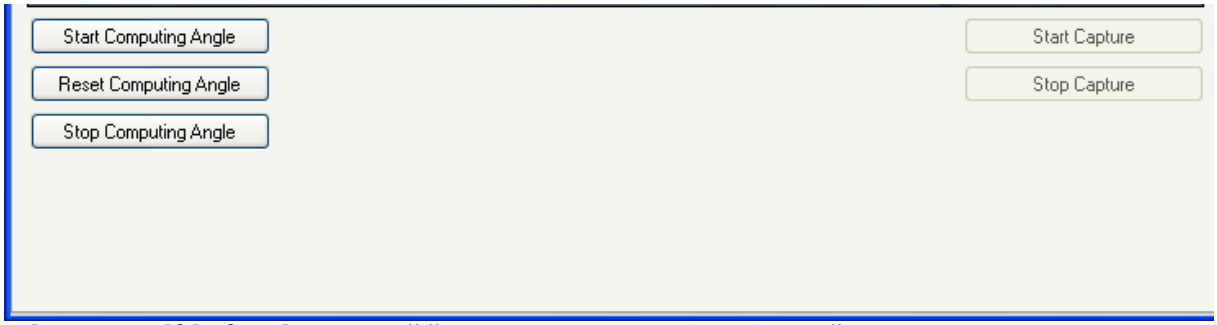
V poslední záložce můžeme nastavit parametry klíčovacího procesu. V horní části záložky můžeme nastavit křivku, podle které se má přepočítávat rozdíl mezi

barevnými kanály na hodnoty masky. Ve spodní části pak můžeme nastavit konstantu, pomocí níž se z obrazu odstraňuje zelená aura, která typicky vzniká kolem objektů popředí (tzv. green spill).
Po nastavení všech parametrů virtuální scény můžeme ještě zkontrolovat obraz infračervené kamery (pro trasování je třeba, aby obraz obsahoval přesně 6 referenčních bodů), který se průběžně zobrazuje v záložce „Tracking“. Zde také můžeme upravit hodnotu prahu, pomocí něhož se detekují jednotlivé objekty v obraze (ovladač „Threshold“), a délku expozice kamery (ovladač „Exposure“).



Obr A.23 Okno pro kontrolu výstupu z infračervené kamery

V této fázi je již vše připraveno ke spuštění trasování kamery. To se provádí pomocí tlačítka „Start Computing Angle“. Trasování můžeme rovněž pozastavit (tlačítko „Stop Computing Angle“) či můžeme trasovací proces resetovat (což může být vhodné, především pokud z nějakého důvodu vypadnou některé referenční body detekované v obraze kamery). Poslední možností, kterou nám virtuální studio nabízí je uložení výsledného obrazu do souboru (název a umístění výstupního souboru jsme zadávali v konfiguračním dialogu před spuštěním virtuálního studia). Ukládání spustíme pomocí tlačítka „Start Capture“. Pokud chceme ukládání ukončit, stačí zmáčknout tlačítko „Stop Capture“.



Obr A.24 Ovládací prvky pro spuštění trasování kamery (vlevo) a uložení výstupního obrazu do souboru (vpravo)

A.4 Překlad a spuštění aplikace

Přiložené cd obsahuje zdrojové soubory naší aplikace spolu se soubory, které jsou nutné pro otevření a překlad této aplikace v programu Visual Studio 2008. Před překladem aplikace se je nutné ujistit, že jsou na cílovém počítači nainstalovány následující komponenty:

Microsoft Directx SDK	(aplikace testována s verzí August 2009)
Optirack SDK	SDK pro komunikaci s hardware TrackIR 5 (aplikace testována s verzí 1.3)
EmguCV	Řízené rozhraní pro knihovnu OpenCV (aplikace testována s verzí 2.1)

Pro všechny tyto komponenty je nutné nastavit příslušné cesty a reference ve VisualStudiu. Po té již je možné aplikaci přeložit.

Na počítači, na kterém chceme naši aplikaci spustit, musí být nainstalovány:

Directx 9.0c
Optitrack SDK
.NET Framework 3.5

Příloha B: Zásady pro vytvoření virtuální scény

Při vytváření scény pro naše virtuální studio je potřeba dodržet několik jednoduchých zásad.

- Velikost objektů scény musí být zadána v metrech
- Ve scéně mohou být pouze reflektorová světla (spot light). Pro jejich správný export je třeba vytvořit dva objekty pojmenované „light+pořadové číslo světla“ (např. light1), a „target+pořadové číslo světla“ (např. target1). První určuje pozici světla a druhý směr, kterým světlo svítí. Světel může být maximálně pět a jsou číslována od 1 do 5. Další parametry světel se ze scény neexportují lze je však upravit přímo ve virtuálním studiu. Pozn. objekty light a target se ve virtuálním studiu nevykreslují, je proto jedno jakou formu jejich reprezentace zvolíme. Podstatné je pouze umístění (těžiště) těchto objektů.
- Objekty na jejichž povrch se má zobrazovat obraz z video souborů musí být otexturovány. A název textury, jež se bude nahrazovat obrazem z videa, musí mít předponu „render_“. Zvýšenou pozornost je dobré věnovat také rozměrům této textury, protože velikost a poměr stran obrazu, který se na objektu vykresluje, se jim přizpůsobuje.

- Protože s jednotlivými objekty scény je možné ve virtuálním studiu manipulovat pouze na základě jejích názvu je vhodné při jejich pojmenování dodržet alespoň určitou míru štábní kultury.