

Využitie oklúzie v rozšírenej realite

Viktor Seč^{1*}

Školiteľ: Zuzana Berger Haladová^{1†}

Katedra aplikovanej informatiky, FMFI UK, Mlynská Dolina 842 48 Bratislava

Abstrakt: Rozšírená realita je počítačovým rozšírením skutočného fyzického sveta o virtuálne objekty v reálnom čase. Táto práca pojednáva o jednotlivých používaných technikách a metódach na dosahovanie rozšírenej reality a o jej možných praktických využitiach.

Súčasťou riešenia práce je tvorba vlastnej aplikácie s rozšírenou realitou obohatenou o oklúziu s reálnymi objektmi.

Kľúčové slová: rozšírená realita, počítačové videnie, rozpoznávanie markerov, oklúzia

1 Úvod

Rozšírená realita, teda počítačom obohatený pohľad na skutočný svet nachádza čoraz väčšie uplatnenie v zábave, medicíne, armáde, reklame a mnohých ďalších priemysloch, pretože sa rozširujú hardvérové možnosti. To, na čo bolo kedysi treba drahé laboratórne vybavenie, ako výkonné počítače, profesionálne kamery a senzory, dnes dokáže takmer každý moderný mobilný telefón s kamerou. Rozšírená realita otvára nové možnosti interakcie medzi virtuálnym a fyzickým svetom, čo môže byť v budúcnosti veľmi zaujímavé.

V práci sú rozobrané jednotlivé možnosti aplikácií tejto technológie aj s konkrétnymi príkladmi. Ďalej sa pojednáva o niektorých metódach používaných na vytvorenie rozšírenej reality.

2 Prehľad problematiky

V tejto kapitole vysvetlíme, čo je rozšírená realita a ako bola definovaná. Popíšeme na čo všeobecne slúži a akými spôsobmi sa používa. Popíšeme kedy nastáva oklúzia a definujeme oklúder.

2.1 Definícia rozšírenej reality

Rozšírená realita (po anglicky *augmented reality*, skrátene *AR*) je počítačom rozšírený pohľad na re-

álny svet. Je to variácia virtuálnej reality, v ktorej používateľ nevníma svet okolo seba a je prenesený do sveta virtuálneho. Tento virtuálny svet je úplne umelý a nezávislý. Oproti tomu, pri rozšírenej realite používateľ nadálej vníma skutočný svet okolo seba doplnený o virtuálne objekty [5]. Ronald T. Azuma definuje rozšírenú realitu troma pravidlami [5].

1. Rozšírená realita musí kombinovať reálne a virtuálne objekty.
2. Aplikácia musí prebiehať v reálnom čase a nejakým spôsobom reagovať na zmeny v prostredí, teda byť interaktívna.
3. Rozšírená realita musí byť registrovaná v trojdimentzionaľnom priestore. To znamená, že musí korektne registrovať pohľad kamery s virtuálnym svetom a správne identifikovať, na ktoré pozície je potrebné vykresliť (po anglicky *render*) virtuálne objekty.

Cieľom rozšírenej reality môže byť v jednoduchom prípade prezentovať používateľovi nejaké informácie (napríklad informácie o určitých skutočných objektoch, ako je ich vzdialenosť, poloha, identifikácia a podobne), alebo vykreslovať neskutočné objekty tak, aby vyzerali ako skutočné a patriace do okolitého prostredia. V druhom prípade je potrebné, aby boli tieto objekty trojdimentzionaľne a vykreslovali sa správne v súlade s perspektívou a skutočnými objektami (napríklad prekrývali objekty za nimi, ale boli prekryté objektmi pred nimi) [4]. V prvom prípade je však podľa Azumovej definície stále potrebné, aby sa dané informácie vykreslovali vo výstupe na správne miesta, závisiace od vstupu kamery, alebo iných senzorov. Príklady oboch sú uvedené v nasledujúcej kapitole.

Rozšírená realita sa, rovnako ako virtuálna realita, nemusí nutne týkať iba vizuálneho obrazu. Teoreticky by mohla ovplyvňovať každý druh senzoričkého vnímania. Je to však obtiažná úloha, pretože na rozdiel od virtuálnej reality, v ktorej stačí tieto umelé zmyslové podnety len generovať, rozšírená realita musí upravovať skutočný svet. To znamená, že

*hello@viktorsec.com

†haladova@fmph.uniba.sk

okrem dopĺňania virtuálnych objektov občas potrebuje retušovať skutočné objekty, aby zanikli. Na to, aby sa to dalo dosiahnuť je potrebné vedieť zablokovať určitú časť pôvodného vnemu [7].

V prípade zraku je potrebné prekresliť skutočný objekt pozadím, ktoré sa nachádza za ním. Pri sluchu je filtrovanie jednotlivých zvukových stôp zo zmixovaného vstupu a navyše v reálnom čase (napríklad odfiltrovanie hlasu niektornej osoby v miestnosti) obviažejším problémom.

Rozšírená realita sa z praktických dôvodov obvykle zameriava na obraz. Týmto aspektom sa zoberá aj tátó práca.

2.2 Účel

Dôvodov pre vývoj rozšírenej reality je niekoľko. Pre používateľa môže byť rozšírená realita jedným z najjednoduchších spôsobov ako získať určité druhy informácií. Toto môže zefektívňovať a zjednodušovať jeho skutočnú prácu. Obzvlášť prakticky vyzerajú napríklad koncepty, pri ktorých má používateľ špeciálne okuliare, ktoré mu zobrazujú požadované dátá zo senzorov a databáz priamo na sklá a používateľovi ostávajú volné ruky na prácu.

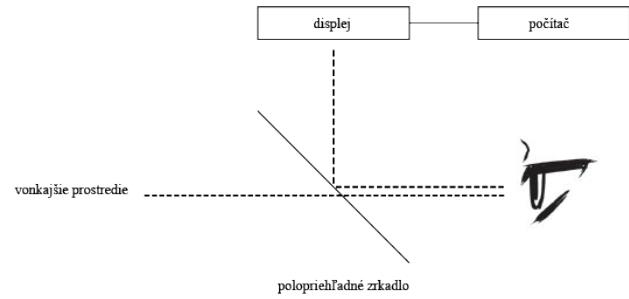
Dobrým príkladom je napríklad aplikácia firmy Boeing, vyvinutá pre mechanikov servisujúcich lietadlá. Keď mechanik odstráni niektorý z krycích panelov na lietadle, môže namieriť kameru tabletu na zväzky káblov a rozvodov, ktoré sa pod ním nachádzajú. Softvér v reálnom čase na obrazovku doplná údaje o tom, ktorý kábel, alebo trubica kam viedie a na čo slúži. Šetrí sa tak množstvo času oproti vyhľadávaniu v hrubých manuáloch.

Rozšírená realita má samozrejme taktiež široké spektrum využitia v zábave alebo marketingu.

2.3 Zariadenia

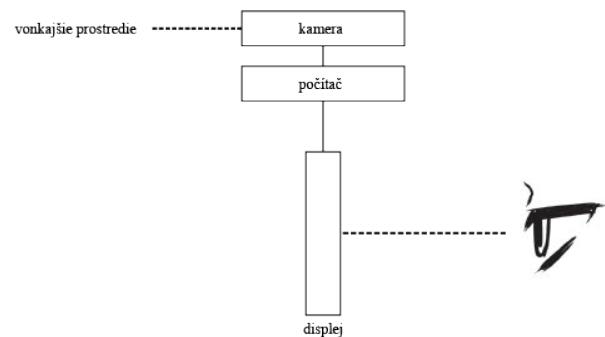
Rozšírená realita môže byť prezentovaná používateľovi buď priamo (napríklad vykreslovaním na priečinný displej, cez ktorý je priamo vidno okolité prostredie), alebo nepriamo, čiže vykreslovaním do záaznamu z videokamery, ktorý je vzápäť po rozšírení prezentovaný používateľovi na nepriehľadný displej.

V prípade priamej optickej rozšírenej reality za použitia priehľadného displeja sa obvykle používa nejaký typ okuliarov, alebo helmy. Tieto okuliare obsahujú šikmú poloreflexnú plochu, cez ktorú je priamo vidno, ale taktiež odráža obraz z malého displeja



Obr. 1: Schéma priamej rozšírenej reality

umiestneného nad ňou, alebo pod ňou [7]. Keď sa cez ne používateľ pozrie, obrazy sa mu skombinujú (znázornené na obrázku 1).



Obr. 2: Schéma sprostredkovanej rozšírenej reality

Pre videom sprostredkovanú rozšírenú realitu sa používajú ľubovoľné zariadenia s obrazovkami, ako sú telefóny, tablety, či počítače. Ani v tomto prípade však nie je vylúčené použitie špeciálnej helmy. Takéto helmy sa označujú ako HMD z anglického *head-mounted display*. Podľa vysvetleného použitia sa delia na *optical see through* (priame) a *video see through* (sprostredkovane). V demo aplikácií, ktorá bola vyvinutá ako súčasť tejto práce, ukazujeme sprostredkovanú rozšírenú realitu pomocou počítača, ku ktorému je pripojená kamera (znázornené na obrázku 2).

2.4 Definícia oklúzie

Oklúzia (po anglicky *occlusion culling*), je proces počas ktorého algoritmus rozhoduje, ktoré objekty alebo prípadne časti objektov sú na scéne viditeľné. Pokial' je nejaky objekt sčasti alebo kompletne skrytý za iným objektom, znamená to, že je okludovaný a teda sčasti alebo vôbec nie je viditeľný. Objekt, ktorý ho zakrýva sa nazýva oklúderom (po anglicky *occluder*).

Oklúzia sa obvykle používa ako optimalizačná metóda (nie je potrebné vykreslovať pixely, ktoré budú napokon prekryté), v rozšírenej realite sa dá využiť tak, aby ilúzia pôsobila reálnejšie.

3 Aplikácie

Rozšírená realita má využitia v medicíne, zábave, športe, inžnierstve, dizajne, vo výučbe a iných oblastiach. V tomto článku uvádzame niekoľko konkrétnych príkladov.

3.1 Záchranári

Tím na Technickej univerzite vo Viedni vyvíja aplikáciu pre záchranárov a požiarnikov. Užívateľovi ukazujú v okuliaroch obraz z termálnej kamery registrovaný s 3D modelom štruktúry budovy, ktorý je rekonštruovaný v reálnom čase (obrázok 3). Vďaka tomu vie ako vyzerá jeho okolie aj v prípade hustého dymu [26].



Obr. 3: Projekt má požiarnikom zabezpečiť životné dôležité informácie v podmienkach slabej viditeľnosti [26]

3.2 Preklady

Na University of California vytvorili mobilnú aplikáciu, ktorá cez rozšírenú realitu prekladá text. Používateľ jednoducho namieri telefón na nápis, ktorý chce preložiť, a aplikácia text v obraze z kamery zdeteguje pomocou optického rozoznávania znakov. Softvér vzápäť text preloží a vykreslí na obrazovku telefónu priamo do videa kamery. Pri tomto vložení dbá nielen na to, aby preložený text umiestnil na správne miesto a v správnej farbe a veľkosti, ale aby aj vyretušoval pôvodný text, ktorý by bol umiestnený pod tým preloženým [11]. Ukážka aplikácie je na obrázku 4.

3.3 Armádne aplikácie

Vojenské operácie sa často vykonávajú v mestskom prostredí. Bojové zóny, v ktorých sa nachá-



Obr. 4: Ukážka aplikácie TranslatAR [11]

dzajú poschodové budovy sú veľmi komplikované a pre úspech misie sú pre vojaka mimoriadne dôležité informácie o okolí. Ked' voják pozera do mapy, ohrozuje sa, pretože dáva nižší pozor na svoje okolie.

Vedci z Naval Research Laboratory v USA tvárajú helmu, ktorá bude vojakom sprostredkovávať najdôležitejšie informácie. Používateľ môže vidieť nad budovami napísané ich mená a plány interiérov, na zemi zas môže vidieť napísané názvy ulíc. Taktiež sa mu môžu zobrazovať ikony na presných lokáciach, kde boli nahlásení ostreľovači [22, 35].

Inou aplikáciou rozšírenej reality na armádne účely je systém rozširujúci videnie pilota lietadla. Úlohy ako zameriavanie ciela, dodávky zbraní a zásob na padákoch, či obyčajný let v nízkej výške vyžadujú, aby pilot presne rozoznával terén pod sebou. Senzory na stíhačke môžu sledovať oblasť, ktorú pilotovi v zornom poli zakrýva samotné lietadlo, alebo mu poskytovať dátá za podmienok slabej viditeľnosti. Všetky tieto dátá sa potom môžu premieť do pilotovej helmy, umožniť mu vidieť to, čo by inak nevidel a zvýrazniť dôležité body [21].

Prvé podobné primitívne systémy vznikli už pred Druhou svetovou vojnou. Spojeneckí piloti používali v niektorých lietadlach napríklad Mark II Gyro Sight, teda gyroskopický zameriavač. Toto zariadenie pilotovi ukazovalo na polopriehľadný displej, kam poletí strela na základe údajov z gyroskopu a meraču rýchlosťi.

Rozšírená realita prenikla na poli letectva aj do civilnej sféry. Na obrázku 5 je pilotná kabína Boeingu pri navádzaní na pristátie.

Podobné aplikácie by mohli mať význam aj pre vojenské a civilné pozemné dopravné prostriedky.

4 Metódy

Pre dosiahnutie rozšírenej reality je potrebné rozoznať, na ktoré miesto v obraze sa má vykresliť daný



Obr. 5: Rozšírená realita v kokpite lietadla Boeing 737-800; Autor: Barend Havenga [14]

virtuálny objekt. Tiež treba správne rozhodnúť, aký má byť tento objekt veľký a ako má byť natočený. Na toto rozhodnutie je treba poznat vzájomnú polohu virtuálneho objektu a kamery, respektíve očí používateľa.

Na riešenie problému sa obvykle používajú metódy počítačového videnia. Na ich aplikáciu je potrebné zariadenie s digitálnou kamerou, ktorej záznam softvér analyzuje.

4.1 Rozpoznávanie pomocou markerov

Najjednoduchší spôsob, ako sa dá tento prístup implementovať, je za pomoci markerov. Markerov sú obvykle jednoduché asymetrické čiernobiele značky, ktoré sa umiestnia do skutočného sveta tak, aby boli vždy v zábere kamery. Je potrebné zaznačiť, aká je presná poloha, na ktorú sa má vykresliť virtuálny objekt voči tomuto statickému markeru. Občas sa vykresluje priamo na marker, ale obvyklejšie je umiestniť marker len niekam do pozadia. Softvér s rozšírenou realitou potom hľadá vo videu tento marker a keď ho odhalí, tak nájde transformáciu, ktorá marker posunie, otočí a zoškáluje do veľkosti ako je vo videu. Touto transformáciou sa potom transformuje aj virtuálny model, ktorý sa vzápäť vykreslí buď späť do videa, alebo zvlášť na priečlný displej.

Technológia má presnosť umiestňovania jednotlivých objektov závislú od kvality snímaného obrazu. Vyskytujú sa však chyby, kedy aplikácia pre zlú viditeľnosť nerozozná marker a nič nevykreslí, alebo rozozná za marker skutočný objekt, ktorý markerom

nie je a vykreslí virtuálny objekt aj keď ho vykresliť nemá. Ďalšiou nevýhodou je samozrejme to, že v momente, keď sa stratí marker zo záberu, prestane sa vykreslovať aj k nemu prislúchajúci objekt. Tento problém sa dá riešiť bud' inštalovaním siete markerov do pozadia, ktoré v sebe majú zakódované svoje súradnice, alebo sledovaním (trackovaním) pohybu kamery¹. Softvéru potom stačí, aby bol v zábere kamery dobre viditeľný vždy aspoň jeden. Pokial' markery neznehodnotia scénu a vývojári potrebujú maximaličovať šance systému na správnu registráciu, môžu sa rozhodnúť použiť pole markerov (po anglicky *marker field*), ktoré pokrýva kompletný povrch scény.

4.1.1 Algoritmus rozpoznávania markerov

Hľadanie markerov sa zvykne robiť nasledovným postupom, ktorý sa zvlášť aplikuje na každý snímok z kamery. Snímok najprv prejde prahovaním, čo znamená, že sa zmení na binárny. Určí sa istý prah jasu a každý bod obrázku, ktorý má vyšší jas sa prefarbi na biely a všetky ostatné sa prefarbia na čierne. Výsledkom je binárny obraz. Ak sa prah nastaví správne malo by byť v tomto obraze jasne vidno aspoň jeden čiernobiely marker na jednofarebnom pozadí [15].

Ako ďalší krok sa označia jednotlivé jednofarebné komponenty a zdetegujú sa ich kontúry. Kontúry sa rozdelia na úsečky a softvér medzi nimi označí prie-sečníky. Tie, ktoré sú blízko pri sebe a súčet uhlov medzi nimi je blízky 180 stupňom sa odignorujú, pretože tvoria takmer rovnú čiaru a výrazne nemenia tvar objektu. Komponenty, ktoré nemajú štyri ostré vrcholy sa vyradia, pretože nemajú štvorcový tvar. Zvyšné komponenty ostanú kandidátmi na markery. Algoritmus ďalej nájde všetky možné homografie, ktoré zobrazia rohy štvorca na rohy týchto komponentov. Výsledkom sú rotačné matice, ktorými sa transformujú uložené obrázky markerov. Týchto markerov môže byť viac, napríklad ak chce program zobrazovať rôzne objekty na rôzne markery. Po tom, čo sa pre každý komponent vypočítajú cez všetky jeho matice všetky markery z pamäte, sa tieto výsledné obrázky prekryjú s originálnym snímkom a porovnajú [23]. Binárny obrázok s komponentmi už ďalej nie je potrebný. Pri porovnaní originálneho snímku s pretransformovaným markerom sa použije niektorá z metód hodnotenia podobnosti. Napríklad

¹Dá sa nameriť priamo z optického toku dát, pomocou informácií z gyroskopu, akcelerometra, magnetometra, alebo ich kombinácie.

sa pre každý pixel vypočíta hodnota rozdielu jasu a potom sa urobí suma všetkých týchto hodnôt. Čím je výsledná hodnota nižšia, tým sú si obrázky podobnejšie.

Pre každého kandidáta na marker sa vyberie ten obrázok, ktorý je mu najpodobnejší a zapamätá sa akou konkrétnou homografiou vznikol. Ak je podobnosť nižšia ako istá hranica, kandidát sa vymaže z výberu a považuje sa za chybu. Výsledkom je zoznam jednotlivých kandidátov, ich súradníc v rámci snímky, k nim prislúchajúce identifikácie markerov (ak sa v aplikácii používa viac markerov) a homografie, určujúce ako na ne niečo premietnuť.

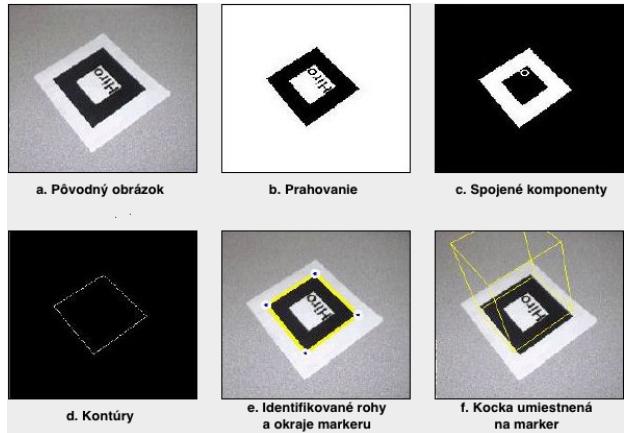
V prípade, že je snahou umiestniť virtuálny objekt priamo na marker, môže sa na jeho 3D model použiť daná homografia, čím sa správne umiestní, naškáluje a zrotuje, a vykreslí bud' do pôvodného snímku, alebo na priečadlný displej. Ak je marker posunutý, považuje sa za počiatok súradnicovej sústavy a virtuálny objekt sa adekvátnie posunie.

Marker samozrejme nemusí byť čiernobiely ani štvorcový, v týchto prípadoch sa algoritmus príslušne upraví. Nakol'ko vypočítavanie celého tohto postupu pre každý snímok v reálnom čase môže byť náročné, často sa využívajú techniky sledovania pohybu. To znamená, že si program pamäta pre objekt jednotlivé polohy a homografie z predchádzajúcich snímkov a prednostne ich vyskúša. Taktiež môže pri plynulom pohybe predpovedať lokácie na nasledovných snímkoch.

Kroky tohto algoritmu sú znázornené na obrázku 6. Jeho implementáciu v knižnici ARToolKit využívame aj v demo aplikácii vytvorenej v rámci tejto práce.

4.2 Rozpoznávanie na základe významných bodov

V prípade, že nie je žiadúce použitie markeru, pretože nie je možné modifikovať prostredie, prípadne je potrebné, aby aplikácia fungovala aj v prostrediach, ktoré na tento účel neboli predpripravené, je potrebné túto úlohu riešiť zložitejšou analýzou. Pri riešení týmto spôsobom musí byť známe ako vyzerá okolie, do ktorého je žiadane vykresliť virtuálny objekt. Toto okolie je potom potrebné rozoznať v zábere kamery. Táto metóda sa obvykle používa na aplikáciu, ktoré napríklad dopisujú v galérií údaje k obrazom a podobne. V tomto prípade slúži obraz ako špeciálny marker.



Obr. 6: Jednotlivé kroky aplikované pri rozpoznávaní markeru, tak ako sú uvedené v dokumentácii knižnice ARToolKit [23]

4.2.1 Rozpoznávanie algoritmom SIFT

Scale-invariant feature transform, alebo SIFT je algoritmus na rozpoznávanie a popisovanie významných bodov vyvinutý Davidom Lowe. Idea je, že objekt, ktorý chceme rozoznať obsahuje významné body, ktoré sa dajú popísat'. Výsledkom je deskriptor, s pomocou ktorého sa dá tento objekt lokalizovať na iných obrázkoch.

Algoritmus generuje z obrazu objektu množstvo vektorov významných bodov. Tieto vektori sú invariantné na rotáciu a škálovanie obrazu. Významné body obvykle ležia na rohoch, hranách a iných kontrastných miestach, vďaka čomu sú dobre viditeľné aj za zhoršených podmienok. Pri detekcii sa potom vyhľadáva v tejto databáze významných bodov [24, 33].

Jednou z výhod algoritmu SIFT je, že dokáže rozpoznať aj objekty, ktoré sú čiastočne zakryté. Jedným z následníkov SIFT je napríklad SURF (celý názov *Speeded Up Robust Features*), ktorý je približne osem krát rýchlejší. SURF je patentovaný Herbertom Bayom [6]. Ďalšími algoritmami na rozpoznávanie významných bodov sú BRIEF [8] (*Binary Robust Independent Elementary Features*), ktorý má podobné alebo lepšie výsledky ako SURF, BRISK (*Binary Robust Invariant Scalable Keypoints*) [20], FREAK (*Fast Retina Keypoint*) [1], SUSSAN [27], FAST [31] a DAISY [30].

4.3 Prehľad softvérových knižníc

Existuje niekoľko softvérových knižníc uľahčujúcich implementáciu aplikácií vytvárajúcich rozšírenú

realitu.

Jednou z prvých takýchto knižníc je ARToolKit vyvinutý Hirokazu Katom v roku 1999 pre jazyky C a C++. Táto knižnica deteguje markery a prepočíta pod akým uhlom ich používateľ vidí. Vývojárom d'alej poskytuje súradnicový systém, ktorý do tohto priestoru transformuje. ARToolKit je k dispozícii zadarmo pod licenciou GNU/GPL [2]. Z tejto knižnice vychádza množstvo d'alších nasledovníkov a dodnes sa používa. Využívame ju za účelom registrácie objektov v aplikácii s rozšírenou realitou, ktorá je súčasťou tejto práce.

ARToolkitPlus bol otvorený tracker, ktorý vychádzal z ARToolKitu. Jeho vývoj sa zastavil v roku 2007 a bol nahradený Studierstube trackerom, ktorý už však nie je verejne prístupný [34].

Studierstube je framework vyvinutý na *Graz University of Technology*. Na trackovanie sa odporúča použiť OpenTracker od rovnakých autorov. Vývoj bol ukončený v roku 2008 [19, 18].

ArUco je minimalistická knižnica pre C++. Rozpoznáva markery, alebo polia markerov a je veľmi rýchla vďaka tomu, že sa opiera o OpenCV [3].

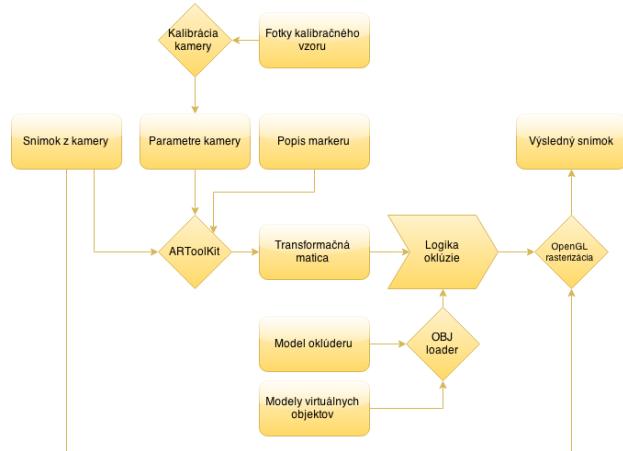
Vuforia je proprietárnej knižnicou, ktorá dokáže rozpoznať obrázky a jednoduché 3D objekty. Dá sa použiť v C++, Java a Objective-C, vďaka čomu je oblúbená na mobilných platformách iOS a Android [25].

5 Implementácia

V rámci riešenia práce sme navrhli aplikáciu na demonštráciu rozšírenej reality obohatenú o oklúzie fyzických objektov.

V tejto kapitole popisujeme proces vývoja demo aplikácie od začiatku po finálnu verziu. Pre takúto demonštráciu je potrebné vyriešiť niekoľko problémov. Získať, alebo vyrobiť samotný oklúder a jeho digitálnu 3D reprezentáciu. Je potrebné vymodelovať objekty, ktoré chceme vykreslovať a naimportovať ich do grafickej knižnice. Tiež potrebujeme nejakým spôsobom registrovať scénu, do ktorej chceme objekty vykreslovať. Potrebujeme nakalibrovať kameru, aby bola ilúzia čo najpresnejšia. Na záver potrebujeme vyriešiť, ako implementovať samotnú oklúziu.

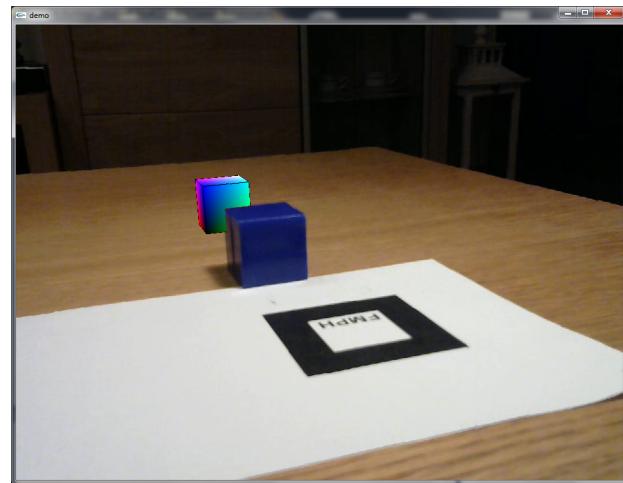
Táto kapitola opisuje všetky kroky schémy znázornenej na obrázku 7.



Obr. 7: Podrobná schéma demo aplikácie

5.1 Zjednodušená demonštrácia

Na obrázku je snímok obrazovky prvého funkčného prototypu našej demo aplikácie. Aplikácia registruje scénu pomocou markeru a dokresľuje na ňu virtuálnu kocku okludovanú fyzickou kockou. Na obrázku 8 sa nachádza snímok obrazovky tohto skorého prototypu. Kroky tohto procesu sú chronologicky popísané v nasledujúcich sekciách.



Obr. 8: Jednoduchá ukážka oklúzie s kockami v našej aplikácii

5.2 3D modely

Aby bolo možné vykreslovať 3D objekty na obrazovku, je potrebné ich reprezentovať v pamäti. Počas behu programu sa o to stará grafická knižnica, ale najjednoduchší spôsob ako do nej dátá naimportovať je uložiť ich do súboru.

Pre prácu s 3D modelmi existujú štandardizované formáty. Koncom deväťdesiatych rokov bol obľúbený formát VRML², ktorý je aj natívne podporovaný v ARToolkit, neskôr ho nahradil následník X3D [9]. Medzi dnes používanejšie formáty patrí napríklad COLLADA vyvíjaná združením Khronos Group [12], jej výhodou je vysoká kompatibilita s väčšinou nástrojov na modelovanie aj hernými enginmi. Iným obľúbeným formátom je STL (*stereolithography*) používaný najmä na inžinierske účely ako je automatické projektovanie (po anglicky *computer-aided design*, alebo *CAD*) a 3D tlač. Výhodou je možnosť ukladania do textových aj binárnych súborov.

Zrejme najpoužívanejším 3D formátom je dnes Wavefront OBJ vyvinutý už neexistujúcou firmou Wavefront Technologies. OBJ je jednoduchý otvorený formát, ktorý sme sa rozhodli použiť.

5.2.1 Načítanie Wavefront OBJ

OBJ sme si vybrali, pretože sa jednoducho spracováva (*parsuje*) a zároveň je všeobecne používaný a kompatibilný.

Formát vie zaznamenávať vrcholy (*vertexy*), ich normály, textúrne súradnice a steny. Steny sú reprezentované n-uholníkmi danými zoznamami už definovaných vrcholov. Tieto vrcholy sú v zoznamoch uvedené v poradí v smere hodinových ručičiek [28].

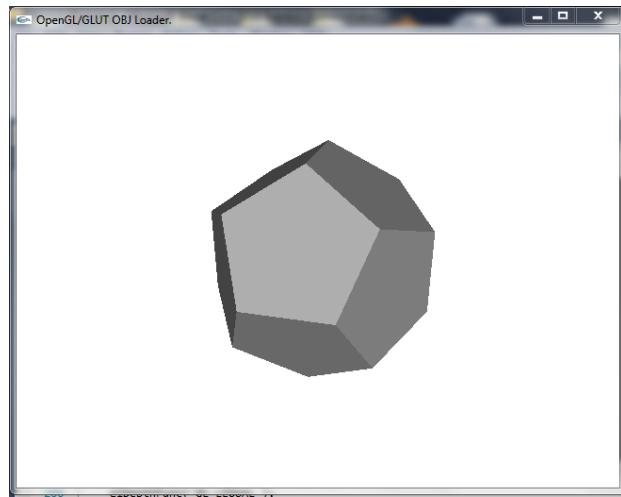
Aby sme si ušetrili starosti pri spracovávaní a vykreslovaní modelu, rozhodli sme sa ho reprezentovať trojuholníkovou sietou (po anglicky *triangulated geometry*). To znamená, že každá stena je rozložená na steny ktoré majú len tri vrcholy. To má tú výhodu, že tieto trojuholníky sa dajú neskôr jednoducho vykreslovať pomocou OpenGL.

Na trianguláciu modelov sme použili otvorený modelovací nástroj Blender, ktorý dokáže exportovať modely do formátu OBJ v tejto podobe.

Takto pripravený súbor načítame, spracujeme a vrcholy, normály a steny si uložíme do polí, z ktorých ich cez OpenGL rasterizujeme (ukážka na obrázku 9).

5.3 Registrácia scény

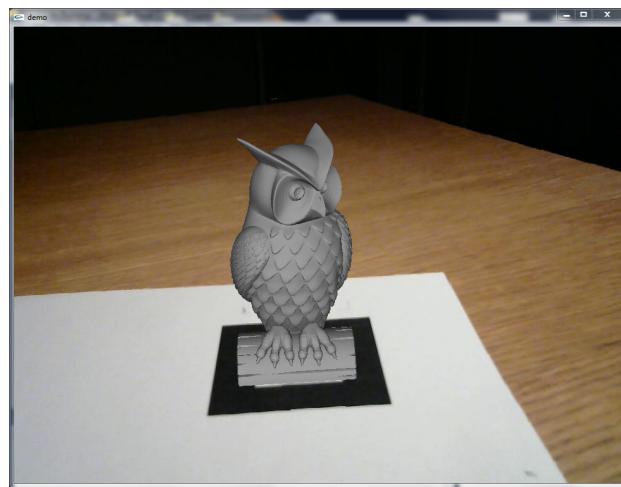
Na registráciu scény sme sa rozhodli použiť slobodnú knižnicu ARToolkit. Vznikla už v roku 1999 [2], ale stále sa používa pre svoju jednoduchú rozšíritelnosť. ARToolkit deteguje markery spôsobom popísan-



Obr. 9: Dvanásťstien načítaný zo súboru OBJ a vykreslený na obrazovku

ným v predchádzajúcej kapitole a nájdete nám transformáciu, ktorou zarovnáme virtuálnu scénu v počítači do reálnej scény z kamery.

Do tejto virtuálnej scény môžeme pomocou OpenGL vykresliť to, čo potrebujeme. V prípade, že vieme aké bude osvetlenie fyzickej scény, môžeme podľa toho nastaviť osvetlenie vo virtuálnej scéne, aby neskutočné objekty pôsobili skutočnejšie. Na obrázku 10 je ukážka modelu vykresleného na marker.



Obr. 10: Model sovy vykreslený a umiestnený na marker v našej demo aplikácii

5.4 Kalibrácia kamery

Pokiaľ vytvárame aplikáciu s rozšírenou realitou dosiahnutou pomocou počítačového videnia a potrebujeme, aby zobrazovala virtuálne objekty čo najpres-

²Číta sa vermal

nejšie, je potrebné nakalibrovať kameru.

Kamery sa líšia kus od kusu a preto je potrebné vykonať meranie, ktorým získame parametre konkrétneho zariadenia. Tieto parametre potom zohľadníme pri rozpoznávaní obrazu.

V prípade bežnej dierkovej kamery (po anglicky *pinhole camera*) sa tieto parametre usporiadavajú do matice parametrov kamery (*camera matrix*), ktorá je vynásobením matice vnútorných (*intrinsic*) parametrov s vonkajšími (*extrinsic*) parametrami, ktoré udávajú transformáciu 3D súradníc svetu na 3D súradnice kamery. Určenie matice vnútorných parametrov kamery sa nazýva kalibrácia kamery.

$$A = \begin{pmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

Matica vnútorných parametrov obsahuje parametre, ktoré sa dajú vypočítať nasledovne.

$$\alpha_x = f \cdot m_x \quad (1)$$

$$\alpha_y = f \cdot m_y \quad (2)$$

Kde m_x a m_y sú škálovacie faktory pixelov a f je ohnisková vzdialenosť. γ je skresľovací koeficient medzi osami x a y a obvykle ho môžeme nastaviť na 0. u_0 a v_0 označujú posunutie optického stredu³ (po anglicky *principal point*) v oboch osiach [13].

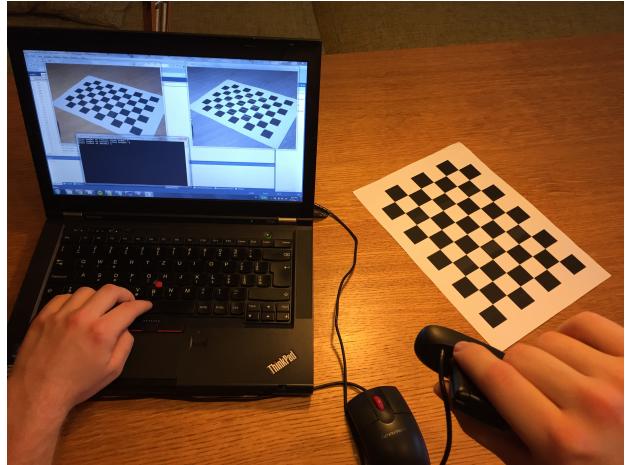
Tieto parametre je možné (pokiaľ sú známe vonkajšie parametre kamery) vypočítať z minimálne jednej fotky kalibráčného vzoru⁴. Vhodné je fotiť napríklad šachovnicu alebo pole kruhov. Od začiatku merania nesmie kamera opticky preostrovovať, ani meniť ohniskovú vzdialenosť (priблиžovať), pretože by sa parametre zmenili.

Po transformácii maticou parametrov kamery vieme zvrátiť deformáciu obrazu.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

Pre kalibráciu kamery sme použili jednoduchý program využívajúci knižnicu pre počítačové vide-

nie OpenCV (obrázok 11). Táto knižnica implementuje šikovné metódy na detekciu kalibračnej siete aj výpočet parametrov z nameraných údajov. Taktiež umožňuje výpočet šošovkového skreslenia spôsobeného použitím bežnej kamery so šošovkou.



Obr. 11: Kalibrácia kamery

5.5 Príprava oklúderu

Predtým ako môžeme implementovať oklúzium, potrebujeme mať vybraný oklúder a jeho digitálnu 3D reprezentáciu. Zvolili sme si na to viacero postupov.

5.5.1 Modelovanie

Najzákladnejší spôsob ako získať 3D model skutočného objektu, je ho jednoducho odmerať a vymodelovať v modelovacom programe. Pri prvom prototype ukázanom v článku 5.1 sme aplikovali zjednodušený postup a zvolili si za oklúder kocku, pretože má len osem vrcholov na jednoducho vymenovateľných súradničiach. Rozhodli sme sa vyniechať načítavanie modelu a kocku popísať priamo v programe. Vďaka tomu, že každé dve kocky sú si podobné stačí po popísaní kocky nájsť len správny škálovací koeficient.

5.5.2 3D skener SMISS

SMISS, teda *Scalable Multifunctional Indoor Scanning System* je zariadenie vyvinuté na Fakulte matematiky, fyziky a informatiky Univerzity Komenského, ktoré dokáže skenovať trojrozmerné objekty [32]. Objekt, ktorý je potrebné nasnímať sa položí na otočný stôl. Na tento stôl je pod uhlom namierená kamera a projektor. Projektor premieta na snímaný

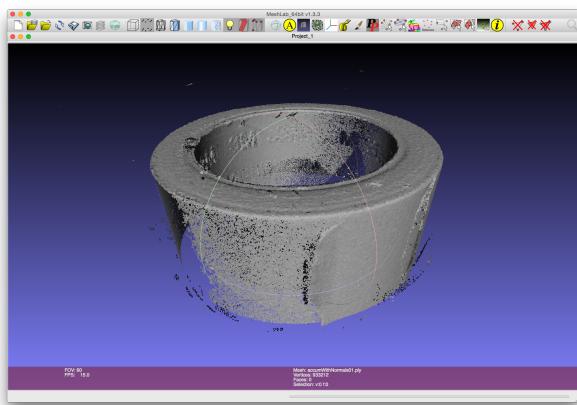
³Optický stred je bod, ktorý je prienikom optickej osi s priebežnou.

⁴Obvykle sa pre zvýšenie presnosti použijú výsledky z viacerých fotiek vytvorených z rôznych uhlov. My sme ich pri kalibrácii vyhotovili dvadsať.

objekt štruktúrované svetlo. Toto svetlo projektuje pruhy, ktorými „rozreže“ skenovaný objekt na jednotlivé roviny, pričom každá rovina je osvetlená a zakódovaná iným vzorom svetla. Každé nasvietenie je odfotené kamerou. Na začiatku skenovania je objekt rozdelený iba na zopár rovín, tie sa ale postupne delia na tenšie rezy, čím vzniká vyššia detailnosť. Keď sú zosnímané všetky požadované nasvietenia, motor stôl pootočí a projektor začne objekt znova nasvecovať z nového uhlu [17, 16].

Počítač z obrazu dekóduje jednotlivé roviny a po zosnímaní zo všetkých strán vytvorí mračno bodov (po anglicky *point cloud*) reprezentujúce objekt. Na výsledné mračno bodov môžeme aplikovať trianguláciu a tým získame polygonálny model, ktorý môžeme použiť ako oklúder. Priemerná prenosť SMISSu je $500 \mu\text{m}$ [16].

Na skeneri sme skenovali lepiacu pásku, tento sken je na obrázku 12.



Obr. 12: Mračno bodov, ktoré je výstupom skenovania SMISSom

5.5.3 3D tlač

Druhou možnosťou, ako získať oklúder a jeho virtuálny 3D model je namodelovať ho vo počítači a potom ho vytlačiť na 3D tlačiarne. Na vytvorenie takéhoto modelu je vhodný napríklad Blender, z ktorého ho potom môžeme vyexportovať do OBJ pre aplikáciu aj STL pre tlač.

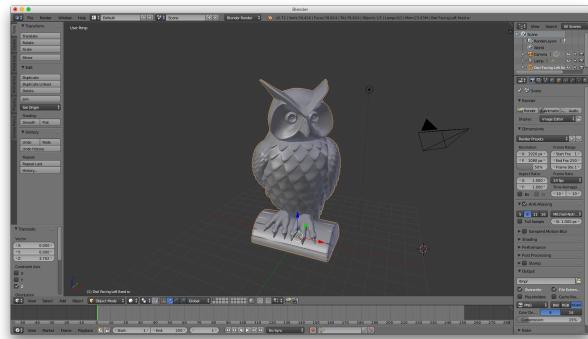
STL model nahráme do *sliceru*, to je program, ktorý zoberie model a „rozreže“ ho na veľký počet 2D vrstiev. Príkladom je slicer MakerBot Desktop, dodávaný k tlačiarňam značky MakerBot. Výška vrstvy závisí od nastaveného rozlíšenia. Tieto dátu sa potom

vložia do tlačiarne a tá začne roztápať plastovú náplň (*filament*) a nanášať ju vrstvu po vrstve na seba.

Inteligentný slicer dokáže do dutého modelu dordobiť siet vnútorných stien, aby sa nerozpadol. V prípade, že tlačený predmet obsahuje časti, ktoré prevírajú von a teda by ich nebolo ako tlačiť do vzduchu, môže slicer dopočítať podpery, ktoré sa vytlačia spolu s modelom a po vytlačení sa odrezú. V prípade použitia tlačiarne s dvomi tlačiacimi hlavami je dokonca možné použiť dva filamenty, z toho jeden vodou rozpustný, ktorým sa tlačia podpery. Po tlači sa model len ponori do vody.

Dnešné 3D tlačiarne zvládajú rozlíšenie okolo $100 \mu\text{m}$, čo je okolo 255 dpi, teda 255 vrstiev na jeden palec výšky. Nevýhodou je, že s kvalitou rýchlo narastá aj čas, ktorý tlačenie potrvá.

Na tlačiarni sme tlačili model sovy z obrázku 13.



Obr. 13: Digitálny model sovy, otvorený v modelovačom programe Blender. Autorom modelu je modelár Tom Cushta [10]

5.6 Oklúzia v rozšírenej realite

Ak chceme v rozšírenej realite zobrazovať virtuálne objekty tak, aby ich úplne, alebo čiastočne prekrývali fyzické objekty na scéne, je potrebné, aby pre každý tento fyzický objekt aplikácia okrem virtuálneho 3D modelu poznala aj jeho veľkosť a polohu na scéne voči markeru, prípadne ho vedela inak rozoznať.

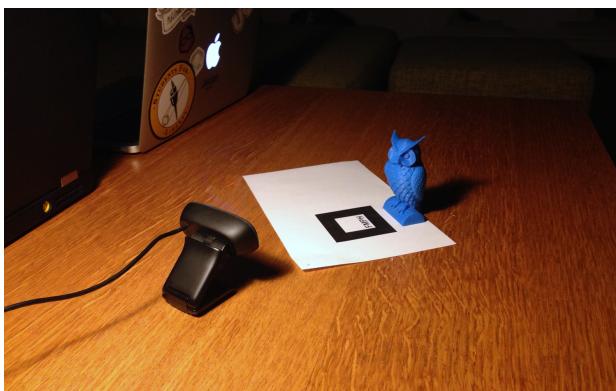
Program následne objekty, prípadne ich časti, ktoré sú prekryté oklúderom nevykreslí a rovnako nevykreslí ani samotný oklúder. Na mieste, kde majú byť objekty prekryté sa teda zobrazí stopa z videa, ktorá je na pozadí. Táto stopa na danom mieste obsahuje pôvodný fyzický objekt zosnímaný kamerou a tým sa vytvorí ilúzia, že virtuálne objekty sú prekryté tými reálnymi. Pri oklúzii je obzvlášť dôležité mať

správne nakalibrovanú kameru, aby sa virtuálne objekty premietali čo najpresnejšie na svoje miesta.

Najprv si vykreslíme pomocou OpenGL oklúder do *stencil bufferu*. Tento buffer je binárnym obrázkom veľkosti vykreslovaného okna a slúži ako úložné miesto na dočasné informácie o danom obrazu. Na každý bod v stencil bufferi zapíšeme true, pokiaľ by sa na dané súradnice v normálnom pohľade vykreslila časť oklúdera. Neskôr pri vykreslovaní virtuálnych objektov, ktoré sa majú nachádzať za oklúderom, robíme pre každý pixel jednoduchú kontrolu. Nachádza sa na týchto súradničach v stencil bufferi hodnota true? Ak sa nachádza, tak pixel nevykreslíme, pretože nemá byť vidieť a pokračujeme ďalej.

5.7 Výsledky

Aplikácia má funkcionality, ktorú sme si špecifikovali. Na obrázku 14 je znázornený príklad scény na ktorej predvádzame aplikáciu.



Obr. 14: Scéna na ktorej demonštrujeme oklúziu. Modrý model sovy je oklúderom.

Aby sme sa uistili, že máme fyzický oklúder položený na správnu relatívnu pozíciu voči markeru, môžeme si v aplikácii zapnúť vykreslovanie oklúdera a pozrieť sa, nakoľko presne sa prekrývajú (znázorené na obrázku 15).

Ak je oklúder správne zarovnaný a vypneme jeho vykreslovanie na obrazovku, dosiahneme oklúziu⁵, ako je vidno na obrázku 16.

5.7.1 Výkon aplikácie

Bez ohľadu na zložitosť (počet vrcholov) skúšaných modelov nám aplikácia bežala na našom hard-

⁵Autorom voľne šíritel'ného modelu stromu v pozadí je theswiss z portálu Thingiverse [29].



Obr. 15: Natočili sme scénu tak, aby sa objekty prekrývali a nechali sme vykreslovať aj virtuálny model oklúdera, ktorý sa pomocou markeru registruje cez skutočnú sovu.



Obr. 16: V predchádzajúcej scéne sme nechali oklúder vykreslovať iba do stencil bufferu a výsledkom je oklúzia s fyzickým objektom.

véri plynulo pri framerate 29 až 30 snímkov za sekundu (*framerate* vyjadruje rýchlosť obnovovania obrazovky). Informácie o použitých modeloch sú uvedené v tabuľke 4.1. Na obrázku 17 sú vykreslené všetky modely naraz, tiež pri framerate 30 snímkov za sekundu.

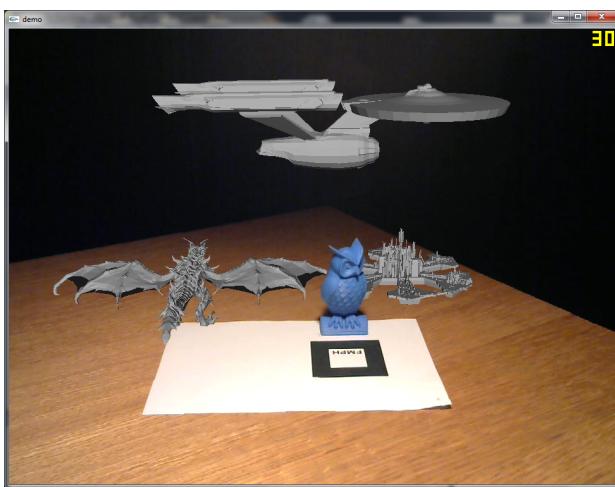
Hodnota 30 snímkov za sekundu je limitovaná našou kamerou, ktorá nezvláda robiť snímky rýchlejšie a aplikácia teda musí po spracovaní snímku čakať na nasledovný snímok.

Aplikáciu sme spúšťali na počítači s procesorom Intel Core i3 taktovaným na 2.4 GHz, 8 GB operačnej pamäte a integrovanou grafickou kartou Intel HD

3000. Snímky z kamery boli spracovávané v rozlíšení 960 na 720 pixelov.

model	počet vrcholov	počet stien
sova (oklúder)	39416	7824
lod' Enterprise	2680	5372
drak Alduin	8160	16074
mesto Atlantis	41437	83190
spolu	91693	112460

Tabuľka 1: Údaje o modeloch použitých pri testovaní výkonu aplikácie



Obr. 17: Vykresľovanie viacerých modelov naraz

5.7.2 Možné problémy a nepresnosť

Detekcia markeru funguje spoločne s aplikáciou obvykle nemá problémy s identifikáciou scény pokiaľ je celý marker viditeľný. Zaznamenali sme iba jeden prípad kedy detekcia zlyhá a to keď je časť markeru v tieni. Táto situácia môže byť nastáť, pokiaľ je scéna nasvetiená ostrým svetlom (napríklad stolou lampou) spoza oklúdera a oklúder zatieňuje časť markeru.

Problémy s nepresnosťou oklúzie môžu nastáť niekoľkými spôsobmi, prípadne ich kombináciou. Fyzický model oklúdera je potrebné relatívne umiestniť voči markeru veľmi presne, čo môže byť obtiažne. Nepresnosť tak tiež môže spôsobiť nesprávne naškálovaný virtuálny model oklúdera. Nenakalibrovaná, alebo nedostatočne presne nakalibrovaná kamera spôsobuje ďalšiu nepresnosť. Tieto nepresnosti sa prejavujú tak, že výrez vo vykresľovanom objekte je posunutý oproti fyzickému oklúderu a nie je s ním presne zarovnaný.

6 Záver

Úlohou práce bolo naprogramovať aplikáciu, ktorá v reálnom čase zaregistruje objekt a vykreslí zaň iný virtuálny objekt, čiastočne prekrytý tým skutočným. Cieľ práce bol splnený.

V práci boli vysvetlené princípy rozšírenej reality, možné aplikácie a techniky na jej dosiahnutie. Ukázali sme, ako vytvoriť aplikáciu s rozšírenou realitou, ktorá demonštruje oklúziu s reálnym svetom a túto aplikáciu sme aj vytvorili.

Tromi rôznymi metódami sme si pripravovali oklúdere, naprogramovali sme načítavanie modelov, kalibrovali kameru, vytvorili vlastný marker, korektnie registrovali scénu, vykreslili virtuálne objekty a zabezpečili ich oklúziu a poskladali to celé do jednej demo aplikácie.

Podobná aplikácia (rozšírená realita obohatená o oklúziu) by mohla mať využitie v umení, zábave, na vyučovaní alebo v ľubovoľnom prípade, v ktorom chceme zvýšiť pocit vnorenia do rozšírenej reality.

Pod'akovanie

Ďakujem svojej školiteľke RNDr. Zuzane Berger Haladovej, Phd. za pomoc, cenné nápady, neustálu prístupnosť a podporu.

Taktiež som vďačný organizácii FabLab Bratislava, ktorá mi sprístupnila 3D tlačiareň a poradila, ako s ňou narábať. Ďakujem tímu FTLab na našej fakulte za prístup k 3D skeneru SMISS.

Literatúra

- [1] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517. Ieee, 2012.
- [2] ARToolKit. Webová stránka artoolkit @ONLINE. <http://www.hitl.washington.edu/artoolkit/>, 2004.
- [3] ArUco. Aruco library @ONLINE. <http://www.uco.es/investiga/grupos/ava/node/26>, 2014.
- [4] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *Computer Graphics and Applications, IEEE*, 21(6):34–47, 2001.
- [5] Ronald T Azuma et al. A survey of augmented reality. *Presence*, 6(4):355–385, 1997.

- [6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer, 2006.
- [7] Oliver Bimber and Ramesh Raskar. *Spatial augmented reality: merging real and virtual worlds*, volume 6. AK Peters Wellesley, MA, 2005.
- [8] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010*, pages 778–792. Springer, 2010.
- [9] Web3D Consortium. What is x3d @ONLINE. <http://www.web3d.org/x3d/what-x3d>, 2015.
- [10] Tom Cushwa. Owl statue @ONLINE. <http://www.thegiverse.com/thing:18218>, 2012.
- [11] Victor Fragoso, Steffen Gauglitz, Shane Zamora, Jim Kleban, and Matthew Turk. Translatar: A mobile augmented reality translator. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 497–502. IEEE, 2011.
- [12] Khronos Group. 3d asset exchange schema @ONLINE. <http://www.khronos.org/collada>, 2015.
- [13] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [14] Barend Havenga. Hud on a boeing 737-800 @ONLINE. http://www.reddit.com/r/pics/comments/lplfb/hud_on_a_boeing_737800, 2012.
- [15] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999.(IWAR'99) Proceedings. 2nd IEEE and ACM International Workshop on*, pages 85–94. IEEE, 1999.
- [16] Tomáš Kovačovský. Hdr smiss–fast high dynamic range 3d scanner.
- [17] Tomáš Kovačovský. *Scalable multifunctional indoor scanning system*. PhD thesis, Citeseer, 2012.
- [18] Tobias Langlotz. Studierstube opentracker @ONLINE. <http://studierstube.icg.tugraz.at/opentracker/>, 2008.
- [19] Tobias Langlotz. Studierstube project @ONLINE. <http://studierstube.icg.tugraz.at/download.php>, 2008.
- [20] Stefan Leutenegger, Margarita Chli, and Roland Yves Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.
- [21] Mark A Livingston, Lawrence J Rosenblum, Dennis G Brown, Gregory S Schmidt, Simon J Julier, Yohan Baillot, J Edward Swan II, Zhuming Ai, and Paul Maassel. Military applications of augmented reality. In *Handbook of augmented reality*, pages 671–706. Springer, 2011.
- [22] Mark A Livingston, Lawrence J Rosenblum, Simon J Julier, Dennis Brown, Yohan Baillot, II Swan, Joseph L Gabbard, Deborah Hix, et al. An augmented reality system for military operations in urban terrain. Technical report, DTIC Document, 2002.
- [23] Julian Looser. Artoolkit documentation @ONLINE. <http://www.hitl.washington.edu/artoolkit/documentation/vision.htm>, 2004.
- [24] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.
- [25] Qualcomm. Vuforia library @ONLINE. <https://www.qualcomm.com/products/vuforia>, 2015.
- [26] Christian Schonauer, Emanuel Vonach, Georg Gerstweiler, and Hannes Kaufmann. 3d building reconstruction and thermal mapping in fire brigade operations. In *Proceedings of the 4th Augmented Human International Conference AH '13*, New York, 2013. ACM. talk: 4th Augmented Human International Conference, Stuttgart; 2013-03-07 – 2013-03-08.
- [27] Stephen M Smith and J Michael Brady. Susan—a new approach to low level image processing. *International journal of computer vision*, 23(1):45–78, 1997.
- [28] Wavefront Technologies. *Object file specification* @ONLINE, 1991.
- [29] theswiss. business card tree @ONLINE. <http://www.thegiverse.com/thing:439474>, 2014.
- [30] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(5):815–830, 2010.
- [31] Oncel Tuzel, Fatih Porikli, and Peter Meer. Region covariance: A fast descriptor for detection and classification. In *Computer Vision–ECCV 2006*, pages 589–600. Springer, 2006.
- [32] FMFI UK. Mgr. tomáš kovačovský, 3d scanner smiss @ONLINE. <https://www.youtube.com/watch?v=TW1hbInC7vc>.
- [33] Marek Vološín. Rozpoznávanie obrazov v robotike. Master’s thesis, Technická univerzita v Košiciach, 2011.
- [34] Daniel Wagner and Dieter Schmalstieg. *Artoolkitplus for pose tracking on mobile devices*. na, 2007.
- [35] Simon Julier Yohan, Simon Julier, Yohan Baillot, Marco Lanzagorta, Dennis Brown, and Lawrence Rosenblum. Bars: Battlefield augmented reality system. In *In NATO Symposium on Information Processing Techniques for Military Systems*. Citeseer, 2000.