Západočeská univerzita v Plzni

INIB-IK

Bakalářská práce

Rozšířená realita

Autor:
Andrea Varáčková

FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

Obsah

1 Úvod do rozšířené reality

Rozšířená realita (Augmented Reality, zkráceně AR) [1] je variace virtuální reality (Virtual reality, VR). VR kompletně pohltí uživatele do uměle vytvořeného prostředí, během čehož nevidí okolní svět. AR naopak nechá uživatele vidět reálný svět, ale vkládá do něj virtuální předměty, jež ho překrývají, nebo se s ním spojí. AR okolní prostředí pouze doplňuje místo jeho úplného nahrazení. Tvoří tak mezičlánek reálného světa a virtuální reality.

Někteří výzkumníci definují AR tak, že je k jeho použití potřeba Head-Mounted Display (HMD), což jsou přístroje, které se nasadí na hlavu a umožňují vidět virtuální objekty nebo prostředí. To ale limituje AR na určité technologie, proto jsou v této práci AR definované jako systémy, splňující tyto tři charakteristiky:

- **Kombinují reálné a virtuální** - Reálné prostředí s virtuálními, umělými objekty se může překrývat a uživatel uvidí obojí.
- **Jsou interaktivní v reálném čase** - Interakce s virtuálními objekty jsou prováděny ihned, aniž by uživatel musel dlouho čekat.
- **Jsou registrované ve 3D** - Virtuální objekty, ať už 2D nebo 3D, jsou vkládány to reálného, 3D prostředí. Musí být vloženy do přesné pozice a hloubky.

Tato definice dovoluje při zachování základů AR i jiné technologie než HMD, například využití mobilního telefonu nebo webkamer [2]. AR umožňuje uživateli získat o prostředí informace, které by pouze svými smysly získat nemohl. Zlepšuje vnímání reálného prostředí a interakci s ním, čímž mu může ulehčovat práci v reálném světě.

1.1 Aplikace

V následujících odstavcích jsou vypsány příklady oborů, jež využívají rozšířenou realitu. AR je v nich využito ke zjednodušení či zpřesnění práce vykonávané pracovníkem v daném oboru. Postupem času se tyto technologie stále zlepšují a přibývá více a více možností pro jejich aplikaci [3].

1.1.1 Lékařství

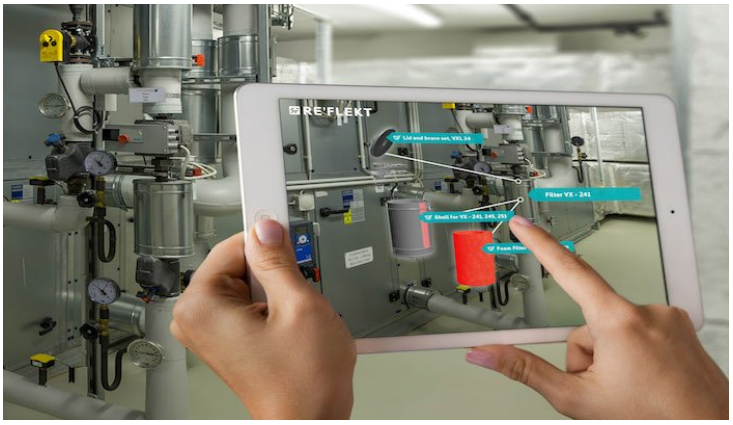
AR se využívá jako pomůcka pro školení chirurgů a vizualizaci operací – viz Obrázek 1. Umožňuje důkladnou vizualizaci pacienta, což má za následek možnost zmenšení operačních zákroků a tím pádem snížení pooperačních traumat.

Další možností využití je kombinace lidského zraku s daty získanými pomocí AR ve stejném čase, "promítání" dat přímo na pacienta a tím usnadnění lokalizace a upřesnění místa zákroku.

Obrázek 1: Příklad AR v lékařství

1.1.2 Výroba a opravy v průmyslu

AR může zobrazovat instrukce nejen jako textové a obrázkové manuály, ale i jako projekci přímo do prostoru, kde je jich potřeba – viz Obrázek 2. Mohou se zobrazovat krok za krokem, být pohyblivá, přesně ukazovat body, které, kde a kdy se mají vykonávat. Díky názorné ukázce postupu je tedy možné dojít ke zrychlení práce, aniž by byla zhoršena její kvalita. AR se dá také využít k efektivnímu zaučování nových pracovníků do pracovního procesu.



Obrázek 2: Příklad AR ve výrobě

1.1.3 Anotace a vizualizace

Další využití rozšířené reality je anotace objektů a prostředí veřejnými informacemi za předpokladu přístupu do veřejných databází. Anotace je možné připnout přímo k objektům, takže při pohledu na něj je s určitým vybavením možné rovnou číst informace k němu připojené – viz Obrázek 3. Využívá se toho například pro orientaci v knihovně, pro popis jednotlivých částí strojů nebo vložení návrhů budov do scenérie, aby bylo možné posoudit, jak by na svém místě skutečně vypadaly.

Obrázek 3: Příklad AR anotace

1.1.4 Plánování cest robotu

Ovládání robotu může být mnohdy příliš náročné, obzvlášť když je ve velké vzdálenosti a odezva je pomalá. V těchto případech je snazší ovládat jen virtuální model, který umožňuje předvídat chování skutečného robotu v reálném čase. Výsledky jsou poté přímo zobrazeny v reálném světě. Pokud jsou žádoucí, uživatel nechá skutečného robota provést ověřený plán.

1.1.5 Zábava

Ve filmařství je možné zasadit herce stojícího před zelenou plochou do libovolného virtuálního 3D prostředí v reálném čase. Další možné využití je aplikace v interaktivních hrách, jakou je například Pokémon GO, kterou je možno vidět na Obrázku 4. Ve hře je přes kameru mobilu vidět reálné prostředí a do něj jsou vloženy virtuální postavičky. Tyto hry se stávají velmi úspěšnými, prolnutí reálného a virtuálního je lákavou zábavou.

Obrázek 4: Příklad AR pro hru

1.1.6 Letectvo

AR pomáhá pilotům při letu zobrazováním informací do jejich zorného pole, aniž by kompletně překrývalo výhled nebo piloty nutilo k uhýbání pohledem. Stejně může pomáhat i při střelbě nebo míření na terč a podobně, což ukazuje výhled z kokpitu obohacen o AR na na Obrázku 5.

Obrázek 5: Příklad AR v letectví

1.2 Charakteristika

AR může do reálného prostředí objekty vkládat, ale i "odstraňovat". Je možné překrýt předmět, který chceme nechat zmizet, přidáním vrstev zobrazujících pozadí daného předmětu.

Rozšířenou realitou je možné nahrazovat více smyslů, než jen zrak. Za použití sluchátek se uživateli pouští zvuky, které se v reálné scéně momentálně nevyskytují. Když jsou sluchátka zvukotěsná, mikrofony se nahrávají okolní zvuky a z nich se dají určité části vyfiltrovat. Tímto uživatel nemusí slyšet vše, co by slyšel ve skutečnosti, ale jen to, co má v daném případě slyšet. Další možností je použití speciálních rukavic se senzory, které umožňují při dotyku změnit hmatové vjemy uživatele.

1.3 Optické versus video technologie

Jsou zde dva hlavní způsoby, kterými je možné AR realizovat: optické a video technologie [4]. Rozdíly mezi nimi, jejich výhody i nedostatky jsou popsané v následujících odstavcích.

Optické HMD jsou předměty ke kombinování virtuálního a reálného. Jsou to Head-Mounted Displaye, které umožňují vidět reálné prostředí, do kterého v reálném čase vkládají virtuální předměty. Video HMD neumožňují uživateli skrz sebe vidět reálný svět.

Optické HMD vkládají propustné kombinátory před oči uživatele. Tyto kombinátory umožňují uživateli vidět skrz ně i reálný svět. Jsou také částečně reflekční, takže uživatel vidí virtuální předměty v odraze z monitorů na hlavě. Optické kombinéry jsou vlastně polopropustná zdcadla, čímž redukují množství světla, které z reálného prostředí propustí, aby mohly do výhledu uživatele pustit i světlo z monitorů. Některé kombinátory zase

propouští světlo pouze o určitých vlnových délkách. Když je přístroj vypnutý, chová se jako sluneční brýle.

Video HMD fungují na principu kamer přidělaných na hlavě. Kamery nahrazují pohled na reálný svět. Video z těchto kamer je kombinováno s grafickými obrazy vytvořenými generátorem scén, kombinující reálné a virtuální.

Složení videa může být provedeno několika způsoby. Jedním z jednoduchých způsobů je chroma-klíčování. V počítačovém grafickém prostředí, kde se vyskytují vytvořené virtuální předměty, je pozadí nastaveno na specifickou (nejčastěji zelenou) barvu, kterou na sobě žádný z virtuálních předmětů nemá. Pak je na zelenou plochu promítnuto video z nahrávky reálného světa, čímž se obrazy složí. Sofistikovanější způsoby využívají informaci o hloubce prostředí, což umožňuje i překrývání virtuálních objektů reálnými.

Dalším způsobem je zobrazování rozšířené reality na monitoru počítače, tabletu nebo mobilního telefonu. V tomto případě pohyblivé nebo statické kamery nahrávají prostředí, generátor scény je pak zkombinuje s virtuálními objekty a výsledek promítne na monitoru. Uživatel nemusí mít na hlavě zobrazovací zařízení, stačí mu dívat se na obrazovku. V praktické části této práce bude zkoumána právě možnost využít k AR kameru.

1.3.1 Výhody optických technologií

Optické směšování obrazů je jednodušší a levnější. Má totiž jen jeden proud videa, virtuální obrazy. Směšování video technologií pracuje s videem reálného světa i virtuálními objekty, což může navodit časové zkreslení, a oba proudy musí být správně synchronizované. Optické technologie se tomuto problému vyhnou.

Optické systémy se potýkají s mírným zkreslením vzniklým průchodem obrazu optickou soustavou bez jeho možnosti úprav. Jejich celková implementace je však jednodušší.

Video směšování limituje to, co uživatel vidí, podle rozlišení zobrazovací jednotky. U optického jsou ovlivněny jen virtuální obrazy, protože uživatel vidí reálný svět bez pomoci videí.

Optické HMD jsou bezpečnější, protože bez přísunu energie uživatel stále vidí realitu, na rozdíl od video HMD, které toto neumožňuje. Když je vypnuto, nelze přes něj vidět.

U video technologií dochází k "posunu" očí. Kamery totiž nejsou na úplně stejném místě, jako oči uživatele, proto se zdá, že uživatel vidí svět z trochu jiné perspektivy, než je zvyklý. Tato nevýhoda se musí opravit pomocí zrcadel, která posunou optický tok do uživatelových očí. Tento posun ale není problémem pro optické displeje. Oko se může otáčet vzhledem k pozici HMD a chyby výsledků jsou malé.

1.3.2 Výhody video technologií

Základní problém optických technologií je, že virtuální objekty nezakryjí dokonale ty reálné, protože optické kombinéry propouští světlo z virtuálního i reálného zdroje. Je složité udělat optické HMD, které by světlo propouštělo jen selektivně. Reálný svět je zaměřen jen na jeden bod optické cesty, na oko uživatele, neprochází tedy žádnými filtry a tak podobně. Proto zde musí být dvě místa, kde je obraz v zaměření, lidské oko a hypotetický filtr. Toto činí optický design mnohem náročnější a komplexnější. Vznikají pak další problémy, jako například nejasnost a průhlednost virtuálních předmětů. Video technologie je v tomto ohledu mnohem flexibilnější, reálné i virtuální obrazy má v digitální podobě, čímž se většiny z těchto problémů zbavuje.

Zkreslení u optických systémů je funkcí radiální vzdálenosti od optické osy. Čím dál od centra obrazu se uživatel dívá, tím větší je zkreslení. Z optických systémů je možné

toto zkreslení odstranit, ale vyžaduje to spoustu výpočtů. S optickými technologiemi je složitější vytvořit širokoúhlý displej. Všechna zkreslení musí být napravena opticky, ne digitálně, protože systém nemá k manipulaci digitalizovaný obraz reálného světa.

Video technologie nabízí způsob, jak redukovat nebo se vyhnout problémům způsobených dočasnými časovými nesoulady mezi reálnými a virtuálními obrazy. Optické HMD poskytují okamžitý pohled na reálný svět, ale pohled na ten virtuální mívá zpoždění. U video technologií je možné posílat se zpožděním video reálného světa, aby bylo sehrané s tím virtuálním. Toto může představovat komplikace například při pohybu uživatele, ale pro některé aplikace to je výhodou.

Optické technologie mají jen jednu informaci o poloze uživatelovy hlavy a to z head trackeru. Video technologie mají k dispozici digitalizovaný obraz reálného světa, ze kterého lze lokaci určovat také. U video technologií je také snazší sjednotit jas reálných a virtuálních obrazů.

1.4 Zaostření a kontrast rozšířené reality

Zaostření může být problémem pro oba typy technologií. Ideálně se má reálné s virtuálním shodovat. Na videu založené systémy kombinují virtuální a reálné obrazy a vkládají je do stejné vzdálenosti na monitor nebo HMD optiku. V závislosti na hloubce ostrosti kamery ale nemusí být některé části reálného světa ostré. Na řešení tohoto problému existují metody, které umožňují zaostřit všechny objekty nezávisle na vzdálenosti. Také se používají samozaostřovací čočky.

U optických technologií je svět viděn přesně tak, jak je ve skutečnosti, ale virtuální předměty jsou všechny promítány do stejné vzdálenosti. To může způsobovat, že není možné obojí vidět ostře zároveň.

Problémy s kontrastem vznikají, protože se zobrazovací prostředky nemohou vyrovnat svým rozsahem lidskému oku a jeho vnímání světa. Pro optické systémy je to problematičtější. Když je okolní prostředí příliš světlé, může dojít ke ztracení virtuálních obrazů a když je reálné prostředí příliš tmavé, může být zastíněno virtuálními objekty.

1.5 Porovnání s virtuální realitou

- Přenosnost

  U virtuální reality musí člověk zůstávat na omezeném místě, nemá příležitost se volně pohybovat v prostoru. Některé AR aplikace však podporují uživatele, který chodí v rozlehlém prostředí. Generátor scény, HMD a sledovací systém musí být samostatné a schopné přežít vystavení prostředí.

- Generátor scény

  Vykreslování není pro AR hlavním problémem. VR systémy mají mnohem větší požadavky na vytváření realistických obrazů, protože naprosto nahrazují realitu virtuálním prostředím. V AR virtuální obrazy realitu pouze doplňují. V mnoha aplikacích ani nemusí být příliš realistické, aby posloužily účelu.

- Zobrazovací zařízení

  Tady má AR opět méně přísné požadavky než VR. Rozlišení monitoru v optickém HMD může být nižší, než by uživatel toleroval v aplikaci VR, protože optický HMD nezmenšuje rozlišení skutečného prostředí.

- Sledování a snímání

  V tomto případě jsou vyšší a přísnější požadavky naopak kladeny na AR. Virtuální reality se tento problém týká jen v případě sledování pohybu hlavy uživatele, aby se posuny úhlu pohledu měnily i ve VR. AR ale musí sledovat přesnou polohu, aby bylo schopno vkládat virtuální předměty přesně na svá místa, jinak by se mohlo stát, že by stůl levitoval uprostřed místnosti. Tento problém je blíže popsán v následující části.

1.6 Registrace

Jedním ze základních problémů limitujících AR je registrace [5]. Je důležité, aby bylo virtuální a reálné správně postaveno a navazovalo na sebe, tzn správné propojení reálného a virtuálního světa. Jinak je ohrožena iluze koexistence těchto dvou světů. Některé aplikace přímo vyžadují naprostou přesnost. Například v případě biopsie musí lékař přesně vědět, kam udělat jehlou vpich.

Chyby registrací pro AR je složité adekvátně kontrolovat kvůli vysokým nárokům a mnoha možným zdrojům chyb. Tyto zdroje mohou být rozděleny do dvou skupin: statické a dynamické. Statické způsobují chyby, když je prostředí i uživatel v klidu. Dynamické nemají žádný efekt, dokud se objekty nezačnou hýbat.

1.6.1 Statické chyby

Hlavní zdroje chyb jsou: optické zkreslení, chyby sledovacího systému, mechanické posunutí a nesprávné sledovací parametry. Jsou blíže popsány v následujících odstavcích.

Optické zkreslení

Optická zkreslení jsou přítomna ve většině kamerových čoček i v optice používané k zobrazování. Zkreslení zatěžuje především širokoúhlé displeje. Objekty kolem centra bývají bez zkreslení, ale kraje mohou být výrazně zkreslené. Například rovné čáry mohou vypadat pokřiveně. V optických HMD s úzkým zobrazováním zorného pole kombinátory nepřidávají téměř žádné zkreslení. Pohled uživatele na reálný svět tedy není deformován.

Optika používaná k zaostřování a zvětšování grafických obrazů z monitorů může také způsobit zkreslení. Mapování zkreslených virtuálních obrazů překrývajících pohled na skutečný nezkreslený svět způsobuje statické registrační chyby. Kamery a displeje mohou mít také nelineární zkreslení.

Většinou se jedná o systémové chyby, které mohou být zmapované a vykompenzované dodatečnou optikou. Tohoto je možné dosáhnout přidáním potřebné optiky na HMD, čímž se zvýší její váha, nebo použít digitální kompenzace. Toho může být docíleno použitím deformačních technik na digitalizovaná videa i na grafické obrazy. Jeden ze způsobů je rozložení obrazů tak, aby se po zobrazení objevily nedotčené.

Chyby sledovacích systémů

Chyby výstupů ze sledovacích a snímacích systémů jsou většinou nejzávažnějším typem registračních chyb. Tato zkreslení nelze snadno měřit a eliminovat, kvůli potřebě dalšího "3-D pravítka" přesnějšího, než je testovaný sledovač. Tyto chyby nebývají systematické, je složité je plně charakterizovat.

Mechanické posunutí

Jde o nesrovnalosti mezi modelem nebo hardwarovou specifikací a reálnými fyzikálními vlastnostmi systému. Mechanická posunutí mohou působit nechtěné změny v umístění nebo rotaci virtuálních objektů, což je těžké kompenzovat. Některé chyby mohou být kalibrovány, ale někdy je efektivnější postavit objekty znovu a správně.

Nesprávné parametry prohlížení

Tyto chyby je možné považovat za speciální případ chyb sladění, které se dají kalibrovat. Parametry prohlížení určují, jak převést hlášenou polohu hlavy nebo kamery do zobrazovacích matic používaných generátorem scén pro kreslení obrazů. Pro HMD systém tyto parametry zahrnují:

- centrum projekce a rozměry výřezu
- posun jak v překladu, tak v orientaci mezi umístěním sledovače hlavy a uživatelovýma očima
- zorné pole

Nesprávné parametry prohlížení způsobují statické chyby. Například, když jsou kamery umístěny nad očima uživatele a upravení posunutím není dostatečné, předměty se zdají být níž. Tyto chyby je možné upravovat manuálně, mimo systém, ale ne vždy jsou výsledky dostatečně robustní pro všechny úhly pohledu. K tomu je zapotřebí veliké množství parametrů.

Další metodou je přímé měření parametrů za použití různých metod. Jednou z možností je měření interpupilární vzdálenosti. Pravítka mohou měřit rozdíl mezi sledovačem a okem uživatele. Díky tomuto je možné získat různé pozorovací parametry. Možné je také nastavení geometrických konstant. Uživatel provádí několik úkolů, čímž je shromážděn dostatek informací k určení sledovacích parametrů.

U systémů založených na videu pak jde především o techniky sloužící ke kalibraci kamer. Takové techniky získávají pozorovací parametry opakovaným snímáním objektu z několika směrů.

1.6.2 Dynamické chyby

Objevují se kvůli systémovému zpoždění nebo zaostávání. Systémové zpoždění end-to-end je časový rozdíl mezi okamžikem, kdy je měřena poloha a orientace, a okamžikem, kdy se obrazy generované na základě naměřených hodnot promítnou na displejích. Dochází k němu, protože každá část systému potřebuje čas na splnění své části. Zpoždění sledovacího zařízení, komunikace generátoru obrazů, zobrazování obrazů na displeje, to vše přispívá do end-to-end zpoždění. Chyby při registraci způsobuje ale pouze při pohybu. Například při rozhlížení se předměty mohou vykreslovat zpožděně, což znamená, že je uživatel uvidí objevovat se na místech odlišných od těch, kam umístěny být měly. Systémová zpoždění velmi narušují iluzi koexistence reálného a virtuálního, protože působí velké registrační chyby. Metody používané ke snižování dynamických registračních chyb se dělí do čtyřech hlavních kategorií:

- Redukce systémových zpoždění
- Redukce zjevných zpoždění
- Shodnost časových toků
- Předvídání budoucích lokací

Redukce systémových zpoždění

Nejpřímější přístup je rovnou redukovat nebo ideálně eliminovat systémová zpoždění. Většina moderních generátorů scény je ale postavena hlavně pro průchodnost, ne minimální latenci. U některých softwarů je možné omezením průchodnosti latenci minimalizovat.

Redukce zjevných zpoždění

Vychýlení obrazu je dobrá technika snížení viditelných zpoždění především pro systémy využívající jen orientaci hlavy. Jde o způsob vložení aktuálnějších výsledků měření orientace do pozdější fáze vykreslování. Je to tedy dopředná technika.

Shodnost časových toků

V systémech AR založených na videu videokamera a digitalizační hardware ukládají přirozené zpoždění pohledu uživatele na reálný svět. Toto je výhodné pro redukci dynamických chyb, protože umožňuje přizpůsobit časové toky reálných a virtuálních obrazů. Dodatečné zpoždění je přidáno do videa reálného světa, aby bylo sjednoceno s videem z generátorů virtuálních obrazů. Protože zpoždění není časově konstantní, musí systém oba toky dynamicky synchronizovat.

Tato metoda ale působí, že reálné i virtuální objekty se zobrazují se zpožděním. Pro malé časové posuny to není velký problém, ale s narůstající dobou mohou nastávat výrazné problémy.

Předvídání

Poslední metodou je předvídání budoucího úhlu pohledu a lokací objektů. Pokud jsou budoucí lokace známy, scéna může být vykreslena spíše pomocí předvídaných lokalit než naměřených míst. Když se scéna objeví, úhel pohledu a objekty se posunou do předem určených lokací a obrazy jsou na správném místě ve správný čas. Přesné predikce požadují systémy vybudovány pro měření a výpočty v reálném čase. Pro zpřesnění se využívá vnějších senzorů.

1.6.3 Techniky založené na vidění

Registraci založenou pouze na informacích ze sledovacího systému je možné přirovnat k regulátoru otevřené smyčky. Systému chybí zpětná vazba o skutečné shodě reálného a virtuálního prostředí. Bez ní je složité dosáhnout perfektní shody. Techniky založené na videu však mohou využívat zpracování obrazu a počítačové vizuální techniky na podporu registrace. Díky digitalizovanému obrazu reálného světa je možné v prostředí detekovat prvky a ty využívat k vynucení registrace. Toto se již dá považovat za uzavřenou smyčku, protože digitalizovaný obraz poskytuje mechanismus pro dodání zpětné vazby do systému.

Není to ale snadné. Tato detekce a shoda musí běžet v reálném čase a být robustní. To často vyžaduje speciální hardware a senzory. Jedna z používaných technik je využití referenčních značek (barevné tečky, LED světla nebo speciální markery). Systém zná

umístění těchto značek. Ty se umístí do prostředí a podle nich se provádí porovnávání a korekce. Vždy musí být viditelný alespoň jeden bod. Tato technika vede v velmi přesným výsledkům.

Místo referenčních značek je k dosažení registrace možné využít odpovídajících šablon. Odpovídající obrazy reálných objektů jsou nasnímány z několika úhlů. Poté jsou využity k nacházení reálných objektů v digitalizovaných obrazech. Jakmile je nalezen, může být překryt virtuálním obrazem.

Jiné přístupy na videu založeném přizpůsobování vylučují potřebu kalibrace. Objekty představují v neeuklidovském afinním referenčním rámci, který umožňuje vykreslování bez znalosti parametrů kamery. Další možností je extrahování kontur z videa reálného světa a použitím optimalizační techniky je sjednotit s vykreslovanými 3D virtuálními objekty.

Tyto dvě metody ale neposkytují přesné informace o hloubce, což může působit problémy při přesném vykreslování. Metody používající referenční značky zase poskytují pouze relativní projektivní vztah mezi objekty a videokamerou. Informace je postačující pro registraci, ale neposkytuje všechny informace pro potřebu různých AR aplikací, například absolutní umístění kamery a objektů.

Pro snížení složitosti problémů je možné přidat dodatečné senzory. Jiná cesta je akceptovat fakt, že systém nebude naprosto robustní a nemusí všechny úkony plnit automaticky. V tomto případě může systém po uživateli požadovat provedení nějakého úkonu. Může tedy požadovat nějaké manuální zákroky, například, když je výhled něčím zastíněný.

1.6.4 Snímání

Přesná registrace a polohování virtuálních objektů do reálného prostředí vyžadují přesné sledování uživatelovy hlavy a snímání polohy ostatních objektů v prostředí. Největší překážkou pro budování AR systémů je potřeba přesných senzorů s dlouhým dosahem a sledovačů, které nahlásí polohu uživatele a okolí. AR vyžaduje v porovnání s VR od senzorů a sledovačů více především ve třech oblastech:

- větší rozmanitost vstupů a šířka pásma
- vyšší přesnost
- větší vzdálenost

Větší rozmanitost vstupů

AR systémy potřebují mnohem větší rozmanitost vstupů a širší pásmo. Výstupy jsou limitovány pěti lidskými smysly. Vstupy mohou přijít ze všeho, co senzory mohou detekovat.

Rozsah dat je speciální část vstupu, která je pro mnoho AR aplikací důležitá. AR systém sice zná vzdálenost od virtuálních objektů, ale nemusí vědět, kde v prostředí jsou umístěny reálné předměty. Systém může předpokládat, že je prostředí statické, takže mu stačí pouze jedno měření na začátku. Často je ale potřeba obsáhnout i dynamická prostředí, kde se předměty pohybují, proto musí být sledovány v reálném čase. V těchto případech se často využívá hloubkové mapy. Ta umožňuje provádět porovnání hloubky reálných a virtuálních objektů pixel po pixelu. Získání hloubkové mapy v reálném čase však není triviální. Mohou být například použity senzory jako laserové dálkoměry. Existuje ale i spousta metod počítačového vidění, které dokáží danou úlohu splnit.

Nakonec mohou být jako vstupy považovány i databáze prostředí, například databáze umístění potrubí v budově, umístění kabelů a drátů a tak dále.

Vysoká přesnost

Ve většině případů je registrace přesná pouze tolik, jako sledovač. Takže je zapotřebí sledovače s přesností na milimetr a malý zlomek stupně po celém jeho rozsahu. Je zde několik typů sledovačů a každý má nějaké nevýhody, které jeho přesnost snižují. Některé mechanické sledovače jsou velmi přesné, ale omezují rozsah prostředí. Magnetické sledovače jsou k nepřesnostem náchylné kvůli výskytu kovu v prostředí. Ultrazvukové sledovače jsou zatíženy šumem a je obtížné dosáhnout požadovanou přesnost na dlouhé vzdálenosti kvůli změnám teploty v prostředí.

Optické technologie mají problémy se zkreslením a kalibrací. Jsou ale nejslibnější, díky výkonným kamerám s vysokým rozlišením. Nejčastěji se ale používají hybridní systémy, kombinace inerciálních a optických technologií.

Dlouhý dosah

Příliš sledovačů není vybudováno pro dlouhý dosah, jelikož ho VR aplikace nevyžadují. Aplikace zachycování pohybu se využívají například při přenesení pohybu herce na kreslenou postavičku. To je dobré pro určení pozice, nikoli orientace. Určení pozice je založeno na vypočtených pozicích. Jen malé odchylky v těchto výpočtech mohou způsobit posun i o několik stupňů, což je pro AR příliš.

Rozšiřitelný systém lze rozvinout tak, aby pokrýval libovolný požadovaný rozsah, přidáním více modulárních komponent. Udělá se celulární sledovací systém, kde uživatele sledují pouze blízké zdroje a senzory. Jak se uživatel pohybuje po okolí, mění se i set senzorů a zdrojů, tím pádem se odstraní problém velké vzdálenosti mezi současně pracujícími zdroji a senzory.

Někdy se pro AR používá i GNSS (global navigation satelite system), ale její výsledky nejsou natolik přesné, aby vždy vyhovovaly.

1.7 Přístup k AR v této práci

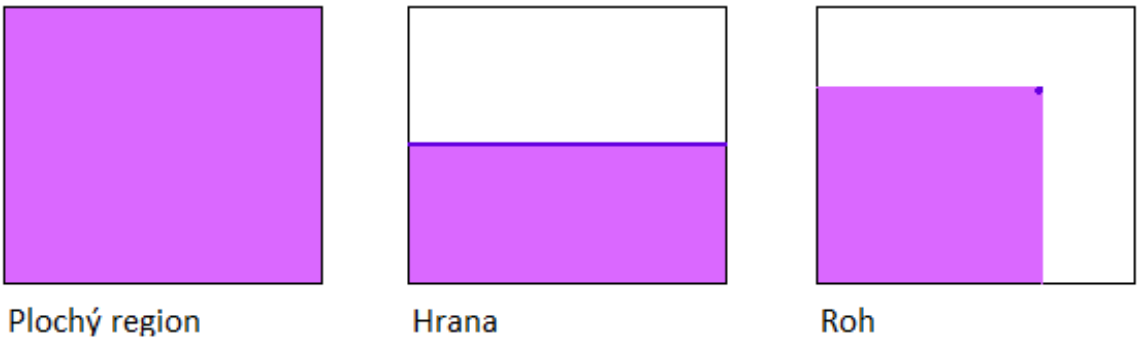
V této práci budu k rozšířené realitě přistupovat dvěma cestami. Pro aplikaci vizuální odometrie (— viz kapitola 4) využiji cestu, která nevyužívá značky, což je náročnější úloha, než když je prostředí označkováno předem. Složitější je především proto, že si musí vyhledávat významné body, které nejsou předem dané, v obrazech z prostředí a to v reálném čase. Systém musí být také robustní vůči větším, rychlejším změnám, například když uživatel otáčí hlavou, nebo zařízením s AR. V této práci nebudu využívat HMD, ale přenosnou webkameru.

Pro vykreslování 3D objektů do prostředí poté použiji značky v podobě šachovnice a aruco kódu. AR využívající značky v obraze pouze vyhledá daný marker a na něj umístí virtuální 3D objekt. Geometrie objektu je řízena geometrií markeru, což má za následek, že jakmile je marker před kamerou zakryt, zmizí i objekt. Jakmile je zase viditelný, objekt se objeví.

V AR, které značky nevyužívá, se geometrie objektu řídí geometrií vytvořenou takzvaným SLAM (současná lokalizace a mapování — viz kapitola 5), který v prostředí hledá významné body a vytváří 3D reprezentaci prostředí. Systém si pak prostředí pamatuje jako 3D model, do něhož vkládá virtuální objekt.

2 Techniky zpracování obrazů

Pro orientaci počítače v prostoru je nutné nalézt významné body. Obraz sám je možné rozdělit na ploché regiony, hrany a rohy — viz Obrázek 6. Ploché regiony jsou nevhodné k nalezení a sledování, protože se nedá s jistotou říci, v jaké části plochy se právě nacházíme. Hrany mají pro sledování lepší vlastnosti než plochy, ale přesto nejsou úplně ideální. Když okno sledující hranu po této hraně popojede dál, sledování nemůže přesně určit, jestli jsme stále na stejném místě či ne. Rohy jsou pro hledání a sledování nejvhodnější. Při posunu po obraze je jisté, že se v rohu již nenacházíme, jak je také zřejmé z Obrázku 6. Rohy v obraze jsou tedy významnými body, v nichž existují dva dominantní směry. Významným bodem však nemusí být pouze roh, může se jednat i o bod nebo oblast dostatečně jasově odlišenou od svého okolí.



Obrázek 6: Části obrazu

V následujících podkapitolách budou popsány jednotlivé techniky [6] založené na hledání právě těchto dominantních bodů.

2.1 Detektory vs deskriptory

Detektory [7] vlastností jsou algoritmy, které vezmou zadaný obraz a jako výstup vrátí lokace významných oblastí nacházejících se v něm. Mezi ně patří například detektor rohů, který vrací polohy těchto konkrétních významných bodů. Detektory však nepodávají žádné bližší informace o nalezených bodech.

Deskriptory [8] vlastností jsou algoritmy, které nejen detekují významné oblasti, ale vracejí i jejich deskriptory, neboli vektory vlastností. Tyto deskriptory obsahují geometrii oblasti zakódovanou do vektoru, který je možné použít k rozlišení jednotlivých významných bodů. Ideálně by tyto informace měly být invariantní vůči transformaci.

2.2 Moravcův detektor rohů

Algoritmus testuje podobnost okolí pixelu. Podobnost se měří pomocí sumy absolutních rozdílů. Malá suma se rovná velké podobnosti. Rohy jsou lokální maxima sum absolutních rozdílů. Tento detektor využívá pravoúhlého okénka, které definuje lokální okolí. Vybere ze směrů ten, který dává minimální odezvu, což znamená největší shodu, funkcí:

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \tag{1}$$

kde I je obraz a w pravoúhlé okénko.

Takto se získá pro každý pixel hodnota, z nichž se pak pomocí potlačení nemaxim vyberou lokální maxima.

2.3 Harrisův detektor rohů

Nahrazuje pravoúhlé okénko, které využíval Moravcův detektor, Gaussovským, což vede k potlačení šumu. Diskretizace analyzovaných směrů je uskutečněna drobnými posuny, jež jsou aproximovány Taylorovým rozvojem. Výpočtem pro každý bod v obraze získáme matici M, nazývanou Harrisovou maticí.

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_u^2(x,y) & I_u I_v(x,y) \\ I_u I_v(x,y) & I_v^2(x,y) \end{bmatrix} \tag{2}$$

Tu dále použijeme pro výpočet:

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \tag{3}$$

Další analýzou Harrisovy matice zjistíme, o jaký typ okolí pixelu se jedná. Okolí rozdělejume na plochý region, hranu a roh. Vlastní vektory matice M odpovídají směru hlavní a vedlejší poloosy elipsy, reprezentující konturu Gaussovského rozložení s kovarianční maticí M. Vlastní čísla popisují jejich velikost a podle nich a tvaru elipsy je možné určit jestli se jedná o roh, hranu nebo plochý region.

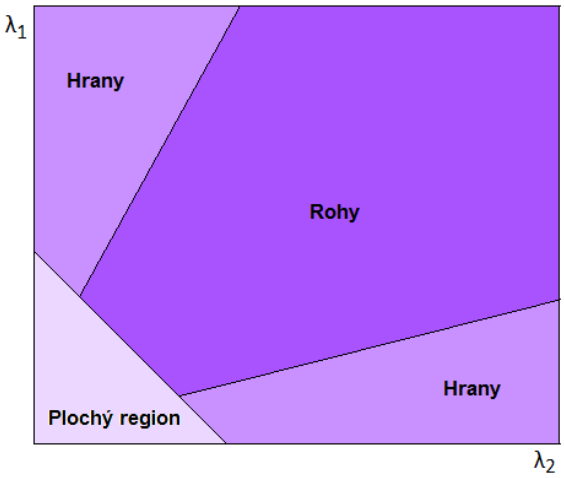
Pro hranu je jeden ze směrů výraznější než druhý. Jinými slovy, jedno z vlastních čísel je výrazně větší než druhé. V případe, že se čísla shodují, záleží na jejich absolutní velikosti. Pokud jsou malá, jedná se o plochý region, pokud ne, pak jde o roh.

Výpočet vlastních čísel je náročný, proto se používá následující aproximace:

$$R = det(M) - k(trace(M))^2 \tag{4}$$

kde k je konstanta a nabývá hodnot mezi 0.04 až 0.1. Z výsledné hodnoty R můžeme vyčíst, o jaký případ se jedná. Tuto závislost typu detekovaného prvku na poměru vlastních čísel je možné vidět na grafu z Obrázku 7.

- Při hodnotě R > 10000 se jedná o roh
- při R < -10000 o hranu
- a v případě -10000 < R > 10000 o plochý region.



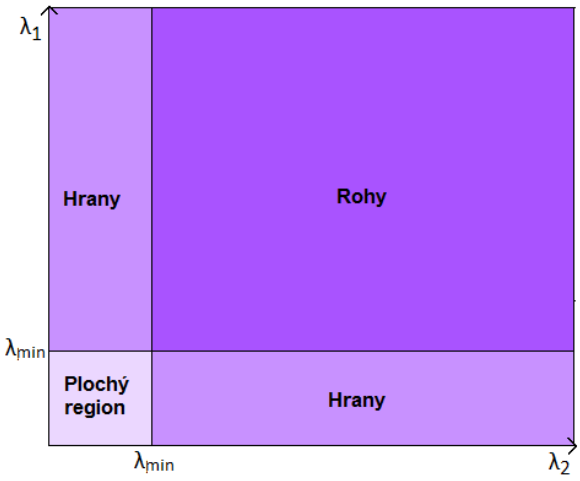
Obrázek 7: Graf pro Harrisův detektor

2.4 Shi-Tomasi detektor rohů

J. Shi a C. Tomasi modifikovali Harrisův detektor rohů ve své práci Good Features to Track. Tato modifikace má lepší výsledky než samotný originál. Výpočet hodnoty R z Harrisova operátoru je v tomto případě nahrazen:

$$R = min(\lambda_1, \lambda_2) \tag{5}$$

Když je R větší než prahová hodnota, pak je bod považován za roh. Kdybychom funkci vykreslili v $\lambda_1 - \lambda_2$ prostoru, jak se to dělá u Harrisova detektoru, dostali bychom obraz, ze kterého je zřejmé, že λ_{min} detekuje roh pouze, když jsou λ_1 a λ_2 větší než minimální hodnota. Na Obrázku 8 je opět zobrazena závislost typu regionu na poměru vlastních čísel, jako tomu bylo v případě Harrisova detektoru.



Obrázek 8: Graf pro Shi-Tomasi detektor

2.5 Scale-Invariant Feature Transform (SIFT)

Harrisův i Shi-Tomasi detektor nalezly stejné rohy i při rotaci obrazu. V malém obrázku může být určitý bod identifikovaný jako roh, ale po zvětšení už ne. Tyto detektory tedy nejsou škálově invariantní.

Na to reagoval výzkum zveřejněný v článku D.G.Lowe [9] představením nové metody nazvané Scale-Invrariant Feature Transform (SIFT). Metoda extrahuje klíčové body a vypočítá jejich deskriptory pomocí následujících kroků:

1. Detekce extrémů za použití prostorového měřítka

   Pro identifikaci klíčových bodů s různým měřítkem není možné použít stejné okno. K detekci větších rohů je třeba větších oken. Proto se používá filtrování prostorového měřítka. Pro obraz je nalezen Laplacián Gaussiánu s různou hodnotou σ. která může být považována za škálovací parametr. Gaussovské jádro s větší hodnotou σ odpovídá většímu rohu, zatímco s menší σ rohu menšímu.

2. Lokalizace klíčových bodů

   Když jsou nalezeny lokace klíčových bodů, k získání přesnějších výsledků je potřeba je vylepšit. K upřesnění polohy extrémů se používá rozvoj Taylorovou řadou ke zvětšení rozsahu prostoru. Pokud je intenzita tohoto extrému menší než práh, vyřadí se.

   Diference Gausiánů má pro okraje větší odezvu, takže okraje musí být také odstraněny. K tomuto účelu se používá Harrisův detektor rohů. V tomto kroku se tedy odstraní všechny klíčové body s nízkým kontrastem a hrany, takže zůstávají pouze silné zájmové body.

3. Přiřazení orientace

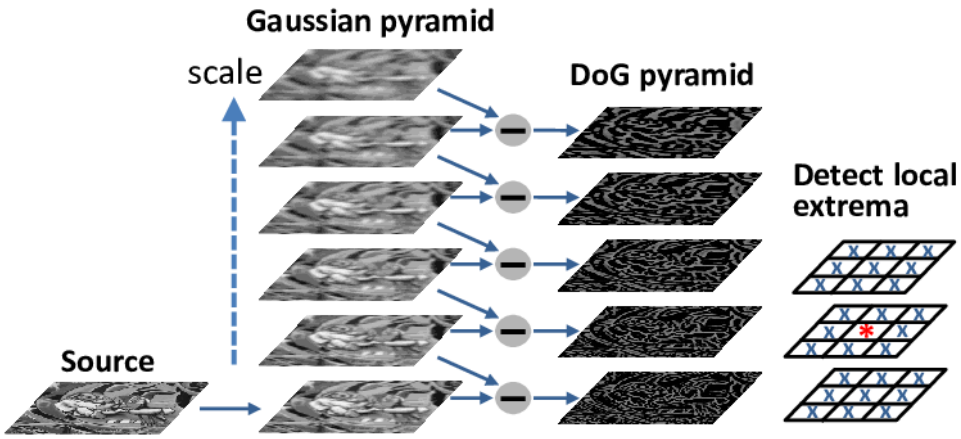
   Každému klíčovému bodu je přidělena orientace k dosažení invariance rotace obrazu. Kolem bodu je v závislosti na měřítku vzato sousední okolí a vypočítá se velikost a směr jeho gradientu. Vytvoří se orientační histogram o délce 36 polí pokrývající 360 stupňů. Je vzat nejvyšší hrot histogramu a do výpočtu orientace jsou zahrnuty i ostatní hroty nad 80%. Díky tomuto získáme klíčové body se stejným měřítkem a lokací, ale odlišnými směry.

4. Deskriptor klíčových bodů

   Kolem klíčového bodu je vzato okolí 16×16, které je rozděleno na 16 dílčích částí 4×4. Pro každou je vytvořen histogram orientace o délce 8 polí. Výsledných 128 hodnot je reprezentováno jako vektor ke zformování deskriptoru klíčových bodů.

5. Shoda klíčových bodů

   Klíčové body mezi dvěma obrazy jsou získávány pomocí porovnávání jejich nejbližších sousedů. V případech, kdy je kvůli vlivu šumu nebo rušení druhá nejlepší shoda velmi blízko té první, je nutné vypočítat poměr mezi vzdálenostmi. Je-li větší než 0.8, jsou tyto body odstraněny a tím je většina falešných shod odstraněna, aniž by tato metoda nesprávně eliminovala shody správné.



Obrázek 9: Algoritmus detekování SIFT

Na Obrázku 9 je diagram průběhu SIFT algoritmu. Je na něm vidět zdroj, pyramida Gaussiánů, pyramida rozdílů Gaussiánů a detekce lokálního extrému.

2.6 Speeded-Up Robust Features (SURF)

Další používanou metodou je SURF. V porovnání s výše uvedeným SIFT je rychlejší, jak je zřejmé z názvu. Pro aproximaci Laplaciánu Gaussiánu používá čtvercový filtr (Box Filter) — viz Obrázek 10.

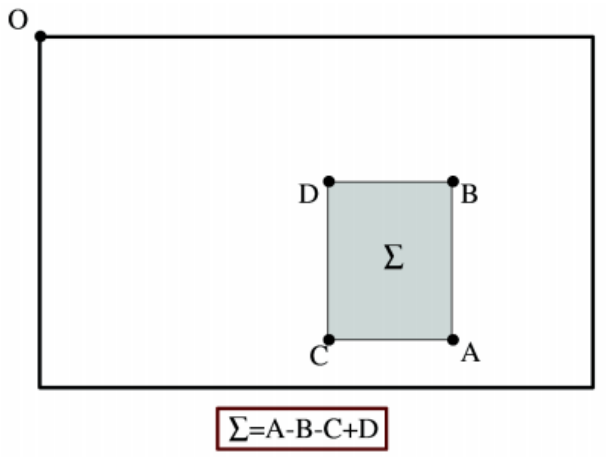
$$\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix} \qquad \frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}$$

Obrázek 10: Porovnání čtvercového (vlevo) a Gaussiánského (vpravo) filtru

Velká výhoda této aproximace je, že konvoluce čtvercového filtru je snadno vypočitatelná pomocí integrálních obrazů, což může být prováděno paralelně pro různá měřítka. SURF také závisí na determinantu matice Hesiánu pro škálu i lokaci, který získáme aplikací právě integrálních obrazů, což je způsob reprezentace obrazu tak, že každý bod x představuje součet hodnot všech pixelů, nacházejících se od toho současného vlevo a nahoře. Matematický zápis popisuje následující rovnice.

$$I_{\Sigma}(x,y) = \sum_{i=0}^{i\leq x}\sum_{j=0}^{j\leq x} I(i,j) \tag{6}$$

Na Obrázku 11 je pak možné vidět integrální obraz a pod ním vzorec pro výpočet plochy.



Obrázek 11: Integrální obraz

SURF využívá pro přiřazení orientace horizontální a vertikální vlnové odezvy pro okolí velikosti, která je určena při implementaci. Připočte také adekvátní Gaussovské váhy. Poté se vykreslí do prostoru a dominantní orientace se odhaduje pomocí sumy všech odezev v posuvném okně o úhlu 60°.

Výhodami SURFu je, že vlnové odezvy mohou být pomocí integrálních obrazů zjištěny velmi snadno i ve velkém měřítku, často ani není vyžadována rotační invariance. SURF proto poskytuje funkcionalitu zvanou Upright-SURF nebo U-SURF. Proces se tedy zrychluje a je robustní až do 15°.

Pro výpočet je vzato okolí klíčového bodu o předem určené velikosti. Je rozděleno do dílčích částí. Pro každou menší část jsou zjištěny horizontální a vertikální vlnové odezvy a jsou zaznamenány do vektoru v. Když jsou výsledky takto zaznamenány, poskytují SURF detektor vlastností s šedesáti čtyřmi dimenzemi.

Existuje i verze SURF detektoru se 128 dimenzemi, který má větší rozlišovací způsobilost, ale zároveň není příliš zvětšena výpočetní složitost. Rozdíl spočívá v tom, že sumy d_x a $|d_x|$ jsou počítány zvlášť pro $d_y < 0$ a $d_y \geq 0$, kde d_x a d_y jsou zaznamenané odezvy. Obdobně je tomu i pro d_y a $|d_y|$, které závisí na hodnotě d_x.

Dalším vylepšením je použití znaménka Laplaciánu pro zvýraznění základních bodů zájmu. Žádné další výpočetní náklady nejsou přidány, protože jsou vypočteny už během detekce. Znaménko Laplaciánu rozlišuje světlá místa od tmavého pozadí a během porovnávání je pozornost věnována pouze funkcím, které mají stejný typ kontrastu. Tyto minimální informace urychlují zjišťování shod bez snižování výkonnosti deskriptoru.

Shrnutí: SURF je až třikrát rychlejší než SIFT, přičemž výkon je porovnatelný. Je také dobrý při práci s obrazy s rozmazáním nebo rotací, ale změna úhlu pohledu a osvětlení mu dělá problémy.

2.7 Features from Accelerated Segment Test (FAST)

Jak už název napovídá, FAST je rychlý algoritmus používaný pro detekci vlastností obrazu. Díky své rychlosti je schopen úlohu zvládat v reálném čase a je tedy vhodný pro mnoho aplikací. Je založen na prozkoumávání okolí pixelu a porovnávání intenzit.

Zde je popis jednotlivých kroků algoritmu:

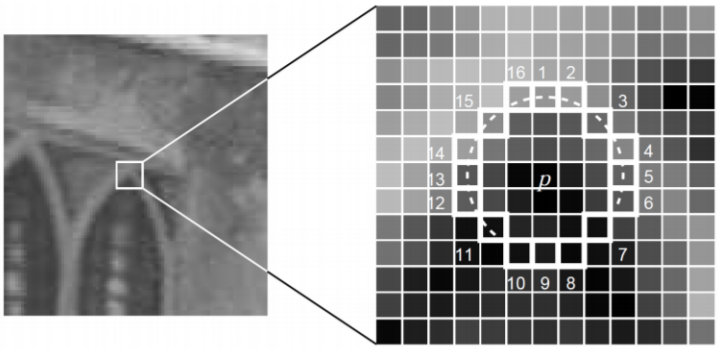
1. V obraze se vybere pixel p s intenzitou I_p.

2. Zvolí se prahová hodnota t.

3. Nyní je k testu potřeba kružnice šestnácti pixelů kolem námi zvoleného pixelu p, viz Obrázek 12.

4. Pixel p je označen jako roh, existuje-li v kružnici soubor n souvislých pixelů, které jsou všechny jasnější než $I_p + t$, nebo tmavší než $I_p - t$. n bylo zvoleno jako 12.

5. Rychlý test se provede následovně. Sledují se jen pixely na pozici 1, 5, 9 a 13. Nejprve jsou testovány 1 a 9, platí-li pro oba stejná ze dvou podmínek uvedených výše. Pokud ano, zkontrolují se i body 5 a 13. Pokud je p roh, alespoň tři z těchto čtyř bodů musí být jasnější než $I_p + t$, nebo tmavší než $I_p - t$. Tímto postupem se vyřadí mnoho nerohových bodů a FAST se nemusí zdržovat zkoumáním všech šestnácti pixelů v jejich okolí.

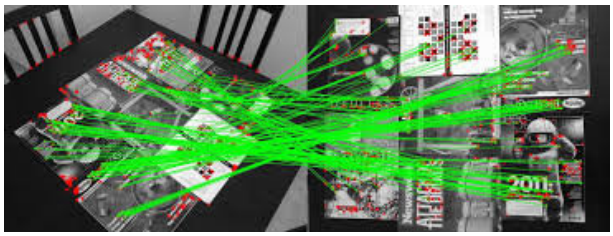


Obrázek 12: Ukázka významného bodu pro FAST

Shrnutí: Jedná se o velmi rychlý algoritmus, ale není robustní pro vysoké úrovně šumu. Záleží na zvoleném prahu, který určuje, v jakém rozsahu rozdílu jasu bude bod prohlášen za roh.

2.8 Oriented FAST and Rotated BRIEF (ORB)

ORB je dobrá alternativa k SIFT a SURF ve výpočetní ceně a odpovídající výkonnosti. Je to vlastně kombinace FAST a BRIEF (Binary Robust Independent Elementary Features) s modifikacemi pro zlepšení výkonu. Nejdříve použije FAST pro nalezení klíčových bodů a potom Harrisův výpočet hodnoty R pro vybrání N nejlepších z těchto bodů. Nalezené a spárované body je možné vidět na Obrázku 13. Používá pyramidu k dosažení větší robustnosti vůči změně měřítka. Problémem je, že FAST nepočítá orientaci. Bylo potřeba udělat modifikace, aby byla zahrnuta i rotační invariance.



Obrázek 13: Příklad shody bodů v obrazech detekovaných ORBem

ORB počítá centroid masky vážený intenzitou s rohem umístěným do středu. Směr vektoru od rohu k centroidu udává orientaci. Pro zlepšení rotační invariance se momenty počítají s x a y, které by měly být v kruhové oblasti s poloměrem r, kde r je velikost masky.

ORB využívá BRIEF deskriptor. Ten má ale problémy s rotací, takže ORB manipuluje s BRIEF podle orientace klíčových bodů. Pro sety vlastností o n binárních testech definuje $2 \times n$ matici S, která obsahuje souřadnice těchto pixelů. Je nalezena matice rotace masky a rotováním matice S se získá její správně natočená podoba.

ORB rozděluje úhel do částí po 12° a vytvoří přehled předpočítaných BRIEF vzorů. Dokud je orientace klíče napříč jednotlivými pohledy konzistentní, sada bodů v natočené matici bude využita k vypočítání jejího deskriptoru.

Důležité vlastnosti BRIEFu jsou velký rozptyl, průměr kolem 0.5 pro každou bitovou funkci a to, že má nekorelované testy, což znamená, že se každý test podílí na výsledku. ORB tedy provádí vyčerpávající hledávání mezi všemi možnými binárními testy a najde ty, které mají vysoký rozptyl, středy kolem 0.5 a které jsou nekorelované. Výsledek se nazývá rBRIEF.

Shrnutí: ORB je rychlejší než SIFT i SURF a jeho deskriptor pracuje lépe než deskriptor SURFu.

2.9 KAZE Features

KAZE Features [10] je metoda detekce a popisu vlastností, která pracuje zcela v nelineárním měřítkovém prostoru. Využívá nelineární difúzi, prostřednictvím které je možné detekovat a popsat vlastnosti v prostorech s nelineárním měřítkem a přitom zachovat důležité detaily obrazu a odstranit nechtěný šum. Vzorec pro nelineární difúzi vypadá následovně:

$$\frac{\sigma L}{\sigma t} = div(c(x, y, t) \cdot \nabla L),$$

kde c je funkce konduktivity, ∇ je gradientní operátor, a L je jas obrazu. KAZE je založen na škálově normalizovaném determinantu Hessiánu, který je počítán na více škálových úrovní. Maxima odezvy detektoru se získávají pomocí pohyblivého okýnka. Popis vlastností zařizuje rotační invarianci nalezením dominantní orientace v kruhovém sousedství kolem každého prvku.

Shrnutí: Body získané pomocí metody KAZE jsou invariantní v měřítku, rotaci, omezené afinitě a mají lepší rozlišitelnost v různých měřítkách, což ale způsobuje mírné zvýšení výpočetního času.

2.9.1 Accelerated-KAZE (AKAZE)

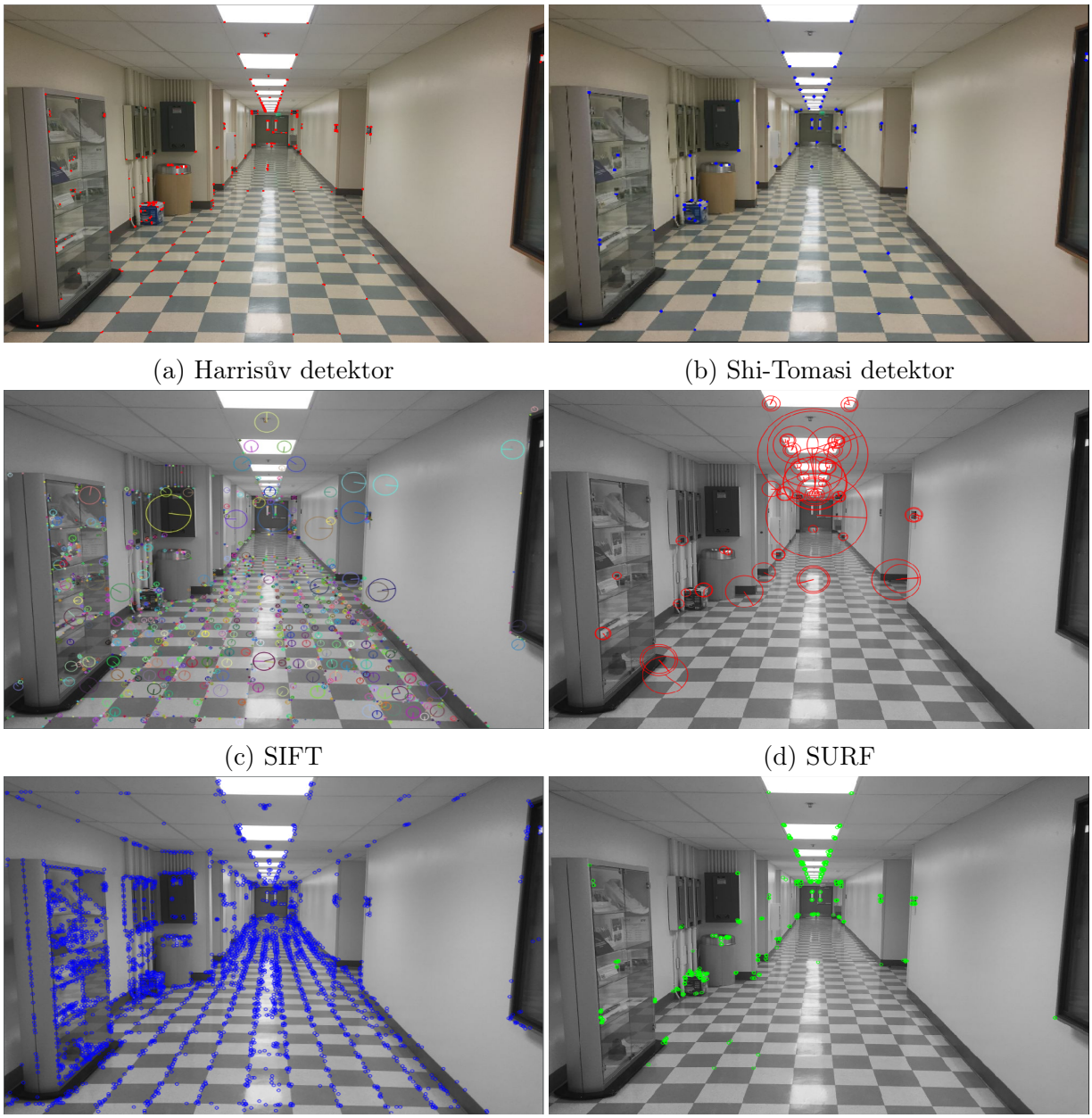
Jak už z názvu vyplývá, AKAZE je zrychlený KAZE algoritmus. Získává srovnatelné výsledky, ale je výrazně rychlejší. Využívá totiž rychlou explicitní difúzi, která výrazně urychlí výpočet pro prostor v nelineárním měřítku. AKAZE detektor je také založen na determinantu Hessiánu, ale kvalita rotační invariance je vylepšená použitím Scharrových filtrů. Deskriptor AKAZE využívá algoritmus Modified Local Difference Binary.

Shrnutí: Významné body z AKAZEjsou také invariantní v měřítku, rotaci, omezené afinitě a mají lepší rozlišitelnost v různých měřítkách. Je však výrazně rychlejší než KAZE.

2.10 Závěr

V této kapitole je uvedeno mnoho deskriptorů a detektorů, každý funguje jinak, hodí se pro něco jiného a jinde se využívá. Pro porovnání výstupů jednotlivých algoritmů je použit obrázek z webové stránky https://www.glassdoor.com/Photos/University-of-Alaska-Fairbanks-Office-Photos-IMG626018.htm. Na něm byly spuštěny Harrisův a Shi-Tomasi detektor, SIFT, SURF, FAST a ORB. Výsledky těchto algoritmů jsou ukázány na Obrázku 14.

V porovnání je možné sledovat, že Harrisův a Shi-Tomasi detektory skutečně body pouze naleznou, na rozdíl od deskriptorů jako je SIFT nebo SURF. Tyto deskriptory jsou invariantní vůči velikosti obrázku, jak je možné vidět na různých velikostech nalezených oblastí. FAST poté vyhledává největší množství bodů.



(a) Harrisův detektor (b) Shi-Tomasi detektor

(c) SIFT (d) SURF

(e) FAST (f) OBR

Obrázek 14: Výsledky jednotlivých algoritmů

3 Párování významných bodů

Pro orientaci v prostoru a další úkony vycházející z ní, je nutné udržovat povědomí o pozici konkrétních významných bodů napříč jednotlivými snímky videa. Když například systém vybere nějaký z významných bodů, na nějž umístí virtuální objekt, je nutné, aby se při pohybu kamery přesouval objekt spolu s daným bodem. K dosažení toho je nezbytné provádět párování významných bodů a k tomu je možné využít několik algoritmů. V následujících podkapitolách jsou jejich příklady.

3.1 Random sample consensus (RANSAC)

Ransac algoritmus [11] je učící technika k odhadu parametrů modelu nebo hypotéz pohybu kamery za pomoci náhodného vzorkování pozorovaných dat. Používá hlasovací schéma k nalezení optimálního výsledku, přičemž se datové prvky využívají k hlasování pro jednu nebo více hypotéz. Pro správnost hlasování jsou nutné dva předpoklady a to, že prvky nebudou hlasovat všechny pro jednu hypotézu a že existuje dostatek funkcí pro dohodu na dobré hypotéze.

Ransac vykonává dva kroky, které se neustále opakují:

- Náhodně se vybere množina prvků z datasetu. Musí být minimálně tak velká, aby bylo možno určit unikátní řešení pro pohyb kamery. Parametry se vypočítávají jen z této množiny.
- Algoritmus zkontroluje, které prvky celého datasetu jsou shodné s hypotézou vytvořenou odhadovanými parametry získanými z prvního kroku. Ta data, která nebudou odpovídat hypotéze odhadovaných parametrů v rámci chyby dané maximální odchylkou, jež vyjadřuje účinky šumu, budou označeny za takzvaný outliner. Jedná se o data, která jsou odskočená od ostatních. To znamená že nevyhovují dostatečně dobře našemu modelu.

Data, která modelu vyhovují jsou označena za inliers. Ta tvoří sadu dat shody. Algoritmus opakuje předešlé dva kroky, dokud nedostane dostatek inliers.

3.2 Metoda nejbližšího souseda

Tato metoda [12] spočívá v hledání nejlepší shody deskriptivních vektorů. Nejprve se vyhledají významné body a každý je popsán deskriptivním vektorem. Poté se vektor vybraného bodu porovná se všemi ostatními za pomoci $arg\,min\|x_i - x_j\|$ a ten s nejlepší shodou je považován za nejbližšího souseda [13]. Je to jedna z nejpřesnějších metod. Existují i výrazně rychlejší aproximální metody, ale u nich je možné, že vynechají některé dobré shody.

3.3 Trackování

Cílem trackování [14] je sledování objektu napříč sekvencí mnoha obrazů. Požadavky na trackování jsou především správné detekování zájmových oblastí, stabilita během sledování, přesnost odhadování následující polohy sledovaného objektu a funkčnost v reálném čase. Nejdůležitějším krokem je volba správných vlastností, které musí být schopné detekovat objekty zájmu, aby bylo zajištěno stabilní trackování. Další je potřeba navrhnout

vyhovující pohybový model, který poté zužuje plochu potřebnou k prohledání v následujícím obraze. Čím menší plochu je nutné prohledat pro shodu vlastností, tím lépe. Tímto je dosaženo funkčnosti systému v reálném čase.

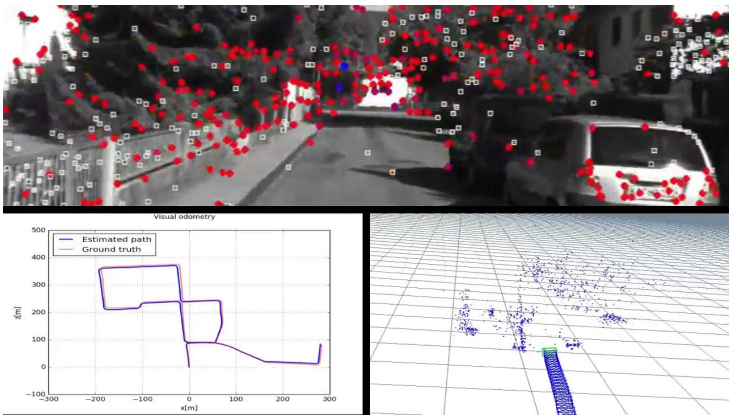
Správný výběr vlastností je prováděn pomoci nějakého z detektorů rohů nebo deskriptorů. Kromě rohů mohou být jako tyto významné body vybírány i bloby.

Pohybový model je nutné použít ke zkrácení času potřebného pro algoritmus. Takový model může být určen přímo uživatelem, nebo učením za pomoci trénovací sekvence.

Shoda a aktualizace stavového modelu je dalším důležitým krokem. Je nutné provést shodu mezi nalezenými vlastnostmi a naměřenými daty a na základě toho model skládající se převážně z polohy a rychlosti tělesa aktualizovat. Existuje tři třídy metod, kterými je možné tohoto docílit a to pravděpodobnostní, deterministické nebo hybridní, které kombinují obě předchozí.

4 Vizuální odometrie

V robotice a počítačovém vidění je vizuální odometrie (VO) [15] proces odhadu pohybu robota, kamery nebo zařízení vybaveného kamerou, získaného z vizuálních dat. Využívá k tomu analýzu snímků kamery, bez nutné apriorní znalosti prostředí, v němž se pohybuje. Používá se v širokém spektru robotických aplikací, přestože zaznamenává pouze trajektorii pohybu, nevytváří mapu. Ukázku takové trajektorie je možné vidět na Obrázku 15.



Obrázek 15: Příklad trajektorie tvořené VO

Odometrie jako taková určovala polohu robotu pomocí polohových senzorů, otáček kol a podobně. Je ale nemožné ji použít například u robotů s nohama a i v případě kol se vyskytují nepřesnosti v souvislosti s podkluzováním nebo problémy způsobenými nerovnostmi povrchu. Vizuální odometrie tyto problémy eliminuje, protože využívá obraz prostředí získávaný z kamer.

Pro odhadování polohy robota je důležité, že je složena ze všech transformací, od časového okamžiku nula, což znamená, že současná poloha je závislá na všech předchozích.

$$X_k = T_k \cdot T_{k-1} \cdot T_{k-2} \cdot ... \cdot T_0 \cdot X_0 \tag{7}$$

kde T je transformace, která se skládá z rotace R a translace t.

V závislosti na počtu zúčastněných kamer se VO [16] dělí na Monocular VO, používající pouze jednu kameru, a Stereo VO, používající kamery dvě. Další dělení je podle metod, které VO využívá. Většina aplikací VO je založena na metodách hledajících konkrétní významné body v obraze, o kterých se píše výše v dokumentu. Díky vývoji je ale již možné používat přímé metody (direct methods), které jako vstupy zpracovávají přímo intenzity jednotlivých pixelů. Existují ovšem i hybridní metody, přechody mezi těmito dvěma typy.

4.1 Detekce a popis významných bodů

V každém obraze je provedena detekce rohů nebo takzvaných blobů, neboli oblastí, což jsou místa, která mají od svého okolí odlišnou intenzitu, barvu a texturu. Každý správný detektor by měl zajistit přesnost lokalizace, opakovatelnost se stejným výsledkem, výpočetní efektivitu, robustnost a invarianci vůči geometrickým i fotometrickým změnám. Tímto se získají body, které jsou relativně stabilní při malých až středních zkresleních obrazu. Je zde spousta detailů, které mění kvalitu a přesnost výsledků, jako například nastavení prahů, výběr filtrů a tak dále. Poté se provede jejich popis pomocí deskriptorů a na základě těchto popisů se může přejít ke kroku hledání shod významných bodů.

4.2 Párování významných bodů

Významné body jsou vždy porovnávány mezi dvojicemi obrazů. V různých přístupech se může lišit velikost plochy, v níž se shody nacházejí. Někdy se deskriptor porovnává se všemi deskriptory v následujícím obraze, jindy je bod porovnán se všemi v jeho okolí, což znamená, že porovnáváme všechny body s vlastnostmi, které jsou mezi sebou v jistém limitu rozdílnosti. Používá se k tomu určité okénko, jehož velikost záleží na nastavení.

Po porovnání podobnosti u jednotlivých prvků je možné, že ve druhém obraze je více bodů, souhlasících s tím v prvním obraze. Proto se provede takzvaná vzájemná kontrola konzistence. Každý prvek je obsažen v nějaké normalizované korelaci prvků ve druhém obraze. Tady se vybere ten, který má s daným prvkem největší normalizovanou korelaci. Pokud je i v seznamu nejvhodnějších párů takto zvoleného prvku, určí se jako jeho shoda.

4.3 Trackování významných bodů

V ohledu na sledování významných bodů je nutné rozdělit VO na dvě podskupiny a to Mono VO a Stereo VO. Každá totiž funguje na odlišném principu. Vhodným algoritmem, používaným v obou případech, může být například RANSAC, který také v praktické části využiji.

Monokulární VO systém

- Nejprve se sledují významné body přes několik obrazů, pak se za pomoci vybraných algoritmů určí odhadované relativní pozice mezi třemi snímky.
- Pomocí prvního a posledního sledování každého bodu, za použití optimální triangulace podle směrové chyby se rozdělí sledované vlastnosti do 3D bodů.
- Trackuje se několika dalšími obrazy, přičemž se vypočítá pozice kamery s ohledem na známé 3D body.
- Pak nastane přerozdělení 3D bodů za použití jejich první a poslední sledované stopy.
- Poslední dva kroky se opakují, dokud je potřeba a následně se opakuje celý proces, přičemž se do něj vkládají takzvané firewally. Kvůli zabránění akumulaci chyb je nutné použít opatření, firewall. Když dojde k chybnému určení pozice, nastane špatné umístění 3D bodů, které se bude jen zhoršovat, protože následující kroky by vycházely z těchto chyb. Proto se použije firewall, což znamená, že následující triangulace 3D bodů nebude počítaná z pozorování za poslední zdí. Místo toho se obraz následující po každé zdi určí jako první.

Stereo VO systém

- Použitím stereo přístupu se proces usnadňuje. Vypadne z něj řešení problému relativní orientace, namísto toho se může rovnou opakovaně rovnou provádět triangulace a pozicování. Tyto relativní pozice už jsou určovány ze známém měřítku, jelikož známe velikost základní hodnoty sterea.
- Začíná se shodou významných bodů mezi pravým a levým obrazem stereo páru, načež se provede triangulace nalezených shod do 3D bodů.

- Pak se významné body trackují několika obrazy. Za použití zvolených algoritmů se určí pozice stereo soupravy. Bodování a iterativní zpřesnění jsou založeny na chybách projekce v obou snímcích stereo páru. Toto se několikrát opakuje.
- Poté se triangulují všechny nové shody z pozorování stereo páru. Od druhého kroku se postup opět několikrát opakován.
- Dále se přerozdělí všechny 3D body a vloží se firewall. Od druhého kroku se proces opakuje, dokud se nedojde k výsledku.
- Pro tento přístup se nevyužívá informace o hloubce přímo z obrazu, které stereo souprava poskytuje, ale určuje se za pomoci vypočítaných 3D bodů. Toto je méně ovlivněno nejasnostmi odhadu hloubky 3D bodů.
- Výhodou oproti Mono VO je, že stereo systém může pracovat správně i bez pohybu kamery, což znamená i větší stabilitu výsledků.

4.4 Výsledky

Testy provedenými v článku Visual Odometry [15] srovnáváním výstupů systému vizuální odometrie s výstupy diferenciálního globálního systému určování polohy (Differential Global Positioning System) a inerciálního navigačního systému (Inertial Navigation System) se došlo k závěru, že odhady polohy VO jsou přesné a spolehlivé i přes množství manévrů na reálných scénářích pohybu pozemních vozidel. Také se prokázala užitečnost mapování a vyhýbání se překážkám, uskutečněné díky VO. K těmto testům nebyla využita žádná apriorní znalost pohybu.

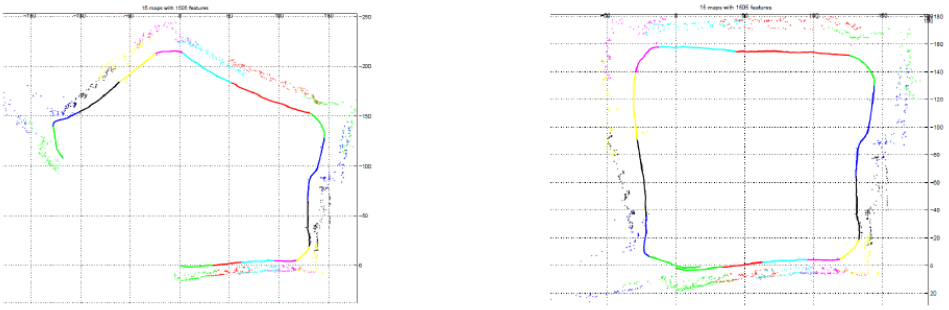
5 Současná lokalizace a mapování

Počítačové vidění je užitečné v mnoha ohledech. Slouží například k navigaci vozidel a robotů. Video senzory a stereo vize se využívají k detekci překážek, k navigaci a odhadu pohybu pro lokalizaci platformy.

Současná lokalizace a mapování neboli Simultaneous Localization and Mapping (SLAM) [17] znamená, že lokalizace a mapování probíhá průběžně, současně. Na konci tohoto procesu je získána trajektorie pohybu kamery a dokonce mapa prostředí. SLAM obecně je často prováděn za použití jiných senzorů než jen kamer.

Vizuální odometrie se dá využít jako základ pro SLAM systém. Zásadní rozdíl je v tom, že VO určuje polohu a vytváří pouze trajektorii, nikoliv i mapu. Je tedy výpočetně méně náročná, nemusí si uchovávat historii o celém pohybu kamery.

SLAM uchovává kompletně celou cestu, kromě trajektorie vytváří i mapu, a dává důraz na konzistenci celé trajekrotie. Visuální odometrie má vůči současné SLAM, nevýhodu toho, že se zaměřuje pouze na lokální konzistenci trajektorie, což znamená, že pokud se časem dostane na začátek, trajektorie se nemusí přesně shodovat, neuzavírá smyčku, jak je možné vidět na Obrázku 16. SLAM si udržuje v paměti všechny minulé polohy, a když se dostane tam, kde už byl, trajektorii upraví, aby seděla a smyčku uzavírala. Na následujícím obrázku je porovnána trajektorie vytvořená VO a trajektorie tvořená SLAM ve stejném prostředí.



Obrázek 16: Trajektorie VO vs SLAM

5.1 Využití SLAM

Využití tato technologie najde například v řízení autonomních robotů a vozidel, nebo v rozšířené realitě. Pro předkládání imaginárních předmětů do reálného prostředí je třeba jeho přesného porozumění, zmapování, a určení polohy pozorovatele.

Je také možné, že doplní systémy GPS pro navigaci za účelem zvýšení přesnosti. SLAM by řešil problémy slepých míst GPS, například uvnitř budov, v tunelech, nebo v jiných místech bez výhledu na nebe. SLAM totiž není závislý na informacích z družic, ale pouze na svých senzorech pro vnímání prostředí, kterými provádí přesná měření fyzikálních veličin světa kolem sebe.

Dalším využitím může být pomoc při záchranných úkolech ve vysoce rizikovém nebo navigačně složitém prostředí. Dále také usnadní planetární, letecké, pozemské a oceánské průzkumy, nebo třeba jisté úkoly v medicíně.

5.2 Vizuální SLAM

Vizuální systémy SLAM (vSLAM) [18] využívají jako jediný zdroj vstupních informací obrazy z kamer, podle kterých určují polohu přístroje v prostředí. Není tedy zapotřebí jiného senzoru než kamery. Už během získávání obrazů vytváří 3D reprezentaci prozkoumávaného prostředí. Cílem těchto systémů je v tedy zmapovat své okolí ve vztahu k jejich vlastní poloze za účelem navigace.

Výhodou tohoto přístupu je, že robot je získáváním obrazů prostředí schopen při dalším nastavení rozpoznávat různé objekty, například lidi, budovy, místa. Kamery jsou také levnější a spotřebovávají méně energie než jiné používané senzory.

I tento přístup má ovšem své nevýhody. Mohou nastávat chyby v datech, zapříčiněné například změnou jasu okolí, nedostatečným rozlišením kamery nebo rozmazáním obrazu v případě rychlého pohybu.

Je mnoho metod, které se pro výpočty polohy stroje a parametrů mapy využívají. Většina systémů SLAM pracuje tak, že sleduje dané body skrz po sobě jdoucí snímky obrazu, aby určovala svou 3D polohu, přičemž tyto informace současně používá k přibližnému odhadu pozice kamery. SLAM systémy založené na pravděpodobnosti k určování pozice kamery využívají pravděpodobnostní rozložení

$$P(x_k, m | Z_{0:k}, U_{0:k}, x_0) \tag{8}$$

kde x je stavový vektor, m lokace orientačního bodu, Z set všech pozorování orientačního bodu, U historie kontrolních vstupů a x_0 je počáteční stavový vektor. Na základě výpočtu pravděpodobností je poté získáván pozorovací a stavový model.

Dá se říct, že SLAM se skládá z několika částí [19]:

- **Inicializace**

  Prvním podstatným krokem je definovat souřadnicový systém pro odhad pohybu kamery a 3D rekonstrukci prostředí, ve kterém se bude pohybovat. Poté probíhá současná lokalizace a mapování, pro nepřetržité odhadování pozice kamery. Pro správný chod je také nutné předem znát kalibraci parametrů kamery.

- **Lokalizace**

  Mapování a lokalizace spolu souvisí. Pomocí hledání významných bodů a jejich párováním se hledá korespondence mezi obrazy a rekonstruovanou mapou, na základě čehož se určuje pozice kamery.

- **Mapování**

  Při mapování pak vzniká mapa pomocí výpočtů 3D struktury neznámého prostředí.

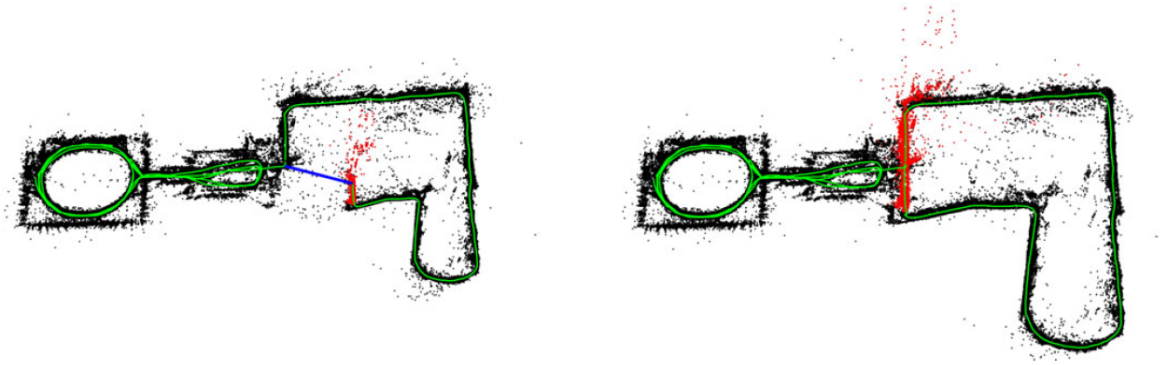
- **Relokalizace**

  Relokalizace je důležitá v případě selhání sledování, k čemuž může dojít například při rozmazání rychlým pohybem nebo jiné chybě. Je potom nutné provést přepočítání pozice kamery vzhledem k mapě. Pokud by se relokalizace neprovedla, systém by přestal fungovat.

- **Optimalizace globální mapy**

  V mapě se během jejího vytváření kumulují chyby vzniklé jejím pohybem. Aby se tyto chyby odstranily a vzniklá mapa se zpřesnila, provádí se takzvaná optimalizace globální mapy, přičemž se provádí její úprava na základě zvážení konzistence

informací o celé mapě. Pokud se v nějakém bodě pohybu robotu dostane na místo, na němž už byl, provede se uzavření smyčky, mapa se upraví tak, aby se trajektorie v tomto bodě setkala. Tato smyčka je poté použita jako zdroj informací pro optimalizaci. Optimalizace mapy je provedena pro opravení pozice kamery a uzavření smyčky je pro získání geometricky konzistentní mapy. Na Obrázku 17 je možné porovnat mapu před a po uzavření smyčky.



Obrázek 17: Uzavření smyčky

Existuje množství různých SLAM systémů [20], z nichž každý najde využití v odlišných úkolech. Příklady některých z nich jsou popsány v nádledujících kapitolách.

5.2.1 Přímé vs nepřímé metody

Metody SLAMu můžeme rozdělit do dvou kategorií podle toho, jakým způsobem využívají informace z přijatého obrazu. Klasifikovat je můžeme do přímých a nepřímých SLAM [21].

Nepřímé systémy SLAM se snaží extrahovat významné body a ty využít k vyhledání kamery, zaznamenávání trajektorie a vytvoření mapy. Těmito body rozumíme geometrické prvky jako rohy, hrany, atd., nebo deskriptory získané například pomocí SIFT, SURF nebo FAST.

Systémy využívající přímé metody pracují přímo s intenzitou pixelů, bez hledání těchto bodů. Přímé metody se snaží přes optimalizaci mapy a parametry kamery získávat informace o hloubce a struktuře prostředí. Výhodou oproti nepřímým metodám je, že čas, který nepřímé metody stráví extrahováním významných bodů, mohou využít k dalším výpočtům a mít stejnou snímkovou frekvenci.

Nepřímé metody však mají mnohem větší toleranci vůči změnám osvětlení prostředí, protože nepracují přímo s intenzitou. Také náročnost nepřímých metod je výrazně nižší. Přímé metody sice nemusí extrahovat významné body, ale musí obecně zpracovat větší množství informací, protože pracují se všemi body v obraze a proto jsou náročnější.

5.2.2 Mono SLAM

Mono SLAM pro získávání obrazů prostředí využívá pouze jednu kameru. Tento přístup využívá pravděpodobnostní mapu založenou na vlastnostech, která sleduje jak odhady, tak nejistotu stavů kamery, kterými jsou poloha, úhlová rychlost, rychlost a orientace, a stav všech sledovaných prvků. Takto vzniklá mapa se neustále vyvíjí, přepočítávají se její funkce a během pohybu se aktualizují odhady, nejistoty a pozorování objektů na základě rozšířeného Kalmanova filtru. V něm jsou formou stavového vektoru zaznamenány polohy

3D bodů a stupeň volnosti. Počáteční mapa je vytvořena pozorováním známého objektu, u kterého je definován globální souřadnicový systém.

Proces se skládá se ze dvou hlavních částí:

- Inicializace mapy na základě známého objektu
- Odhadování pozice 3D bodů a pohybu kamery pomocí Kalmanova filtru.

Problém tohoto systému spočívá v jeho náročnosti, která výrazně roste, čím je prostředí větší.

5.2.3 PTAM

Paralelní sledování a mapování neboli Parallel Tracking and Mapping (PTAM) je kamerový sledovací systém pro rozšířenou realitu. Jako SLAM nevyžaduje žádné značky, předem připravené mapy ani známé šablony. Jde o jednu z imlplementací Mono SLAM.

Sledování kamery a mapování prostoru je spuštěno paralelně na různých vláknech vícejádrového procesoru, přesněji využívá vlákna dvě. ORB SLAM, o němž se bude zmiňovat jedna z následujících podkapitol, využívá dokonce tři.

Protože se PTAM využívá pro rozšířenou realitu, je důležité, aby byly mapy vykresleny s velmi dobrou grafikou, co nejpřesnější a nejbohatší. K dosažení dostatečného množství času pro tento úkol se mapa neaktualizuje po každém snímku, ale pouze u klíčových.

PTAM má ale stále ještě několik problémů, které jeho použití pro AR komplikují. Největším problémem je, že systém pro výpočty vyžaduje silný hardware, který ve většině mobilních zařízeních zatím není. Určování ruční neboli volně pohyblivé kamery, je také mnohem složitější, než sledování kamery pevně připojené k robotu. Další problémy mohou nastat, když dojde k samo-okluzi, což je místo, kde část objektu zakrývá jinou část stejného objektu, která je sledována z kamery.

5.2.4 RGB-D SLAM

Tento SLAM [22] využívá RGB-D kamery. To jsou zařízení pro snímání hloubky, která pracují ve spojení s kamerou RGB. Dokáží rozšířit obraz o informace o hloubce. Jsou využívána především ve vnitřních prostorách, protože mají omezení, do kterého jsou schopna hloubku určovat. SLAM poté pro určování polohy v prostředí využívá vytvořené hloubkové mapy.

5.2.5 ORB-SLAM

ORB-SLAM je monokulární SLAM systém založený na významných bodech, který pracuje v reálném čase, v malém i velkém, vnitřním i venkovním prostředí. Systém je robustní pro častý zmatený pohyb. Umožňuje uzavření a přemístění smyčky základní trajektorie v mapě. Nejdůležitější změnou oproti PTAMu je, že ORB-SLAM využívá tři vlákna vícejádrového procesoru. Z názvu je zřejmé že pro vyhledávání a popis vlastností využívá ORB algoritmus.

5.2.6 ORB-SLAM2

ORB-SLAM2 je kompletní Open-Source SLAM systém pro monokulární, stereo a RGB-D kamery, včetně možnosti opětovného použití mapy, uzavření smyčky a přemístění, jehož

autory jsou Raul Mur-Artal a Juan D. Tardos. Systém pracuje v reálném čase na standardních procesorech v široké škále prostředí. Umožňuje přesný odhad trajektorie s přesným měřítkem. Metoda je velice přesná, většinou je nejpřesnějším řešením SLAM.

5.3 Shrnutí

Přístroj využívající vSLAM je schopen zároveň přijímat obrazy, vytvářet mapu a určovat vlastní polohu v prostředí. Tyto schopnosti se využívají v mnoha situacích ulehčujících práci nebo jiné úlohy.

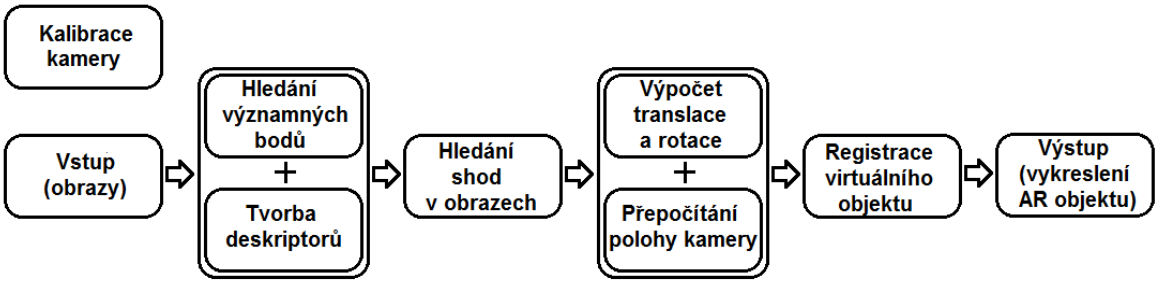
Snímat polohu kamery a okolí v okolí bez znalosti předem daných bodů je neuvěřitelně obtížné. Systémy vSLAM se však při řešení této výzvy ukázaly jako velmi efektivní a objevují se jako jedna z nejdokonalejších dostupných technologií vestavěného vidění. Proto jsou také vhodné pro použití v AR. Kromě faktu, že pracují v reálném čase, jsou velmi dobře schopné odhadovat polohu kamery, což je pro AR systémy důležitým požadavkem, jak je zmíněno v kapitolách výše. AR je pak schopno v reálném čase mísit reálné obrazy s virtuálními objekty a ty přesně umisťovat na své místo.

6 Praktická část

6.1 Návrh AR systému

Cílem této práce je návrh systému rozšířené reality, jehož součástí je vizuální odometrie a zobrazení virtuálního předmětu, 2D i 3D, do skutečného prostředí. Při jejím návrhu jsem vycházela z metod popsaných v předchozích kapitolách. Všechny kódy pro Python použité v této práci jsou uloženy v GitHub repozitáři[1].

K vytvoření funkčního systému pro rozšířenou realitu je nutné výše zmíněné metody a algoritmy kombinovat. Je potřeba provést detekci a popis důležitých bodů, určovat polohu kamery a do prostředí vložit virtuální objekt. Pro lepší ilustraci procesu poslouží graf s návrhem AR systému — viz Obrázek 18.



Obrázek 18: Schéma AR

Na schématu je možné vidět pouze jednu iteraci programu. Jednotlivé kroky, kromě kalibrace kamery, probíhají v cyklu stále dokola. Jsou blíže popsány a samostatně otestovány v následujících kapitolách.

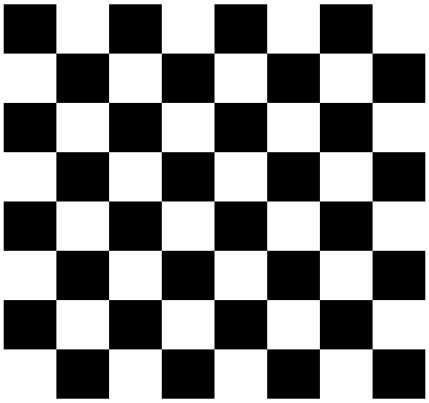
6.2 Kalibrace kamery

Prvním důležitým krokem je kalibrace kamery. Čočka totiž způsobuje zakřivení, které se musí kompenzovat výpočty. Kalibrací je tedy možné získat parametry kamery, které se používají v dalších bodech a bez nichž by nebylo možné provést následující úkony správně a přesně. Mezi ně patří matice kamery, popisující ohniskovou vzdálenost a optická centra, a parametry zakřivení. Parametrů zakřivení je celkem pět a matice kamery vypadá následovně:

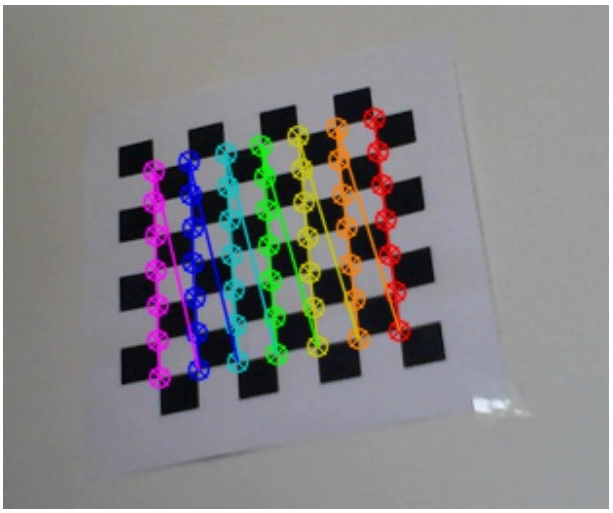
$$C_M = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

kde hodnoty f_x a f_y značí ohniska a c_x a c_y optická centra. K dosažení tohoto výsledku se využívá vytištěný vzor šachovnice – viz Obrázek 19a, za jehož pomoci je možné určit kalibrační parametry. Základní kód i s vysvětlením je možné nalézt v oficiálních tutoriálech OpenCV pro Python. Kalibrace může být provedena mimo hlavní program, dále budou potřeba jen samotné parametry, které se nemění. Na Obrázku 19b je zobrazeno zvýraznění vnitřních rohů šachovnice, které slouží pro kalibraci.

[1] https://github.com/AnVarackova/BakalarskaPraceAR



(a) Vzorová šachovnice



(b) Šachovnice po nalezení rohů

Obrázek 19: Kalibrace kamery

6.3 Vykreslení objektů

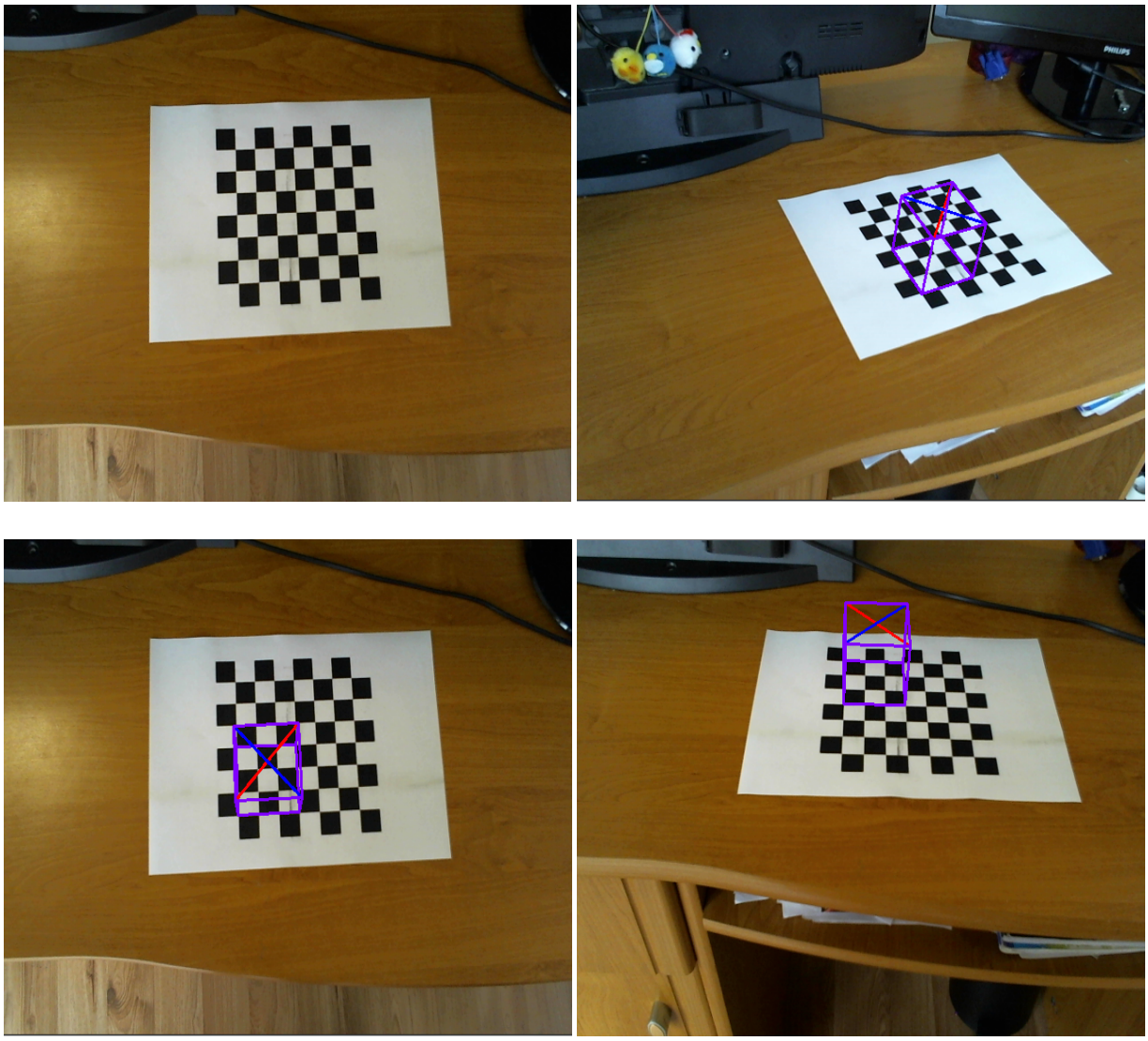
Jak je popsáno v kapitole 1.7, existují dvě cesty jak přistupovat k vnímání prostředí. Jedna využívá značky a druhá se orientuje pouze podle předem neoznačených významných bodů v obraze. Pro tento úkol značky využívám. Vytvořila jsem dva různé algoritmy, první detekuje v obraze šachovnici a druhý aruco kód. Volba virtuálního objektu, který má být vykreslen do reálné scény, je libovolná a může se jednat o 2D nebo 3D objekt. Pro tuto práci použiji oba druhy pro lepší ilustraci. Po zvolení objektu je nutné provést registraci, kterou je zde myšleno nalezení správného místa, kam virtuální obraz umístit. Registrace je blíže popsána v kapitole 1.6.. K samotnému zobrazení objektu do obrazu reálného prostředí slouží rendering, neboli vykreslování, což je proces převádějící počítačový model do zdánlivě reálného obrazu.

V prvním kroku do obou algoritmů přicházejí obrazy z kamery, pro kterou již mám předem zjištěné kalibrační parametry a matici kamery. Kód pro získání kamerových snímků je založený na OpenCV tutoriálu.

6.3.1 Šachovnice

V tomto případě jsem jako marker použila vytištěnou šachovnici, stejnou jako pro kalibraci, a položila ji na stůl. Poté jsem nasnímala několik snímků z různých úhlů a vzdáleností od značky. V OpenCV knihovně jsou již vytvořeny funkce pro automatické nalezení vzoru šachovnice, které jsem používala i v kódu pro kalibraci. Dále je třeba napsat funkci, která vytvoří zvolený objekt, a poté ho vykreslit do obrazu. Pro ukázku jsem si vybrala obrys krychle s modrou a červenou úhlopříčkou vrchní strany, aby bylo možné pozorovat i její natočení.

Na Obrázku 20 jsou zobrazeny výsledky tohoto algoritmu. Na prvním obrázku je vidět mnou zvolená značka na stole a poté několik pohledů na vykreslenou 3D krychli. Je zřejmé, že šachovnicový marker není nejpřesnější. Kromě výskytu krychle v různých rozích je možné si podle modré a červené orientační čáry všimnout i rozdílného natočení. Je to způsobeno tím, že je šachovnice symetrická. Proto algoritmus nedokáže rozhodnout vždy stejně, kde je její počáteční a kde koncový bod.



Obrázek 20: Reálné prostředí se šachovnicí a virtuálním objektem

6.3.2 Aruco kód

V této části jsem jako marker zvolila aruco kód, což je obdoba QR kódu. Jeho výhodou je, že je snadné určit jeho přesné natočení a tedy i natočení vkládaného objektu. Jako základ pro svůj kód jsem použila již existující kód[2] na GitHubu. V tomto případě jsem nejprve zvlášť načetla obrázek. který jsem chtěla zobrazit, a soubor s kódy, později vytištěnými a vloženými do reálného prostředí, jak je možné vidět v první části obrázku 20. Poté je nutné změnit velikost obrázku, aby dobře překryl vytištěný kód. V knihovnách OpenCV je opět možné nalézt již hotové funkce pro detekci aruco kódů, které jsem použila.

V předchozím úkolu jsem krychli vkládala pouze do obrázků, ale v tomto případě je vložený obrázek vidět již přes webkameru ve vykreslovaném videu. Z Obrázku 21 je zřejmé, že natočení virtuálního obrazu zůstává stejné, ať se uživatel dívá z jakékoliv strany.

[2]https://github.com/marcelogdeandrade/AugmentedReality

Obrázek 21: Reálné prostředí s aruco kódem a s virtuálním objektem

6.4 Aplikace vizuální odometrie/SLAMu

V tomto bodě jsem se zaměřila převážně na vizuální odometrii, což je podmnožina vizuálního SLAMu. Pro účel této práce jsem použila pouze jednu kameru. Princip mono-VO je vysvětlen v kapitole 4, proto v následujících odstavcích jen stručně popíšu, jak program funguje. Jako základ jsem použila kód[3] na GitHubu psaný v jazyce C. Přepsala jsem ho do Pythonu a upravila. Pro doladění jsem se inspirovala i kódem[4] psaným v Pythonu.

Jako vstup algoritmu je proud obrazů, které přicházejí z kamery. V nich jsou nalezeny významné body, k čemuž se využívá nějakého z detektorů popsaných v kapitole Techniky zpracování obrazů. Pro tuto práci je použit FAST algoritmus.

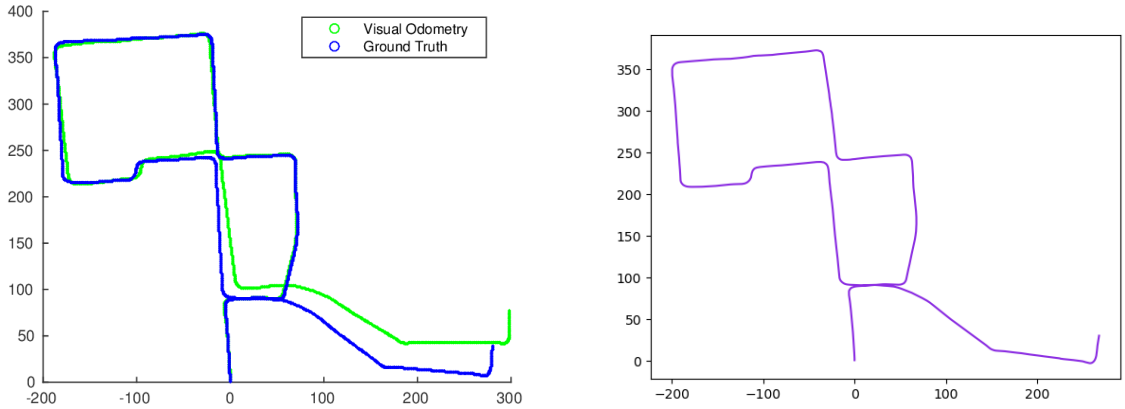
Dalším krokem je nalézt shody mezi obrazy. Vizuální odometrie v kombinaci s RANSAC algoritmem umožňuje hledat tyto shody a na základě nich určovat rotaci a translaci mezi nimi. Z těchto informací je pak možné přepočítávat polohu kamery, protože translace vypovídá o posunu a rotace o změně úhlu.

[3] https://github.com/avisingh599/mono-vo
[4] https://github.com/uoip/monoVO-python

V průběhu se vykresluje trajektorie a ukládají se polohy kamery, aby se mohly na konci vykreslit souřadnice pohybu. Nejprve jsem pro ověření správnosti algoritmu použila dataset[5], pro který jsou již známé výsledky. Na obrázku 22 je ukázka dvou snímků.

Obrázek 22: Ukázka z kitti datasetu

Pro tento dataset jsou známy parametry kamery i to, jak by měla vypadat výsledná trajektorie. Ta je s mým výsledkem porovnána na Obrázku 23.



(a) Výstup kódu z GitHubu

(b) Výstup mého algoritmu

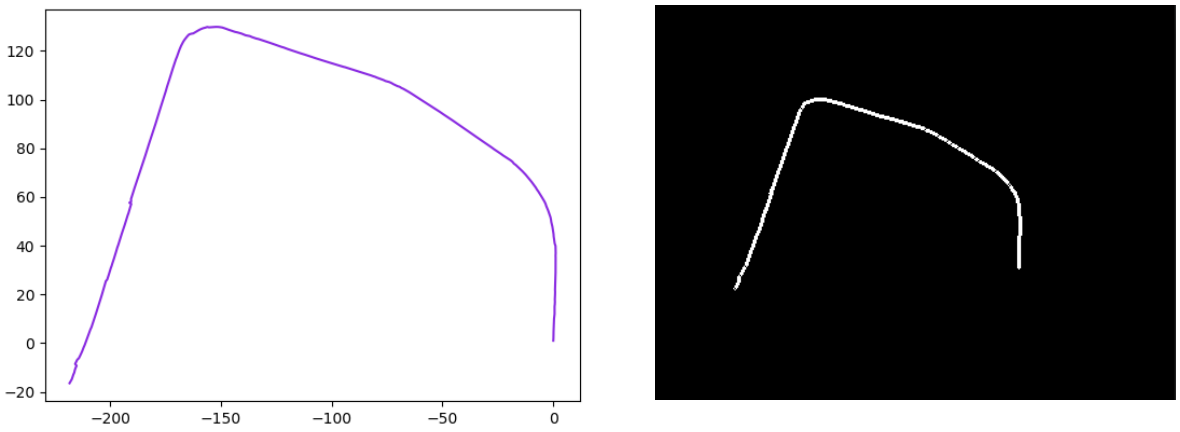
Obrázek 23: Porovnání

Po ověření správnosti jsem použila vlastní data získané během jízdy ulicí. Zvolila jsem jednodušší trajektorii než ve výše uvedeném příkladu, Data obsahují jízdu rovně následovanou jednou mírnou a jednou ostrou pravotočivou zatáčkou. Pro představu je na Obrázku 24 ukázka 4 fotek z mého datasetu.

Obrázek 24: Ukázka vlastního datasetu

[5]http://www.cvlibs.net/datasets/kitti/eval_odometry.php

V kódu pro mono-VO jsem změnila parametry kamery, aby odpovídaly mnou použitému zařízení. Místo výpočtu měřítka v každém kroku, bylo měřítko nastaveno fixně na hodnotu 1, což má za následek, že nemusí odpovídat vzdálenosti, ale tvar trajektorie zůstane stejný. Na Obrázku 25 je vidět výsledek algoritmu pro má data, jak vykreslená trajektorie tvořící se v průběhu programu, tak graf spojující všechny uložené body, vykreslený po jeho konci.



Obrázek 25: Trajektorie pro vlastní dataset

7 Závěr

Ve své práci jsem se věnovala především sepsání rešerše o rozšířené realitě a o tom, jaké jsou možnosti jejího využití. Popsala jsem různé podskupiny rozšířené reality, jako je vizuální odometrie, SLAM nebo i vykreslování virtuálních objektů. Vysvětlila a porovnala jsem také různé algoritmy a metody pro získávání významných bodů v obraze, jejich popis a sledování.

V praktické části jsem navrhla systém AR — viz Obrázek 17 a testovala její části implementačně. Na obou úkolech týkajících se vykreslování virtuálních objektů jsem chtěla především ilustrovat, že je možné vykreslovat více typů objektů a také to, že velmi záleží na volbě markeru. Vzor, který je z více úhlů stejný není dostatečně přesný, na rozdíl od kódů rozlišitelných z různých stran. Ve druhé části jsem vytvořila aplikaci vizuální odometrie, jejíž výstup je trajektorie pohybu kamery. Ověřila jsem její přesnost na oficiálním datasetu a poté jsem vykreslila trajektorii z vlastních dat. Všechny kódy k praktické části jsou uloženy na internetu v repozitáři na GitHubu[6].

Tuto práci použiji jako základ pro diplomovou práci, jejíž cílem bude navrhnout komplexnější AR systém. Jednak plánuji implementovat AR systém bez využití markerů, aby fungoval pouze na základě hledání předem neznámých významných bodů z obrazových dat jako například systémy PTAM či ORBSLAM2 uvedené v kapitole 5. Dalším aspektem je měřítko, které jsem v aktuální práci neřešila do hloubky, a které je obtížné získat použitím pouze jedné kamery. Proto bych ráda využila stereovizi, neboli dvě kamery, ze které lze měřítko vypočítat. Díky tomu bude možné tvořit mapy, které přibližně odpovídají realitě nejen tvarem trajektorie, ale i měřítkem.

[6] https://github.com/AnVarackova/BakalarskaPraceAR

Reference

[1] R. T. Azuma, "A survey of augmented reality," *Presence: Teleoperators & Virtual Environments*, vol. 6, no. 4, pp. 355–385, 1997.

[2] D. Chatzopoulos, C. Bermejo, Z. Huang, and P. Hui, "Mobile augmented reality survey: From where we are to where we go," *Ieee Access*, vol. 5, pp. 6917–6950, 2017.

[3] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre, "Recent advances in augmented reality," *IEEE computer graphics and applications*, vol. 21, no. 6, pp. 34–47, 2001.

[4] D. Van Krevelen and R. Poelman, "A survey of augmented reality technologies, applications and limitations," *International journal of virtual reality*, vol. 9, no. 2, pp. 1–20, 2010.

[5] E. Marchand, H. Uchiyama, and F. Spindler, "Pose estimation for augmented reality: a hands-on survey," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 12, pp. 2633–2651, 2015.

[6] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*. IEEE, 2018, pp. 1–10.

[7] M. Hrúz, "Corner detection, sift, surf," http://www.kky.zcu.cz/uploads/courses/mpv/03/materialy03.pdf.

[8] OpenCV, "Feature detection and description," https://docs.opencv.org/3.4.0/db/d27/tutorial_py_table_of_contents feature2d.html.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[10] Q. Zhu and Z. Lei, "Kaze algorithm applied in augmented reality," in *2015 International conference on Applied Science and Engineering Innovation*. Atlantis Press, 2015.

[11] O. Chum, J. Matas, and J. Kittler, "Locally optimized ransac," in *Joint Pattern Recognition Symposium*. Springer, 2003, pp. 236–243.

[12] D. Askey, A. Bertapelli, and C. Rawley, "Nearest neighbor edge selection from feature tracking," Nov. 7 2002, uS Patent App. 09/847,864.

[13] C. Schmid, "Efficient visual search of local features," https://lear.inrialpes.fr/ verbeek/mlcr.slides. 11.12/visual_search_gre1.pdf.

[14] R. P. Pflugfelder, "A comparison of visual feature tracking methods for traffic monitoring," *ÖGAI Journal*, vol. 19, no. 4, pp. 15–22, 2000.

[15] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1. Ieee, 2004, pp. I–I.

[16] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[17] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[18] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial intelligence review*, vol. 43, no. 1, pp. 55–81, 2015.

[19] H. Lategahn, A. Geiger, and B. Kitt, "Visual slam for autonomous ground vehicles," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 1732–1737.

[20] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual slam algorithms: a survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, p. 16, 2017.

[21] G. Silveira, E. Malis, and P. Rives, "An efficient direct approach to visual slam," *IEEE transactions on robotics*, vol. 24, no. 5, pp. 969–979, 2008.

[22] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 2100–2106.

Zdroje obrázků

Obrázek 1 https://arvr247.com/the-promises-of-augmented-reality-in-medical-science/

Obrázek 2 https://www.engineering.com/AdvancedManufacturing/ArticleID/14904/What-Can-Augmented-Reality-Do-for-Manufacturing.aspx

Obrázek 3 https://www.quora.com/q/dqltreumazaxmfvq/Augmented-Reality-Apps-Changing-the-Way-People-Consume-Content-and-Interact

Obrázek 4 https://cxocard.com/what-is-augmented-reality-gaming/

Obrázek 5 http://www.absolute-knowledge.com/what-is-augmented-reality/

Obrázek 9 https://www.researchgate.net/figure/Diagram-of-SIFT-keypoint-detection-algorithm-showing-one-octave-with-6-Gaussian-image-fig-1256546531

Obrázek 10 https://www.opencv-srf.com/2018/03/homogeneous-blur.htmla https://www.opencv-srf.com/2018/03/gaussian-blur.html

Obrázek 11 https://cs.wikipedia.org/wiki/Integr%C3%A1ln%C3%AD_obraz#/media/Soubor:Integralni_obraz.png

Obrázek 12 https://docs.opencv.org/3.4.0/df/d0c/tutorial-py-fast.html

Obrázek 13 https://www.researchgate.net/figure/An-example-of-ORB-descriptor-similar-to-SIFT-and-SURF-keypoints-detect-identical-fig3-329571493

Obrázek 15 https://i.ytimg.com/vi/tP1GFapGalQ/maxresdefault.jpg

Obrázek 16 https://blog.csdn.net/wb790238030/article/details/90770801

Obrázek 17 https://sites.google.com/a/korea.ac.kr/intelligent-robot-laboratory/research/navigation/visual-slam